TPC Benchmark™ C

Full Disclosure Report for

PRIMERGY N400

Using Microsoft SQL Server 2000 Enterprise Edition

and Microsoft Windows 2000 Advanced Server

September 5, 2000

First Edition

First Edition September 5, 2000

Fujitsu Siemens believes that the information in this document is accurate as of the publication date. The information in this document is subject to change without notice. We assume no responsibility for any errors that may appear in this document. The pricing information in this document is believed to accurately reflect the current prices as of the publication date. However, we provide no warranty of the pricing information in this document.

Benchmark results are highly dependent upon workload, specific application requirements, system design and implementation. Relative system performance will vary as a result of these and other factors. Therefore, TPC Benchmark™ C should not be used as a substitute for a specific customer application benchmark when critical capacity planning and/or product evaluation decisions are contemlated.

All performance data contained in this report were obtained in a rigorously controlled environment. Results obtained in other operating environments may vary significantly. We do not warrant or represent that a user can or will achieve similar performance expressed in transactions per minute (tpmC) or normalized price/performance (€/tpmC). No warranty of system performance or price/performance is expressed or implied in this report.

Copyright © 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

Permission is hereby granted to reproduce this document in whole or in part provided the copyright notice printed above is set forth in full text on the title page of each item reproduced.

PRIMERGY N400, PRIMERGY 870 and PRIMERGY 170 are trademarks of Fujitsu Siemens Computers GmbH.

Microsoft, Windows 2000, SQL Server and Benchcraft are registered trademarks of Microsoft Corporation.

Pentium®III XEON is a registered trademark of Intel.

TPC Benchmark™ is a trademark of the Transaction Processing Performance Council (TPC).

Other product names mentioned in this document may be trademarks and/or registered trademarks of their respective companies.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

Preface

The Transaction Processing Performance Council (TPC), of which Fujitsu Siemens Computers GmbH is a member, is an organization of computer companies, dedicated to the development of objective, industry-wide performance metrics in the area of transaction processing. Fujitsu Siemens Computers GmbH is involved in this effort, participating on the council and utilizing TPC benchmarks in performance evaluation.

The TPC Benchmark™ C Standard Specification was developed by the Transaction Processing Performance Council. This benchmark exercises the system components necessary to perform tasks associated with that class of on-line transaction processing (OLTP) environments emphasizing a mixture of read-only and update intensive transactions. This is a complex OLTP application environment exercising a breadth of system components associated by such environments characterized by:

- The simultaneous execution of multiple transaction types that span a breadth of complexity
- On-line and deferred transaction execution modes
- Multiple on-line terminal sessions
- Moderate system and application execution time
- Significant disk input/output
- Transaction integrity (ACID properties)
- Non-uniform distribution of data access through primary and secondary keys
- Databases consisting of many tables with a wide variety of sizes, attributes, and relationships
- Contention on data access and update

This benchmark defines four on-line transactions and one deferred transaction, intended to emulate functions that are common to many OLTP applications. However, this benchmark does not reflect the entire range of OLTP requirements. The extent to which a customer can achieve the results reported by a vendor is highly dependent on how closely TPC-C approximates the customer application. The relative performance of systems derived from this benchmark does not necessarily hold for other workloads or environments. Extrapolations to any other environment are not recommended.

Benchmark results are highly dependent upon workload, specific application requirements, system design and implementation. Relative system performance will vary as a result of these and other factors. Therefore, TPC-C should not be used as a substitute for a specific customer application benchmarking when critical capacity planning and/or product evaluation decisions are contemplated.

The performance metric reported by TPC-C is a "business throughput" measuring the number of orders processed per minute. Multiple transactions are used to simulate the business activity of processing an order , and each transaction is subjected to a response time constraint. The performance metric for this benchmark is expressed in transactions-per-minute-C (tpmC). To be compliant with the TPC-C standard, all references to tpmC results must include the tpmC rate, the associated price-per-tpmC, and the availability date of the priced configuration.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

Summary

This report documents the TPC Benchmark™ C results achieved by the Fujitsu Siemens Computers GmbH using Microsoft SQL Server 2000 Enterprise Edition.

The TPC Benchmark™ C tests were run on a PRIMERGY N400 system using the Windows 2000 Advanced Server operating system.

The results, summarized below, show the number of TPC Benchmark™ C transactions per minute (tpmC) and the price per tpmC (€/tpmC).

Software	Hardware	tpmC	€/tpmC
Microsoft SQL Server 2000 Enterprise Edition, Windows 2000 Advanced Server	Fujitsu Siemens Computers GmbH PRIMERGY N400	34,150.87	17.81€

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

FUJITSU COMPUTERS SIEMENS

PRIMERGY N400

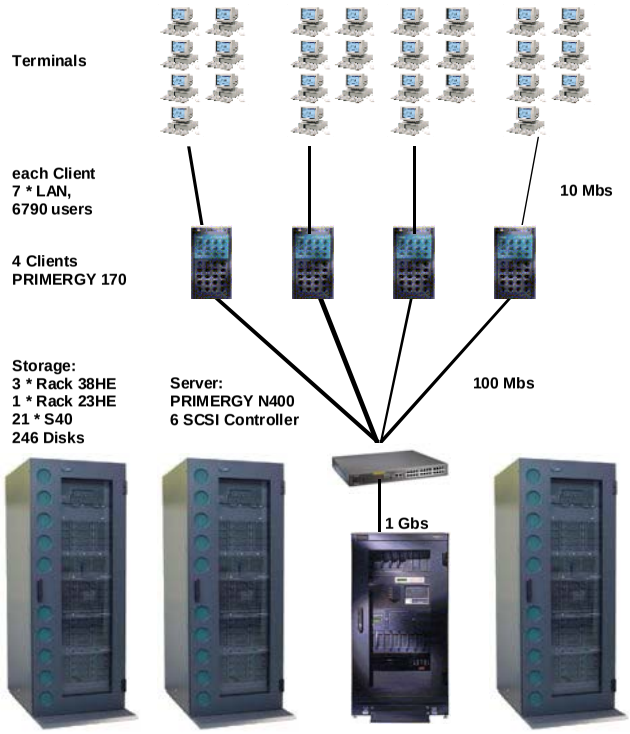
C/S with 4 PRIMERGY 170

TPC-C REV 3.5
EXECUTIVE SUMMARY

Report Date: September 5, 2000

Total System Cost	TPC-C Throughput	Price/Performance	Availability Date
€ 608,226	34,150.87 tpmC	€17.81/tpmC	November 1, 2000

Processors	Database Manager	Operating-System	Other Software	Number of Users
4 Intel Pentium® III Xeon 700 MHz	Microsoft SQL Server 2000 Enterprise Edition	Microsoft Windows 2000 Advanced Server	Windows 2000 Server, IIS 5.0 and COM+	27,160



System Components	Qty/Srv.	1 PRIMERGY N400	Qty/Client	4 PRIMERGY 170
Processors	4	Intel Pentium® III Xeon 700 MHz with 2 MB SLC	1	Intel Pentium® III 750 MHz with256 KB SLC
Memory	8	GB	256	MB
Disk Controller	6	Mylex eXtremeRAID 2000	1	SCSI Controller
Disk Drives	145	9 GB	1	9 GB
	96	18 GB		
	6	36 GB		
Total GB of Storage	1	2,789 GB	1	9 GB

FUJITSU COMPUTERS SIEMENS

PRIMERGY N400

C/S with 4 PRIMERGY 170

TPC-C REV 3.5
EXECUTIVE SUMMARY

Report Date: September 5, 2000

Description	Part Number	Third Party Brand	Pricing	Unit Price	Qty.	Extended Price	5yr Maint. Price
PRIMERGY N400 GE Rack 700MHz 2MB	SNP:SY-K596V202-P		1	10.530 Euro	1	10.530 Euro	
Pentium III Xeon Processor 700MHZ/2MB	SNP:SY-F2327E702-P		1	6.615 Euro	3	19.845 Euro	
Memory 2 GB SDRAM 100MHZ (4 Mod.)	SNP:SY-F2174E545-P		1	7.680 Euro	4	30.720 Euro	
Gigabit Ethernet PCI 32/64	D:RM6T5-CL18		1	3.107 Euro	1	3.107 Euro	
Tape Drv SLR50 , 25GB	SNP:SY-F1835E2-A		1	1.575 Euro	1	1.575 Euro	
9GB/10K LVD-SCSI, Hot Plug	SNP:SY-F1899E109-P		1	480 Euro	1	480 Euro	
23 HE Rack	SNP:SY-K476V201-P		1	1.800 Euro	1	1.800 Euro	
Primergy S40 ES 1 channel	SNP:SY-K538V501-P		1	2.655 Euro	1	2.655 Euro	
S40 3 channels kit	SNP:SY-F2015E3-P		1	414 Euro	1	414 Euro	
36GB/10K, Hot Plug	SNP:SY-F1899E136-P		1	1.643 Euro	6	9.858 Euro	
RAID-Ctrl, PCI, 4-Ch, BBU	SNP:SY-F2190-E128		1	3.150 Euro	6	18.900 Euro	
APC USV 3000VA	SNP:PS-E399L1-P		1	1.598 Euro	1	1.598 Euro	
CD-ROM ATAPI 1/2"	SNP:SY-2313E1-P		1	135 Euro	1	135 Euro	
Power Supply Module 500W (add)	SNP:PS-F596E1-P		1	405 Euro	1	405 Euro	
Tastatur KBPC B Light Basic (D)	S26381-K271-V320		1	29 Euro	1	29 Euro	
Country Pack	SNP:SY-F1699B831-P		1	40 Euro	1	40 Euro	
Monitor 154V	S26361-K605-V150		1	240 Euro	1	240 Euro	
	Server Hardware Subtotal		1			102.331 Euro	20.604 Euro
Data Center Rack 38 HE	SNP:SY-K614V102-P		1	1.755 Euro	3	5.265 Euro	5.675 Euro
18GB/10K, Hot Plug (+10% spares)	SNP:SY-F1899E118-P		1	840 Euro	106	89.040 Euro	
9GB/10K, Hot Plug (+10% spares)	SNP:SY-F1899E109-P		1	480 Euro	159	76.320 Euro	
Primergy S40 ES 1 channel (+ 10% spares)	SNP:SY-K538V501-P		1	2.655 Euro	22	58.410 Euro	
SCSI Cable UHD-HD 5m (+10% spares)	SNP:SY-F1947L50-A		1	135 Euro	22	2.970 Euro	
	Storage Subtotal		1			232.005 Euro	
PRIMERGY 170 GE F, PIII 750, SCSI	SNP:SY-K549V146-A		1	1.463 Euro	4	5.852 Euro	5.400 Euro
Keyboard KBPC	S26381-K271-V320		1	29 Euro	4	116 Euro	
Country Pack	SNP:SY-F1699B401-A		1	40 Euro	4	160 Euro	
Memory 128 MB SDRAM ECC	S26361-F1840E515		1	428 Euro	8	3.424 Euro	
HD 9GB, U2W	SNP:SY-F2069E9-A		1	428 Euro	4	1.712 Euro	
Fast Ethernet 2x 10/100 Duralink64 PCI	SNP:SY-F2217E1-P		1	405 Euro	16	6.480 Euro	
Monitor 154V	S26361-K605-V150		1	240 Euro	4	960 Euro	440 Euro
	Client Hardware Subtotal		1			18.704 Euro	
Microsoft Windows 2000 Adv. Server, 25 CAI	C10-00164		1	2.737 Euro	1	2.737 Euro	
MS SQL-Server 2000 Ent.Edit. Per Proc Lic.	810-00845		1	19.718 Euro	4	78.872 Euro	
	Server Software Subtotal					81.609 Euro	17.260 Euro
Microsoft Windows 2000 Server, incl. 25 CAL	C11-00313		1	843 Euro	4	3.372 Euro	
Microsoft Visual C++ Professional 6.0	048-00426		1	482 Euro	1	482 Euro	
	Client Software Subtotal					3.854 Euro	Incl. Above(svr)
8xTP, 1x Coax 10Mbit Hub	Artk-Nr 354	K&M	2	31 Euro	3758	116.498 Euro	
8x10/0100Mbit Switch	DES-3226G	D-Link	3	876 Euro	3	2.628 Euro	
Gigabit Uplink (+10% spare)	DES-3251G	D-Link	3	406 Euro	3	1.218 Euro	
	User Connectivity Subtotal					120.344 Euro	
	Total					558.847 Euro	49.379 Euro

Five-Year Cost of Ownership	608.226 Euro
tpmC Rating	34.150,87
Euro / tpmC	17,81 Euro

1=Fujitsu-Siemens, 2=K&M Elektronik, 3=D-Link

Prices used in TPC bechmarks reflect the actual prices a customer would pay for a one-time purchase of the stated components. Individually negotiated discounts are not permitted. Special prices based on assumptions about past or future purchases are not permitted. All discounts reflect standard pricing policies for the listed components. For complete details, see the pricing section of the TPC benchmark pricing specifications. If you find that the stated prices are not available according to these terms, please inform the TPC at pricing@tpc.org. Thank you.

Five-Year Cost of Ownership:	€608,226
tpmC Rating:	34,150.87
€ / tpmC:	17.81

Note: The benchmark results and test methodology were audited by Bradley J. Askins of InfoSizing

Numerical Quantities Summary

MQTh, computed Maximum Qualified Throughput			34,150.87 tpmC
% throughput difference, reported & reproducibility runs 0.16 %			
Response Times (in seconds)	**90th percentile**	**Average**	**Maximum**
- New-Order	**0.40**	**0.28**	**6.86**
- Payment	**0.31**	**0.20**	**2.00**
- Order-Status	**0.34**	**0.22**	**8.82**
- Delivery (interactive portion)	**0.12**	**0.11**	**0.79**
- Delivery (deferred portion)	**0.35**	**0.20**	**0.86**
- Stock-Level	**1.48**	**1.02**	**2.81**
- Menu	**0.12**	**0.11**	**0.90**
Transaction Mix, in percent of total transactions			
- New-Order			**44.83 %**
- Payment			**43.00 %**
- Order-Status			**4.06 %**
- Delivery			**4.05 %**
- Stock-Level			**4.06 %**
Emulation Delay (in seconds)		**Response Time**	**Menu**
- New-Order		**0.1**	**0.1**
- Payment		**0.1**	**0.1**
- Order-Status		**0.1**	**0.1**
- Delivery (interactive)		**0.1**	**0.1**
- Stock-Level		**0.1**	**0.1**
Keying/Think Times (in seconds)	**Minimum**	**Average**	**Maximum**
- New-Order	**18.00/0.00**	**18.01/12.05**	**18.04/120.50**
- Payment	**3.00/0.00**	**3.01/12.04**	**3.04/120.50**
- Order-Status	**2.00/0.00**	**2.01/10.01**	**2.03/100.50**
- Delivery (interactive)	**2.00/0.00**	**2.01/ 5.09**	**2.03/ 50.49**
- Stock-Level	**2.00/0.00**	**2.01/ 5.04**	**2.03/ 50.50**
Test Duration and Checkpointing			
- Ramp-up time		**27 minutes**	
- Measurement interval		**30 minutes**	
- Number of checkpoints		**1**	
- Checkpoint interval		**30 minutes**	
- Transactions during measurement interval (all types)		**2,378,050**	

Contents

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

Introduction

This is the Full Disclosure Report for the TPC Benchmark™ C running on the Fujitsu Siemens Computers system PRIMERGY N400. It meets the requirements of the TPC Benchmark™ C Standard Revision 3.5.

System Overview

This report documents the compliance of the Fujitsu Siemens Computers GmbH TPC Benchmark™ C tests using Microsoft SQL Server 2000 Enterprise Edition Relational Database Management System.

The TPC Benchmark™ C tests were carried out on a PRIMERGY N400. The PRIMERGY N400 is a powerful Server with a motherboard based on the ServerWorks chipset that holds up to 4 Intel Pentium® III Xeon 700 MHz processors with 2 MB L2 cache. The system was equipped with 8 GB of ECC SDRAM memory. 6 of the 8 PCI-Slots were used for SCSI RAID controllers, 1 was used for an Alteon Gigabit Ethernet adapter with uplink switch 1GB / 100 MB.
The client machines were 4 PRIMERGY 170 with 1 Intel Pentium® III 750 MHz. They all included 256 MB ECC SDRAM memory and 4 Adaptec ANA 62022 dual-port ethernet adapters.

The server operating system was Windows 2000 Advanced Server. The client operating system was Windows 2000 Server.

Full Disclosure

From Clause 8.1 of the TPC Benchmark™ C Standard Specification:

> The intent of this disclosure is for a customer to be able to replicate the results of this benchmark given the appropriate documentation and products.

Fujitsu Siemens Computers believes that this full disclosure report meets the stated intention. Fujitsu Siemens Computers has strived to maintain the integrity of the Specification by adhering not only to the letter of the Specification, but also to its spirit.

Report Format

*The format of this document follows Clause 8 of the TPC Benchmark™ C specification (*TPC Benchmark™ C Standard Specification*, Revision 3., Transaction Processing Performance Council) which describes the full disclosure report requirements for the test.*

Each section of this report begins with the specification requirement printed in *italic type*. It is followed by plain type text that explains how the test complies with the requirement. Sections which require extensive listings reference appropriate appendices.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

Report organization:

- General Items
- Clause 1 Related Items - Logical Database Design
- Clause 2 Related Items - Transaction and Terminal Profiles
- Clause 3 Related Items - Transaction and System Properties
- Clause 4 Related Items - Scaling and Database Population
- Clause 5 Related Items - Performance Metrics and Response Time
- Clause 6 Related Items - SUT, Driver, and Communication Definition
- Clause 7 Related Items - Pricing
- Clause 8 Related Items - Audit
- Appendix A - Application Source Code
- Appendix B - Database Details
- Appendix C - Tunable Parameters and Options
- Appendix D – Space Calculation
- Appendix E - Price Quotations
- Appendix F - Attestation Letter

Additional Copies

Additional copies of this report are available upon request from Fujitsu Siemens Computers GmbH:

Fujitsu SiemensComputers
SHV Server DS 5
PRIMERGY Server Performance Lab
Mr. Bathe
Heinz-Nixdorf-Ring 1
33106 Paderborn
Germany

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

1. General Items

1.1 Application Code

The application program (as defined in Clause 2.1.7) must be disclosed. This includes, but is not limited to, the code implementing the five transactions and the terminal input and output functions. [Clause 8.1.1.4]

The source code of the application program is provided in Appendix A - Application Source Code.

1.2 Benchmark Sponsor

A statement identifying the benchmark sponsor(s) and other participating companies must be provided. [Clause 8.1.1.5]

This benchmark was sponsored and executed by Fujitsu Siemens Computers GmbH.
The benchmark was developed and engineered by Fujitsu Siemens Computers GmbH and Microsoft Corporation. Testing took place at Fujitsu Siemens Computers NT-benchmark laboratories in Paderborn, Germany.

1.3 Parameter Settings

Settings must be provided for all customer-tunable parameters and options which have been changed from the defaults found in actual products, including but not limited to:

- *Database tuning options.*
- *Recovery/commit options.*
- *Consistency/locking options.*
- *Operating system and application configuration parameters.*

[Clause 8.1.1.6]

The significant parameters and system configuration files are provided in Appendix C - Tunable Parameters and Options.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

1.4 Configuration Diagrams

Diagrams of both measured and priced configurations must be provided, accompanied by a description of the differences. This includes, but is not limited to:

- *Number and type of processors.*
- *Size of allocated memory, and any specific mapping/partitioning of memory unique to the test.*
- *Number and type of disk units (and controllers, if applicable).*
- *Number of channels or bus connections to disk units, including their protocol type.*
- *Number of LAN (e.g., Ethernet) connections, including routers, workstations, terminals, etc., that were physically used in the test or are incorporated into the pricing structure (see Clause 8.1.8).*
- *Type and the run-time execution location of software components (e.g., DBMS, client processes, transaction monitors, software drivers, etc.).*

[Clause 8.1.1.7]

SUT Configuration

The PRIMERGY N400 server system included:

4	Intel Pentium® III Xeon 700 MHz with 2 MB Second Level Cache
8	GB memory
6	Mylex eXtremRAID 2000 SCSI controllers
145	disks 9 GB measured
96	disks 18 GB measured
145	disks 9 GB priced
96	disks 18 GB priced
6	disks 36 GB
1	LAN

Client Configuration

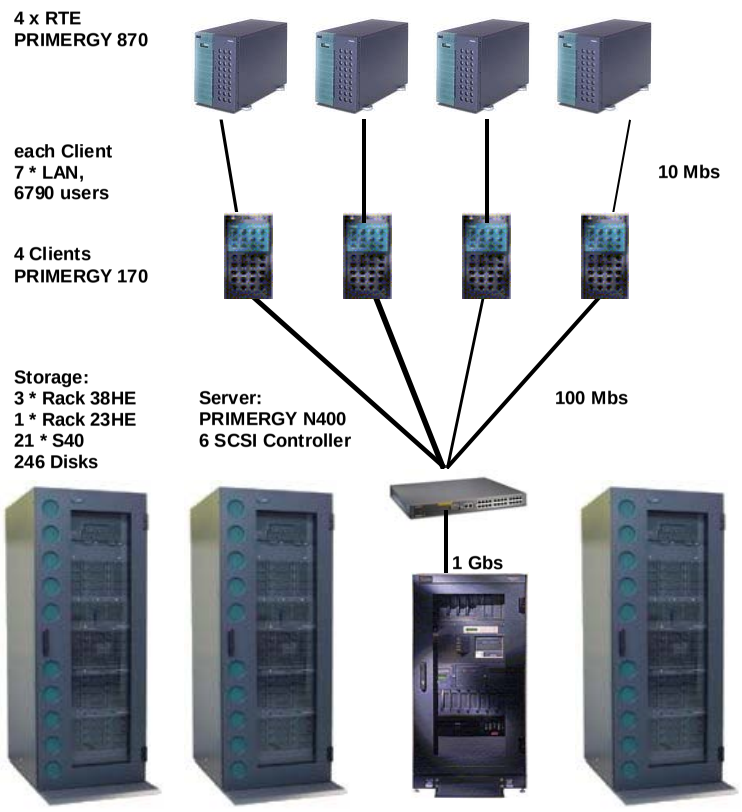
The PRIMERGY 170 client systems included:

1	Intel Pentium® III 750 MHz with 256 KB SLC
256	MB memory
1	SCSI controller
1	disk 9 GB
4	dual port LAN

The benchmarked and priced system configurations are shown in Figure 1 and Figure 2 in accordance with Clause 8.1.1.7.

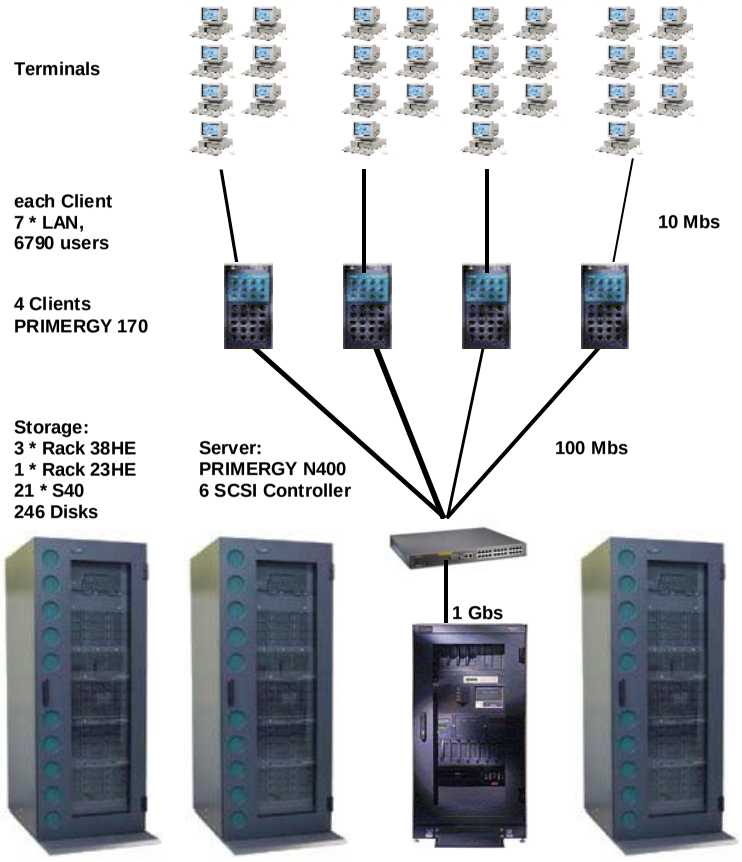
© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

FIGURE 1: BENCHMARK SYSTEM CONFIGURATION PRIMERGY N400



© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

FIGURE 2: PRICED SYSTEM CONFIGURATION PRIMERGY N400



© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

2. Clause 1 Related Items - Logical Database Design

2.1 Table Definitions

Listings must be provided for all table definition statements and all other statements used to set-up the database. [Clause 8.1.2.1]

The programs that defined, created, and populated the Microsoft SQL Server 2000 Enterprise Edition database for this TPC benchmark™ C are listed in Appendix B - Database Details.

2.2 Physical Organization of Database

The physical organization of tables and indices, within the database, must be disclosed. [Clause 8.1.2.2]

FIGURE 3: PHYSICAL ORGANIZATION OF THE DATABASE

TPCC RAID Configuration PRIMERGY N400

Controller	Channel 0	Channel 1	Channel 2	Channel 3	RAID	Drive
eXtremeRAID 2000 #0	0-0 1-0 2-0	0-1 1-1 2-1			SPAN 0 to 3 RAID1	L:
extremeRAID 2000 #1	0-0 0-1 0-2 0-3 0-4 0-5 0-10 0-11 0-12 0-13 0-14 0-15	1-0 1-1 1-2 1-3 1-4 1-5 1-10 1-11 1-12 1-13 1-14 1-15	2-0 2-1 2-2 2-3 2-4 2-5 2-10 2-11 2-12 2-13 2-14 2-15	3-0 3-1 3-2 3-3 3-4 3-5 3-10 3-11 3-12 3-13 3-14 3-15	SPAN 0 to 3 RAID0	E: N:
eXtremeRAID 2000 #2	0-0 0-1 0-2 0-3 0-4 0-5 0-10 0-11 0-12 0-13 0-14 0-15	1-0 1-1 1-2 1-3 1-4 1-5 1-10 1-11 1-12 1-13 1-14 1-15	2-0 2-1 2-2 2-3 2-4 2-5 2-10 2-11 2-12 2-13 2-14 2-15	3-0 3-1 3-2 3-3 3-4 3-5 3-10 3-11 3-12 3-13 3-14 3-15	SPAN 0 to 3 RAID0	F: O:
eXtremeRAID 2000 #3	0-0 0-1 0-2 0-3	1-0 1-1 1-2 1-3	2-0 2-1 2-2 2-3	3-0 3-1 3-2 3-3	SPAN 0 to 3 RAID0	G: P:

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

	0-4 0-5 0-10 0-11 0-12 0-13 0-14 0-15	1-4 1-5 1-10 1-11 1-12 1-13 1-14 1-15	2-4 2-5 2-10 2-11 2-12 2-13 2-14 2-15	3-4 3-5 3-10 3-11 3-12 3-13 3-14 3-15		
eXtremeRAID 2000 #4	0-0 0-1 0-2 0-3 0-4 0-5 0-10 0-11 0-12 0-13 0-14 0-15	1-0 1-1 1-2 1-3 1-4 1-5 1-10 1-11 1-12 1-13 1-14 1-15	2-0 2-1 2-2 2-3 2-4 2-5 2-10 2-11 2-12 2-13 2-14 2-15	3-0 3-1 3-2 3-3 3-4 3-5 3-10 3-11 3-12 3-13 3-14 3-15	SPAN 0 to 3 RAID0	H: Q: X:
eXtremeRAID 2000 #5	0-0 0-1 0-2 0-3 0-4 0-5 0-10 0-11 0-12 0-13 0-14 0-15	1-0 1-1 1-2 1-3 1-4 1-5 1-10 1-11 1-12 1-13 1-14 1-15	2-0 2-1 2-2 2-3 2-4 2-5 2-10 2-11 2-12 2-13 2-14 2-15	3-0 3-1 3-2 3-3 3-4 3-5 3-10 3-11 3-12 3-13 3-14 3-15	SPAN 0 to 3 RAID0	I: R: Y:

Disk Types:

- 9 GB: Fujitsu MAG3091LC
- 18 GB: Fujitsu MAG3182LC
- 36 GB: Fujitsu MAF3364LC
-

Caching:

No caching on controller or disks, except log drives write cache enabled (no single point of failure)

All controllers were configurated with write cache disabled. Write cache was enabled on the log drives and disabled on the data drives.
Space was allocated to Microsoft SQL Server 2000 Enterprise Edition on SUT disks according to the data in section 5.2. The size of the datafile on each disk drive was calculated to provide even distribution on load across the disk drives. The Windows Disk Manager was used to create raw devices for data/log and NTFS partitions for dump devices. For further information see Appendix B (Disk Usage) and Figure 4 in 5.2 (Distribution of Tables and Log). No attempt was made to alter the default physical organization of the database tables and indices chosen by Microsoft SQL Server 2000 Enterprise Edition.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

2.3 Insert and Delete Operations

It must be ascertained that insert and/or delete operations to any of the tables can occur concurrently with the TPC-C transaction mix. Furthermore, any restriction in the SUT database implementation that precludes inserts beyond the limits defined in Clause 1.4.11 must be disclosed. This includes the maximum number of rows that can be inserted and the maximum key value for these new rows. [Clause 8.1.2.3]

There were no restrictions on insert and delete operations to any tables.

2.4 Database Partitioning

While there are a few restrictions placed upon horizontal or vertical partitioning of tables and rows in the TPC benchmark™ C (see Clause 1.6), any such partitioning must be disclosed. [Clause 8.1.2.4]

There was no partitioning used in this implementation.

2.5 Replication of Tables

Replication of tables, if used, must be disclosed (see Clause 1.4.6). [Clause 8.1.2.5]

Replication of tables was not used in this implementation.

2.6 Additional and/or Duplicated Attributes

Additional and/or duplicated attributes in any table must be disclosed along with a statement on the impact on performance (see Clause 1.4.7). [Clause 8.1.2.6]

No additional and/or duplicated attributes were used.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

3. Clause 2 Related Items - Transaction and Terminal Profiles

3.1 Random Number Generator

The method of verification for the random number generation must be described. [Clause 8.1.3.1]

The random number generation was done in Microsoft BenchCraft, which was audited independently.

3.2 Input/Output Screen Layout

The actual layouts of the terminal input/output screens must be disclosed. [Clause 8.1.3.2]

The screen layout corresponded exactly to those of the TPC-C Standard Specification (specified in Clause 2.4.3, 2.5.3, 2.6.3, 2.7.3, and 2.8.3).

3.3 Configured Terminal Features

The method used to verify that the emulated terminals provide all the features described in Clause 2.2.2.4 must be explained. Although not specifically priced, the type and model of the terminals used for the demonstration in 8.1.3.3 must be disclosed and commercially available (including supporting software and maintenance). [Clause 8.1.3.3]

All of the requirements in clause 2.2.2.4. are supported. This was verified by manually exercising each specification on a PRIMERGY 870-40.

3.4 Presentation Managers or Intelligent Terminals

Any usage of presentation managers or intelligent terminals must be explained. [Clause 8.1.3.4]

Application code running on the client machines implemented the TPC-C user interface. No presentation manager software or intelligent terminal features were used. The source code for the forms application is listed in Appendix A - Application Source Code.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

3.5 Transaction Statistics

The numerical quantities which are required are listed in the following table. [Clause 8.1.3.5 to 8.1.3.11]

	Statistics	Percentage
New-Order	Home order-lines	99.00%
	Remote order-lines	1.00%
	Rolled back transactions	0.99%
	Average items per order	10.00
Payment	Home transactions	84.96%
	Remote transactions	15.04%
	Non-primary key access	60.02%
Order-Status	Non-primary key access	59.92
Delivery	Skipped transactions	0
Transaction Mix	New-Order	44.83 %
	Payment	43.00 %
	Order-Status	4.06 %
	Delivery	4.05 %
	Stock-Level	4.06 %

3.6 Queueing Mechanism

The queuing mechanism used to defer the execution of the Delivery transaction must be disclosed. [Clause 8.1.12]

Deferred deliveries are queued by making an entry in an array within the client application process (tpcc.dll). The queued delivery transactions are processed and logged asynchronously by background threads within the application.
The source code is listed in Appendix A - Application Source Code.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

4. Clause 3 Related Items - Transaction and System Properties

ACID Tests

The results of the ACID tests must disclosed along with a description of how the ACID requirements were met. This includes disclosing which case was followed for the execution of Isolation Test 7. [Clause 8.1.4.1]

All ACID tests were performed successfully. The following sections describe the requirements of each of the tests as described in Clause 3 and the approach used to satisfy them.

All ACID tests were performed on the PRIMERGY N400 system using the fully scaled database, except for the test of durable media failure.
The durability test was performed on a database scaled to 15 warehouses. This test would also pass on a fully scaled database.

4.1 Atomicity

The system under test must guarantee that database transactions are atomic; the system will either perform all individual operations on the data, or will assure that no partially-completed operations leave any effects on the data. [Clause 3.2.1]

Commit Transaction

Perform the Payment transaction for a randomly selected warehouse, district, and customer (by customer number as specified in Clause 2.5.1.2) and verify that the records in the CUSTOMER, DISTRICT, and WAREHOUSE tables have been changed appropriately. [Clause 3.2.2.1]

The following steps demonstrated atomicity for completed (COMMIT) transactions:

- A row was randomly selected from the warehouse, district and customer table.
- the current balance was noted.
- A payment transaction was executed with the above identifiers and a known amount.
- The transaction was committed.
- It was verified, that the rows contain the correct updated balances.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

Rollback Transaction

Perform the Payment transaction for a randomly selected warehouse, district, and customer (by customer number as specified in Clause 2.5.1.2) and substitute a ROLLBACK of the transaction for the COMMIT of the transaction. Verify that the records in the CUSTOMER, DISTRICT, and WAREHOUSE tables have NOT been changed. [Clause 3.2.2.2]

The following steps demonstrated atomicity for aborted (ROLLBACK) transactions:

- A row was randomly selected from the warehouse, district and customer table.
- the current balance was noted.
- A payment transaction was executed with the above identifiers and a known amount.
- The transaction was rolled back.
- It was verified, that the rows contain the original balances.

4.2 Consistency

Consistency is the property of the application that requires any execution of a database transaction to take the database from one consistent state to another, assuming that the database is initially in a consistent state. [Clause 3.3.1]

Consistency conditions 1 - 4 were tested by issuing queries to the database. The results of the queries verified that the database was consistent for all these tests. The tests were performed before and after the performance run on the same database that was used for the benchmark.

4.3 Isolation

Operations of concurrent transactions must yield results which are indistinguishable from the results which would be obtained by forcing each transaction to be serially executed to completion in some order.

We ran all of the seven isolation tests as described in clause 3.4.2.1 to 3.4.2.7 and additionally the two phantom protection tests. The tests were executed using shell scripts to issue queries to the database. The results of the queries verified that the required isolation had been met.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

4.4 Durability

The tested system must guarantee durability: the ability to preserve the effects of committed transactions and insure database consistency after recovery from any one of the failures listed in Clause 3.5.3. [Clause 3.5]

List of single failures:

1 *Permanent irrecoverable failure of any single durable medium containing TPC-C database tables or recovery log data.*
2 *Instantaneous interruption (system crash / system hang) in processing which requires system reboot to recover.*
3 *Failure of all or part of memory (loss of contents).*

[Clause 3.5.3]

The intent of these tests is to demonstrate that all transactions whose output messages have been received at the terminal or RTE have in fact been committed in spite of any single failure from the list in Clause 3.5.3 and that all consistency conditions are still met after the database is recovered.

It is required that the system crash test(s) and the loss of memory test(s) described in Clause 3.5.3.2 and 3.5.3.3 be performed under full terminal load and a fully scaled database. The durable media failure test(s) described in Clause 3.5.3.1 may be performed on a subset of the SUT configuration and database. For the SUT subset, all multiple hardware components, such as processors and disk / controllers in the full SUT configuration, must be represented by the greater of 10% of the configuration or two of each of the multiple hardware components. The database must be scaled to at least 10% of the fully scaled database, with a minimum of two warehouses. ... Furthermore, the standard driving mechanism must be used in this test. The test sponsor must state that to the best of their knowledge, a fully scaled test would also pass all durability tests. [Clause 3.5.4]

The failure of all or part of memory test and the system crash test were combined with the loss of log disk and performed under full load and by using a fully scaled database.

The full hardware configuration of the SUT (in accordance with Clause 3.5.4) and the same test procedure was used during all durability tests, except the test for loss of data.

- The current count of the total number of orders was determined by summing up the D_NEXT_O_ID fields of all rows in the DISTRICT table before the test.
- After 6 min in steady state we pulled off one of the log disks. As we use hardware-mirrored diskpairs with the SCSI-controller, execution continued.
- After additional 6 min we powered of the server to emulate the loss of memory. After server system reboot, SQL-Server starts with recovering the database tpcc. After completion, we computed the sum of D_NEXT_O_ID from district. Client and RTE systems were interrupted and evaluation started on the RTE. The difference of all D_NEXT_O_ID between RTE an server was in the permitted scope.

The durable media failure test for loss of data disk was performed with 48 of the 240 data disks and a database scaled to 15 warehouses under the load of 150 users. We used one RTE and one client system. To the best of the test sponsor's

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

knowledge, a fully loaded and fully scaled database would also pass this durability test.

- The database was backed up.
- The current count of the total number of orders was determined by summing up the D_NEXT_O_ID fields of all rows in the DISTRICT table before the test.
- After 5 min in steady state we pulled of one of the data disks.
- SQL-Server recognized the loss of a device. We dumped the transaction log and removed the database with dropdevice. Then we shut down SQL-Server and the system.
- We replaced the disk and made it online.
- We restarted SQL-Server, no tpcc database and none of its devices were present. We recreated the database, loaded dump and load transaction log
- After completion, we computed the sum of D_NEXT_O_ID from district.
- Client and RTE systems were interrupted and evaluation started on the RTE. The difference of all D_NEXT_O_ID between RTE an server was in the permitted scope.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

5. Clause 4 Related Items - Scaling and Database Population

5.1 Initial Cardinality of Tables

The cardinality (e.g., the number of rows) of each table, as it existed at the start of the benchmark run (see Clause 4.2), must be disclosed. If the database was over-scaled and inactive rows of the WAREHOUSE table were deleted (see Clause 4.2.2), the cardinality of the WAREHOUSE table as initially configured and the number of rows deleted must be disclosed. [Clause 8.1.5.1]

The database for the PRIMERGY N400 system was scaled for 2716 warehouses. The performance run used all warehouses. In accordance with Clause 4.2, the following number of records were loaded in the specified tables:

Table	Number of Records
Warehouse	2716
District	27,160
Customer	81,480,000
History	81,480,000
Order	81,480,000
New-Order	24,444,000
Order-Line	814,797,124
Stock	271,600,000
Item	100,000
Deleted Warehouses	0

The following constant values were used during the database build and benchmark test for the NURand function:

Constant C	Value
C_LAST (build)	123
C_LAST (run)	233

5.2 Distribution of Tables and Log

The distribution of tables and logs across all media must be explicitly depicted for the tested and priced systems. [Clause 8.1.5.2]

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

FIGURE 4: LOGICAL ORGANIZATION OF THE DATABASE

Disk #	Controller	Disktype	RAID Configuration	Drive Letter	Size MB	Filegroup or Filesystem
0	onboard	9 GB	-	system C:	9000	NTFS
1	eXtremeRAID 2000 #1	6 x 36GB	RAID 1	L:	82500	log
2	eXtremeRAID 2000 #2	48 x 9 GB	RAID 0	E: N:	31000 18000	cs1 misc1
3	eXtremeRAID 2000 #3	48 x 9 GB	RAID 0	F: O:	31000 18000	cs2 misc2
4	eXtremeRAID 2000 #4	48 x 9 GB	RAID 0	G: P:	31000 18000	cs3 misc3
5	eXtremeRAID 2000 #5	48 x 18 GB	RAID 0	H: Q: X:	31000 18000 200000	cs4 misc4 backup1
6	eXtremeRAID 2000 #6	48 x 18 GB	RAID0	I: R: Y:	31000 18000 200000	cs5 misc5 backup2

5.3 Database Model, Interface, and Access Language

A statement must be provided that describes:

1. The data model implemented by the DBMS used (e.g., relational, network, hierarchical)

2. The database interface (e.g., embedded, call level) and access language (e.g., SQL, DL/1, COBOL read/write) used to implement the TPC-C transactions. If more than one interface/access language is used to implement TPC-C, each interface / access language must be described and a list of which interface/access language is used with which transaction type must be disclosed.

[Clause 8.1.5.3]

Microsoft SQL Server 2000 Enterprise Edition is a Relational DataBase Management System. The interface used was Microsoft SQL Server 2000 Enterprise Edition stored procedures accessed with Remote Procedure Calls embedded in C code.

5.4 Database Partitions/Replications Mapping

The mapping of database partitions/replications must be explicitly described. [Clause 8.1.5.4]

There was no partitioning and/or replication used in this implementation.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

5.5 180 day space Calculation

Details of the 180-day space computations along with proof that the database is configured to subtain 8 hours of growth for the dynamic tables (Order, Order-Line, and History) must be disclosed (see Clause 4.2.3). [Clause 8.1.5.5]

Calculations of space requirements in the priced configurations for the 180-day period are provided in Appendix D – Space Calculation.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

6. Clause 5 Related Items - Performance Metrics and Response Time

6.1 Measured tpmC

Measured tpmC must be reported. [Clause 8.1.6.1]

During the 30 minutes measurement period on the PRIMERGY N400 the throughput measured was 34,150.87 tpmC.

6.2 Response Times

Ninetieth percentile, maximum and average response times must be reported for all transaction types as well as for the Menu response time. [Clause 8.1.6.2]

Type	Average	Maximum	90 Percentile
New-Order	0.28	6.86	0.40
Payment	0.20	2.00	0.31
Order-Status	0.22	8.82	0.34
Interactive Delivery	0.11	0.79	0.12
Deferred Delivery	0.20	0.86	0.35
Stock-Level	1.02	2.81	1.48
Menu	0.11	0.90	0.12

6.3 Keying and Think Times

The minimum, the average, and the maximum keying and think times must be reported for each transaction type. [Clause 8.1.6.3]

Keying Times			
Type	**Average**	**Maximum**	**Minimum**
New-Order	18.01	18.04	18.00
Payment	3.01	3.04	3.00
Order-Status	2.01	2.03	2.00
Delivery	2.01	2.03	2.00
Stock-Level	2.01	2.03	2.00

Think Times			
Type	**Average**	**Maximum**	**Minimum**
New-Order	12.05	120.50	0.00
Payment	12.04	120.50	0.00
Order-Status	10.01	100.50	0.00
Delivery	5.09	50.49	0.00
Stock-Level	5.04	50.50	0.00

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

6.4 Graphs

Response Time frequency distribution curves (see Clause 5.6.1) must be reported for each transaction type. [Clause 8.1.6.4]

The performance curve for response times versus throughput (see Clause 5.6.2) must be reported for the New-Order transaction. [Clause 8.1.6.5]

Think Time frequency distribution curves (see Clause 5.6.3) must be reported for each transaction type. [Clause 8.1.6.6]

Keying Time frequency distribution curves (see Clause 5.6.4) must be reported for each transaction type. [Clause 8.1.6.7]

A graph of throughput versus elapsed time (see Clause 5.6.5) must be reported for the New-Order transaction. [Clause 8.1.6.8]

FIGURE 5: NEW-ORDER RESPONSE TIME DISTRIBUTION

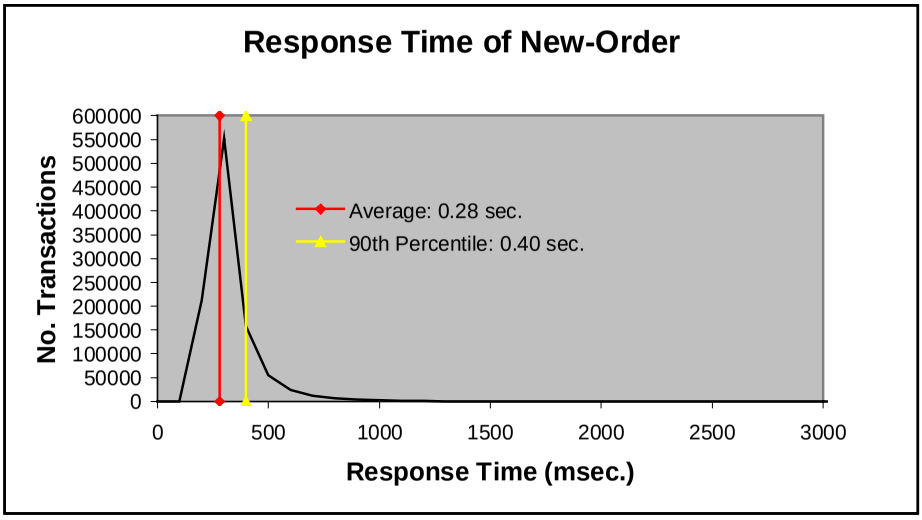
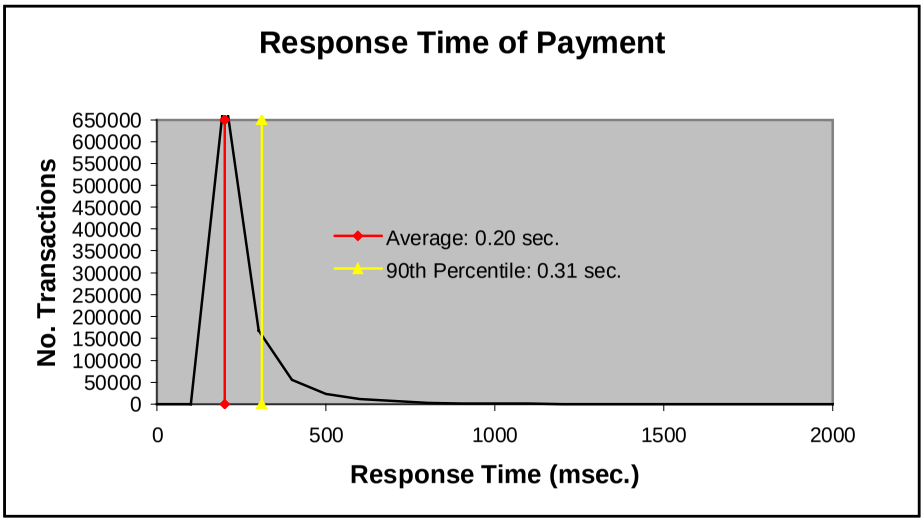


FIGURE 6: PAYMENT RESPONSE TIME DISTRIBUTION



© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

FIGURE 7: ORDER-STATUS RESPONSE TIME DISTRIBUTION

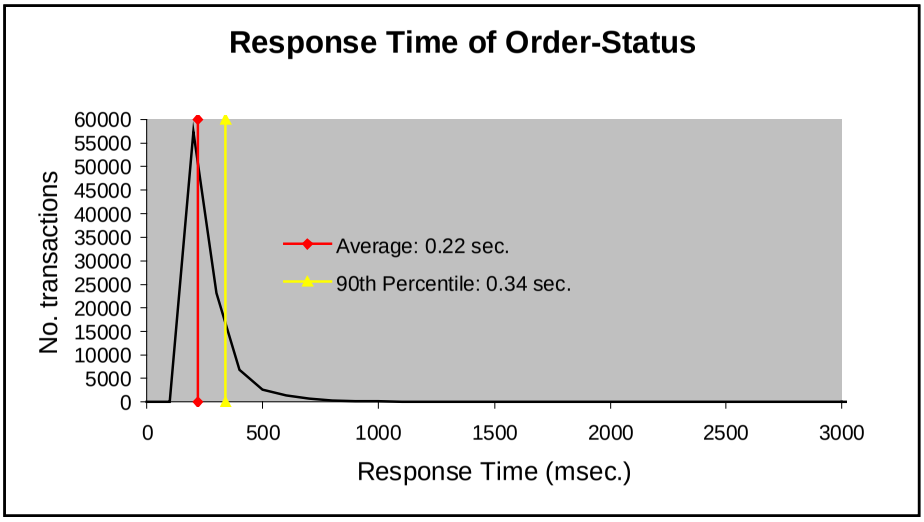
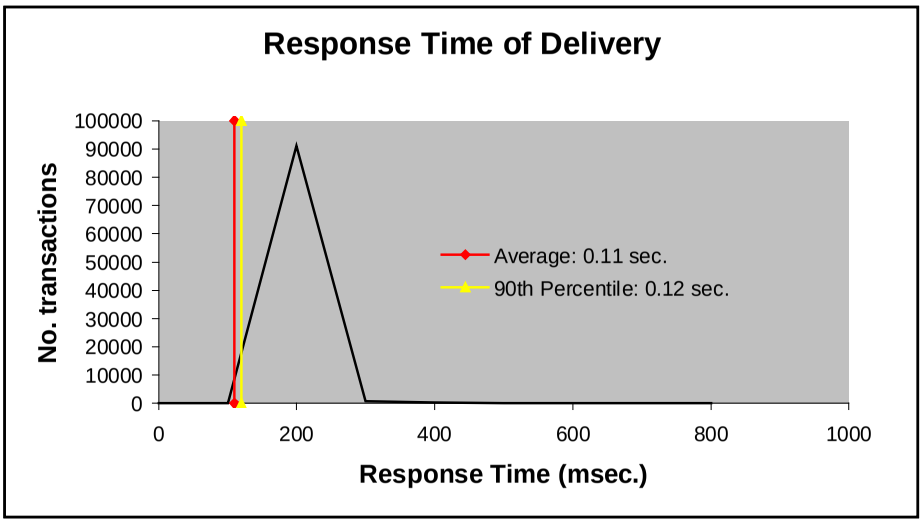


FIGURE 8: DELIVERY RESPONSE TIME DISTRIBUTION



© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

FIGURE 9: STOCK-LEVEL RESPONSE TIME DISTRIBUTION

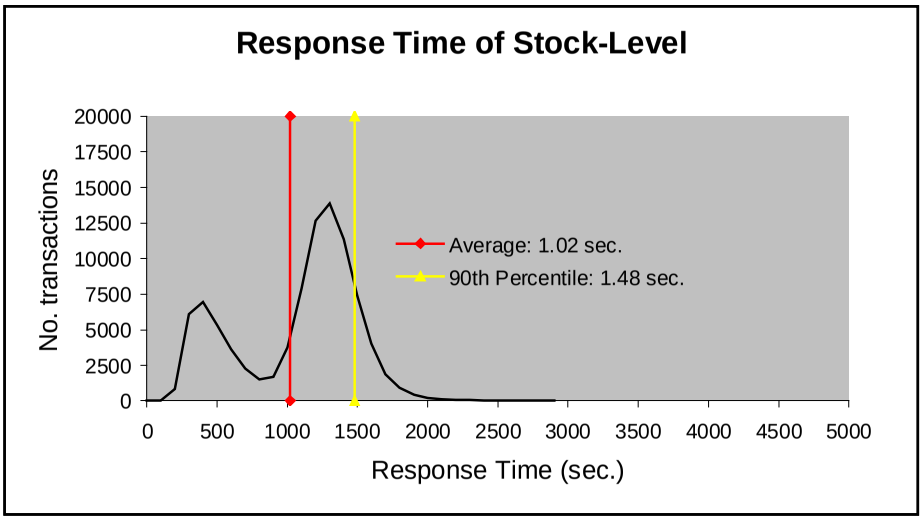
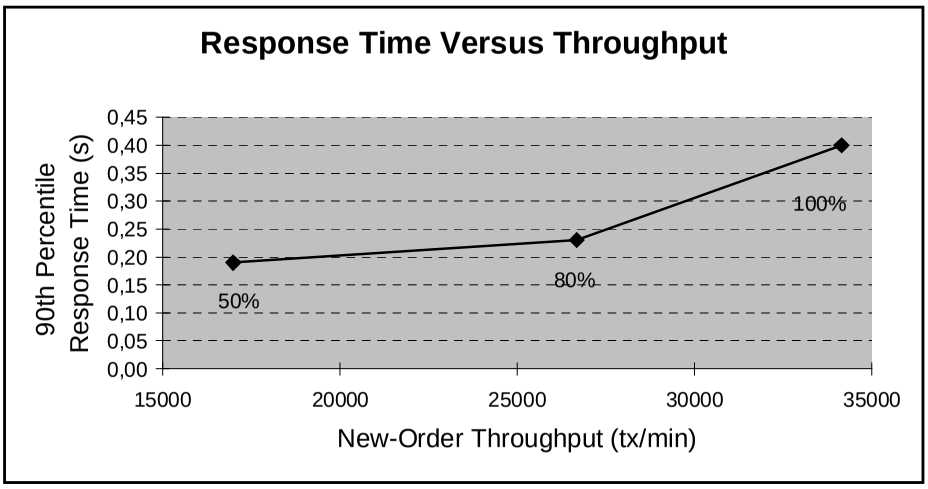


FIGURE 10:RESPONSE TIME VERSUS THROUGHPUT



© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

FIGURE 11:NEW-ORDER THINK TIME DISTRIBUTION

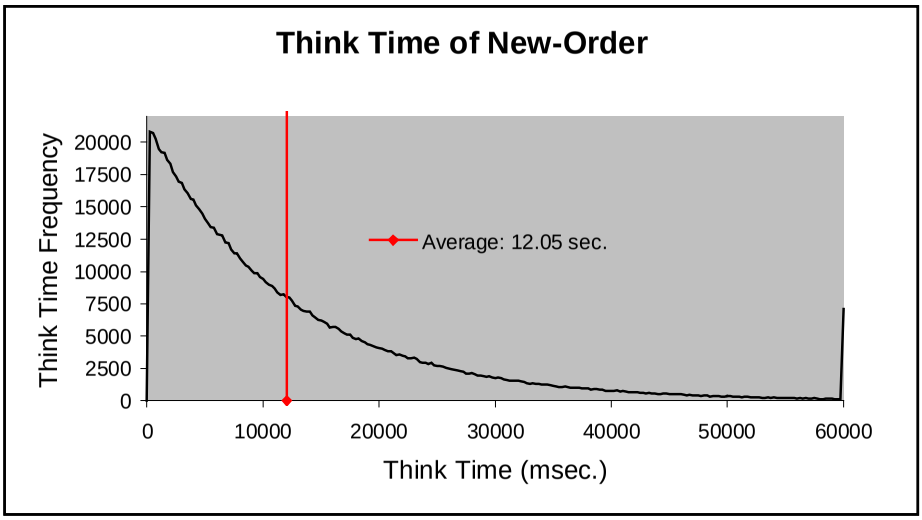
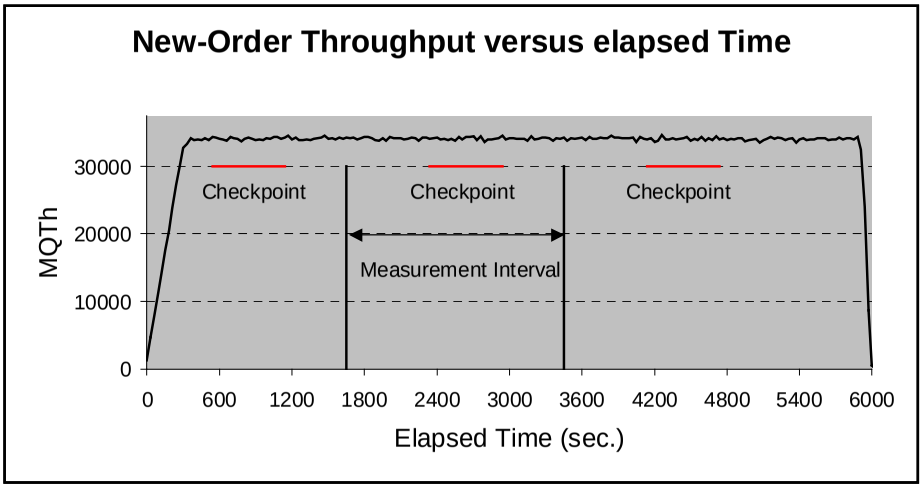


FIGURE 12:THROUGHPUT VERSUS ELAPSED TIME



6.5 Steady State Determination

The method used to determine that the SUT had reached a steady state prior to commencing the measurement interval (see Clause 5.5) must be described. [Clause 8.1.6.9]

In all test runs, steady state was achieved before the measurement period began. Steady state was determined to occur based on a visual inspection of tpmC versus time (see graph in section 6.4).

The graph in section 6.4 illustrates that the measurement period was within the steady state period for the run. One checkpoint occurred during the measurement period.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

6.6 Work Performed

A description of how the work normally performed during a sustained test (for example checkpointing, writing redo/undo log records, etc.), actually occurred during the measurement interval must be reported. [Clause 8.1.6.10]

The RTE generated the required input data to choose a transaction from the menu. This data was timestamped and captured in RTE log files before being transmitted. There was one log file for each user. The input screen for the requested transaction was returned and it was also captured and timestamped in the RTE log files. The difference between these two timestamps was the menu response time.

The RTE generated the required input data for the chosen transaction. It waited to complete the minimum required key time before transmitting the input screen. The transmission was timestamped and captured in RTE log files. The return of the screen with the required response data was timestamped and captured in the RTE log files. The difference between these two timestamps was the response time for that transaction.

The RTE then waited the required think time interval before repeating the process starting at selecting a transaction from the menu.

The RTE transmissions were sent to Internet Information Server running on the client machines through Ethernet LANs. Internet Information Server handled all screen I/O as well as all requests to the database on the server. Internet Information Server communicated with the database server over COM+ which was used as transaction monitor.

All database operations like update, select, delete and insert are performed by one of the TPC-C back end programs. The TPC-C backend program commits the transaction after all the corresponding operations are done.

Modified database buffers are migrated to disk a least-recently-used basis independent of transaction commits. In addition, every block modification is protected by log records. Asynchronously the log buffers are flushed to a log file on disk either when the transaction is committed or when the log buffer's fill state reaches it's limit. The log buffer's always flushed by a commit before it become full.

To perform checkpoints at specific intervals, we wrote a script to schedule multiple checkpoints at specific intervals. By setting the trace flag #3502, SQL Server logged the checkpoint beginning and ending time in the ERRORLOG file. The script included a wait time between each checkpoint equal to the measurement interval which was 30 minutes. The checkpoint script was started manually after the RTE had all users logged in and sending transactions.

At each checkpoint, Microsoft SQL Server wrote to disk all memory pages that had been updated but not yet physically written to disk. Upon completion of the checkpoint, Microsoft SQL Server wrote a special record to the recovery log to indicate that all disk operations had been satisfied to this point.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

6.7 Reproducibility

A description of the method used to determine the reproducibility of the measurement results must be reported. [Clause 8.1.6.11]

The PRIMERGY N400 system test was run twice to ensure the reproducibility of the results. The reproducibility test run under exactly the same conditions as the reported test. All tests conform to the TPC rules.

The tpmC result from the reproducibility test was within 0.16% of the reported tpmC.

In the following, both results are shown to document the reproducibility:

	tpmC
reported test	34,150.87
reproducibility test	34,095.13

6.8 Duration of Measurement

A statement of the duration of the measurement interval for the reported Maximum Qualified Throughput (tpmC) must be included. [Clause 8.1.6.12]

The measurement interval of the PRIMERGY N400 system test was 30 minutes. This measurement interval corresponds to the amount of time from the beginning of one checkpoint to the beginning of the next (which, actually, is less than the amount of time it takes to fill a log file).

6.9 Regulation of Transaction Mix

The method of regulation of the transaction mix (e.g., card decks or weighted random distribution) must be described. If weighted distribution is used and the RTE adjusts the weights associated with each transaction type, the maximum adjustments to the weight from the initial value must be disclosed. [Clause 8.1.6.13]

The transaction mix was regulated by weighted distribution. The chosen weights meet the required minimum percentages of the mix which are described in Clause 5.2.3 of the Standard Specifications. No adjustments were made by the RTE.

6.10 Transaction Mix

The percentage of the total mix for each transaction type must be disclosed. [Clause 8.1.6.14]

	Percentage
New-Order	44.83 %
Payment	43.00 %
Order-Status	4.06 %
Delivery	4.05 %
Stock-Level	4.06 %

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

6.11 Transaction Statistics

The percentage of New-Order transactions rolled back as a result of invalid item number must be disclosed. [Clause 8.1.6.15]

The average number of order-lines entered per New-Order transaction must be disclosed. [Clause 8.1.6.16]

The percentage of remote order-lines entered per New-Order transaction must be disclosed. [Clause 8.1.6.17]

The percentage of remote Payment transactions must be disclosed. [Clause 8.1.6.18]

The percentage of customer selections by customer last name in the Payment and Order-Status transactions must be disclosed. [Clause 8.1.6.19]

The percentage of Delivery transactions skipped due to there being fewer than necessary orders in the New-Order table must be disclosed. [Clause 8.1.6.20]

The numerical quantities which are required in Clause 8.1.6.15 to 8.1.6.20 are already listed in a table above (see section 3.5).

6.12 Checkpoint Statistics

The number of checkpoints in the Measurement Interval, the time in seconds from the start of the Measurement Interval to the first checkpoint and the Checkpoint Interval must be disclosed. [Clause 8.1.6.21]

There was 1 checkpoints before and one during the measurement interval. The checkpoint occurred 11:40 minutes after the start of the measurement interval. The checkpoint interval was 30 minutes. The duration of the checkpoint during measurement was 10 minutes.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

7. Clause 6 Related Items - SUT, Driver, and Communication Definition

7.1 RTE Inputs

If the RTE is commercially available, then its inputs must be specified. Otherwise, a description must be supplied of what inputs (e.g., scripts) to the RTE had been used. [Clause 8.1.7.1]

Microsoft Benchcraft was used as the RTE to emulate the terminals. Its input parameters are shown in Appendix C - Tunable Parameters and Options.

We used COM+ to simulate terminal users, generate random data, record response times and statistical data. Its input parameters are shown in Appendix C - Tunable Parameters and Options.

7.2 Functionality and Performance of Emulated Components

It must be demonstrated that the functionality and performance of the components being emulated in the Driver System are equivalent to that of the priced system. The results of the test described in Clause 6.6.3.4 must be disclosed. [Clause 8.1.7.2]

The Driver System consisted of a PRIMERGY 870-40. This driver was attached to the client machine through an Ethernet LAN. Since this is exactly the same connectivity as configured in the priced system, no component was emulated. Therefore, the test described in Clause 6.6.3.4 was not required.

7.3 Functional Diagrams of the Benchmarked and Proposed Configuration

A complete functional diagram of both the benchmark configuration and the configuration of the proposed (target) system must be disclosed. A detailed list of all software and hardware functionality being performed on the Driver System, and its interface to the SUT must be disclosed (see Clause 6.6.3.6). [Clause 8.1.7.3]

Figure 1 and Figure 2 in section 1.4 show the functional diagrams of the benchmark configuration and the priced configuration.

7.4 Network Configurations of the Tested and Proposed Services

The network configurations of both the tested services and the proposed (target) services which are being represented and a thorough explanation of exactly which parts of the proposed configuration are being replaced with the Driver System must be disclosed (see Clause 6.6.4). [Clause 8.1.7.4]

Figure 1 and Figure 2 in section 1.4 show the network setup of both configurations. The driver replaces the workstations.

In both configurations one 1Gbs/100Mbs ethernet LAN segment was used to connect the server with the 4 clients.

In the measured configuration 4 x 7 10 Mbs ethernet LAN segments were used to connect each of the 4 clients with the 4 RTE systems.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

In the priced configuration 4 x 7 10 Mbs ethernet LAN segments were used to connect 27,160 workstations. Each client has 6790 users connected with 7 ethernet segments.

7.5 Network Bandwidth

The bandwidth of the network(s) used in the tested / priced configuration must be disclosed. [Clause 8.1.7.5]

The Ethernet used in the local area network (LAN) between the emulated user system and the front-end system complies with the IEEE 802.3 standard. Its bandwith is 10 Mbps. Between front-end and SUT the bandwidth is 1 Gbps/100 Mbps.

7.6 Operator Intervention

If the configuration requires operator intervention (see Clause 6.6.6), the mechanism and the frequency of this intervention must be disclosed. [Clause 8.1.7.6]

The PRIMERGY N400 requires no operator intervention to sustain the reported throughput.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

8. Clause 7 Related Items - Pricing

8.1 System Pricing

A detailed list of hardware and software used in the priced system must be reported. Each separately orderable item must have vendor part number, description, and release/revision level, and either general availability status or committed delivery date. If package-pricing is used, vendor part number of the package and a description uniquely identifying each of the components of the package must be disclosed. Pricing source(s) and effective date(s) of price(s) must also be reported. [Clause 8.1.8.1]

The total 5-year price of the entire configuration must be reported, including: hardware, software, and maintenance charges. Separate component pricing is recommended. The basis of all discounts used must be disclosed. [Clause 8..1.8.2]

The details of the hardware and software are reported in the summary in front of this report. The spreadsheet used to determine the 5-year price and the speadsheet used to describe the priced configuration can be found in Appendix D – Space Calculation.

8.2 Availability Dates

The committed delivery date for general availability (availability date) of products used in the price calculations must be reported. When the priced system includes products with different availability dates, the reported availability date for the priced system must be the date at which all components are committed to be available. [Clause 8.1.8.3]

All hardware and software components used in the price calculations of the PRIMERGY N400 system will be generally available from Fujitsu Siemens Computers GmbH as of November 1, 2000.

8.3 Throughput and Price/Performance

A statement of the measured tpmC, as well as the respective calculations for 5-year pricing, price/performance (price/tpmC), and the availability date must be included. [Clause 8.1.8.4]

PRIMERGY N400 system was measured at 34,150.87 tpmC with Microsoft SQL Server 2000 Enterprise Edition with a 5-year system price of €608,226. The respective price/performance for the PRIMERGY N400 is €17.81/tpmC. The priced PRIMERGY N400 will be available as of November 1, 2000.

8.4 Country Specific Pricing

Additional Clause 7 related items may be included in the Full Disclosure Report for each country specific priced configuration. Country specifiv pricing is subject to Clause 7.1.7 [Clause 8.1.8.5]

The system is being priced for Germany.

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

8.5 Usage Pricing

For any usage pricing, the sponsor must diclose:

- *Usage level at which the component was priced.*
- *A statement of the company policy allowing such pricing.*

[Clause 8.1.8.6]

The component pricing based on usage is shown below:

- One Microsoft SQL Server 2000 Enterprise Edition
- One Windows 2000 Advanced Server
- 4 Microsoft Windows 2000 Server license (includes 5 client access licenses)
- One Microsoft Visual C++ Professional 6.0

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

9. Clause 8 Related Items - Audit

The auditor's name, address, phone number, and a copy of the auditor's attestation letter indicating compliance must be included in the Full Disclosure Report.

A review of the pricing model is required to ensure that all components required are priced (see Clause 9.2.8). The auditor is not required to review the final Full Disclosure Report or the final pricing prior to issuing the attestation letter. [Clause 8.1.9]

The benchmark test of the PRIMERGY N400 system with Microsoft SQL Server 2000 Enterprise Edition was independently audited by:

Bradley Askins, a TPC certified auditor of Infosizing.
The attestation letter is included in Appendix F.

Requests for this TPC-C Full Disclosure Report should be sent to:

Transaction Processing Performance Council
c/o Shanley Public Relations
777 North First Street, Suite 6000
San Jose, CA 95112-6311

or

FUJITSU SIEMENS COMPUTERS
SHV SERVER DS5
PRIMERGY Server Performance Lab
Mr. Bathe
Heinz-Nixdorf-Ring 1
33106 Paderborn
Germany

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

Appendix A - Application Source Code

```
LIBRARY TPCC.DLL

EXPORTS

       GetExtensionVersion    @1
       HttpExtensionProc      @2
       TerminateExtension     @3


/*     FILE:          TPCC.H
 *                           Microsoft TPC-C Kit Ver. 4.20.000
 *                           Copyright Microsoft, 1999
 *              All Rights Reserved
 *
 *                           Version 4.10.000 audited by Richard Gimarc,
Performance Metrics, 3/17/99
 *
 *     PURPOSE:       Header file for ISAPI TPCC.DLL, defines structures
and functions used in the isapi tpcc.dll.
 *
 */

//VERSION RESOURCE DEFINES
#define _APS_NEXT_RESOURCE_VALUE                              101
#define _APS_NEXT_COMMAND_VALUE                                      40001
#define _APS_NEXT_CONTROL_VALUE                                      1000
#define _APS_NEXT_SYMED_VALUE                                 101

#define TP_MAX_RETRIES                                               50


//note that the welcome form must be processed first as terminal ids
assigned here, once the
//terminal id is assigned then the forms can be processed in any order.
#define WELCOME_FORM                                          1
       //beginning form no term id assigned, form id
#define MAIN_MENU_FORM                                        2
       //term id assigned main menu form id
#define NEW_ORDER_FORM                                        3
       //new order form id
#define PAYMENT_FORM                                          4
       //payment form id
#define DELIVERY_FORM                                         5
       //delivery form id
#define ORDER_STATUS_FORM                                     6
       //order status id
#define STOCK_LEVEL_FORM                                      7
       //stock level form id

//This macro is used to prevent the compiler error unused formal
parameter
#define UNUSEDPARAM(x) (x = x)


//This structure defines the data necessary to keep distinct for each
terminal or client connection.
typedef struct _CLIENTDATA
{
       int                    iNextFree;
       //index of next free element or -1 if this entry in use.
       int                    w_id;
       //warehouse id assigned at welcome form
       int                    d_id;
       //district id assigned at welcome form

       int                    iSyncId;
       //syncronization id
       int                    iTickCount;                          //time
of last access;

       CTPCC_BASE     *pTxn;

} CLIENTDATA, *PCLIENTDATA;


//This structure is used to define the operational interface for terminal
id support
typedef struct _TERM
{
       int                    iNumEntries;
       //total allocated terminal array entries
       int                    iFreeList;
              //next available terminal array element or -1 if none
       int                    iMasterSyncId;
       //syncronization id
       CLIENTDATA     *pClientData;
       //pointer to allocated client data
} TERM;

typedef TERM *PTERM;
       //pointer to terminal structure type
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
enum WEBERROR
{
        NO_ERR,
        ERR_COMMAND_UNDEFINED,
        ERR_D_ID_INVALID,
        ERR_DELIVERY_CARRIER_ID_RANGE,
        ERR_DELIVERY_CARRIER_INVALID,
        ERR_DELIVERY_MISSING_OCD_KEY,
        ERR_DELIVERY_THREAD_FAILED,
        ERR_GETPROCADDR_FAILED,
        ERR_HTML_ILL_FORMED,
        ERR_INVALID_SYNC_CONNECTION,
        ERR_INVALID_TERMID,
        ERR_LOADDLL_FAILED,
        ERR_MAX_CONNECTIONS_EXCEEDED,
        ERR_MEM_ALLOC_FAILED,
        ERR_MISSING_REGISTRY_ENTRIES,
        ERR_NEWORDER_CUSTOMER_INVALID,
        ERR_NEWORDER_CUSTOMER_KEY,
        ERR_NEWORDER_DISTRICT_INVALID,
        ERR_NEWORDER_FORM_MISSING_DID,
        ERR_NEWORDER_ITEMID_INVALID,
        ERR_NEWORDER_ITEMID_RANGE,
        ERR_NEWORDER_ITEMID_WITHOUT_SUPPW,
        ERR_NEWORDER_MISSING_IID_KEY,
        ERR_NEWORDER_MISSING_QTY_KEY,
        ERR_NEWORDER_MISSING_SUPPW_KEY,
        ERR_NEWORDER_NOITEMS_ENTERED,
        ERR_NEWORDER_QTY_INVALID,
        ERR_NEWORDER_QTY_RANGE,
        ERR_NEWORDER_QTY_WITHOUT_SUPPW,
        ERR_NEWORDER_SUPPW_INVALID,
        ERR_NO_SERVER_SPECIFIED,
        ERR_ORDERSTATUS_CID_AND_CLT,
        ERR_ORDERSTATUS_CID_INVALID,
        ERR_ORDERSTATUS_CLT_RANGE,
        ERR_ORDERSTATUS_DID_INVALID,
        ERR_ORDERSTATUS_MISSING_CID_CLT,
        ERR_ORDERSTATUS_MISSING_CID_KEY,
        ERR_ORDERSTATUS_MISSING_CLT_KEY,
        ERR_ORDERSTATUS_MISSING_DID_KEY,
        ERR_PAYMENT_CDI_INVALID,
        ERR_PAYMENT_CID_AND_CLT,
        ERR_PAYMENT_CUSTOMER_INVALID,
        ERR_PAYMENT_CWI_INVALID,
        ERR_PAYMENT_DISTRICT_INVALID,
        ERR_PAYMENT_HAM_INVALID,
        ERR_PAYMENT_HAM_RANGE,
        ERR_PAYMENT_LAST_NAME_TO_LONG,
        ERR_PAYMENT_MISSING_CDI_KEY,
        ERR_PAYMENT_MISSING_CID_CLT,
        ERR_PAYMENT_MISSING_CID_KEY,
        ERR_PAYMENT_MISSING_CLT,
        ERR_PAYMENT_MISSING_CLT_KEY,
        ERR_PAYMENT_MISSING_CWI_KEY,
        ERR_PAYMENT_MISSING_DID_KEY,
        ERR_PAYMENT_MISSING_HAM_KEY,
        ERR_STOCKLEVEL_MISSING_THRESHOLD_KEY,
        ERR_STOCKLEVEL_THRESHOLD_INVALID,
        ERR_STOCKLEVEL_THRESHOLD_RANGE,
        ERR_VERSION_MISMATCH,
        ERR_W_ID_INVALID
};

class CWEBCLNT_ERR : public CBaseErr
{
        public:

                CWEBCLNT_ERR(WEBERROR Err)
                {
                        m_Error = Err;
                        m_szTextDetail = NULL;
                        m_SystemErr = 0;
                        m_szErrorText = NULL;
                };

                CWEBCLNT_ERR(WEBERROR Err, char *szTextDetail, DWORD
dwSystemErr)
                {
                        m_Error = Err;
                        m_szTextDetail = new char[strlen(szTextDetail)+1];
                        strcpy( m_szTextDetail, szTextDetail );
                        m_SystemErr = dwSystemErr;
                        m_szErrorText = NULL;
                };

                ~CWEBCLNT_ERR()
                {
                        if (m_szTextDetail != NULL)
                                delete [] m_szTextDetail;
                        if (m_szErrorText != NULL)
                                delete [] m_szErrorText;
                };

                WEBERROR        m_Error;
                char            *m_szTextDetail;       //
                char            *m_szErrorText;
                DWORD           m_SystemErr;

                int ErrorType() {return ERR_TYPE_WEBDLL;};
                int ErrorNum() {return m_Error;};
                char *ErrorText();

};
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
//These constants have already been defined in engstut.h, but since we do
//not want to include it in the delisrv executable
#define TXN_EVENT_START               2
#define TXN_EVENT_STOP        4
#define TXN_EVENT_WARNING     6              //used to record a warning
into the log

//function prototypes

BOOL APIENTRY DllMain(HANDLE hModule, DWORD ul_reason_for_call, LPVOID
lpReserved);
void WriteMessageToEventLog(LPTSTR lpszMsg);
void ProcessQueryString(EXTENSION_CONTROL_BLOCK *pECB, int *pCmd, int
*pFormId, int *pTermId, int *pSyncId);
void WelcomeForm(EXTENSION_CONTROL_BLOCK *pECB, char *szBuffer);
void SubmitCmd(EXTENSION_CONTROL_BLOCK *pECB, char *szBuffer);
void BeginCmd(EXTENSION_CONTROL_BLOCK *pECB, int iFormId, int iTermId);
void ProcessCmd(EXTENSION_CONTROL_BLOCK *pECB, int iFormId, int iTermId);
void StatsCmd(EXTENSION_CONTROL_BLOCK *pECB, char *szBuffer);
void ErrorMessage(EXTENSION_CONTROL_BLOCK *pECB, int iError, int
iErrorType, char *szMsg, int iTermId);
void GetKeyValue(char **pQueryString, char *pKey, char *pValue, int iMax,
WEBERROR err);
int  GetIntKeyValue(char **pQueryString, char *pKey, WEBERROR NoKeyErr,
WEBERROR NotIntErr);
void TermInit(void);
void TermDeleteAll(void);
int  TermAdd(void);
void TermDelete(int id);
void ErrorForm(EXTENSION_CONTROL_BLOCK *pECB, int iType, int iErrorNum,
int iTermId, int iSyncId, char *szErrorText, char *szBuffer );
void MakeMainMenuForm(int iTermId, int iSyncId, char *szForm);
void MakeStockLevelForm(int iTermId, STOCK_LEVEL_DATA *pStockLevelData,
BOOL bInput, char *szForm);
void MakeNewOrderForm(int iTermId, NEW_ORDER_DATA *pNewOrderData, BOOL
bInput, char *szForm);
void MakePaymentForm(int iTermId, PAYMENT_DATA *pPaymentData, BOOL
bInput, char *szForm);
void MakeOrderStatusForm(int iTermId, ORDER_STATUS_DATA
*pOrderStatusData, BOOL bInput, char *szForm);
void MakeDeliveryForm(int iTermId, DELIVERY_DATA *pDeliveryData, BOOL
bInput, char *szForm);
void ProcessNewOrderForm(EXTENSION_CONTROL_BLOCK    *pECB, int iTermId,
char *szBuffer);
void ProcessPaymentForm(EXTENSION_CONTROL_BLOCK *pECB, int iTermId, char
*szBuffer);
void ProcessOrderStatusForm(EXTENSION_CONTROL_BLOCK *pECB, int iTermId,
char *szBuffer);
void ProcessDeliveryForm(EXTENSION_CONTROL_BLOCK *pECB, int iTermId, char
*szBuffer);
void ProcessStockLevelForm(EXTENSION_CONTROL_BLOCK *pECB, int iTermId,
char *szBuffer);
void GetNewOrderData(LPSTR lpszQueryString, NEW_ORDER_DATA
*pNewOrderData);
void GetPaymentData(LPSTR lpszQueryString, PAYMENT_DATA *pPaymentData);
void GetOrderStatusData(LPSTR lpszQueryString, ORDER_STATUS_DATA
*pOrderStatusData);
BOOL PostDeliveryInfo(short w_id, short o_carrier_id);
BOOL IsNumeric(char *ptr);
BOOL IsDecimal(char *ptr);
void DeliveryWorkerThread(void *ptr);

//Microsoft Developer Studio generated resource script.
//
#include "resource.h"

#define APSTUDIO_READONLY_SYMBOLS
////////////////////////////////////////////////////////////////////////////
////
//
// Generated from the TEXTINCLUDE 2 resource.
//
#include "afxres.h"

////////////////////////////////////////////////////////////////////////////
////
#undef APSTUDIO_READONLY_SYMBOLS

////////////////////////////////////////////////////////////////////////////
////
// English (U.S.) resources

#if !defined(AFX_RESOURCE_DLL) || defined(AFX_TARG_ENU)
#ifdef _WIN32
LANGUAGE LANG_ENGLISH, SUBLANG_ENGLISH_US
#pragma code_page(1252)
#endif //_WIN32

#ifndef _MAC
////////////////////////////////////////////////////////////////////////////
////
//
// Version
//

VS_VERSION_INFO VERSIONINFO
 FILEVERSION 0,4,0,0
 PRODUCTVERSION 0,4,0,0
 FILEFLAGSMASK 0x3fL
#ifdef _DEBUG
 FILEFLAGS 0x1L
#else
 FILEFLAGS 0x0L
#endif
 FILEOS 0x40004L
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
 FILETYPE 0x2L
 FILESUBTYPE 0x0L
BEGIN
    BLOCK "StringFileInfo"
    BEGIN
        BLOCK "040904b0"
        BEGIN
            VALUE "Comments", "TPC-C HTML DLL Server (DBLIB)\0"
            VALUE "CompanyName", "Microsoft\0"
            VALUE "FileDescription", "TPC-C HTML DLL Server (DBLIB)\0"
            VALUE "FileVersion", "0, 4, 0, 0\0"
            VALUE "InternalName", "tpcc\0"
            VALUE "LegalCopyright", "Copyright © 1997\0"
            VALUE "OriginalFilename", "tpcc.dll\0"
            VALUE "ProductName", "Microsoft tpcc\0"
            VALUE "ProductVersion", "0, 4, 0, 0\0"
        END
    END
    BLOCK "VarFileInfo"
    BEGIN
        VALUE "Translation", 0x409, 1200
    END
END

#endif    // !_MAC


#ifdef APSTUDIO_INVOKED
//////////////////////////////////////////////////////////////////////////
////
//
// TEXTINCLUDE
//

1 TEXTINCLUDE DISCARDABLE
BEGIN
    "resource.h\0"
END

2 TEXTINCLUDE DISCARDABLE
BEGIN
    "#include ""afxres.h""\r\n"
    "\0"
END

3 TEXTINCLUDE DISCARDABLE
BEGIN
    "\r\n"
    "\0"
END

#endif    // APSTUDIO_INVOKED


//////////////////////////////////////////////////////////////////////////
////
//
// Dialog
//

IDD_DIALOG1 DIALOG DISCARDABLE  0, 0, 186, 95
STYLE DS_MODALFRAME | WS_POPUP | WS_CAPTION | WS_SYSMENU
CAPTION "Dialog"
FONT 8, "MS Sans Serif"
BEGIN
    DEFPUSHBUTTON   "OK",IDOK,129,7,50,14
    PUSHBUTTON      "Cancel",IDCANCEL,129,24,50,14
END


//////////////////////////////////////////////////////////////////////////
////
//
// DESIGNINFO
//

#ifdef APSTUDIO_INVOKED
GUIDELINES DESIGNINFO DISCARDABLE
BEGIN
    IDD_DIALOG1, DIALOG
    BEGIN
        LEFTMARGIN, 7
        RIGHTMARGIN, 179
        TOPMARGIN, 7
        BOTTOMMARGIN, 88
    END
END
#endif    // APSTUDIO_INVOKED

#endif    // English (U.S.) resources
//////////////////////////////////////////////////////////////////////////
////



#ifndef APSTUDIO_INVOKED
//////////////////////////////////////////////////////////////////////////
////
//
// Generated from the TEXTINCLUDE 3 resource.
//


//////////////////////////////////////////////////////////////////////////
////
#endif    // not APSTUDIO_INVOKED
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
/*      FILE:           TPCC.C
 *                              Microsoft TPC-C Kit Ver. 4.20.000
 *                              Copyright Microsoft, 1999
 *              All Rights Reserved
 *
 *                              Version 4.10.000 audited by Richard Gimarc,
Performance Metrics, 3/17/99
 *
 *      PURPOSE:        Main module for TPCC.DLL which is an ISAPI service
dll.
 *      Contact:        Charles Levine (clevine@microsoft.com)
 *
 *  Change history:
 *              4.20.000 - reworked error handling; added options for COM
and Encina txn monitors
 */

#include <windows.h>
#include <process.h>
#include <tchar.h>
#include <stdio.h>
#include <stdarg.h>
#include <malloc.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <sys\timeb.h>
#include <io.h>
#include <assert.h>

#include <sqltypes.h>

#ifdef ICECAP
#include <icapexp.h>
#endif

#include "..\..\common\src\trans.h"          //tpckit transaction header
contains defination of structures specific to TPC-C
#include "..\..\common\src\error.h"
#include "..\..\common\src\txn_base.h"
#include "..\..\common\src\ReadRegistry.h"

#include "..\..\common\txnlog\include\rtetime.h"
#include "..\..\common\txnlog\include\spinlock.h"
#include "..\..\common\txnlog\include\txnlog.h"

// Database layer includes
#include "..\..\db_dblib_dll\src\tpcc_dblib.h"              // DBLIB
implementation of TPC-C txns
#include "..\..\db_odbc_dll\src\tpcc_odbc.h"        // ODBC
implementation of TPC-C txns

// Txn monitor layer includes
#include "..\..\tm_com_dll\src\tpcc_com.h"                  // COM
Services implementation on TPC-C txns
#include "..\..\tm_tuxedo_dll\src\tpcc_tux.h"               // interface
to Tuxedo libraries
#include "..\..\tm_encina_dll\src\tpcc_enc.h"               // interface
to Encina libraries

#include "httpext.h"                          //ISAPI DLL information
header
#include "tpcc.h"                                     //this dlls specific
structure, value e.t. header.


#define LEN_ERR_STRING 256

// defines for Make<Txn>Form calls to distinguish input and output
flavors
#define OUTPUT_FORM    0
#define INPUT_FORM     1

char           szMyComputerName[MAX_COMPUTERNAME_LENGTH+1];

//Terminal client id structure
TERM    Term = { 0, 0, 0, NULL };

// The WEBCLIENT_VERSION string specifies the version level of this web
client interface.
// The RTE must be synchronized with the interface level on login,
otherwise the login
// will fail.  This is a sanity check to catch problems resulting from
mismatched versions
// of the RTE and web client.
#define WEBCLIENT_VERSION "410"

static  CRITICAL_SECTION              TermCriticalSection;

static  HINSTANCE hLibInstanceTm = NULL;
static  HINSTANCE hLibInstanceDb = NULL;

TYPE_CTPCC_DBLIB       *pCTPCC_DBLIB_new;
TYPE_CTPCC_ODBC                *pCTPCC_ODBC_new;
TYPE_CTPCC_TUXEDO      *pCTPCC_TUXEDO_new;
TYPE_CTPCC_ENCINA   *pCTPCC_ENCINA_new;
TYPE_CTPCC_ENCINA   *pCTPCC_ENCINA_post_init;
TYPE_CTPCC_COM         *pCTPCC_COM_new;

// For deferred Delivery txns:

CTxnLog                               *txnDelilog = NULL;
       //used to log delivery transaction information
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
HANDLE                                  hWorkerSemaphore       =
INVALID_HANDLE_VALUE;
HANDLE                                  hDoneEvent                     =
INVALID_HANDLE_VALUE;
HANDLE                                  *pDeliHandles          = NULL;

// configuration settings from registry
TPCCREGISTRYDATA              Reg;


DWORD                                   dwNumDeliveryThreads = 4;
CRITICAL_SECTION              DelBuffCriticalSection;
       //critical section for delivery transactions cache
DELIVERY_TRANSACTION   *pDelBuff                       = NULL;
DWORD                                   dwDelBuffSize          = 100;
       // size of circular buffer for delivery txns
DWORD                                   dwDelBuffFreeCount;
              // number of buffers free
DWORD                                   dwDelBuffBusyIndex     = 0;
       // index position of entry waiting to be delivered
DWORD                                   dwDelBuffFreeIndex     = 0;
       // index position of unused entry

#include "..\..\common\src\ReadRegistry.cpp"

/* FUNCTION: DllMain
 *
 * PURPOSE:    This function is the entry point for the DLL.  This
implementation is based on the
 *                     fact that DLL_PROCESS_ATTACH is only called from
the inet service once.
 *
 * ARGUMENTS:  HANDLE  hModule                            module handle
 *                             DWORD   ul_reason_for_call     reason for
call
 *                             LPVOID  lpReserved
       reserved for future use
 *
 * RETURNS:            BOOL    FALSE                              errors
occured in initialization
 *                                         TRUE
       DLL successfully initialized
 */

BOOL APIENTRY DllMain(HANDLE hModule, DWORD ul_reason_for_call, LPVOID
lpReserved)
{
       DWORD  i;
       char szEvent[LEN_ERR_STRING] = "\0";
       char szLogFile[128];
       char szDllName[128];

       try
       {
              switch( ul_reason_for_call )
              {
                     case DLL_PROCESS_ATTACH:
                            {
                                   DWORD dwSize =
MAX_COMPUTERNAME_LENGTH+1;
                                   GetComputerName(szMyComputerName,
&dwSize);
                                   szMyComputerName[dwSize] = 0;
                            }


       DisableThreadLibraryCalls((HMODULE)hModule);

       InitializeCriticalSection(&TermCriticalSection);

                            if ( ReadTPCCRegistrySettings( &Reg ) )
                                   throw new CWEBCLNT_ERR(
ERR_MISSING_REGISTRY_ENTRIES );

                            dwDelBuffSize = min(
Reg.dwMaxPendingDeliveries, 10000 );  // min with 10000 as a sanity
constraint
                            dwNumDeliveryThreads = min(
Reg.dwNumberOfDeliveryThreads, 100 );  // min with 100 as a sanity
constraint

                            TermInit();

                            // load DLL for txn monitor
                            if (Reg.eTxnMon == TUXEDO)
                            {
                                   strcpy( szDllName, Reg.szPath );
                                   strcat( szDllName,
"tpcc_tuxedo.dll");
                                   hLibInstanceTm = LoadLibrary(
szDllName );
                                   if (hLibInstanceTm == NULL)
                                          throw new CWEBCLNT_ERR(
ERR_LOADDLL_FAILED, szDllName, GetLastError() );

                                   // get function pointer to wrapper
for class constructor
                                   pCTPCC_TUXEDO_new =
(TYPE_CTPCC_TUXEDO*) GetProcAddress(hLibInstanceTm,"CTPCC_TUXEDO_new");
                                   if (pCTPCC_TUXEDO_new == NULL)
                                          throw new CWEBCLNT_ERR(
ERR_GETPROCADDR_FAILED, szDllName, GetLastError() );
                            }
                            else if (Reg.eTxnMon == ENCINA)
                            {
                                   strcpy( szDllName, Reg.szPath );
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                                        strcat( szDllName,
"tpcc_encina.dll");
                                        hLibInstanceTm = LoadLibrary(
szDllName );
                                        if (hLibInstanceTm == NULL)
                                                throw new CWEBCLNT_ERR(
ERR_LOADDLL_FAILED, szDllName, GetLastError() );
                                        // get function pointer to wrapper
for class constructor
                                        pCTPCC_ENCINA_new =
(TYPE_CTPCC_ENCINA*) GetProcAddress(hLibInstanceTm,"CTPCC_ENCINA_new");
                                        pCTPCC_ENCINA_post_init =
(TYPE_CTPCC_ENCINA*)
GetProcAddress(hLibInstanceTm,"CTPCC_ENCINA_post_init");
                                        if (pCTPCC_ENCINA_new == NULL)
                                                throw new CWEBCLNT_ERR(
ERR_GETPROCADDR_FAILED, szDllName, GetLastError() );
                                }
                                else if (Reg.eTxnMon == COM)
                                {
                                        strcpy( szDllName, Reg.szPath );
                                        strcat( szDllName, "tpcc_com.dll");
                                        hLibInstanceTm = LoadLibrary(
szDllName );
                                        if (hLibInstanceTm == NULL)
                                                throw new CWEBCLNT_ERR(
ERR_LOADDLL_FAILED, szDllName, GetLastError() );

                                        // get function pointer to wrapper
for class constructor
                                        pCTPCC_COM_new = (TYPE_CTPCC_COM*)
GetProcAddress(hLibInstanceTm,"CTPCC_COM_new");
                                        if (pCTPCC_COM_new == NULL)
                                                throw new CWEBCLNT_ERR(
ERR_GETPROCADDR_FAILED, szDllName, GetLastError() );
                                }

                                // load DLL for database connection
                                if ((Reg.eTxnMon == None) ||
(dwNumDeliveryThreads > 0))
                                {
                                        if (Reg.eDB_Protocol == DBLIB)
                                        {
                                                strcpy( szDllName,
Reg.szPath );
                                                strcat( szDllName,
"tpcc_dblib.dll");
                                                hLibInstanceDb =
LoadLibrary( szDllName );
                                                if (hLibInstanceDb == NULL)
                                                        throw new
CWEBCLNT_ERR( ERR_LOADDLL_FAILED, szDllName, GetLastError() );
                                                // get function pointer to
wrapper for class constructor
                                                pCTPCC_DBLIB_new =
(TYPE_CTPCC_DBLIB*) GetProcAddress(hLibInstanceDb,"CTPCC_DBLIB_new");
                                                if (pCTPCC_DBLIB_new ==
NULL)
                                                        throw new
CWEBCLNT_ERR( ERR_GETPROCADDR_FAILED, szDllName, GetLastError() );
                                        }
                                        else if (Reg.eDB_Protocol == ODBC)
                                        {
                                                strcpy( szDllName,
Reg.szPath );
                                                strcat( szDllName,
"tpcc_odbc.dll");
                                                hLibInstanceDb =
LoadLibrary( szDllName );
                                                if (hLibInstanceDb == NULL)
                                                        throw new
CWEBCLNT_ERR( ERR_LOADDLL_FAILED, szDllName, GetLastError() );

                                                // get function pointer to
wrapper for class constructor
                                                pCTPCC_ODBC_new =
(TYPE_CTPCC_ODBC*) GetProcAddress(hLibInstanceDb,"CTPCC_ODBC_new");
                                                if (pCTPCC_ODBC_new == NULL)
                                                        throw new
CWEBCLNT_ERR( ERR_GETPROCADDR_FAILED, szDllName, GetLastError() );
                                        }
                                }

                                if (dwNumDeliveryThreads)
                                {
                                        // for deferred delivery txns:
                                        hDoneEvent = CreateEvent( NULL, TRUE
/* manual reset */, FALSE /* initially not signalled */, NULL );

        InitializeCriticalSection(&DelBuffCriticalSection);
                                        hWorkerSemaphore = CreateSemaphore(
NULL, 0, dwDelBuffSize, NULL );
                                        dwDelBuffFreeCount = dwDelBuffSize;

                                        InitJulianTime(NULL);

                                        // create unique log file name based
on delilog-yymmdd-hhmm.log
                                        SYSTEMTIME Time;
                                        GetLocalTime( &Time );
                                        wsprintf( szLogFile, "%sdelivery-
%2.2d%2.2d%2.2d-%2.2d%2.2d.log",
                                                        Reg.szPath,
Time.wYear % 100, Time.wMonth, Time.wDay, Time.wHour, Time.wMinute );
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                                        txnDelilog = new CTxnLog(szLogFile,
TXN_LOG_WRITE);

                                        //write event into txn log for START
                                        txnDelilog-
>WriteCtrlRecToLog(TXN_EVENT_START, szMyComputerName,
sizeof(szMyComputerName));

                                        // allocate structures for delivery
buffers and thread mgmt
                                        pDeliHandles = new
HANDLE[dwNumDeliveryThreads];
                                        pDelBuff = new
DELIVERY_TRANSACTION[dwDelBuffSize];
                                        // launch DeliveryWorkerThread to
perform actual delivery txns
                                        for(i=0; i<dwNumDeliveryThreads;
i++)
                                        {
                                                pDeliHandles[i] = (HANDLE)
_beginthread( DeliveryWorkerThread, 0, NULL );
                                                if (pDeliHandles[i] ==
INVALID_HANDLE_VALUE)
                                                        throw new
CWEBCLNT_ERR( ERR_DELIVERY_THREAD_FAILED );
                                        }
                                }

                                break;

                        case DLL_PROCESS_DETACH:
                                if (dwNumDeliveryThreads)
                                {
                                        if (txnDelilog != NULL)
                                        {
                                                //write event into txn log
for STOP
                                                txnDelilog-
>WriteCtrlRecToLog(TXN_EVENT_STOP, szMyComputerName,
sizeof(szMyComputerName));

                                                // This will do a clean
shutdown of the delivery log file
                                                CTxnLog *txnDelilogLocal =
txnDelilog;
                                                txnDelilog= NULL;
                                                delete txnDelilogLocal;
                                        }

                                        delete [] pDeliHandles;
                                        delete [] pDelBuff;

                                        CloseHandle( hWorkerSemaphore );
                                        CloseHandle( hDoneEvent );

                                DeleteCriticalSection(&DelBuffCriticalSection);
                                }


                                DeleteCriticalSection(&TermCriticalSection);

                                if (hLibInstanceTm != NULL)
                                        FreeLibrary( hLibInstanceTm );
                                hLibInstanceTm = NULL;

                                if (hLibInstanceDb != NULL)
                                        FreeLibrary( hLibInstanceDb );
                                hLibInstanceDb = NULL;

                                Sleep(500);
                                break;

                        default:
                                /* nothing */;
                        }
                }
                catch (CBaseErr *e)
                {
                        WriteMessageToEventLog( e->ErrorText() );
                        delete e;
                        TerminateExtension(0);
                        return FALSE;
                }
                catch (...)
                {
                        WriteMessageToEventLog(TEXT("Unhandled exception.  DLL
could not load."));
                        TerminateExtension(0);
                        return FALSE;
                }

                return TRUE;
}


/* FUNCTION: GetExtensionVersion
 *
 * PURPOSE:    This function is called by the inet service when the DLL
is first loaded.
 *
 * ARGUMENTS:  HSE_VERSION_INFO      *pVer   passed in structure in which
to place expected version number.
 *
 * RETURNS:           TRUE    inet service expected return value.
 */
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
BOOL WINAPI GetExtensionVersion(HSE_VERSION_INFO *pVer)
{
	pVer->dwExtensionVersion = MAKELONG(HSE_VERSION_MINOR,
HSE_VERSION_MAJOR);
	lstrcpyn(pVer->lpszExtensionDesc, "TPC-C Server.",
HSE_MAX_EXT_DLL_NAME_LEN);

	// TODO: why do we need this here instead of in the DLL attach?
	if (Reg.eTxnMon == ENCINA)
		pCTPCC_ENCINA_post_init();

	return TRUE;
}

/* FUNCTION: TerminateExtension
 *
 * PURPOSE:    This function is called by the inet service when the DLL
is about to be unloaded.
 *                     Release all resources in anticipation of being
unloaded.
 *
 * RETURNS:            TRUE    inet service expected return value.
 */

BOOL WINAPI TerminateExtension( DWORD dwFlags )
{
	if (pDeliHandles)
	{
		SetEvent( hDoneEvent );
		for(DWORD i=0; i<dwNumDeliveryThreads; i++)
			WaitForSingleObject( pDeliHandles[i], INFINITE );
	}

	TermDeleteAll();
	return TRUE;
}


/* FUNCTION: HttpExtensionProc
 *
 * PURPOSE:    This function is the main entry point for the TPCC DLL.
The internet service
 *                     calls this function passing in the http string.
 *
 * ARGUMENTS:  EXTENSION_CONTROL_BLOCK       *pECB   structure pointer to
passed in internet
 *
 *                     service information.
 *
 * RETURNS:            DWORD   HSE_STATUS_SUCCESS
 *
	HSE_STATUS_SUCCESS_AND_KEEP_CONN      keep connect valid comment
sent
 *
 * COMMENTS:   None
 *
 */

DWORD WINAPI HttpExtensionProc(EXTENSION_CONTROL_BLOCK *pECB)
{
	int                     iCmd, FormId, TermId, iSyncId;
	char            szBuffer[4096];

	int                     lpbSize;
	static char     szHeader[] = "200 Ok";
	DWORD           dwSize = 6;             // initial value is
strlen(szHeader)
	char            szHeader1[4096];


#ifdef ICECAP
	StartCAP();
#endif

	try
	{
		//process http query
		ProcessQueryString(pECB, &iCmd, &FormId, &TermId,
&iSyncId);

		if (TermId != 0)
		{
			if ( TermId < 0 || TermId >= Term.iNumEntries ||
Term.pClientData[TermId].iNextFree != -1 )
			{
				// debugging...
				char szTmp[128];
				wsprintf( szTmp, "Invalid term ID; TermId =
%d", TermId );
				WriteMessageToEventLog( szTmp );

				throw new CWEBCLNT_ERR( ERR_INVALID_TERMID
);
			}

			//must have a valid syncid here since termid is
valid
			if (iSyncId != Term.pClientData[TermId].iSyncId)
				throw new CWEBCLNT_ERR(
ERR_INVALID_SYNC_CONNECTION );

			//set use time
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                        Term.pClientData[TermId].iTickCount =
GetTickCount();
                }


                switch(iCmd)
                {
                case 0:
                        WelcomeForm(pECB, szBuffer);
                        break;
                case 1:
                        switch( FormId )
                        {
                                case WELCOME_FORM:
                                case MAIN_MENU_FORM:
                                        break;
                                case NEW_ORDER_FORM:
                                        ProcessNewOrderForm(pECB, TermId,
szBuffer);
                                        break;
                                case PAYMENT_FORM:
                                        ProcessPaymentForm(pECB, TermId,
szBuffer);
                                        break;
                                case DELIVERY_FORM:
                                        ProcessDeliveryForm(pECB, TermId,
szBuffer);
                                        break;
                                case ORDER_STATUS_FORM:
                                        ProcessOrderStatusForm(pECB, TermId,
szBuffer);
                                        break;
                                case STOCK_LEVEL_FORM:
                                        ProcessStockLevelForm(pECB, TermId,
szBuffer);
                                        break;
                        }
                        break;
                case 2:
                        // new-order selected from menu; display new-order
input form
                        MakeNewOrderForm(TermId, NULL, INPUT_FORM,
szBuffer);
                        break;
                case 3:
                        // payment selected from menu; display payment
input form
                        MakePaymentForm(TermId, NULL, INPUT_FORM,
szBuffer);
                        break;
                case 4:
                        // delivery selected from menu; display delivery
input form
                        MakeDeliveryForm(TermId, NULL, INPUT_FORM,
szBuffer);
                        break;
                case 5:
                        // order-status selected from menu; display order-
status input form
                        MakeOrderStatusForm(TermId, NULL, INPUT_FORM,
szBuffer);
                        break;
                case 6:
                        // stock-level selected from menu; display stock-
level input form
                        MakeStockLevelForm(TermId, NULL, INPUT_FORM,
szBuffer);
                        break;
                case 7:
                        // ExitCmd
                        TermDelete(TermId);
                        WelcomeForm(pECB, szBuffer);
                        break;
                case 8:
                        SubmitCmd(pECB, szBuffer);
                        break;
                case 9:
                        // menu
                        MakeMainMenuForm(TermId,
Term.pClientData[TermId].iSyncId, szBuffer);
                        break;
                case 10:
                        // CMD=Clear
                        // resets all connections; should only be used when
no other connections are active
                        TermDeleteAll();
                        TermInit();
                        WelcomeForm(pECB, szBuffer);
                        break;
                case 11:        // CMD=Stats
                        StatsCmd(pECB, szBuffer);
                        break;
                }
        }
        catch (CBaseErr *e)
        {
                ErrorForm( pECB, e->ErrorType(), e->ErrorNum(), TermId,
iSyncId, e->ErrorText(), szBuffer );
                delete e;
        }
        catch (...)
        {
                ErrorForm( pECB, ERR_TYPE_WEBDLL, 0, TermId, iSyncId,
"Error: Unhandled exception in Web Client.", szBuffer );
        }
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
#ifdef ICECAP
        StopCAP();
#endif


        lpbSize = strlen(szBuffer);
        wsprintf(szHeader1,
                        "Content-Type: text/html\r\n"
                        "Content-Length: %d\r\n"
                        "Connection: Keep-Alive\r\n\r\n" , lpbSize);
        strcat( szHeader1, szBuffer );

        (*pECB->ServerSupportFunction)(pECB->ConnID,
HSE_REQ_SEND_RESPONSE_HEADER, szHeader, (LPDWORD) &dwSize,
(LPDWORD)szHeader1);

        //finish up and keep connection
        pECB->dwHttpStatusCode = 200;
        return HSE_STATUS_SUCCESS_AND_KEEP_CONN;
}


void WriteMessageToEventLog(LPTSTR lpszMsg)
{
    TCHAR   szMsg[256];
    HANDLE  hEventSource;
    LPTSTR  lpszStrings[2];

    // Use event logging to log the error.
    //
    hEventSource = RegisterEventSource(NULL, TEXT("TPCC.DLL"));

    _stprintf(szMsg, TEXT("Error in TPCC.DLL: "));
    lpszStrings[0] = szMsg;
    lpszStrings[1] = lpszMsg;

    if (hEventSource != NULL)
        {
        ReportEvent(hEventSource, // handle of event source
            EVENTLOG_ERROR_TYPE,  // event type
            0,                    // event category
            0,                    // event ID
            NULL,                 // current user's SID
            2,                    // strings in lpszStrings
            0,                    // no bytes of raw data
            (LPCTSTR *)lpszStrings,    // array of error strings
            NULL);                // no raw data

        (VOID) DeregisterEventSource(hEventSource);
    }
}
```

```
/* FUNCTION: DeliveryWorkerThread
 *
 * PURPOSE:    This function processes deferred delivery txns.  There are
typically several
 *                     threads running this routine.  The number of
threads is determined by an entry
 *                     read from the registry.  The thread waits for work
by waiting on semaphore.
 *                     When a delivery txn is posted, the semaphore is
released.  After processing
 *                     the delivery txn, information is logged to record
the txn status and execution
 *                     time.
 */

/*static*/ void DeliveryWorkerThread(void *ptr)
{
        CTPCC_BASE              *pTxn = NULL;

        DELIVERY_TRANSACTION          delivery;
        PDELIVERY_DATA                        pDeliveryData;
        TXN_RECORD_TPCC_DELIV_DEF     txnDeliRec;

        DWORD                                 index;
        HANDLE                                handles[2];

        SYSTEMTIME              trans_end;              //delivery
transaction finished time
        SYSTEMTIME              trans_start;    //delivery transaction start
time

        assert(txnDelilog != NULL);

        try
        {
                if (Reg.eDB_Protocol == ODBC)
                        pTxn = pCTPCC_ODBC_new( Reg.szDbServer,
Reg.szDbUser, Reg.szDbPassword, szMyComputerName, Reg.szDbName );
                else if (Reg.eDB_Protocol == DBLIB)
                        pTxn = pCTPCC_DBLIB_new( Reg.szDbServer,
Reg.szDbUser, Reg.szDbPassword, szMyComputerName, Reg.szDbName );
                pDeliveryData = pTxn->BuffAddr_Delivery();
        }
        catch (CBaseErr *e)
        {
                char szTmp[1024];
                wsprintf( szTmp, "Error in Delivery Txn thread.  Could not
connect to database.  "
                                "%s.  Server=%s, User=%s, Password=%s,
Database=%s",
                                e->ErrorText(), Reg.szDbServer,
Reg.szDbUser, Reg.szDbPassword, Reg.szDbName );
                WriteMessageToEventLog( szTmp );
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                delete e;
                goto ErrorExit;
        }
        catch (...)
        {
                WriteMessageToEventLog(TEXT("Unhandled exception caught in
DeliveryWorkerThread."));
                goto ErrorExit;
        }

        while (TRUE)
        {
                try
                {
                        //while delivery thread running, i.e. user has not
requested termination
                        while (TRUE)
                        {
                                // need to wait for multiple objects:
program exit or worker semaphore;
                                handles[0] = hDoneEvent;
                                handles[1] = hWorkerSemaphore;
                                index = WaitForMultipleObjects( 2,
&handles[0], FALSE, INFINITE );
                                if (index == WAIT_OBJECT_0)
                                        goto ErrorExit;

                                ZeroMemory(&txnDeliRec,
sizeof(txnDeliRec));
                                txnDeliRec.TxnType =
TXN_REC_TYPE_TPCC_DELIV_DEF;

                                // make a local copy of current entry from
delivery buffer and increment buffer index

        EnterCriticalSection(&DelBuffCriticalSection);
                                delivery = *(pDelBuff+dwDelBuffBusyIndex);
                                dwDelBuffFreeCount++;
                                dwDelBuffBusyIndex++;
                                if (dwDelBuffBusyIndex == dwDelBuffSize)
        // wrap-around if at end of buffer
                                        dwDelBuffBusyIndex = 0;


        LeaveCriticalSection(&DelBuffCriticalSection);

                                pDeliveryData->w_id = delivery.w_id;
                                pDeliveryData->o_carrier_id =
delivery.o_carrier_id;

                                txnDeliRec.w_id = pDeliveryData->w_id;
                                txnDeliRec.o_carrier_id = pDeliveryData-
>o_carrier_id;
                                txnDeliRec.TxnStartT0 =
Get64BitTime(&delivery.queue);

                                GetLocalTime( &trans_start );
                                pTxn->Delivery();
                                GetLocalTime( &trans_end );

                                //log txn
                                txnDeliRec.TxnStatus = ERR_SUCCESS;
                                for (int i=0; i<10; i++)
                                        txnDeliRec.o_id[i] = pDeliveryData-
>o_id[i];
                                txnDeliRec.DeltaT4 =
(int)(Get64BitTime(&trans_end) - txnDeliRec.TxnStartT0);
                                txnDeliRec.DeltaTxnExec =
(int)(Get64BitTime(&trans_end) - Get64BitTime(&trans_start));

                                if (txnDelilog != NULL)
                                        txnDelilog->WriteToLog(&txnDeliRec);
                        }
                }
                catch (CBaseErr *e)
                {
                        char szTmp[1024];
                        wsprintf( szTmp, "Error in Delivery Txn thread.
%s", e->ErrorText() );
                        WriteMessageToEventLog( szTmp );
                        delete e;

                        // log the error txn
                        txnDeliRec.TxnStatus = e->ErrorType();
                        if (txnDelilog != NULL)
                                txnDelilog->WriteToLog(&txnDeliRec);
                }
                catch (...)
                {
                        // unhandled exception; shouldn't happen; not much
we can do...
                        WriteMessageToEventLog(TEXT("Unhandled exception
caught in DeliveryWorkerThread."));
                }
        }

ErrorExit:
        delete pTxn;
        _endthread();
}

/* FUNCTION: PostDeliveryInfo
 *
 * PURPOSE:    This function enters the delivery txn into the deferred
delivery buffer.
 *
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
 * RETURNS:            BOOL    FALSE   delivery information posted
successfully
 *                                     TRUE    error cannot post
delivery info
 */

BOOL PostDeliveryInfo(short w_id, short o_carrier_id)
{
        BOOL bError;

        EnterCriticalSection(&DelBuffCriticalSection);
        if (dwDelBuffFreeCount > 0)
        {
                bError = FALSE;
                (pDelBuff+dwDelBuffFreeIndex)->w_id                  =
w_id;
                (pDelBuff+dwDelBuffFreeIndex)->o_carrier_id  =
o_carrier_id;
                GetLocalTime(&(pDelBuff+dwDelBuffFreeIndex)->queue);

                dwDelBuffFreeCount--;
                dwDelBuffFreeIndex++;
                if (dwDelBuffFreeIndex == dwDelBuffSize)
                        dwDelBuffFreeIndex = 0;              // wrap-
around if at end of buffer
        }
        else
                // No free buffers.  Return an error, which indicates that
the delivery buffer is full.
                // Most likely, the number of delivery worker threads
needs to be increased to keep up
                // with the txn rate.
                bError = TRUE;
        LeaveCriticalSection(&DelBuffCriticalSection);

        if (!bError)
                // increment worker semaphore to wake up a worker thread
                ReleaseSemaphore( hWorkerSemaphore, 1, NULL );

        return bError;
}

/* FUNCTION: ProcessQueryString
 *
 * PURPOSE:    This function extracts the relevent information out of the
http command passed in from
 *                     the browser.
 *
 * COMMENTS:   If this is the initial connection i.e. client is at
welcome screen then
 *                             there will not be a terminal id or current
form id.  If this is the case
 *                             then the pTermid and pFormid return values
are undefined.
 */

void ProcessQueryString(EXTENSION_CONTROL_BLOCK *pECB, int *pCmd, int
*pFormId, int *pTermId, int *pSyncId)
{
        char *ptr = pECB->lpszQueryString;
        char szBuffer[25];
        int i;

        //allowable client command strings i.e. CMD=command
        static char *szCmds[] =
        {
                "Process", "..NewOrder..", "..Payment..", "..Delivery..",
"..Order-Status..", "..Stock-Level..",
                "..Exit..", "Submit", "Menu", "Clear", "Stats", ""
        };

        *pCmd    = 0;          // default is the login screen
        *pTermId = 0;

        // if no params (i.e., empty query string), then return login
screen
        if (strlen(pECB->lpszQueryString) == 0)
                return;

        // parse FORMID, TERMID, and SYNCID
        *pFormId = GetIntKeyValue(&ptr, "FORMID", NO_ERR, NO_ERR);
        *pTermId = GetIntKeyValue(&ptr, "TERMID", NO_ERR, NO_ERR);
        *pSyncId = GetIntKeyValue(&ptr, "SYNCID", NO_ERR, NO_ERR);

        // parse CMD
        GetKeyValue(&ptr, "CMD", szBuffer, sizeof(szBuffer),
ERR_COMMAND_UNDEFINED);

        // see which command it matches
        for(i=0; ; i++)
        {
                if (szCmds[i][0] == 0)
                        // no more; no match; return error
                        throw new CWEBCLNT_ERR( ERR_COMMAND_UNDEFINED );
                if ( !strcmp(szCmds[i], szBuffer) )
                {
                        *pCmd = i+1;
                        break;
                }
        }
}

/* FUNCTION: void WelcomeForm
 *
 */
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
void WelcomeForm(EXTENSION_CONTROL_BLOCK *pECB, char *szBuffer)
{
	char szTmp[1024];

	//welcome to tpc-c html form buffer, this is first form client
sees.
	strcpy( szBuffer,	"<HTML><HEAD><TITLE>TPC-C Web
Client</TITLE></HEAD><BODY>"
					"<B><BIG>Microsoft TPC-C Web
Client (ver 4.20)</BIG></B> <BR> <BR>"
					"<font face=\"Courier
New\"><PRE>"
					"Compiled: "__DATE__",
"__TIME__" <BR>"
					"Source:   "__FILE__"
("__TIMESTAMP__") <BR>"
					"</PRE></font>"
					"<FORM ACTION=\"tpcc.dll\"
METHOD=\"GET\">"
					"<INPUT TYPE=\"hidden\"
NAME=\"STATUSID\" VALUE=\"0\">"
					"<INPUT TYPE=\"hidden\"
NAME=\"ERROR\" VALUE=\"0\">"
					"<INPUT TYPE=\"hidden\"
NAME=\"FORMID\" VALUE=\"1\">"
					"<INPUT TYPE=\"hidden\"
NAME=\"TERMID\" VALUE=\"0\">"
					"<INPUT TYPE=\"hidden\"
NAME=\"SYNCID\" VALUE=\"0\">"
					"<INPUT TYPE=\"hidden\"
NAME=\"VERSION\" VALUE=\"" WEBCLIENT_VERSION "\">"
			);

	sprintf( szTmp,	"Configuration Settings: <BR><font
face=\"Courier New\" color=\"blue\"><PRE>"
				"Txn Monitor		=
<B>%s</B><BR>"
				"Database protocol	=
<B>%s</B><BR>"
				"Max Connections	=
<B>%d</B><BR>"
				"# of Delivery Threads	=
<B>%d</B><BR>"
				"Max Pending Deliveries =
<B>%d</B><BR>"
			, szTxnMonNames[Reg.eTxnMon],
szDBNames[Reg.eDB_Protocol],
			Reg.dwMaxConnections, dwNumDeliveryThreads,
dwDelBuffSize );
	strcat( szBuffer, szTmp);

	if (Reg.eTxnMon == COM)
	{
		sprintf( szTmp,	"COM Single Pool	=
<B>%s</B><BR>",
				Reg.bCOM_SinglePool ? "YES" : "NO" );
		strcat( szBuffer, szTmp);
	}
	strcat( szBuffer, "</PRE></font>");

	if (Reg.eTxnMon == None)
		// connection options may be specified when not using a
txn monitor
		sprintf( szTmp,	"Please enter your database options
for this connection:<BR>"
						"<font face=\"Courier New\"
color=\"blue\"><PRE>"
						"DB Server    = <INPUT
NAME=\"db_server\" SIZE=20 VALUE=\"%s\"><BR>"
						"DB User ID   = <INPUT
NAME=\"db_user\" SIZE=20 VALUE=\"%s\"><BR>"
						"DB Password  = <INPUT
NAME=\"db_passwd\" SIZE=20 VALUE=\"%s\"><BR>"
						"DB Name      = <INPUT
NAME=\"db_name\" SIZE=20 VALUE=\"%s\"><BR>"
						"</PRE></font>"
				, Reg.szDbServer, Reg.szDbUser,
Reg.szDbPassword, Reg.szDbName );
	else
		// if using a txn monitor, connection options are
determined from registry; can't
		// set per user.  show options fyi
		sprintf( szTmp,	"Database options which will be used
by the transaction monitor:<BR>"
						"<font face=\"Courier New\"
color=\"blue\"><PRE>"
						"DB Server		=
<B>%s</B><BR>"
						"DB User ID		=
<B>%s</B><BR>"
						"DB Password		=
<B>%s</B><BR>"
						"DB Name		=
<B>%s</B><BR>"
						"</PRE></font>"
				, Reg.szDbServer, Reg.szDbUser,
Reg.szDbPassword, Reg.szDbName );
	strcat( szBuffer, szTmp);

	sprintf( szTmp,	"Please enter your Warehouse and District
for this session:<BR>"
					"<font face=\"Courier New\"
color=\"blue\"><PRE>" );
	strcat( szBuffer, szTmp);
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        strcat( szBuffer,      "Warehouse ID = <INPUT NAME=\"w_id\"
SIZE=4><BR>"
                                                "District ID  = <INPUT
NAME=\"d_id\" SIZE=2><BR>"
                                                "</PRE></font><HR>"
                                                "<INPUT TYPE=\"submit\"
NAME=\"CMD\" VALUE=\"Submit\">"
                                                "</FORM></BODY></HTML>");
}

/* FUNCTION: SubmitCmd
 *
 * PURPOSE:    This function allocated a new terminal id in the Term
structure array.
 *
 */

void SubmitCmd(EXTENSION_CONTROL_BLOCK *pECB, char *szBuffer)
{
        int             iNewTerm;
        char    *ptr = pECB->lpszQueryString;

        char    szVersion[32]  = { 0 };
        char    szServer[32]   = { 0 };
        char    szUser[32]             = "sa";
        char    szPassword[32] = { 0 };
        char    szDatabase[32] = "tpcc";

        // validate version field; the version field ensures that the RTE
is synchronized with the web client
        GetKeyValue(&ptr, "VERSION", szVersion, sizeof(szVersion),
ERR_VERSION_MISMATCH);
        if ( strcmp( szVersion, WEBCLIENT_VERSION ) )
                throw new CWEBCLNT_ERR( ERR_VERSION_MISMATCH );

        if (Reg.eTxnMon == None)
        {
                // parse Server name
                GetKeyValue(&ptr, "db_server", szServer, sizeof(szServer),
ERR_NO_SERVER_SPECIFIED);
                // parse User name
                GetKeyValue(&ptr, "db_user", szUser, sizeof(szUser),
NO_ERR);
                // parse Password
                GetKeyValue(&ptr, "db_passwd", szPassword,
sizeof(szPassword), NO_ERR);          // parse Database name
                // parse Database name

                GetKeyValue(&ptr, "db_name", szDatabase,
sizeof(szDatabase), NO_ERR);
        }

        // parse warehouse ID
        int w_id = GetIntKeyValue(&ptr, "w_id", ERR_HTML_ILL_FORMED,
ERR_W_ID_INVALID);
        if ( w_id < 1 )
                throw new CWEBCLNT_ERR( ERR_W_ID_INVALID );

        // parse district ID
        int d_id = GetIntKeyValue(&ptr, "d_id", ERR_HTML_ILL_FORMED,
ERR_D_ID_INVALID);
        if ( d_id < 1 || d_id > 10 )
                throw new CWEBCLNT_ERR( ERR_D_ID_INVALID );


        iNewTerm = TermAdd();

        Term.pClientData[iNewTerm].w_id = w_id;
        Term.pClientData[iNewTerm].d_id = d_id;

        try
        {
                if (Reg.eTxnMon == TUXEDO)
                        Term.pClientData[iNewTerm].pTxn =
pCTPCC_TUXEDO_new();
                else if (Reg.eTxnMon == ENCINA)
                        Term.pClientData[iNewTerm].pTxn =
pCTPCC_ENCINA_new();
                else if (Reg.eTxnMon == COM)
                        Term.pClientData[iNewTerm].pTxn = pCTPCC_COM_new(
Reg.bCOM_SinglePool );
                else if (Reg.eDB_Protocol == ODBC)
                        Term.pClientData[iNewTerm].pTxn = pCTPCC_ODBC_new(
szServer, szUser, szPassword, szMyComputerName, szDatabase );
                else if (Reg.eDB_Protocol == DBLIB)
                        Term.pClientData[iNewTerm].pTxn = pCTPCC_DBLIB_new(
szServer, szUser, szPassword, szMyComputerName, szDatabase );
        }
        catch (...)
        {
                TermDelete(iNewTerm);
                throw;          // pass exception upward
        }

        MakeMainMenuForm(iNewTerm, Term.pClientData[iNewTerm].iSyncId,
szBuffer);
}

/* FUNCTION: StatsCmd
 *
 * PURPOSE:    This function returns to the browser the total number of
active terminal ids.
 *                      This routine is for development/debugging purposes.
 *
 */
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
void StatsCmd(EXTENSION_CONTROL_BLOCK *pECB, char *szBuffer)
{
        int i;
        int     iTotal;

        EnterCriticalSection(&TermCriticalSection);

        iTotal = 0;
        for(i=0; i<Term.iNumEntries; i++)
        {
                if (Term.pClientData[i].iNextFree == -1)
                        iTotal++;
        }

        LeaveCriticalSection(&TermCriticalSection);

        wsprintf( szBuffer,
                        "<HTML><HEAD><TITLE>TPC-C Web Client
Stats</TITLE></HEAD>"
                        "<BODY><B><BIG> Total Active Connections: %d
</BIG></B><BR></BODY></HTML>"
                        , iTotal );
}

char *CWEBCLNT_ERR::ErrorText()
{
        static SERRORMSG errorMsgs[] =
        {
                {       ERR_COMMAND_UNDEFINED,
        "Command undefined."
                                        },
                {       ERR_D_ID_INVALID,
                "Invalid District ID Must be 1 to 10."
                                        },
                {       ERR_DELIVERY_CARRIER_ID_RANGE,
        "Delivery Carrier ID out of range must be 1 - 10."
                        },
                {       ERR_DELIVERY_CARRIER_INVALID,
        "Delivery Carrier ID invalid must be numeric 1 - 10."
                        },
                {       ERR_DELIVERY_MISSING_OCD_KEY,
        "Delivery missing Carrier ID key \"OCD*\"."
                                },
                {       ERR_DELIVERY_THREAD_FAILED,
        "Could not start delivery worker thread."
                                },
                {       ERR_GETPROCADDR_FAILED,
                "Could not map proc in DLL.  GetProcAddr error.  DLL="
                                },
                {       ERR_HTML_ILL_FORMED,
        "Required key field is missing from HTML string."
                        },
                {       ERR_INVALID_SYNC_CONNECTION,
        "Invalid Terminal Sync ID."
                                                },
                {       ERR_INVALID_TERMID,
                "Invalid Terminal ID."
                                                        },
                {       ERR_LOADDLL_FAILED,
                "Load of DLL failed.  DLL="
                                                        },
                {       ERR_MAX_CONNECTIONS_EXCEEDED,                   "No
connections available.  Max Connections is probably too low."   },
                {       ERR_MISSING_REGISTRY_ENTRIES,
        "Required registry entries are missing.  Rerun INSTALL to
correct."       },
                {       ERR_NEWORDER_CUSTOMER_INVALID,
        "New Order customer id invalid data type, range = 1 to 3000."
                },
                {       ERR_NEWORDER_CUSTOMER_KEY,
        "New Order missing Customer key \"CID*\"."
                                },
                {       ERR_NEWORDER_DISTRICT_INVALID,
        "New Order District ID Invalid range 1 - 10."
                                },
                {       ERR_NEWORDER_FORM_MISSING_DID,
        "New Order missing District key \"DID*\"."
                                },
                {       ERR_NEWORDER_ITEMID_INVALID,                    "New
Order Item Id is wrong data type, must be numeric."                     },
                {       ERR_NEWORDER_ITEMID_RANGE,
        "New Order Item Id is out of range. Range = 1 to 999999."
                },
                {       ERR_NEWORDER_ITEMID_WITHOUT_SUPPW,              "New
Order Item_Id field entered without a corresponding Supp_W."            },
                {       ERR_NEWORDER_MISSING_IID_KEY,                   "New
Order missing Item Id key \"IID*\"."
                },
                {       ERR_NEWORDER_MISSING_QTY_KEY,                   "New
Order Missing Qty key \"Qty##*\"."
                                },
                {       ERR_NEWORDER_MISSING_SUPPW_KEY,
        "New Order missing Supp_W key \"SP##*\"."
                                },
                {       ERR_NEWORDER_NOITEMS_ENTERED,                   "New
Order No order lines entered."
                                },
                {       ERR_NEWORDER_QTY_INVALID,
        "New Order Qty invalid must be numeric range 1 - 99."
                                },
                {       ERR_NEWORDER_QTY_RANGE,
                "New Order Qty is out of range. Range = 1 to 99."
                                },
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                {       ERR_NEWORDER_QTY_WITHOUT_SUPPW,
        "New Order Qty field entered without a corresponding Supp_W."
                },
                {       ERR_NEWORDER_SUPPW_INVALID,
        "New Order Supp_W invalid data type must be numeric."
                        },
                {       ERR_NO_SERVER_SPECIFIED,
        "No Server name specified."
                                        },
                {       ERR_ORDERSTATUS_CID_AND_CLT,                    "Order
Status Only Customer ID or Last Name may be entered, not both."     },
                {       ERR_ORDERSTATUS_CID_INVALID,                    "Order
Status Customer ID invalid, range must be numeric 1 - 3000."        },
                {       ERR_ORDERSTATUS_CLT_RANGE,
        "Order Status Customer last name longer than 16 characters."
                },
                {       ERR_ORDERSTATUS_DID_INVALID,                    "Order
Status District invalid, value must be numeric 1 - 10."             },
                {       ERR_ORDERSTATUS_MISSING_CID_CLT,                "Order
Status Either Customer ID or Last Name must be entered."            },
                {       ERR_ORDERSTATUS_MISSING_CID_KEY,                "Order
Status missing Customer key \"CID*\"."
                },
                {       ERR_ORDERSTATUS_MISSING_CLT_KEY,                "Order
Status missing Customer Last Name key \"CLT*\"."
        },
                {       ERR_ORDERSTATUS_MISSING_DID_KEY,                "Order
Status missing District key \"DID*\"."
                },
                {       ERR_PAYMENT_CDI_INVALID,
        "Payment Customer district invalid must be numeric."
                },
                {       ERR_PAYMENT_CID_AND_CLT,
        "Payment Only Customer ID or Last Name may be entered, not both."
        },
                {       ERR_PAYMENT_CUSTOMER_INVALID,
        "Payment Customer data type invalid, must be numeric."
                        },
                {       ERR_PAYMENT_CWI_INVALID,
        "Payment Customer Warehouse invalid, must be numeric."
                        },
                {       ERR_PAYMENT_DISTRICT_INVALID,
        "Payment District ID is invalid, must be 1 - 10."
                        },
                {       ERR_PAYMENT_HAM_INVALID,
        "Payment Amount invalid data type must be numeric."
                        },
                {       ERR_PAYMENT_HAM_RANGE,
        "Payment Amount out of range, 0 - 9999.99."
                                },
                {       ERR_PAYMENT_LAST_NAME_TO_LONG,
        "Payment Customer last name longer than 16 characters."
                        },
                {       ERR_PAYMENT_MISSING_CDI_KEY,
        "Payment missing Customer district key \"CDI*\"."
                        },
                {       ERR_PAYMENT_MISSING_CID_CLT,
        "Payment Either Customer ID or Last Name must be entered."
                },
                {       ERR_PAYMENT_MISSING_CID_KEY,
        "Payment missing Customer Key \"CID*\"."
                                },
                {       ERR_PAYMENT_MISSING_CLT_KEY,
        "Payment missing Customer Last Name key \"CLT*\"."
                        },
                {       ERR_PAYMENT_MISSING_CWI_KEY,
        "Payment missing Customer Warehouse key \"CWI*\"."
                        },
                {       ERR_PAYMENT_MISSING_DID_KEY,
        "Payment missing District Key \"DID*\"."
                                },
                {       ERR_PAYMENT_MISSING_HAM_KEY,
        "Payment missing Amount key \"HAM*\"."
                                        },
                {       ERR_STOCKLEVEL_MISSING_THRESHOLD_KEY, "Stock Level;
missing Threshold key \"TT*\"."
        },
                {       ERR_STOCKLEVEL_THRESHOLD_INVALID,               "Stock
Level; Threshold value must be in the range = 1 - 99."              },
                {       ERR_STOCKLEVEL_THRESHOLD_RANGE,
        "Stock Level Threshold out of range, range must be 1 - 99."
                },
                {       ERR_VERSION_MISMATCH,
        "Invalid version field.  RTE and Web Client are probably out of
sync." },
                {       ERR_W_ID_INVALID,
                "Invalid Warehouse ID."
                                                                },
                {       0,
                                ""


                        }
        };

        char szTmp[256];
        int i = 0;
        while (TRUE)
        {
                if (errorMsgs[i].szMsg[0] == 0)
                {
                        strcpy( szTmp, "Unknown error number." );
                        break;
                }
                if (m_Error == errorMsgs[i].iError)
                {
                        strcpy( szTmp, errorMsgs[i].szMsg );
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                        break;
                }
                i++;
        }

        if (m_szTextDetail)
                strcat( szTmp, m_szTextDetail );
        if (m_SystemErr)
                wsprintf( szTmp+strlen(szTmp), "  Error=%d", m_SystemErr
);

        m_szErrorText = new char[strlen(szTmp)+1];
        strcpy( m_szErrorText, szTmp );
        return m_szErrorText;

}

/* FUNCTION: GetKeyValue
 *
 * PURPOSE:    This function parses a http formatted string for specific
key values.
 *
 * ARGUMENTS:  char                  *pQueryString  http string from
client browser
 *                           char                  *pKey
        key value to look for
 *                           char                  *pValue
        character array into which to place key's value
 *                           int                             iMax
                maximum length of key value array.
 *                           WEBERROR              err
                error value to throw
 *
 * RETURNS:            nothing.
 *
 * ERROR:              if (the pKey value is not found) then
 *                                   if (err == 0)
 *                                           return (empty string)
 *                                   else
 *                                           throw CWEBCLNT_ERR(err)
 *
 * COMMENTS:   http keys are formatted either KEY=value& or KEY=value\0.
This DLL formats
 *                           TPC-C input fields in such a manner that
the keys can be extracted in the
 *                           above manner.
 */

void GetKeyValue(char **pQueryString, char *pKey, char *pValue, int iMax,
WEBERROR err)
{
        char *ptr;

        if ( !(ptr=strstr(*pQueryString, pKey)) )
                goto ErrorExit;
        ptr += strlen(pKey);
        if ( *ptr != '=' )
                goto ErrorExit;
        ptr++;

        iMax--;  // one position is for terminating null
        while( *ptr && *ptr != '&' && iMax)
        {
                *pValue++ = *ptr++;
                iMax--;
        }
        *pValue = 0;  // terminating null

        *pQueryString = ptr;
        return;

ErrorExit:
        if (err != NO_ERR)
                throw new CWEBCLNT_ERR( err );
        *pValue = 0;  // return empty result string
}

/* FUNCTION: GetIntKeyValue
 *
 * PURPOSE:    This function parses a http formatted string for a
specific key value.
 *
 * ARGUMENTS:  char                  *pQueryString  http string from
client browser
 *                           char                  *pKey
        key value to look for
 *                           WEBERROR              NoKeyErr
        error value to throw if key not found
 *                           WEBERROR              NotIntErr
        error value to throw if value not numeric
 *
 * RETURNS:            integer
 *
 * ERROR:              if (the pKey value is not found) then
 *                                   if (NoKeyErr != NO_ERR)
 *                                           throw CWEBCLNT_ERR(err)
 *                                   else
 *                                           return 0
 *                           else if (non-numeric char found) then
 *                                   if (NotIntErr != NO_ERR) then
 *                                           throw CWEBCLNT_ERR(err)
 *                                   else
 *                                           return 0
 *
 * COMMENTS:   http keys are formatted either KEY=value& or KEY=value\0.
This DLL formats
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
 *                              TPC-C input fields in such a manner that
the keys can be extracted in the
 *                              above manner.
 */

int GetIntKeyValue(char **pQueryString, char *pKey, WEBERROR NoKeyErr,
WEBERROR NotIntErr)
{
        char *ptr0;
        char *ptr;

        if ( !(ptr=strstr(*pQueryString, pKey)) )
                goto ErrorNoKey;
        ptr += strlen(pKey);
        if ( *ptr != '=' )
                goto ErrorNoKey;
        ptr++;

        ptr0 = ptr;             // remember starting point
        // scan string until a terminator (null or &) or a non-digit
        while( *ptr && *ptr != '&' && isdigit(*ptr) )
                ptr++;

        // make sure we stopped scanning for the right reason
        if ((ptr0 == ptr) || (*ptr && *ptr != '&'))
        {
                if (NotIntErr != NO_ERR)
                        throw new CWEBCLNT_ERR( NoKeyErr );
                return 0;
        }

        *pQueryString = ptr;
        return atoi(ptr0);

ErrorNoKey:
        if (NoKeyErr != NO_ERR)
                throw new CWEBCLNT_ERR( NoKeyErr );
        return 0;
}

/* FUNCTION: TermInit
 *
 * PURPOSE:    This function initializes the client terminal structure;
it is called when the TPCC.DLL
 *                      is first loaded by the inet service.
 *
 */

void TermInit(void)
{
        EnterCriticalSection(&TermCriticalSection);

        Term.iMasterSyncId     = 1;
        Term.iNumEntries       = Reg.dwMaxConnections+1;

        Term.pClientData       = NULL;
        Term.pClientData       = (PCLIENTDATA)malloc(Term.iNumEntries *
sizeof(CLIENTDATA));
        if (Term.pClientData == NULL)
        {
                LeaveCriticalSection(&TermCriticalSection);
                throw new CWEBCLNT_ERR( ERR_MEM_ALLOC_FAILED );
        }

        ZeroMemory( Term.pClientData, Term.iNumEntries *
sizeof(CLIENTDATA) );

        Term.iFreeList         = Term.iNumEntries-1;
        // build free list
        // note: Term.pClientData[0].iNextFree gets set to -1, which marks
it as "in use".
        //       This is intentional, as the zero entry is used as an
anchor and never
        //              allocated as an actual terminal.
        for(int i=0; i<Term.iNumEntries; i++)
                Term.pClientData[i].iNextFree = i-1;

        LeaveCriticalSection(&TermCriticalSection);
}

/* FUNCTION: TermDeleteAll
 *
 * PURPOSE:    This function frees allocated resources associated with
the terminal structure.
 *
 * ARGUMENTS:  none
 *
 * RETURNS:            None
 *
 * COMMENTS:   This function is called only when the inet service unloads
the TPCC.DLL
 *
 */

void TermDeleteAll(void)
{
        EnterCriticalSection(&TermCriticalSection);

        for(int i=1; i<Term.iNumEntries; i++)
        {
                if (Term.pClientData[i].iNextFree == -1)
                        delete Term.pClientData[i].pTxn;
        }

        Term.iFreeList         = 0;
        Term.iNumEntries       = 0;
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        if ( Term.pClientData )
                free(Term.pClientData);
        Term.pClientData       = NULL;

        LeaveCriticalSection(&TermCriticalSection);
}

/* FUNCTION: TermAdd
 *
 * PURPOSE:    This function assigns a terminal id which is used to
identify a client browser.
 *
 * RETURNS:            int             assigned terminal id
 *
 */

int TermAdd(void)
{
        DWORD   i;
        int             iNewTerm, iTickCount;

        if (Term.iNumEntries == 0)
                return -1;

        EnterCriticalSection(&TermCriticalSection);
        if (Term.iFreeList != 0)
        {       // position is available
                iNewTerm = Term.iFreeList;
                Term.iFreeList = Term.pClientData[iNewTerm].iNextFree;
                Term.pClientData[iNewTerm].iNextFree = -1;  // indicates
this position is in use
        }
        else
        {       // no open slots, so find the slot that hasn't been used
in the longest time and reuse it
                for(iNewTerm=1, i=1, iTickCount=0x7FFFFFFF;
i<Reg.dwMaxConnections; i++)
                {
                        if (iTickCount > Term.pClientData[i].iTickCount)
                        {
                                iTickCount =
Term.pClientData[i].iTickCount;
                                iNewTerm = i;
                        }
                }
                // if oldest term is less than one minute old, it probably
means that more connections
                // are being attempted than were specified as "Max
Connections" at install.  In this case,
                // do not bump existing connection; instead, return error
to requestor.
                if ((GetTickCount() - iTickCount) < 60000)
                {
                        LeaveCriticalSection(&TermCriticalSection);
                        throw new CWEBCLNT_ERR(
ERR_MAX_CONNECTIONS_EXCEEDED );
                }
        }

        Term.pClientData[iNewTerm].iTickCount = GetTickCount();
        Term.pClientData[iNewTerm].iSyncId = Term.iMasterSyncId++;
        Term.pClientData[iNewTerm].pTxn = NULL;

        LeaveCriticalSection(&TermCriticalSection);
        return iNewTerm;
}

/* FUNCTION: TermDelete
 *
 * PURPOSE:    This function makes a terminal entry in the Term array
available for reuse.
 *
 * ARGUMENTS:  int                                             id
        Terminal id of client exiting
 *
 */

void TermDelete(int id)
{
        if ( id > 0 && id < Term.iNumEntries )
        {
                delete Term.pClientData[id].pTxn;

                // put onto free list
                EnterCriticalSection(&TermCriticalSection);

                Term.pClientData[id].iNextFree = Term.iFreeList;
                Term.iFreeList = id;

                LeaveCriticalSection(&TermCriticalSection);
        }
}


/* FUNCTION: MakeErrorForm
 */

void ErrorForm(EXTENSION_CONTROL_BLOCK *pECB, int iType, int iErrorNum,
int iTermId, int iSyncId, char *szErrorText, char *szBuffer )
{
        wsprintf(szBuffer,
                "<HTML><HEAD><TITLE>TPC-C Error</TITLE></HEAD><BODY>"
                "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">"
                "<INPUT TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"%d\">"
                "<INPUT TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"%d\">"
                "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"%d\">"
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%d\">"
                "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">"
                "<BOLD>An Error Occurred</BOLD><BR><BR>"
                "%s"
                "<BR><BR><HR>"
                "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..NewOrder..\">"
                "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Payment..\">"
                "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Delivery..\">"
                "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Order-
Status..\">"
                "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Stock-
Level..\">"
                "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Exit..\">"
                "</FORM></BODY></HTML>"
                , iType, iErrorNum, MAIN_MENU_FORM, iTermId, iSyncId,
szErrorText );
}

/* FUNCTION: MakeMainMenuForm
 */

void MakeMainMenuForm(int iTermId, int iSyncId, char *szForm)
{
        wsprintf(szForm,
                "<HTML><HEAD><TITLE>TPC-C Main Menu</TITLE></HEAD><BODY>"
                "Select Desired Transaction.<BR><HR>"
                "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">"
                "<INPUT TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"0\">"
                "<INPUT TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"0\">"
                "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"%d\">"
                "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%d\">"
                "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">"
                "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..NewOrder..\">"
                "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Payment..\">"
                "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Delivery..\">"
                "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Order-
Status..\">"
                "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Stock-
Level..\">"
                "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Exit..\">"
                "</FORM></BODY></HTML>"
                , MAIN_MENU_FORM, iTermId, iSyncId);
}

/* FUNCTION: MakeStockLevelForm
 *
 * PURPOSE:    This function constructs the Stock Level HTML page.
 *
 * COMMENTS:   The internal client buffer is created when the terminal id
is assigned and should not
 *                              be freed except when the client terminal id
is no longer needed.
 */

void MakeStockLevelForm(int iTermId, STOCK_LEVEL_DATA *pStockLevelData,
BOOL bInput, char *szForm)
{
        int     c;

        c = wsprintf(szForm,
                "<HTML><HEAD><TITLE>TPC-C Stock Level</TITLE></HEAD><FORM
ACTION=\"tpcc.dll\" METHOD=\"GET\">"
                "<INPUT TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"0\">"
                "<INPUT TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"0\">"
                "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"%d\">"
                "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%d\">"
                "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">"
                "<PRE><font face=\"Courier\">
Stock-Level<BR>"
                "Warehouse: %4.4d   District: %2.2d<BR> <BR>",
                STOCK_LEVEL_FORM, iTermId,
Term.pClientData[iTermId].iSyncId,
                Term.pClientData[iTermId].w_id,
Term.pClientData[iTermId].d_id);

        if ( bInput )
        {
                strcpy(szForm+c,
                        "Stock Level Threshold: <INPUT NAME=\"TT*\"
SIZE=2><BR> <BR>"
                        "low stock:    </font><BR> <BR> <BR> <BR> <BR> <BR>
<BR> <BR> <BR> <BR>"
                        " <BR> <BR> <BR> <BR> <BR> <BR> <BR></PRE><HR>"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"Process\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"Menu\">"
                        "</FORM></HTML>" );
        }
        else
        {
                wsprintf(szForm+c,
                        "Stock Level Threshold: %2.2d<BR> <BR>"
                        "low stock: %3.3d</font> <BR> <BR> <BR> <BR> <BR>
<BR> <BR> <BR> <BR>"
                        " <BR> <BR> <BR> <BR> <BR> <BR> <BR>
<BR></PRE><HR>"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..NewOrder..\">"
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Payment..\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Delivery..\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Order-Status..\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Stock-Level..\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Exit..\">"
                        "</FORM></HTML>"
                        , pStockLevelData->threshold, pStockLevelData-
>low_stock);
        }
}

/* FUNCTION: MakeNewOrderForm
 *
 * COMMENTS:   The internal client buffer is created when the terminal id
is assigned and should not
 *                             be freed except when the client terminal id
is no longer needed.
 */

void MakeNewOrderForm(int iTermId, NEW_ORDER_DATA *pNewOrderData, BOOL
bInput, char *szForm)
{
        int             i, c;
        BOOL    bValid;
        static  char szBR[] = " <BR> <BR> <BR> <BR> <BR> <BR> <BR> <BR>
<BR> <BR> <BR> <BR> <BR> <BR>";

        if (!bInput)
                assert( pNewOrderData->exec_status_code == eOK ||
pNewOrderData->exec_status_code == eInvalidItem );

        bValid = (bInput || (pNewOrderData->exec_status_code == eOK));

        c = wsprintf(szForm,
                "<HTML><HEAD><TITLE>TPC-C New Order</TITLE></HEAD><BODY>"
                "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">"
                "<INPUT TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"%d\">"
                "<INPUT TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"0\">"
                "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"%d\">"
                "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%d\">"
                "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">"
                "<PRE><font face=\"Courier\">
New Order<BR>"
                , bValid ? 0 : ERR_BAD_ITEM_ID, NEW_ORDER_FORM, iTermId,
Term.pClientData[iTermId].iSyncId);

        if ( bInput )
        {
                c += wsprintf(szForm+c, "Warehouse: %4.4d   ",
Term.pClientData[iTermId].w_id );

                strcpy( szForm+c,
                        "District: <INPUT NAME=\"DID*\" SIZE=1>
Date:<BR>"
                        "Customer:  <INPUT NAME=\"CID*\" SIZE=4>   Name:
Credit:      %Disc:<BR>"
                        "Order Number:           Number of Lines:
W_tax:          D_tax:<BR> <BR>"
                        " Supp_W  Item_Id  Item Name                 Qty
Stock  B/G  Price    Amount<BR>"
                        "  <INPUT NAME=\"SP00*\" SIZE=4>  <INPUT
NAME=\"IID00*\" SIZE=6>                              <INPUT NAME=\"Qty00*\"
SIZE=1><BR>"
                        "  <INPUT NAME=\"SP01*\" SIZE=4>  <INPUT
NAME=\"IID01*\" SIZE=6>                              <INPUT NAME=\"Qty01*\"
SIZE=1><BR>"
                        "  <INPUT NAME=\"SP02*\" SIZE=4>  <INPUT
NAME=\"IID02*\" SIZE=6>                              <INPUT NAME=\"Qty02*\"
SIZE=1><BR>"
                        "  <INPUT NAME=\"SP03*\" SIZE=4>  <INPUT
NAME=\"IID03*\" SIZE=6>                              <INPUT NAME=\"Qty03*\"
SIZE=1><BR>"
                        "  <INPUT NAME=\"SP04*\" SIZE=4>  <INPUT
NAME=\"IID04*\" SIZE=6>                              <INPUT NAME=\"Qty04*\"
SIZE=1><BR>"
                        "  <INPUT NAME=\"SP05*\" SIZE=4>  <INPUT
NAME=\"IID05*\" SIZE=6>                              <INPUT NAME=\"Qty05*\"
SIZE=1><BR>"
                        "  <INPUT NAME=\"SP06*\" SIZE=4>  <INPUT
NAME=\"IID06*\" SIZE=6>                              <INPUT NAME=\"Qty06*\"
SIZE=1><BR>"
                        "  <INPUT NAME=\"SP07*\" SIZE=4>  <INPUT
NAME=\"IID07*\" SIZE=6>                              <INPUT NAME=\"Qty07*\"
SIZE=1><BR>"
                        "  <INPUT NAME=\"SP08*\" SIZE=4>  <INPUT
NAME=\"IID08*\" SIZE=6>                              <INPUT NAME=\"Qty08*\"
SIZE=1><BR>"
                        "  <INPUT NAME=\"SP09*\" SIZE=4>  <INPUT
NAME=\"IID09*\" SIZE=6>                              <INPUT NAME=\"Qty09*\"
SIZE=1><BR>"
                        "  <INPUT NAME=\"SP10*\" SIZE=4>  <INPUT
NAME=\"IID10*\" SIZE=6>                              <INPUT NAME=\"Qty10*\"
SIZE=1><BR>"
                        "  <INPUT NAME=\"SP11*\" SIZE=4>  <INPUT
NAME=\"IID11*\" SIZE=6>                              <INPUT NAME=\"Qty11*\"
SIZE=1><BR>"
                        "  <INPUT NAME=\"SP12*\" SIZE=4>  <INPUT
NAME=\"IID12*\" SIZE=6>                              <INPUT NAME=\"Qty12*\"
SIZE=1><BR>"
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                        "  <INPUT NAME=\"SP13*\" SIZE=4>  <INPUT
NAME=\"IID13*\" SIZE=6>                              <INPUT NAME=\"Qty13*\"
SIZE=1><BR>"
                        "  <INPUT NAME=\"SP14*\" SIZE=4>  <INPUT
NAME=\"IID14*\" SIZE=6>                              <INPUT NAME=\"Qty14*\"
SIZE=1><BR>"
                        "Execution Status:
Total:<BR>"
                        "</font></PRE><HR>"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"Process\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"Menu\">"
                        "</FORM></HTML>"
                        );
        }
        else
        {
                c += wsprintf(szFormat+c, "Warehouse: %4.4d   District:
%2.2d                        Date: ",
                        pNewOrderData->w_id,
                        pNewOrderData->d_id);

                if ( bValid )
                {
                        c += wsprintf(szForm+c, "%2.2d-%2.2d-%4.4d
%2.2d:%2.2d:%2.2d",
                                pNewOrderData->o_entry_d.day,
                                pNewOrderData->o_entry_d.month,
                                pNewOrderData->o_entry_d.year,
                                pNewOrderData->o_entry_d.hour,
                                pNewOrderData->o_entry_d.minute,
                                pNewOrderData->o_entry_d.second);
                }

                c += wsprintf(szForm+c, "<BR>Customer:  %4.4d   Name: %-
16s   Credit: %-2s   ",
                        pNewOrderData->c_id, pNewOrderData->c_last,
pNewOrderData->c_credit);

                if ( bValid )
                {
                        c += sprintf(szForm+c,
                                        "%%Disc: %5.2f
<BR>"
                                        "Order Number: %8.8d  Number
of Lines: %2.2d        W_tax: %5.2f   D_tax: %5.2f  <BR> <BR>"
                                        " Supp_W  Item_Id  Item Name
Qty  Stock  B/G  Price    Amount<BR>",
                                100.0*pNewOrderData->c_discount,
                                pNewOrderData->o_id,
                                pNewOrderData->o_ol_cnt,
                                100.0 * pNewOrderData->w_tax,
                                100.0 * pNewOrderData->d_tax);

                        for(i=0; i<pNewOrderData->o_ol_cnt; i++)
                        {
                                c += sprintf(szForm+c, "  %4.4d   %6.6d
%-24s  %2.2d    %3.3d    %1.1s   $%6.2f  $%7.2f  <BR>",
                                        pNewOrderData->OL[i].ol_supply_w_id,
                                        pNewOrderData->OL[i].ol_i_id,
                                        pNewOrderData->OL[i].ol_i_name,
                                        pNewOrderData->OL[i].ol_quantity,
                                        pNewOrderData->OL[i].ol_stock,
                                        pNewOrderData-
>OL[i].ol_brand_generic,
                                        pNewOrderData->OL[i].ol_i_price,
                                        pNewOrderData->OL[i].ol_amount );
                        }
                }
                else
                {
                        c += wsprintf(szForm+c,
                                "%Disc:<BR>"
                                "Order Number: %8.8d  Number of Lines:
W_tax:         D_tax:<BR> <BR>"
                                " Supp_W  Item_Id  Item Name
Qty  Stock  B/G  Price    Amount<BR>"
                                , pNewOrderData->o_id);

                        i = 0;
                }

                strncpy( szForm+c, szBR, (15-i)*5 );
                c += (15-i)*5;

                if ( bValid )
                        c += sprintf(szForm+c, "Execution Status:
Transaction commited.                        Total:  $%8.2f  ",
                                pNewOrderData->total_amount);
                else
                        c += wsprintf(szForm+c, "Execution Status: Item
number is not valid.                  Total:");

                strcpy(szForm+c,
                        " <BR></font></PRE><HR>"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..NewOrder..\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Payment..\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Delivery..\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Order-Status..\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Stock-Level..\">"
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Exit..\">"
                        "</FORM></HTML>"
                        );
        }
}

/* FUNCTION: MakePaymentForm
 *
 * COMMENTS:   The internal client buffer is created when the terminal id
is assigned and should not
 *                           be freed except when the client terminal id
is no longer needed.
 */

void MakePaymentForm(int iTermId, PAYMENT_DATA *pPaymentData, BOOL
bInput, char *szForm)
{
        int c;

        c = wsprintf(szForm,
                "<HTML><HEAD><TITLE>TPC-C Payment</TITLE></HEAD><BODY>"
                "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">"
                "<INPUT TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"0\">"
                "<INPUT TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"0\">"
                "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"%d\">"
                "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%d\">"
                "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">"
                "<PRE><font face=\"Courier\">
Payment<BR>"
                "Date: "
                , PAYMENT_FORM, iTermId,
Term.pClientData[iTermId].iSyncId);

        if ( !bInput )
        {
                c += wsprintf(szForm+c, "%2.2d-%2.2d-%4.4d
%2.2d:%2.2d:%2.2d",
                        pPaymentData->h_date.day,
                        pPaymentData->h_date.month,
                        pPaymentData->h_date.year,
                        pPaymentData->h_date.hour,
                        pPaymentData->h_date.minute,
                        pPaymentData->h_date.second);
        }

        if ( bInput )
        {
                c += wsprintf(szForm+c,
                        "<BR> <BR>Warehouse: %4.4d"
                        "                          District: <INPUT
NAME=\"DID*\" SIZE=1><BR> <BR> <BR> <BR> <BR>"
                        "Customer: <INPUT NAME=\"CID*\" SIZE=4>"
                        "Cust-Warehouse: <INPUT NAME=\"CWI*\" SIZE=4>  "
                        "Cust-District: <INPUT NAME=\"CDI*\" SIZE=1><BR>"
                        "Name:                    <INPUT NAME=\"CLT*\"
SIZE=16>                      Since:<BR>"
                        "
Credit:<BR>"
                        "
Disc:<BR>"
                        "
Phone:<BR> <BR>"
                        "Amount Paid:          $<INPUT NAME=\"HAM*\"
SIZE=7>      New Cust-Balance:<BR>"
                        "Credit Limit:<BR> <BR>Cust-Data: <BR> <BR> <BR>
<BR> <BR></font></PRE><HR>"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"Process\"><INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"Menu\">"
                        "</BODY></FORM></HTML>"
                        , Term.pClientData[iTermId].w_id);
        }
        else
        {
                c += wsprintf(szForm+c,
                        "<BR> <BR>Warehouse: %4.4d
District: %2.2d<BR>"
                        "%-20s                    %-20s<BR>"
                        "%-20s                    %-20s<BR>"
                        "%-20s %-2s %5.5s-%4.4s       %-20s %-2s %5.5s-
%4.4s<BR> <BR>"
                        "Customer: %4.4d  Cust-Warehouse: %4.4d  Cust-
District: %2.2d<BR>"
                        "Name:   %-16s %-2s %-16s     Since:  %2.2d-%2.2d-
%4.4d<BR>"
                        "        %-20s                    Credit: %-
2s<BR>"

                        , Term.pClientData[iTermId].w_id, pPaymentData-
>d_id
                        , pPaymentData->w_street_1, pPaymentData-
>d_street_1
                        , pPaymentData->w_street_2, pPaymentData-
>d_street_2
                        , pPaymentData->w_city, pPaymentData->w_state,
pPaymentData->w_zip, pPaymentData->w_zip+5
                        , pPaymentData->d_city, pPaymentData->d_state,
pPaymentData->d_zip, pPaymentData->d_zip+5
                        , pPaymentData->c_id, pPaymentData->c_w_id,
        pPaymentData->c_d_id
                        , pPaymentData->c_first, pPaymentData->c_middle,
pPaymentData->c_last
                        , pPaymentData->c_since.day, pPaymentData-
>c_since.month,        pPaymentData->c_since.year
                        , pPaymentData->c_street_1, pPaymentData->c_credit
                        );
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        c += sprintf(szForm+c,
                        "        %-20s                        %%Disc:
%5.2f<BR>",
                        pPaymentData->c_street_2, 100.0*pPaymentData-
>c_discount);

                c += wsprintf(szForm+c,
                        "        %-20s %-2s %5.5s-%4.4s       Phone:
%6.6s-%3.3s-%3.3s-%4.4s<BR> <BR>",
                        pPaymentData->c_city, pPaymentData->c_state,
pPaymentData->c_zip, pPaymentData->c_zip+5,
                        pPaymentData->c_phone, pPaymentData->c_phone+6,
pPaymentData->c_phone+9, pPaymentData->c_phone+12 );

                c += sprintf(szForm+c,
                        "Amount Paid:          $%7.2f      New Cust-
Balance: $%14.2f<BR>"
                        "Credit Limit:   $%13.2f<BR> <BR>"
                        , pPaymentData->h_amount, pPaymentData->c_balance
                        , pPaymentData->c_credit_lim
                        );

                if ( pPaymentData->c_credit[0] == 'B' && pPaymentData-
>c_credit[1] == 'C' )
                        c += wsprintf(szForm+c,
                                        "Cust-Data: %-50.50s<BR>
%-50.50s<BR>           %-50.50s<BR>           %-50.50s<BR>",
                                        pPaymentData->c_data, pPaymentData-
>c_data+50, pPaymentData->c_data+100, pPaymentData->c_data+150 );
                else
                        strcpy(szForm+c, "Cust-Data: <BR> <BR> <BR> <BR>");

                strcat(szForm, " <BR></font></PRE><HR>"
                                        "<INPUT TYPE=\"submit\"
NAME=\"CMD\" VALUE=\"..NewOrder..\">"
                                        "<INPUT TYPE=\"submit\"
NAME=\"CMD\" VALUE=\"..Payment..\">"
                                        "<INPUT TYPE=\"submit\"
NAME=\"CMD\" VALUE=\"..Delivery..\">"
                                        "<INPUT TYPE=\"submit\"
NAME=\"CMD\" VALUE=\"..Order-Status..\">"
                                        "<INPUT TYPE=\"submit\"
NAME=\"CMD\" VALUE=\"..Stock-Level..\">"
                                        "<INPUT TYPE=\"submit\"
NAME=\"CMD\" VALUE=\"..Exit..\">"
                                        "</BODY></FORM></HTML>");
        }

}

/* FUNCTION: MakeOrderStatusForm
 *
 * COMMENTS:   The internal client buffer is created when the terminal id
is assigned and should not
 *                              be freed except when the client terminal id
is no longer needed.
 */

void MakeOrderStatusForm(int iTermId, ORDER_STATUS_DATA
*pOrderStatusData, BOOL bInput, char *szForm)
{
        int             i, c;
        static char szBR[] = " <BR> <BR> <BR> <BR> <BR> <BR> <BR> <BR>
<BR> <BR> <BR> <BR> <BR> <BR> <BR>";

        c = wsprintf(szForm,
                "<HTML><HEAD><TITLE>TPC-C Order-
Status</TITLE></HEAD><BODY>"
                "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">"
                "<INPUT TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"0\">"
                "<INPUT TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"0\">"
                "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"%d\">"
                "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%d\">"
                "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">"
                "<PRE><font face=\"Courier\">
Order-Status<BR>"
                "Warehouse: %4.4d   ",
                ORDER_STATUS_FORM, iTermId,
Term.pClientData[iTermId].iSyncId, Term.pClientData[iTermId].w_id);

        if ( bInput )
        {
                strcpy(szForm+c,
                        "District: <INPUT NAME=\"DID*\" SIZE=1><BR>"
                        "Customer: <INPUT NAME=\"CID*\" SIZE=4>   Name:
<INPUT NAME=\"CLT*\" SIZE=23><BR>"
                        "Cust-Balance:<BR> <BR>"
                        "Order-Number:            Entry-Date:
Carrier-Number:<BR>"
                        "Supply-W     Item-Id    Qty     Amount
Delivery-Date<BR> <BR> <BR> <BR>"
                        " <BR> <BR> <BR> <BR> <BR> <BR> <BR> <BR> <BR> <BR>
<BR></font></PRE>"
                        "<HR><INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"Process\"><INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"Menu\">"
                        "</BODY></FORM></HTML>" );
        }
        else
        {
                c += wsprintf(szForm+c,
                        "District: %2.2d<BR>"
                        "Customer: %4.4d   Name: %-16s %-2s %-16s<BR>",
                        pOrderStatusData->d_id, pOrderStatusData->c_id,
                        pOrderStatusData->c_first, pOrderStatusData-
>c_middle, pOrderStatusData->c_last);
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                c += sprintf(szForm+c, "Cust-Balance: $%9.2f<BR> <BR>",
                        pOrderStatusData->c_balance);

                c += wsprintf(szForm+c,
                        "Order-Number: %8.8d   Entry-Date: %2.2d-%2.2d-
%4.4d %2.2d:%2.2d:%2.2d   Carrier-Number: %2.2d<BR>"
                        "Supply-W     Item-Id    Qty     Amount
Delivery-Date<BR>",
                        pOrderStatusData->o_id,
                        pOrderStatusData->o_entry_d.day,
                        pOrderStatusData->o_entry_d.month,
                        pOrderStatusData->o_entry_d.year,
                        pOrderStatusData->o_entry_d.hour,
                        pOrderStatusData->o_entry_d.minute,
                        pOrderStatusData->o_entry_d.second,
                        pOrderStatusData->o_carrier_id);

                for(i=0; i< pOrderStatusData->o_ol_cnt; i++)
                {
                        c += sprintf(szForm+c, "  %4.4d       %6.6d
%2.2d     $%8.2f      %2.2d-%2.2d-%4.4d<BR>",
                                pOrderStatusData->OL[i].ol_supply_w_id,
                                pOrderStatusData->OL[i].ol_i_id,
                                pOrderStatusData->OL[i].ol_quantity,
                                pOrderStatusData->OL[i].ol_amount,
                                pOrderStatusData->OL[i].ol_delivery_d.day,
                                pOrderStatusData-
>OL[i].ol_delivery_d.month,
                                pOrderStatusData-
>OL[i].ol_delivery_d.year);
                }

                strncpy( szForm+c, szBR, (15-i)*5 );
                c += (15-i)*5;

                strcpy(szForm+c,
                        "</font></PRE><HR><INPUT TYPE=\"submit\"
NAME=\"CMD\" VALUE=\"..NewOrder..\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Payment..\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Delivery..\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Order-Status..\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Stock-Level..\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Exit..\">"
                        "</BODY></FORM></HTML>" );
        }
}
```

```
/* FUNCTION: MakeDeliveryForm
 *
 * COMMENTS:   The internal client buffer is created when the terminal id
is assigned and should not
 *                              be freed except when the client terminal id
is no longer needed.
 */

void MakeDeliveryForm(int iTermId, DELIVERY_DATA *pDeliveryData, BOOL
bInput, char *szForm)
{
        int     c;

        c = wsprintf(szForm,
                "<HTML><HEAD><TITLE>TPC-C Delivery</TITLE></HEAD><BODY>"
                "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">"
                "<INPUT TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"%d\">"
                "<INPUT TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"0\">"
                "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"%d\">"
                "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%d\">"
                "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%d\">"
                "<PRE><font face=\"Courier\">
Delivery<BR>"
                "Warehouse: %4.4d<BR> <BR>",
                (!bInput && (pDeliveryData->exec_status_code != eOK)) ?
ERR_TYPE_DELIVERY_POST : 0,
                DELIVERY_FORM, iTermId, Term.pClientData[iTermId].iSyncId,
Term.pClientData[iTermId].w_id);

        if ( bInput )
        {
                strcpy( szForm+c,
                        "Carrier Number: <INPUT NAME=\"OCD*\" SIZE=1><BR>
<BR>"
                        "Execution Status: <BR> <BR> <BR> <BR> <BR> <BR>
<BR> <BR>"
                        " <BR> <BR> <BR> <BR> <BR> <BR> <BR> <BR>
</font></PRE><HR>"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"Process\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"Menu\">"
                        "</BODY></FORM></HTML>" );
        }
        else
        {
                wsprintf( szForm+c,
                        "Carrier Number: %2.2d<BR> <BR>"
                        "Execution Status: %s <BR> <BR> <BR> <BR> <BR> <BR>
<BR> <BR>"
                        " <BR> <BR> <BR> <BR> <BR> <BR> <BR> <BR>
</font></PRE>"
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                        "<HR><INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..NewOrder..\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Payment..\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Delivery..\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Order-Status..\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Stock-Level..\">"
                        "<INPUT TYPE=\"submit\" NAME=\"CMD\"
VALUE=\"..Exit..\">"
                        "</BODY></FORM></HTML>"

                        , pDeliveryData->o_carrier_id,
                        (pDeliveryData->exec_status_code == eOK) ?
"Delivery has been queued." : "Delivery Post Failed     "
                        );
        }
}

/* FUNCTION: ProcessNewOrderForm
 *
 * PURPOSE:    This function gets and validates the input data from the
new order form
 *                      filling in the required input variables. it then
calls the SQLNewOrder
 *                      transaction, constructs the output form and writes
it back to client
 *                      browser.
 */

void ProcessNewOrderForm(EXTENSION_CONTROL_BLOCK     *pECB, int iTermId,
char *szBuffer)
{
        PNEW_ORDER_DATA               pNewOrder;

        pNewOrder = Term.pClientData[iTermId].pTxn->BuffAddr_NewOrder();

        ZeroMemory(pNewOrder, sizeof(NEW_ORDER_DATA));
        pNewOrder->w_id = Term.pClientData[iTermId].w_id;
        GetNewOrderData(pECB->lpszQueryString, pNewOrder);

        Term.pClientData[iTermId].pTxn->NewOrder();

        pNewOrder = Term.pClientData[iTermId].pTxn->BuffAddr_NewOrder();
        MakeNewOrderForm(iTermId, pNewOrder, OUTPUT_FORM, szBuffer );
}

/* FUNCTION: void ProcessPaymentForm
 *
 * PURPOSE:    This function gets and validates the input data from the
payment form
 *                      filling in the required input variables. It then
calls the SQLPayment
 *                      transaction, constructs the output form and writes
it back to client
 *                      browser.
 *
 * ARGUMENTS:  EXTENSION_CONTROL_BLOCK       *pECB   passed in structure
pointer from inetsrv.
 *                              int
        iTermId client browser terminal id
 *
 */

void ProcessPaymentForm(EXTENSION_CONTROL_BLOCK *pECB, int iTermId, char
*szBuffer)
{
        PPAYMENT_DATA  pPayment;

        pPayment = Term.pClientData[iTermId].pTxn->BuffAddr_Payment();
        ZeroMemory(pPayment, sizeof(PAYMENT_DATA));
        pPayment->w_id = Term.pClientData[iTermId].w_id;
        GetPaymentData(pECB->lpszQueryString, pPayment);

        Term.pClientData[iTermId].pTxn->Payment();

        pPayment = Term.pClientData[iTermId].pTxn->BuffAddr_Payment();
        MakePaymentForm(iTermId, pPayment, OUTPUT_FORM, szBuffer);
}

/* FUNCTION: ProcessOrderStatusForm
 *
 * PURPOSE:    This function gets and validates the input data from the
Order Status
 *                      form filling in the required input variables. It
then calls the
 *                      SQLOrderStatus transaction, constructs the output
form and writes it
 *                      back to client browser.
 *
 * ARGUMENTS:  EXTENSION_CONTROL_BLOCK       *pECB   passed in structure
pointer from inetsrv.
 *                              int
        iTermId client browser terminal id
 *
 */

void ProcessOrderStatusForm(EXTENSION_CONTROL_BLOCK *pECB, int iTermId,
char *szBuffer)
{
        PORDER_STATUS_DATA     pOrderStatus;

        pOrderStatus = Term.pClientData[iTermId].pTxn-
>BuffAddr_OrderStatus();
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
	ZeroMemory(pOrderStatus, sizeof(ORDER_STATUS_DATA));
	pOrderStatus->w_id = Term.pClientData[iTermId].w_id;
	GetOrderStatusData(pECB->lpszQueryString, pOrderStatus);

	Term.pClientData[iTermId].pTxn->OrderStatus();

	pOrderStatus = Term.pClientData[iTermId].pTxn-
>BuffAddr_OrderStatus();
	MakeOrderStatusForm(iTermId, pOrderStatus, OUTPUT_FORM, szBuffer);
}

/* FUNCTION: ProcessDeliveryForm
 *
 * PURPOSE:    This function gets and validates the input data from the
delivery form
 *                     filling in the required input variables. It then
calls the PostDeliveryInfo
 *                     Api, The client is then informed that the
transaction has been posted.
 *
 * ARGUMENTS:  EXTENSION_CONTROL_BLOCK       *pECB   passed in structure
pointer from inetsrv.
 *                             int
	iTermId client browser terminal id
 *
 */

void ProcessDeliveryForm(EXTENSION_CONTROL_BLOCK *pECB, int iTermId, char
*szBuffer)
{
	char    *ptr = pECB->lpszQueryString;

	PDELIVERY_DATA pDelivery;

	pDelivery = Term.pClientData[iTermId].pTxn->BuffAddr_Delivery();
	ZeroMemory(pDelivery, sizeof(DELIVERY_DATA));
	pDelivery->w_id = Term.pClientData[iTermId].w_id;

	pDelivery->o_carrier_id       = GetIntKeyValue(&ptr, "OCD*",
ERR_DELIVERY_MISSING_OCD_KEY, ERR_DELIVERY_CARRIER_INVALID);
	if ( pDelivery->o_carrier_id > 10 || pDelivery->o_carrier_id < 1 )
		throw new CWEBCLNT_ERR( ERR_DELIVERY_CARRIER_ID_RANGE );

	if (dwNumDeliveryThreads)
	{
		//post delivery info
		if ( PostDeliveryInfo(pDelivery->w_id, pDelivery-
>o_carrier_id) )
			pDelivery->exec_status_code = eDeliveryFailed;
		else
			pDelivery->exec_status_code = eOK;
	}
	else // delivery is done synchronously if no delivery threads
configured
		Term.pClientData[iTermId].pTxn->Delivery();

	pDelivery = Term.pClientData[iTermId].pTxn->BuffAddr_Delivery();
	MakeDeliveryForm(iTermId, pDelivery, OUTPUT_FORM, szBuffer);
}

/* FUNCTION: ProcessStockLevelForm
 *
 * PURPOSE:    This function gets and validates the input data from the
Stock Level
 *                     form filling in the required input variables. It
then calls the
 *                     SQLStockLevel transaction, constructs the output
form and writes it
 *                     back to client browser.
 *
 * ARGUMENTS:  EXTENSION_CONTROL_BLOCK       *pECB   passed in structure
pointer from inetsrv.
 *                             int
	iTermId client browser terminal id
 *
 */

void ProcessStockLevelForm(EXTENSION_CONTROL_BLOCK *pECB, int iTermId,
char *szBuffer)
{
	char           *ptr = pECB->lpszQueryString;

	PSTOCK_LEVEL_DATA     pStockLevel;

	pStockLevel = Term.pClientData[iTermId].pTxn-
>BuffAddr_StockLevel();
	ZeroMemory( pStockLevel, sizeof(STOCK_LEVEL_DATA) );

	pStockLevel->w_id = Term.pClientData[iTermId].w_id;
	pStockLevel->d_id = Term.pClientData[iTermId].d_id;

	pStockLevel->threshold = GetIntKeyValue(&ptr, "TT*",
ERR_STOCKLEVEL_MISSING_THRESHOLD_KEY, ERR_STOCKLEVEL_THRESHOLD_INVALID);
	if ( pStockLevel->threshold >= 100 || pStockLevel->threshold < 0 )
		throw new CWEBCLNT_ERR( ERR_STOCKLEVEL_THRESHOLD_RANGE );

	Term.pClientData[iTermId].pTxn->StockLevel();

	pStockLevel = Term.pClientData[iTermId].pTxn-
>BuffAddr_StockLevel();
	MakeStockLevelForm(iTermId, pStockLevel, OUTPUT_FORM, szBuffer);
}

/* FUNCTION: GetNewOrderData
 *
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
 * PURPOSE:    This function extracts and validates the new order form
data from an http command string.
 *
 * ARGUMENTS:  LPSTR                  lpszQueryString               client
browser http command string
 *                           NEW_ORDER_DATA *pNewOrderData
       pointer to new order data structure
 *
 */

void GetNewOrderData(LPSTR lpszQueryString, NEW_ORDER_DATA
*pNewOrderData)
{
       char    szTmp[26];
       int             i;
       short   items;
       int             ol_i_id, ol_quantity;
       char    *ptr = lpszQueryString;

       static char szSP[MAX_OL_NEW_ORDER_ITEMS][6] =
              { "SP00*", "SP01*", "SP02*", "SP03*", "SP04*",
                "SP05*", "SP06*", "SP07*", "SP08*", "SP09*",
                "SP10*", "SP11*", "SP12*", "SP13*", "SP14*" };
       static char szIID[MAX_OL_NEW_ORDER_ITEMS][7] =
              { "IID00*", "IID01*", "IID02*", "IID03*", "IID04*",
                "IID05*", "IID06*", "IID07*", "IID08*", "IID09*",
                "IID10*", "IID11*", "IID12*", "IID13*", "IID14*" };
       static char szQty[MAX_OL_NEW_ORDER_ITEMS][7] =
              { "Qty00*", "Qty01*", "Qty02*", "Qty03*", "Qty04*",
                "Qty05*", "Qty06*", "Qty07*", "Qty08*", "Qty09*",
                "Qty10*", "Qty11*", "Qty12*", "Qty13*", "Qty14*" };

       pNewOrderData->d_id = GetIntKeyValue(&ptr, "DID*",
ERR_NEWORDER_FORM_MISSING_DID, ERR_NEWORDER_DISTRICT_INVALID);
       pNewOrderData->c_id = GetIntKeyValue(&ptr, "CID*",
ERR_NEWORDER_CUSTOMER_KEY, ERR_NEWORDER_CUSTOMER_INVALID);

       for(i=0, items=0; i<MAX_OL_NEW_ORDER_ITEMS; i++)
       {
              GetKeyValue(&ptr, szSP[i], szTmp, sizeof(szTmp),
ERR_NEWORDER_MISSING_SUPPW_KEY);
              if ( szTmp[0] )
              {
                     if ( !IsNumeric(szTmp) )
                            throw new CWEBCLNT_ERR(
ERR_NEWORDER_SUPPW_INVALID );
                     pNewOrderData->OL[items].ol_supply_w_id =
(short)atoi(szTmp);

                     ol_i_id = pNewOrderData->OL[items].ol_i_id =
                            GetIntKeyValue(&ptr, szIID[i],
ERR_NEWORDER_MISSING_IID_KEY, ERR_NEWORDER_ITEMID_INVALID);
                     if ( ol_i_id > 999999 || ol_i_id < 1 )
                            throw new CWEBCLNT_ERR(
ERR_NEWORDER_ITEMID_RANGE );

                     ol_quantity = pNewOrderData->OL[items].ol_quantity
=
                            GetIntKeyValue(&ptr, szQty[i],
ERR_NEWORDER_MISSING_QTY_KEY, ERR_NEWORDER_QTY_INVALID);
                     if ( ol_quantity > 99 || ol_quantity < 1 )
                            throw new CWEBCLNT_ERR(
ERR_NEWORDER_QTY_RANGE );

                     items++;
              }
              else
              {      // nothing entered for supply warehouse, so item id
and qty must also be blank
                     GetKeyValue(&ptr, szIID[i], szTmp, sizeof(szTmp),
ERR_NEWORDER_MISSING_IID_KEY);
                     if ( szTmp[0] )
                            throw new CWEBCLNT_ERR(
ERR_NEWORDER_ITEMID_WITHOUT_SUPPW );

                     GetKeyValue(&ptr, szQty[i], szTmp, sizeof(szTmp),
ERR_NEWORDER_MISSING_QTY_KEY);
                     if ( szTmp[0] )
                            throw new CWEBCLNT_ERR(
ERR_NEWORDER_QTY_WITHOUT_SUPPW );
              }
       }
       if ( items == 0 )
              throw new CWEBCLNT_ERR( ERR_NEWORDER_NOITEMS_ENTERED );

       pNewOrderData->o_ol_cnt = items;
}

/* FUNCTION: GetPaymentData
 *
 * PURPOSE:    This function extracts and validates the payment form data
from an http command string.
 *
 * ARGUMENTS:  LPSTR                  lpszQueryString               client
browser http command string
 *                           PAYMENT_DATA   *pPaymentData
       pointer to payment data structure
 */

void GetPaymentData(LPSTR lpszQueryString, PAYMENT_DATA *pPaymentData)
{
       char    szTmp[26];
       char    *ptr = lpszQueryString;
       BOOL    bCustIdBlank;
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
	pPaymentData->d_id = GetIntKeyValue(&ptr, "DID*",
ERR_PAYMENT_MISSING_DID_KEY, ERR_PAYMENT_DISTRICT_INVALID);

	GetKeyValue(&ptr, "CID*", szTmp, sizeof(szTmp),
ERR_PAYMENT_MISSING_CID_KEY);
	if ( szTmp[0] == 0 )
	{
		bCustIdBlank = TRUE;
		pPaymentData->c_id = 0;
	}
	else
	{	// parse customer id and verify that last name was NOT
entered
		bCustIdBlank = FALSE;
		if ( !IsNumeric(szTmp) )
			throw new CWEBCLNT_ERR(
ERR_PAYMENT_CUSTOMER_INVALID );
		pPaymentData->c_id = atoi(szTmp);
	}

	pPaymentData->c_w_id = GetIntKeyValue(&ptr, "CWI*",
ERR_PAYMENT_MISSING_CWI_KEY, ERR_PAYMENT_CWI_INVALID);
	pPaymentData->c_d_id = GetIntKeyValue(&ptr, "CDI*",
ERR_PAYMENT_MISSING_CDI_KEY, ERR_PAYMENT_CDI_INVALID);

	if ( bCustIdBlank )
	{	// customer id is blank, so last name must be entered
		GetKeyValue(&ptr, "CLT*", szTmp, sizeof(szTmp),
ERR_PAYMENT_MISSING_CLT_KEY);
		if ( szTmp[0] == 0 )
			throw new CWEBCLNT_ERR( ERR_PAYMENT_MISSING_CID_CLT
);

		_strupr( szTmp );
		if ( strlen(pPaymentData->c_last) > LAST_NAME_LEN )
			throw new CWEBCLNT_ERR(
ERR_PAYMENT_LAST_NAME_TO_LONG );
		strcpy(pPaymentData->c_last, szTmp);
	}
	else
	{	// parse customer id and verify that last name was NOT
entered
		GetKeyValue(&ptr, "CLT*", szTmp, sizeof(szTmp),
ERR_PAYMENT_MISSING_CLT_KEY);
		if ( szTmp[0] != 0 )
			throw new CWEBCLNT_ERR( ERR_PAYMENT_CID_AND_CLT );
	}

	GetKeyValue(&ptr, "HAM*", szTmp, sizeof(szTmp),
ERR_PAYMENT_MISSING_HAM_KEY);
	if (!IsDecimal(szTmp))
		throw new CWEBCLNT_ERR( ERR_PAYMENT_HAM_INVALID );
	pPaymentData->h_amount = atof(szTmp);
	if ( pPaymentData->h_amount >= 10000.00 || pPaymentData->h_amount
< 0 )
		throw new CWEBCLNT_ERR( ERR_PAYMENT_HAM_RANGE );
}

/* FUNCTION: GetOrderStatusData
 *
 * PURPOSE:    This function extracts and validates the payment form data
from an http command string.
 *
 */
void GetOrderStatusData(LPSTR lpszQueryString, ORDER_STATUS_DATA
*pOrderStatusData)
{
	char	szTmp[26];
	char	*ptr = lpszQueryString;

	pOrderStatusData->d_id = GetIntKeyValue(&ptr, "DID*",
ERR_ORDERSTATUS_MISSING_DID_KEY, ERR_ORDERSTATUS_DID_INVALID);

	GetKeyValue(&ptr, "CID*", szTmp, sizeof(szTmp),
ERR_ORDERSTATUS_MISSING_CID_KEY);
	if ( szTmp[0] == 0 )
	{	// customer id is blank, so last name must be entered
		pOrderStatusData->c_id = 0;
		GetKeyValue(&ptr, "CLT*", szTmp, sizeof(szTmp),
ERR_ORDERSTATUS_MISSING_CLT_KEY);
		if ( szTmp[0] == 0 )
			throw new CWEBCLNT_ERR(
ERR_ORDERSTATUS_MISSING_CID_CLT );

		_strupr( szTmp );
		if ( strlen(pOrderStatusData->c_last) > LAST_NAME_LEN )
			throw new CWEBCLNT_ERR( ERR_ORDERSTATUS_CLT_RANGE
);
		strcpy(pOrderStatusData->c_last, szTmp);
	}
	else
	{	// parse customer id and verify that last name was NOT
entered
		if ( !IsNumeric(szTmp) )
			throw new CWEBCLNT_ERR( ERR_ORDERSTATUS_CID_INVALID
);
		pOrderStatusData->c_id = atoi(szTmp);
		GetKeyValue(&ptr, "CLT*", szTmp, sizeof(szTmp),
ERR_ORDERSTATUS_MISSING_CLT_KEY);
		if ( szTmp[0] != 0 )
			throw new CWEBCLNT_ERR( ERR_ORDERSTATUS_CID_AND_CLT
);
	}

}
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
/* FUNCTION: BOOL IsNumeric(char *ptr)
 *
 * PURPOSE:    This function determines if a string is numeric. It fails
if any characters other
 *                     than numeric and null terminator are present.
 *
 * ARGUMENTS:  char                  *ptr    pointer to string to check.
 *
 * RETURNS:            BOOL    FALSE   if string is not all numeric
 *                                       TRUE    if string contains
only numeric characters i.e. '0' - '9'
 */

BOOL IsNumeric(char *ptr)
{
        if ( *ptr == 0 )
                return FALSE;

        while( *ptr && isdigit(*ptr) )
                ptr++;
        return ( !*ptr );
}

/* FUNCTION: BOOL IsDecimal(char *ptr)
 *
 * PURPOSE:    This function determines if a string is a non-negative
decimal value.
 *          It fails if any characters other than a series of numbers
followed by
 *                     a decimal point, another series of numbers, and a
null terminator are present.
 *
 * ARGUMENTS:  char                  *ptr    pointer to string to check.
 *
 * RETURNS:            BOOL    FALSE   if string is not a valid non-
negative decimal value
 *                                       TRUE    if string is OK
 */

BOOL IsDecimal(char *ptr)
{
        char *dotptr;
        BOOL  bValid;

        if ( *ptr == 0 )
                return FALSE;

        // find decimal point
        dotptr = strchr( ptr, '.' );
        if (dotptr == NULL)
                // no decimal point, so just check for numeric
                return IsNumeric(ptr);
        *dotptr = 0;  // temporarily replace decimal with a terminator
        if ( *ptr != 0 )
                bValid = IsNumeric(ptr);
        // string starts with decimal point
        else if (*(dotptr+1) == 0)
                return FALSE;  // nothing but a decimal point is bad
        else
                bValid = TRUE;

        if (*(dotptr+1) != 0)
                // check text after decimal point
                bValid &= IsNumeric(dotptr+1);

        *dotptr = '.';  // replace decimal point
        return bValid;
}

//{{NO_DEPENDENCIES}}
// Microsoft Developer Studio generated include file.
// Used by tpcc.rc
//
#define IDD_DIALOG1                     101

// Next default values for new objects
//
#ifdef APSTUDIO_INVOKED
#ifndef APSTUDIO_READONLY_SYMBOLS
#define _APS_NEXT_RESOURCE_VALUE        102
#define _APS_NEXT_COMMAND_VALUE         40001
#define _APS_NEXT_CONTROL_VALUE         1000
#define _APS_NEXT_SYMED_VALUE           101
#endif
#endif

/*      FILE:           READREGISTRY.CPP
 *                              Microsoft TPC-C Kit Ver. 4.20.000
 *                              Copyright Microsoft, 1999
 *              All Rights Reserved
 *
 *                              not yet audited
 *
 *      PURPOSE:        Implementation for TPC-C Tuxedo class.
 *      Contact:        Charles Levine (clevine@microsoft.com)
 *
 *  Change history:
 *              4.20.000 - first version
 */


/* FUNCTION: ReadTPCCRegistrySettings
 *
 * PURPOSE:    This function reads the NT registry for startup
parameters. There parameters are
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
 *                      under the TPCC key.
 *
 * RETURNS      FALSE = no errors
 *                      TRUE  = error reading registry
 */
BOOL ReadTPCCRegistrySettings( TPCCREGISTRYDATA *pReg )
{
        HKEY    hKey;
        DWORD   size;
        DWORD   type;
        DWORD   dwTmp;
        char    szTmp[256];

        if ( RegOpenKeyEx(HKEY_LOCAL_MACHINE, "SOFTWARE\\Microsoft\\TPCC",
0, KEY_READ, &hKey) != ERROR_SUCCESS )
                return TRUE;

        // determine database protocol to use; may be either ODBC or DBLIB
        pReg->eDB_Protocol = Unspecified;
        size = sizeof(szTmp);
        if ( RegQueryValueEx(hKey, "DB_Protocol", 0, &type, (BYTE
*)&szTmp, &size) == ERROR_SUCCESS )
        {
                if ( !stricmp(szTmp, szDBNames[ODBC]) )
                        pReg->eDB_Protocol = ODBC;
                else if ( !stricmp(szTmp, szDBNames[DBLIB]) )
                        pReg->eDB_Protocol = DBLIB;
        }

        pReg->eTxnMon = None;
        // determine txn monitor to use; may be either TUXEDO, or blank
        size = sizeof(szTmp);
        if ( RegQueryValueEx(hKey, "TxnMonitor", 0, &type, (BYTE *)&szTmp,
&size) == ERROR_SUCCESS )
        {
                if ( !stricmp(szTmp, szTxnMonNames[TUXEDO]) )
                        pReg->eTxnMon = TUXEDO;
                else if ( !stricmp(szTmp, szTxnMonNames[ENCINA]) )
                        pReg->eTxnMon = ENCINA;
                else if ( !stricmp(szTmp, szTxnMonNames[COM]) )
                        pReg->eTxnMon = COM;
        }

        pReg->bCOM_SinglePool = FALSE;
        size = sizeof(szTmp);
        if ( RegQueryValueEx(hKey, "COM_SinglePool", 0, &type, (BYTE
*)&szTmp, &size) == ERROR_SUCCESS )
        {
                if ( !stricmp(szTmp, "YES") )
                        pReg->bCOM_SinglePool = TRUE;
        }

        pReg->dwMaxConnections = 0;
        size = sizeof(dwTmp);
        if ( ( RegQueryValueEx(hKey, "MaxConnections", 0, &type,
(LPBYTE)&dwTmp, &size) == ERROR_SUCCESS )
                && (type == REG_DWORD) )
                pReg->dwMaxConnections = dwTmp;

        pReg->dwMaxPendingDeliveries = 0;
        size = sizeof(dwTmp);
        if ( ( RegQueryValueEx(hKey, "MaxPendingDeliveries", 0, &type,
(LPBYTE)&dwTmp, &size) == ERROR_SUCCESS )
                && (type == REG_DWORD) )
                pReg->dwMaxPendingDeliveries = dwTmp;

        pReg->dwNumberOfDeliveryThreads = 0;
        size = sizeof(dwTmp);
        if ( ( RegQueryValueEx(hKey, "NumberOfDeliveryThreads", 0, &type,
(LPBYTE)&dwTmp, &size) == ERROR_SUCCESS )
                && (type == REG_DWORD) )
                pReg->dwNumberOfDeliveryThreads = dwTmp;

        size = sizeof( pReg->szPath );
        if ( RegQueryValueEx(hKey, "Path", 0, &type, (BYTE *)&pReg-
>szPath, &size) != ERROR_SUCCESS )
                pReg->szPath[0] = 0;

        size = sizeof( pReg->szDbServer );
        if ( RegQueryValueEx(hKey, "DbServer", 0, &type, (BYTE *)&pReg-
>szDbServer, &size) != ERROR_SUCCESS )
                pReg->szDbServer[0] = 0;

        size = sizeof( pReg->szDbName );
        if ( RegQueryValueEx(hKey, "DbName", 0, &type, (BYTE *)&pReg-
>szDbName, &size) != ERROR_SUCCESS )
                pReg->szDbName[0] = 0;

        size = sizeof( pReg->szDbUser );
        if ( RegQueryValueEx(hKey, "DbUser", 0, &type, (BYTE *)&pReg-
>szDbUser, &size) != ERROR_SUCCESS )
                pReg->szDbUser[0] = 0;

        size = sizeof( pReg->szDbPassword );
        if ( RegQueryValueEx(hKey, "DbPassword", 0, &type, (BYTE *)&pReg-
>szDbPassword, &size) != ERROR_SUCCESS )
                pReg->szDbPassword[0] = 0;

        RegCloseKey(hKey);

        return FALSE;
}

/*      FILE:           ReadRegistry.h
 *                              Microsoft TPC-C Kit Ver. 4.20.000
 *                              Copyright Microsoft, 1999
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
 *              All Rights Reserved
 *
 *                              not audited
 *
 *      PURPOSE:        Header for registry related code.
 *
 *  Change history:
 *              4.20.000 - first version
 */

enum DBPROTOCOL { Unspecified, ODBC, DBLIB };
const char *szDBNames[] = { "Unspecified", "ODBC", "DBLIB" };

enum TXNMON { None, TUXEDO, ENCINA, COM };
const char *szTxnMonNames[] = { "NONE", "TUXEDO", "ENCINA", "COM" };

//This structure defines the data necessary to keep distinct for each
terminal or client connection.
typedef struct _TPCCREGISTRYDATA
{
        enum DBPROTOCOL eDB_Protocol;
        enum TXNMON eTxnMon;
        BOOL bCOM_SinglePool;
        DWORD dwMaxConnections;
        DWORD dwMaxPendingDeliveries;
        DWORD dwNumberOfDeliveryThreads;
        char szPath[128];
        char szDbServer[32];
        char szDbName[32];
        char szDbUser[32];
        char szDbPassword[32];
} TPCCREGISTRYDATA, *PTPCCREGISTRYDATA;

BOOL ReadTPCCRegistrySettings( TPCCREGISTRYDATA *pReg );

/*      FILE:           ERROR.H
 *                              Microsoft TPC-C Kit Ver. 4.20.000
 *                              Copyright Microsoft, 1999
 *              All Rights Reserved
 *
 *                              Version 4.10.000 audited by Richard Gimarc,
Performance Metrics, 3/17/99
 *
 *      PURPOSE:        Header file for error exception classes.
 *
 *  Change history:
 *              4.20.000 - updated rev number to match kit
 */

#pragma once

#ifndef _INC_STRING
        #include <string.h>
#endif

const int m_szMsg_size = 512;
const int m_szApp_size = 64;
const int m_szLoc_size = 64;

//error message structure used in ErrorText routines
typedef struct _SERRORMSG
{
        int             iError;                         //error id of message
        char    szMsg[256];                     //message to sent to browser
} SERRORMSG;

#define ERR_FATAL_LEVEL                         1
#define ERR_WARNING_LEVEL               2
#define ERR_INFORMATION_LEVEL 3


#define ERR_TYPE_LOGIC                                                  -1
                //logic error in program; internal error
#define ERR_SUCCESS
        0               //success (a non-error error)
#define ERR_BAD_ITEM_ID
        1               //expected abort record in txnRecord
#define ERR_TYPE_DELIVERY_POST                                          2
                //expected delivery post failed
#define ERR_TYPE_WEBDLL
        3               //tpcc web generated error
#define ERR_TYPE_SQL                                                    4
                //sql server generated error
#define ERR_TYPE_DBLIB                                                  5
                //dblib generated error
#define ERR_TYPE_ODBC                                                   6
                //odbc generated error
#define ERR_TYPE_SOCKET
        7               //error on communication socket client rte only
#define ERR_TYPE_DEADLOCK                                               8
                //dblib and odbc only deadlock condition
#define ERR_TYPE_COM                                                    9
                //error from COM call
#define ERR_TYPE_TUXEDO
        10              //tuxedo error
#define ERR_TYPE_OS
        11              //operating system error
#define ERR_TYPE_MEMORY
        12              //memory allocation error
#define ERR_TYPE_TPCC_ODBC                                              13
                //error from tpcc odbc txn module
#define ERR_TYPE_TPCC_DBLIB                                             14
                //error from tpcc dblib txn module
#define ERR_TYPE_DELISRV                                                15
                //delivery server error
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
#define ERR_TYPE_TXNLOG
        16              //txn log error
#define ERR_TYPE_BCCONN
        17              //Benchcraft connection class
#define ERR_TYPE_TPCC_CONN                                      18
                //Benchcraft connection class
#define ERR_TYPE_ENCINA
        19              //Encina error
#define ERR_TYPE_COMPONENT                                      20
                //error from COM component

class CBaseErr
{
public:
        char    *m_szApp;
        char    *m_szMsg;
        char    *m_szLoc;       // code location where the error occurred
        int             m_idMsg;

        CBaseErr(void)
        {
                m_idMsg         = 0;
                m_szMsg         = new char[m_szMsg_size];
                m_szApp         = new char[m_szApp_size];
                m_szLoc         = NULL;

                m_szMsg[0]      = 0;
                m_szApp[0]      = 0;

                GetModuleFileName(GetModuleHandle(NULL), m_szApp,
m_szApp_size);
        }

        ~CBaseErr(void)
        {
                if (m_szMsg)
                        delete [] m_szMsg;
                if (m_szApp)
                        delete [] m_szApp;
                if (m_szLoc)
                        delete [] m_szLoc;
        };

        CBaseErr(int idMsg)
        {
                m_idMsg         = idMsg;
                m_szApp         = new char[m_szApp_size];
                m_szMsg         = new char[m_szMsg_size];
                m_szLoc         = NULL;

                GetModuleFileName(GetModuleHandle(NULL), m_szApp,
m_szApp_size);
                LoadString(GetModuleHandle(NULL), idMsg, m_szMsg,
m_szMsg_size);
        }

        CBaseErr(LPCTSTR szMsg)
        {
                m_idMsg         = 0;
                m_szApp         = new char[m_szApp_size];
                m_szMsg         = new char[m_szMsg_size];
                m_szLoc         = NULL;

                GetModuleFileName(GetModuleHandle(NULL), m_szApp,
m_szApp_size);
                strcpy(m_szMsg, szMsg);
        }

        void SetError(char *szMsg, LPCTSTR szLocation)
        {
                if (szMsg != NULL)
                        strcpy(m_szMsg, szMsg);
                else
                        m_szMsg[0] = 0;

                if (szLocation != NULL)
                {
                        delete [] m_szLoc;
                        m_szLoc = new char[strlen(szLocation)+1];
                        strcpy(m_szLoc, szLocation);
                }
                else
                {
                        delete [] m_szLoc;
                        m_szLoc = NULL;
                }
        }

        virtual void Draw(HWND hwnd, LPCTSTR szStr = NULL)
        {
                int             j;
                char    szTmp[512];

                if (szStr)
                        j = wsprintf(szTmp, "%s\n",szStr);
                if (m_szLoc)
                        j += wsprintf(szTmp+j, "Location=%s\n",m_szLoc);
                if (m_szMsg)
                        j += wsprintf(szTmp+j, "%s\n", m_szMsg);

                ::MessageBox(hwnd, szTmp, m_szApp, MB_OK);
        }

        char *GetApp(void) { return m_szApp; }
        char *GetMsg(void) { return m_szMsg; }
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        char *GetLocation(void) { return m_szLoc; }

        virtual int ErrorType() = 0;  // a value which distinguishes the
kind of error that occurred
        virtual int ErrorNum() = 0;          // an error value specific
to the error type
        virtual char *ErrorText() = 0;       // a string (i.e., human
readable) representation of the error

};

class CSocketErr : public CBaseErr
{
public:
        enum Action
        {
                eNone,
                eSend,
                eSocket,
                eConnect
        };

        CSocketErr(Action eAction, LPCTSTR szLocation);
        CSocketErr(int iError) { m_errId = iError; };
        int            m_errId;
        Action  m_eAction;

        int ErrorType() { return ERR_TYPE_SOCKET;};
        int ErrorNum() { return m_errId;};
        char *ErrorText(void);
};

class CSystemErr : public CBaseErr
{
public:
        enum Action
        {
                eNone,
                eTransactNamedPipe,
                eWaitNamedPipe,
                eSetNamedPipeHandleState,
                eCreateFile,
                eCreateProcess,
                eCallNamedPipe,
                eCreateEvent,
                eCreateThread,
                eVirtualAlloc,
                eReadFile,
                eWriteFile,
                eMapViewOfFile,
                eCreateFileMapping,
                eInitializeSecurityDescriptor,
                eSetSecurityDescriptorDacl,
                eCreateNamedPipe,
                eConnectNamedPipe,
        };

        CSystemErr(Action eAction, LPCTSTR szLocation);

        void Draw(HWND hwnd, LPCTSTR szStr = NULL);

        int            m_errId;
        Action  m_eAction;

        int ErrorType() { return ERR_TYPE_OS;}
        int ErrorNum() { return m_errId;}
        char *ErrorText() { return "";}              // TODO: need to code
error text
};

class CMemoryErr : public CBaseErr
{
public:
        CMemoryErr(void);

        int ErrorType() { return ERR_TYPE_MEMORY;}
        int ErrorNum() { return 0;}
        char *ErrorText() { return "";}              // TODO: need to code
error text
};


/*      FILE:           TRANS.H
 *                              Microsoft TPC-C Kit Ver. 4.20.000
 *                              Copyright Microsoft, 1999
 *              All Rights Reserved
 *
 *                              Version 4.10.000 audited by Richard Gimarc,
Performance Metrics, 3/17/99
 *
 *      PURPOSE:        Header file for TPC-C structure templates.
 *
 *  Change history:
 *              4.20.000 - updated rev number to match kit
 */
#pragma once

// String length constants
#define SERVER_NAME_LEN            20
#define DATABASE_NAME_LEN          20
#define PASSWORD_LEN               20
#define TABLE_NAME_LEN             20
#define I_DATA_LEN                 50
#define I_NAME_LEN                 24
#define BRAND_LEN                   1
#define LAST_NAME_LEN              16
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
#define W_NAME_LEN                10
#define ADDRESS_LEN               20
#define STATE_LEN                  2
#define ZIP_LEN                    9
#define S_DIST_LEN                24
#define S_DATA_LEN                50
#define D_NAME_LEN                10
#define FIRST_NAME_LEN            16
#define MIDDLE_NAME_LEN            2
#define PHONE_LEN                 16
#define DATETIME_LEN              30
#define CREDIT_LEN                 2
#define C_DATA_LEN               250
#define H_DATA_LEN                24
#define DIST_INFO_LEN             24
#define MAX_OL_NEW_ORDER_ITEMS    15
#define MAX_OL_ORDER_STATUS_ITEMS 15
#define STATUS_LEN                25
#define OL_DIST_INFO_LEN               24

// TIMESTAMP_STRUCT is provided by the ODBC header file sqltypes.h, but
is not available
// when compiling with dblib, so redefined here.  Note: we are using the
symbol "__SQLTYPES"
// (declared in sqltypes.h) as a way to determine if TIMESTAMP_STRUCT has
been declared.
#ifndef __SQLTYPES
       typedef struct
       {
                      short                 /* SQLSMALLINT */     year;
                      unsigned short /* SQLUSMALLINT */     month;
                      unsigned short /* SQLUSMALLINT */     day;
                      unsigned short /* SQLUSMALLINT */     hour;
                      unsigned short /* SQLUSMALLINT */     minute;
                      unsigned short /* SQLUSMALLINT */     second;
                      unsigned long  /* SQLUINTEGER */      fraction;
       } TIMESTAMP_STRUCT;
#endif

// possible values for exec_status_code after transaction completes
enum EXEC_STATUS
{
       eOK,                  // 0    "Transaction commited."
       eInvalidItem,  // 1    "Item number is not valid."
       eDeliveryFailed       // 2    "Delivery Post Failed."
};

// transaction structures
typedef struct
{
       // input params
       short                         ol_supply_w_id;
       long                          ol_i_id;
       short                         ol_quantity;

       // output params
       char                          ol_i_name[I_NAME_LEN+1];
       char                          ol_brand_generic[BRAND_LEN+1];
       double                        ol_i_price;
       double                        ol_amount;
       short                         ol_stock;
} OL_NEW_ORDER_DATA;

typedef struct
{
       // input params
       short                 w_id;
       short                 d_id;
       long                  c_id;
       short                 o_ol_cnt;

       // output params
       EXEC_STATUS                   exec_status_code;
       char                  c_last[LAST_NAME_LEN+1];
       char                  c_credit[CREDIT_LEN+1];
       double                c_discount;
       double                w_tax;
       double                d_tax;
       long                  o_id;
       short                         o_commit_flag;
       TIMESTAMP_STRUCT      o_entry_d;
       short                 o_all_local;
       double                total_amount;
       OL_NEW_ORDER_DATA     OL[MAX_OL_NEW_ORDER_ITEMS];
} NEW_ORDER_DATA, *PNEW_ORDER_DATA;

typedef struct
{      // input params
       short                          w_id;
       short                          d_id;
       long                           c_id;
       short                          c_d_id;
       short                          c_w_id;
       double                         h_amount;
       char                    c_last[LAST_NAME_LEN+1];

       // output params
       EXEC_STATUS                    exec_status_code;
       TIMESTAMP_STRUCT       h_date;
       char                           w_street_1[ADDRESS_LEN+1];
       char                           w_street_2[ADDRESS_LEN+1];
       char                           w_city[ADDRESS_LEN+1];
       char                           w_state[STATE_LEN+1];
       char                           w_zip[ZIP_LEN+1];
       char                           d_street_1[ADDRESS_LEN+1];
       char                           d_street_2[ADDRESS_LEN+1];
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        char                            d_city[ADDRESS_LEN+1];
        char                            d_state[STATE_LEN+1];
        char                          d_zip[ZIP_LEN+1];
        char                            c_first[FIRST_NAME_LEN+1];
        char                            c_middle[MIDDLE_NAME_LEN + 1];
        char                            c_street_1[ADDRESS_LEN+1];
        char                            c_street_2[ADDRESS_LEN+1];
        char                            c_city[ADDRESS_LEN+1];
        char                            c_state[STATE_LEN+1];
        char                            c_zip[ZIP_LEN+1];
        char                            c_phone[PHONE_LEN+1];
        TIMESTAMP_STRUCT        c_since;
        char                            c_credit[CREDIT_LEN+1];
        double                          c_credit_lim;
        double                          c_discount;
        double                          c_balance;
        char                            c_data[200+1];
} PAYMENT_DATA, *PPAYMENT_DATA;

typedef struct
{
        long                            ol_i_id;
        short                           ol_supply_w_id;
        short                           ol_quantity;
        double                          ol_amount;
        TIMESTAMP_STRUCT        ol_delivery_d;
} OL_ORDER_STATUS_DATA;

typedef struct
{       // input params
        short                   w_id;
        short                   d_id;
        long                    c_id;
        char                    c_last[LAST_NAME_LEN+1];

        // output params
        EXEC_STATUS                     exec_status_code;
        char                    c_first[FIRST_NAME_LEN+1];
        char                    c_middle[MIDDLE_NAME_LEN+1];
        double                  c_balance;
        long                    o_id;
        TIMESTAMP_STRUCT          o_entry_d;
        short                   o_carrier_id;
        OL_ORDER_STATUS_DATA OL[MAX_OL_ORDER_STATUS_ITEMS];
        short                   o_ol_cnt;
} ORDER_STATUS_DATA, *PORDER_STATUS_DATA;


typedef struct
{       // input params
        short                    w_id;
        short                    o_carrier_id;

        // output params
        EXEC_STATUS                     exec_status_code;
        SYSTEMTIME              queue_time;
        long                            o_id[10];               // id's of
delivered orders for districts 1 to 10
} DELIVERY_DATA, *PDELIVERY_DATA;

//This structure is used for posting delivery transactions and for
writing them to the delivery server.
typedef struct _DELIVERY_TRANSACTION
{
        SYSTEMTIME      queue;                  //time delivery transaction
queued
        short           w_id;                   //delivery warehouse
        short           o_carrier_id;  //carrier id
} DELIVERY_TRANSACTION;

typedef struct
{       // input params
        short                           w_id;
        short                           d_id;
        short                           threshold;

        // output params
        EXEC_STATUS                     exec_status_code;
        long                            low_stock;
} STOCK_LEVEL_DATA, *PSTOCK_LEVEL_DATA;

/*      FILE:           TXN_BASE.H
 *                              Microsoft TPC-C Kit Ver. 4.20.000
 *                              Copyright Microsoft, 1999
 *              All Rights Reserved
 *
 *                              Version 4.10.000 audited by Richard Gimarc,
Performance Metrics, 3/17/99
 *
 *      PURPOSE:        Header file for TPC-C txn class implementation.
 *
 *  Change history:
 *              4.20.000 - updated rev number to match kit
 */

#pragma once

// need to declare functions for import, unless define has already been
created
// by the DLL's .cpp module for export.
#ifndef DllDecl
#define DllDecl  __declspec( dllimport )
#endif

class DllDecl CTPCC_BASE
{
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        public:
                CTPCC_BASE(void) {};
                virtual ~CTPCC_BASE(void) {};

                virtual PNEW_ORDER_DATA              BuffAddr_NewOrder()
                = 0;
                virtual PPAYMENT_DATA         BuffAddr_Payment()
        = 0;
                virtual PDELIVERY_DATA        BuffAddr_Delivery()
        = 0;
                virtual PSTOCK_LEVEL_DATA     BuffAddr_StockLevel()  = 0;
                virtual PORDER_STATUS_DATA    BuffAddr_OrderStatus() = 0;

                virtual void NewOrder         () = 0;
                virtual void Payment          () = 0;
                virtual void Delivery         () = 0;
                virtual void StockLevel               () = 0;
                virtual void OrderStatus      () = 0;
};

/*      FILE:           TPCC_DBLIB.CPP
 *                              Microsoft TPC-C Kit Ver. 4.20.000
 *                              Copyright Microsoft, 1999
 *              All Rights Reserved
 *
 *                              Version 4.10.000 audited by Richard Gimarc,
Performance Metrics, 3/17/99
 *
 *      PURPOSE:        Implements dblib calls for TPC-C txns.
 *      Contact:        Charles Levine (clevine@microsoft.com)
 *
 *  Change history:
 *              4.20.000 - updated rev number to match kit
 *              4.10.001 - not deleting error class in catch handler on
deadlock retry;
 *                                 not a functional bug, but a memory leak
 *                               - had to tweak some declarations to
compile with latest SDK; no functional change
 */

#include <windows.h>
#include <stdio.h>
#include <assert.h>

#define DBNTWIN32
#include <sqlfront.h>
#include <sqldb.h>

#ifdef ICECAP
#include <icapexp.h>
#endif

// need to declare functions for export
#define DllDecl  __declspec( dllexport )

#include "..\..\common\src\error.h"
#include "..\..\common\src\trans.h"
#include "..\..\common\src\txn_base.h"
#include "tpcc_dblib.h"

#define DEFCLPACKSIZE                 4096

// version string; must match return value from tpcc_version stored proc
const char      sVersion[] = "4.10.000";

const                   iMaxRetries = 10;             // how many retries
on deadlock
static  long    iConnectionCount = 0;  // number of current dblib
connections

BOOL APIENTRY DllMain(HMODULE hModule, DWORD ul_reason_for_call, LPVOID
lpReserved)
{
        switch( ul_reason_for_call )
        {
                case DLL_PROCESS_ATTACH:
                        DisableThreadLibraryCalls(hModule);
                        dbinit();             // initialize dblib
                        break;

                case DLL_PROCESS_DETACH:
                        dbexit();             // close all dblib
structures/connections
                        break;

                default:
                        /* nothing */;
        }
        return TRUE;
}


int err_handler(DBPROCESS *dbproc, int severity, int dberr, int oserr,
LPCSTR dberrstr, LPCSTR oserrstr)
{
        CTPCC_DBLIB                   *pConn;

        assert(dbproc != NULL);
        pConn = (CTPCC_DBLIB*)dbgetuserdata(dbproc);

        if (pConn != NULL)
        {
                pConn->SetDbLibError( severity, dberr, oserr, dberrstr,
oserrstr );
        }
        return INT_CANCEL;
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
}

/* FUNCTION: int msg_handler(DBPROCESS *dbproc, DBINT msgno, int
msgstate, int severity, char *msgtext)
 *
 * PURPOSE:    This function handles DB-Library SQL Server error messages
 *
 * ARGUMENTS:  DBPROCESS             *dbproc               DBPROCESS id
pointer
 *                           DBINT                 msgno
       message number
 *                           int                           msgstate
              message state
 *                           int                           severity
              message severity
 *                           char                  *msgtext
       printable message description
 *
 * RETURNS:           int                           INT_CONTINUE
       continue if error is SQLETIME else INT_CANCEL action
 *                                                        INT_CANCEL
              cancel operation
 *
 * COMMENTS:   This function also sets the dead lock dbproc variable if
necessary.
 *
 */

// typedef INT (SQLAPI *DBMSGHANDLE_PROC)(PDBPROCESS, DBINT, INT, INT,
LPCSTR, LPCSTR, LPCSTR, DBUSMALLINT);

int msg_handler(DBPROCESS *dbproc, DBINT msgno, int msgstate, int
severity,
                           LPCSTR msgtext, LPCSTR srvname, LPCSTR
procname, DBUSMALLINT line)
{
       CTPCC_DBLIB                  *pConn;

       assert(dbproc != NULL);
       pConn = (CTPCC_DBLIB*)dbgetuserdata(dbproc);

       if (pConn != NULL)
       {
              pConn->SetSqlError( msgno, msgstate, severity, msgtext );
       }

       return 0;
}

/* FUNCTION: void UtilStrCpy(char * pDest, char * pSrc, int n)
 *
 * PURPOSE:    This function copies n characters from string pSrc to pDst
and places a
 *                     null character at the end of the destination
string.
 *
 * ARGUMENTS:  char                  *pDest  destination string pointer
 *                                   char                  *pSrc   source string
pointer
 *                                   int                           n
       number of characters to copy
 *
 * RETURNS:           None
 *
 * COMMENTS:   Unlike strncpy this function ensures that the result
string is
 *                                   always null terminated.
 *
 */

inline static void UtilStrCpy(char * pDest, const BYTE * pSrc, int n)
{
    strncpy(pDest, (char *)pSrc, n);
    pDest[n] = '\0';

       return;
}

/* FUNCTION: CTPCC_DBLIB_ERR::ErrorText
 *
 */

char* CTPCC_DBLIB_ERR::ErrorText(void)
{
       int i;

       static SERRORMSG errorMsgs[] =
       {
              { ERR_WRONG_SP_VERSION,               "Wrong version of
stored procs on database server"      },
              { ERR_INVALID_CUST,                   "Invalid Customer
id,name."                                                  },
              { ERR_NO_SUCH_ORDER,          "No orders found for
customer."                                                 },
              { 0,                                                 ""

                     }
       };

       static char szNotFound[] = "Unknown error number.";

       for(i=0; errorMsgs[i].szMsg[0]; i++)
       {
              if ( m_errno == errorMsgs[i].iError )
                     break;
       }
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        if ( !errorMsgs[i].szMsg[0] )
                return szNotFound;
        else
                return errorMsgs[i].szMsg;
}

// wrapper routine for class constructor
__declspec(dllexport) CTPCC_DBLIB* CTPCC_DBLIB_new(
        LPCSTR szServer,              // name of SQL server
        LPCSTR szUser,                // user name for login
        LPCSTR szPassword,            // password for login
        LPCSTR szHost,                // workstation name; shows up in
sp_who; max 30 chars, only first 10 kept by SQL Server
        LPCSTR szDatabase )           // name of database to use
{
        return new CTPCC_DBLIB( szServer, szUser, szPassword, szHost,
szDatabase );
}


CTPCC_DBLIB::CTPCC_DBLIB (
        LPCSTR szServer,              // name of SQL server
        LPCSTR szUser,                // user name for login
        LPCSTR szPassword,            // password for login
        LPCSTR szHost,                // workstation name; shows up in
sp_who; max 30 chars, only first 10 kept by SQL Server
        LPCSTR szDatabase )           // name of database to use
{
        LOGINREC       *login;
        const BYTE     *pData;

        // initialization
        m_dbproc = NULL;
        m_DbLibErr = (CDBLIBERR*)NULL;
        m_SqlErr = (CSQLERR*)NULL;

        m_MaxRetries = 10;            // how many retries on deadlock

        // increase max number of connections if getting close
        if ( dbgetmaxprocs() < (iConnectionCount+5) )
        {
                if ( dbsetmaxprocs(iConnectionCount+10) == FAIL )
                        ThrowError(CDBLIBERR::eDbSetMaxProcs);
        }

        // allocate a login structure
        login = dblogin();
        if (login == NULL)
                ThrowError(CDBLIBERR::eLogin);
        InterlockedIncrement( &iConnectionCount );

        // register error and message handler functions
        if (dbprocerrhandle(login, err_handler) == NULL)
                ThrowError(CDBLIBERR::eDbProcHandler);

        if (dbprocmsghandle(login, msg_handler) == NULL)
                ThrowError(CDBLIBERR::eDbProcHandler);


        DBSETLUSER(login, szUser);
        DBSETLPWD(login, szPassword);
        DBSETLHOST(login, szHost);
        DBSETLPACKET(login, (unsigned short)DEFCLPACKSIZE);
        DBSETLVERSION(login, DBVER60);                // use dblib ver 6.0
client behavior

        // set time to wait for login
        if (dbsetlogintime(60) == FAIL)
                ThrowError(CDBLIBERR::eDbSet);

        // set time to wait for statement execution
        if (dbsettime(180) == FAIL)
                ThrowError(CDBLIBERR::eDbSet);

        m_dbproc = dbopen(login, szServer);

        // deallocate login structure before checking for success
        dbfreelogin( login );

        if (m_dbproc == NULL)
                ThrowError(CDBLIBERR::eDbOpen);

        // save address of class instance so that the message and error
handler
        // can get to data.
        dbsetuserdata(m_dbproc, (LPVOID)this);

        // Use the the right database
        if (dbuse(m_dbproc, szDatabase) == FAIL)
                ThrowError(CDBLIBERR::eDbUse);

        dbcmd(m_dbproc, "set nocount on ");                   // do not
return row counts
        dbcmd(m_dbproc, "set XACT_ABORT ON");         // rollback
transaction on abort

        if (dbsqlexec(m_dbproc) == FAIL)
                ThrowError(CDBLIBERR::eDbSqlExec);

        DiscardNextResults(2);

        // verify that version of stored procs on server is correct
        dbrpcinit(m_dbproc, "tpcc_version", 0);

        if (dbrpcexec(m_dbproc) == FAIL)
                ThrowError(CDBLIBERR::eDbRpcExec);
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        if (dbresults(m_dbproc) != SUCCEED)
                ThrowError(CDBLIBERR::eDbResults);

        if (dbnextrow(m_dbproc) != REG_ROW)
                ThrowError(CDBLIBERR::eDbNextRow);

        char szSrvVersion[16];
        pData=dbdata(m_dbproc, 1);
        if (pData)
                UtilStrCpy(szSrvVersion, pData, dbdatlen(m_dbproc, 1));
        else
                szSrvVersion[0]=0;
        if (strcmp(szSrvVersion,sVersion))
                throw new CTPCC_DBLIB_ERR(
CTPCC_DBLIB_ERR::ERR_WRONG_SP_VERSION );


        DiscardNextRows(0);
        DiscardNextResults(0);
}


CTPCC_DBLIB::~CTPCC_DBLIB( void )
{
        // close db connection and deallocate resources
        dbclose(m_dbproc);
        InterlockedDecrement( &iConnectionCount );
        if (m_DbLibErr != NULL)
                delete m_DbLibErr;
        if (m_SqlErr != NULL)
                delete m_SqlErr;
}

void CTPCC_DBLIB::SetDbLibError(int severity, int dberr, int oserr,
LPCSTR dberrstr, LPCSTR oserrstr)
{
        delete m_DbLibErr;
        m_DbLibErr = new CDBLIBERR(CDBLIBERR::eUnknown, severity, dberr,
oserr);

        if (dberrstr != NULL)
        {
                m_DbLibErr->m_dberrstr = new char[ strlen(dberrstr)+1 ];
                strcpy( m_DbLibErr->m_dberrstr, dberrstr );
        }

        if (oserrstr != NULL)
                m_DbLibErr->m_oserrstr = new char[ strlen(oserrstr)+1 ];
                strcpy( m_DbLibErr->m_oserrstr, oserrstr );
        }

}
```

```
void CTPCC_DBLIB::SetSqlError( int /*DBINT*/ msgno, int msgstate, int
severity, LPCSTR msgtext )
{
        if (m_SqlErr == NULL)
                m_SqlErr = new CSQLERR();

        m_SqlErr->m_msgno = msgno;
        m_SqlErr->m_msgstate = msgstate;
        m_SqlErr->m_severity = severity;

        delete [] m_SqlErr->m_msgtext;
        if (msgtext != NULL)
        {
                m_SqlErr->m_msgtext = new char[ strlen(msgtext)+1 ];
                strcpy( m_SqlErr->m_msgtext, msgtext );
        }
}

void CTPCC_DBLIB::ThrowError( CDBLIBERR::ACTION eAction )
{
        // discard anything still in return buffer
        DiscardNextRows(-1);
        DiscardNextResults(-1);

        // check for SQL Server error first;  if yes, throw it and ignore
any DBLib error.
        if (m_SqlErr != NULL)
        {
                CSQLERR        *pSqlErr;
                pSqlErr = m_SqlErr;
                m_SqlErr = NULL;       // clear our pointer to instance;
catch handler will delete
                throw pSqlErr;
        }

        CDBLIBERR      *pDbLibErr;
        if (m_DbLibErr == NULL)
                // this case isn't expected to happen, since it means that
an error was returned
                // but the error handlers were not called.
                pDbLibErr = new CDBLIBERR(eAction);
        else
        {
                pDbLibErr = m_DbLibErr;
                pDbLibErr->m_eAction = eAction;
                m_DbLibErr = NULL;             // clear our pointer to
instance; catch handler will delete
        }

        throw pDbLibErr;
}
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
// Read and discard rows until no more.  Throw an exception if number of
rows read doesn't
// match number of rows expected.  The row count will be ignored if the
expected count value
// passed in is negative.  A typical use of this routine is to verify
that there are no more
// rows to be read.
void CTPCC_DBLIB::DiscardNextRows(int iExpectedCount)
{
        int             iRowsRead = 0;
        RETCODE rc;

        while (TRUE)
        {
                rc = dbnextrow(m_dbproc);
                if (rc == NO_MORE_ROWS)
                        break;
                if (rc == FAIL)
                {
                        if (iExpectedCount >= 0)
                                ThrowError(CDBLIBERR::eDbNextRow);
                        else
                                break;
                }
                iRowsRead++;
        }

        if ((iExpectedCount >= 0) &&
                (iExpectedCount != iRowsRead))
                ThrowError(CDBLIBERR::eWrongRowCount);
}

// Read and discard results until no more.  Throw an exception if number
of result sets read doesn't
// match number expected.  The result set count will be ignored if the
expected count value
// passed in is negative.  A typical use of this routine is to verify
that there are no more
// result sets to be read.
void CTPCC_DBLIB::DiscardNextResults(int iExpectedCount)
{
        int             iResultsRead = 0;
        RETCODE rc;

        while (TRUE)
        {
                rc = dbresults(m_dbproc);
                if (rc == NO_MORE_RESULTS)
                        break;
                if (rc == FAIL)
                {
                        if (iExpectedCount >= 0)
                                ThrowError(CDBLIBERR::eDbResults);
                        else
                                break;
                }

                DiscardNextRows(-1);
                iResultsRead++;
        }

        if ((iExpectedCount >= 0) &&
                (iExpectedCount != iResultsRead))
                ThrowError(CDBLIBERR::eWrongRowCount);
}

void CTPCC_DBLIB::StockLevel()
{
        int                     iTryCount = 0;
        const BYTE      *pData;

        ResetError();

        while (TRUE)
        {
                try
                {
                        dbrpcinit(m_dbproc, "tpcc_stocklevel", 0);

                        dbrpcparam(m_dbproc, NULL, 0, SQLINT2, -1, -1,
(BYTE *) &m_txn.StockLevel.w_id);             // @w_id smallint
                        dbrpcparam(m_dbproc, NULL, 0, SQLINT1, -1, -1,
(BYTE *) &m_txn.StockLevel.d_id);             // @d_id tinyint
                        dbrpcparam(m_dbproc, NULL, 0, SQLINT2, -1, -1,
(BYTE *) &m_txn.StockLevel.threshold);   // @threshhold smallint

                        if (dbrpcexec(m_dbproc) == FAIL)
                                ThrowError(CDBLIBERR::eDbRpcExec);

                        if (dbresults(m_dbproc) != SUCCEED)
                                ThrowError(CDBLIBERR::eDbResults);

                        if (dbnextrow(m_dbproc) != REG_ROW)
                                ThrowError(CDBLIBERR::eDbNextRow);

                        if (pData=dbdata(m_dbproc, 1))
                                m_txn.StockLevel.low_stock = *((long *)
pData);

                        DiscardNextRows(0);
                        DiscardNextResults(0);

                        m_txn.StockLevel.exec_status_code = eOK;
                        return;
                }
                catch (CSQLERR *e)
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
		{
			if ((e->m_msgno != 1205) || (++iTryCount > 
iMaxRetries))
				throw;

			// hit deadlock; backoff for increasingly longer 
period
			delete e;
			Sleep(10 * iTryCount);
		}
	}	// while (TRUE)
}


void CTPCC_DBLIB::NewOrder()
{
	int			i;
	DBINT		commit_flag;
	DBDATETIME	datetime;
	DBDATEREC	daterec;

	int			iTryCount = 0;
	const BYTE	*pData;

	ResetError();

	while (TRUE)
	{
		try
		{
			dbrpcinit(m_dbproc, "tpcc_neworder", 0);

			dbrpcparam(m_dbproc, NULL, 0, SQLINT2, -1, -1, 
(BYTE *) &m_txn.NewOrder.w_id);
			dbrpcparam(m_dbproc, NULL, 0, SQLINT1, -1, -1, 
(BYTE *) &m_txn.NewOrder.d_id);
			dbrpcparam(m_dbproc, NULL, 0, SQLINT4, -1, -1, 
(BYTE *) &m_txn.NewOrder.c_id);
			dbrpcparam(m_dbproc, NULL, 0, SQLINT1, -1, -1, 
(BYTE *) &m_txn.NewOrder.o_ol_cnt);

			// check whether any order lines are for a remote 
warehouse
			m_txn.NewOrder.o_all_local = 1;
			for (i = 0; i < m_txn.NewOrder.o_ol_cnt; i++)
			{
				if (m_txn.NewOrder.OL[i].ol_supply_w_id != 
m_txn.NewOrder.w_id)
				{
					m_txn.NewOrder.o_all_local = 0;  // 
at least one remote warehouse
					break;
				}
			}
			dbrpcparam(m_dbproc, NULL, 0, SQLINT1, -1, -1, 
(BYTE *) &m_txn.NewOrder.o_all_local);

			for (i = 0; i < m_txn.NewOrder.o_ol_cnt; i++)
			{
				dbrpcparam(m_dbproc, NULL, 0, SQLINT4, -1, 
-1, (BYTE *) &m_txn.NewOrder.OL[i].ol_i_id);
				dbrpcparam(m_dbproc, NULL, 0, SQLINT2, -1, 
-1, (BYTE *) &m_txn.NewOrder.OL[i].ol_supply_w_id);
				dbrpcparam(m_dbproc, NULL, 0, SQLINT2, -1, 
-1, (BYTE *) &m_txn.NewOrder.OL[i].ol_quantity);
			}

			if (dbrpcexec(m_dbproc) == FAIL)
				ThrowError(CDBLIBERR::eDbRpcExec);


			// Get order line results
			m_txn.NewOrder.total_amount = 0;
			for (i = 0; i<m_txn.NewOrder.o_ol_cnt; i++)
			{
				if (dbresults(m_dbproc) != SUCCEED)
					ThrowError(CDBLIBERR::eDbResults);

				if (dbnumcols(m_dbproc) != 5)

	ThrowError(CDBLIBERR::eWrongNumCols);

				if (dbnextrow(m_dbproc) != REG_ROW)
					ThrowError(CDBLIBERR::eDbNextRow);

				if(pData=dbdata(m_dbproc, 1))

	UtilStrCpy(m_txn.NewOrder.OL[i].ol_i_name, pData, 
dbdatlen(m_dbproc, 1));
				if(pData=dbdata(m_dbproc, 2))
					m_txn.NewOrder.OL[i].ol_stock = 
(*(DBSMALLINT *) pData);
				if(pData=dbdata(m_dbproc, 3))

	UtilStrCpy(m_txn.NewOrder.OL[i].ol_brand_generic, pData, 
dbdatlen(m_dbproc, 3));
				if(pData=dbdata(m_dbproc, 4))
					dbconvert(m_dbproc, SQLNUMERIC, 
pData, dbdatlen(m_dbproc,4),
					SQLFLT8, (BYTE 
*)&m_txn.NewOrder.OL[i].ol_i_price, 8);
				if(pData=dbdata(m_dbproc, 5))

					dbconvert(m_dbproc, SQLNUMERIC, 
pData, dbdatlen(m_dbproc,5),
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                                        SQLFLT8, (BYTE
*)&m_txn.NewOrder.OL[i].ol_amount, 8);

                                m_txn.NewOrder.total_amount =
m_txn.NewOrder.total_amount + m_txn.NewOrder.OL[i].ol_amount;

                                DiscardNextRows(0);
                        }


                        // get remaining values for w_tax, d_tax, o_id,
c_last, c_discount, c_credit, o_entry_d, commit_flag
                        if (dbresults(m_dbproc) != SUCCEED)
                                ThrowError(CDBLIBERR::eDbResults);

                        if (dbnextrow(m_dbproc) != REG_ROW)
                                ThrowError(CDBLIBERR::eDbNextRow);

                        if (dbnumcols(m_dbproc) != 8)
                                ThrowError(CDBLIBERR::eWrongNumCols);

                        if (pData=dbdata(m_dbproc, 1))


                                dbconvert(m_dbproc, SQLNUMERIC, pData,
dbdatlen(m_dbproc,1), SQLFLT8, (BYTE *)&m_txn.NewOrder.w_tax, 8);
                        if (pData=dbdata(m_dbproc, 2))

                                dbconvert(m_dbproc, SQLNUMERIC, pData,
dbdatlen(m_dbproc,2), SQLFLT8, (BYTE *)&m_txn.NewOrder.d_tax, 8);
                        if (pData=dbdata(m_dbproc, 3))
                                m_txn.NewOrder.o_id = (*(DBINT *) pData);
                        if (pData=dbdata(m_dbproc, 4))
                                UtilStrCpy(m_txn.NewOrder.c_last, pData,
dbdatlen(m_dbproc, 4));
                        if (pData=dbdata(m_dbproc, 5))
                                dbconvert(m_dbproc, SQLNUMERIC, pData,
dbdatlen(m_dbproc,5), SQLFLT8, (BYTE *)&m_txn.NewOrder.c_discount, 8);
                        if (pData=dbdata(m_dbproc, 6))
                                UtilStrCpy(m_txn.NewOrder.c_credit, pData,
dbdatlen(m_dbproc, 6));
                        if (pData=dbdata(m_dbproc, 7))
                        {
                                datetime = *((DBDATETIME *) pData);
                                dbdatecrack(m_dbproc, &daterec, &datetime);
                                m_txn.NewOrder.o_entry_d.year   =
daterec.year;
                                m_txn.NewOrder.o_entry_d.month  =
daterec.month;
                                m_txn.NewOrder.o_entry_d.day    =
daterec.day;
                                m_txn.NewOrder.o_entry_d.hour   =
daterec.hour;
                                m_txn.NewOrder.o_entry_d.minute =
daterec.minute;
                                m_txn.NewOrder.o_entry_d.second =
daterec.second;
                        }
                        if (pData=dbdata(m_dbproc, 8))
                                commit_flag = (*(DBTINYINT *) pData);

                        DiscardNextRows(0);
                        DiscardNextResults(0);


                        if (commit_flag == 1)
                        {
                                m_txn.NewOrder.total_amount *= ((1 +
m_txn.NewOrder.w_tax + m_txn.NewOrder.d_tax) * (1 -
m_txn.NewOrder.c_discount));
                                m_txn.NewOrder.exec_status_code = eOK;
                        }
                        else
                                m_txn.NewOrder.exec_status_code =
eInvalidItem;

                        return;
                }
                catch (CSQLERR *e)
                {
                        if ((e->m_msgno != 1205) || (++iTryCount >
iMaxRetries))
                                throw;

                        // hit deadlock; backoff for increasingly longer
period
                        delete e;
                        Sleep(10 * iTryCount);
                }
        }       // while (TRUE)
}


void CTPCC_DBLIB::Payment()
{
        DBDATETIME      datetime;
        DBDATEREC       daterec;

        int                     iTryCount = 0;
        const BYTE      *pData;

        ResetError();

        while (TRUE)
        {
                try
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
            {
                    dbrpcinit(m_dbproc, "tpcc_payment", 0);

                    dbrpcparam(m_dbproc, NULL, 0, SQLINT2, -1, -1,
(BYTE *) &m_txn.Payment.w_id);
                    dbrpcparam(m_dbproc, NULL, 0, SQLINT2, -1, -1,
(BYTE *) &m_txn.Payment.c_w_id);
                    dbrpcparam(m_dbproc, NULL, 0, SQLFLT8, -1, -1,
(BYTE *) &m_txn.Payment.h_amount);
                    dbrpcparam(m_dbproc, NULL, 0, SQLINT1, -1, -1,
(BYTE *) &m_txn.Payment.d_id);
                    dbrpcparam(m_dbproc, NULL, 0, SQLINT1, -1, -1,
(BYTE *) &m_txn.Payment.c_d_id);
                    dbrpcparam(m_dbproc, NULL, 0, SQLINT4, -1, -1,
(BYTE *) &m_txn.Payment.c_id);

                    // if customer id is zero, then payment is by name
                    if (m_txn.Payment.c_id == 0)
                            dbrpcparam(m_dbproc, NULL, 0, SQLCHAR, -1,
strlen(m_txn.Payment.c_last), (unsigned char *)m_txn.Payment.c_last);

                    if (dbrpcexec(m_dbproc) == FAIL)
                            ThrowError(CDBLIBERR::eDbRpcExec);

                    if (dbresults(m_dbproc) != SUCCEED)
                            ThrowError(CDBLIBERR::eDbResults);

                    if (dbnextrow(m_dbproc) != REG_ROW)
                            ThrowError(CDBLIBERR::eDbNextRow);

                    if (dbnumcols(m_dbproc) != 27)
                            ThrowError(CDBLIBERR::eWrongNumCols);

                    if (pData=dbdata(m_dbproc, 1))
                            m_txn.Payment.c_id = *((DBINT *) pData);
                    if (pData=dbdata(m_dbproc, 2))
                            UtilStrCpy(m_txn.Payment.c_last, pData,
dbdatlen(m_dbproc, 2));
                    if (pData=dbdata(m_dbproc, 3))
                    {
                            datetime = *((DBDATETIME *) pData);
                            dbdatecrack(m_dbproc, &daterec, &datetime);
                            m_txn.Payment.h_date.year   = daterec.year;
                            m_txn.Payment.h_date.month  =
daterec.month;
                            m_txn.Payment.h_date.day    = daterec.day;
                            m_txn.Payment.h_date.hour   = daterec.hour;
                            m_txn.Payment.h_date.minute =
daterec.minute;
                            m_txn.Payment.h_date.second =
daterec.second;
                    }
                    if (pData=dbdata(m_dbproc, 4))
                            UtilStrCpy(m_txn.Payment.w_street_1, pData,
dbdatlen(m_dbproc, 4));
                    if (pData=dbdata(m_dbproc, 5))
                            UtilStrCpy(m_txn.Payment.w_street_2, pData,
dbdatlen(m_dbproc, 5));
                    if (pData=dbdata(m_dbproc, 6))
                            UtilStrCpy(m_txn.Payment.w_city, pData,
dbdatlen(m_dbproc, 6));
                    if (pData=dbdata(m_dbproc, 7))
                            UtilStrCpy(m_txn.Payment.w_state, pData,
dbdatlen(m_dbproc, 7));
                    if (pData=dbdata(m_dbproc, 8))
                            UtilStrCpy(m_txn.Payment.w_zip, pData,
dbdatlen(m_dbproc, 8));
                    if (pData=dbdata(m_dbproc, 9))
                            UtilStrCpy(m_txn.Payment.d_street_1, pData,
dbdatlen(m_dbproc, 9));
                    if (pData=dbdata(m_dbproc, 10))
                            UtilStrCpy(m_txn.Payment.d_street_2, pData,
dbdatlen(m_dbproc, 10));
                    if (pData=dbdata(m_dbproc, 11))
                            UtilStrCpy(m_txn.Payment.d_city, pData,
dbdatlen(m_dbproc, 11));
                    if (pData=dbdata(m_dbproc, 12))
                            UtilStrCpy(m_txn.Payment.d_state, pData,
dbdatlen(m_dbproc, 12));
                    if (pData=dbdata(m_dbproc, 13))
                            UtilStrCpy(m_txn.Payment.d_zip, pData,
dbdatlen(m_dbproc, 13));
                    if (pData=dbdata(m_dbproc, 14))
                            UtilStrCpy(m_txn.Payment.c_first, pData,
dbdatlen(m_dbproc, 14));
                    if (pData=dbdata(m_dbproc, 15))
                            UtilStrCpy(m_txn.Payment.c_middle, pData,
dbdatlen(m_dbproc, 15));
                    if (pData=dbdata(m_dbproc, 16))
                            UtilStrCpy(m_txn.Payment.c_street_1, pData,
dbdatlen(m_dbproc, 16));
                    if (pData=dbdata(m_dbproc, 17))
                            UtilStrCpy(m_txn.Payment.c_street_2, pData,
dbdatlen(m_dbproc, 17));
                    if (pData=dbdata(m_dbproc, 18))
                            UtilStrCpy(m_txn.Payment.c_city, pData,
dbdatlen(m_dbproc, 18));
                    if (pData=dbdata(m_dbproc, 19))
                            UtilStrCpy(m_txn.Payment.c_state, pData,
dbdatlen(m_dbproc, 19));
                    if (pData=dbdata(m_dbproc, 20))
                            UtilStrCpy(m_txn.Payment.c_zip, pData,
dbdatlen(m_dbproc, 20));
                    if (pData=dbdata(m_dbproc, 21))
                            UtilStrCpy(m_txn.Payment.c_phone, pData,
dbdatlen(m_dbproc, 21));
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                        if (pData=dbdata(m_dbproc, 22))
                        {
                                datetime = *((DBDATETIME *) pData);
                                dbdatecrack(m_dbproc, &daterec, &datetime);
                                m_txn.Payment.c_since.year   =
daterec.year;
                                m_txn.Payment.c_since.month  =
daterec.month;
                                m_txn.Payment.c_since.day    = daterec.day;
                                m_txn.Payment.c_since.hour   =
daterec.hour;
                                m_txn.Payment.c_since.minute =
daterec.minute;
                                m_txn.Payment.c_since.second =
daterec.second;
                        }
                        if(pData=dbdata(m_dbproc, 23))
                                UtilStrCpy(m_txn.Payment.c_credit, pData,
dbdatlen(m_dbproc, 23));
                        if(pData=dbdata(m_dbproc, 24))
                                dbconvert(m_dbproc, SQLNUMERIC, pData,
dbdatlen(m_dbproc,24), SQLFLT8, (BYTE *)&m_txn.Payment.c_credit_lim, 8);
                        if(pData=dbdata(m_dbproc, 25))
                                dbconvert(m_dbproc, SQLNUMERIC, pData,
dbdatlen(m_dbproc,25), SQLFLT8, (BYTE *)&m_txn.Payment.c_discount, 8);
                        if(pData=dbdata(m_dbproc, 26))
                                dbconvert(m_dbproc, SQLNUMERIC, pData,
dbdatlen(m_dbproc,26), SQLFLT8, (BYTE *)&m_txn.Payment.c_balance, 8);
                        if(pData=dbdata(m_dbproc, 27))
                                UtilStrCpy(m_txn.Payment.c_data, pData,
dbdatlen(m_dbproc, 27));

                        DiscardNextRows(0);
                        DiscardNextResults(0);


                        if (m_txn.Payment.c_id == 0)
                                throw new CTPCC_DBLIB_ERR(
CTPCC_DBLIB_ERR::ERR_INVALID_CUST );
                        else
                                m_txn.Payment.exec_status_code = eOK;

                        return;
                }
                catch (CSQLERR *e)
                {
                        if ((e->m_msgno != 1205) || (++iTryCount >
iMaxRetries))
                                throw;

                        // hit deadlock; backoff for increasingly longer
period
                        delete e;
                        Sleep(10 * iTryCount);
                }
        }       // while (TRUE)
}


void CTPCC_DBLIB::OrderStatus()
{
        int                     i;
        DBDATETIME      datetime;
        DBDATEREC       daterec;

        int                     iTryCount = 0;
        RETCODE         rc;
        const BYTE      *pData;

        ResetError();

        while (TRUE)
        {
                try
                {
                        dbrpcinit(m_dbproc, "tpcc_orderstatus", 0);

                        dbrpcparam(m_dbproc, NULL, 0, SQLINT2, -1, -1,
(BYTE *) &m_txn.OrderStatus.w_id);
                        dbrpcparam(m_dbproc, NULL, 0, SQLINT1, -1, -1,
(BYTE *) &m_txn.OrderStatus.d_id);
                        dbrpcparam(m_dbproc, NULL, 0, SQLINT4, -1, -1,
(BYTE *) &m_txn.OrderStatus.c_id);

                        // if customer id is zero, then order status is by
name
                        if (m_txn.OrderStatus.c_id == 0)
                                dbrpcparam(m_dbproc, NULL, 0, SQLCHAR, -1,
strlen(m_txn.OrderStatus.c_last), (unsigned char
*)m_txn.OrderStatus.c_last);


                        if (dbrpcexec(m_dbproc) == FAIL)
                                ThrowError(CDBLIBERR::eDbRpcExec);

                        // Get order lines
                        if (dbresults(m_dbproc) != SUCCEED)
                        {
                                if ((m_DbLibErr == NULL) && (m_SqlErr ==
NULL))
                                        throw new CTPCC_DBLIB_ERR(
CTPCC_DBLIB_ERR::ERR_NO_SUCH_ORDER );
                                else
                                        ThrowError(CDBLIBERR::eDbResults);
                        }
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                        if (dbnumcols(m_dbproc) != 5)
                                ThrowError(CDBLIBERR::eWrongNumCols);

                        i = 0;
                        while (TRUE)
                        {
                                rc = dbnextrow(m_dbproc);
                                if (rc == NO_MORE_ROWS)
                                        break;
                                if (rc != REG_ROW)
                                        ThrowError(CDBLIBERR::eDbNextRow);

                                if(pData=dbdata(m_dbproc, 1))
        m_txn.OrderStatus.OL[i].ol_supply_w_id = (*(DBSMALLINT *) pData);
                                if(pData=dbdata(m_dbproc, 2))
                                        m_txn.OrderStatus.OL[i].ol_i_id =
(*(DBINT *) pData);
                                if(pData=dbdata(m_dbproc, 3))
                                        m_txn.OrderStatus.OL[i].ol_quantity
= (*(DBSMALLINT *) pData);
                                if(pData=dbdata(m_dbproc, 4))
                                        dbconvert(m_dbproc, SQLNUMERIC,
pData, dbdatlen(m_dbproc,4),
                                                SQLFLT8, (BYTE
*)&m_txn.OrderStatus.OL[i].ol_amount, 8);
                                if(pData=dbdata(m_dbproc, 5))
                                {
                                        datetime = *((DBDATETIME *) pData);
                                        dbdatecrack(m_dbproc, &daterec,
&datetime);

        m_txn.OrderStatus.OL[i].ol_delivery_d.year   = daterec.year;

        m_txn.OrderStatus.OL[i].ol_delivery_d.month  = daterec.month;

        m_txn.OrderStatus.OL[i].ol_delivery_d.day    = daterec.day;

        m_txn.OrderStatus.OL[i].ol_delivery_d.hour   = daterec.hour;

        m_txn.OrderStatus.OL[i].ol_delivery_d.minute = daterec.minute;

        m_txn.OrderStatus.OL[i].ol_delivery_d.second = daterec.second;
                                }
                                i++;
                        }
                        m_txn.OrderStatus.o_ol_cnt = i;


                        if (dbresults(m_dbproc) != SUCCEED)
                                ThrowError(CDBLIBERR::eDbResults);

                        if (dbnextrow(m_dbproc) != REG_ROW)
                                ThrowError(CDBLIBERR::eDbNextRow);

                        if (dbnumcols(m_dbproc) != 8)
                                ThrowError(CDBLIBERR::eWrongNumCols);

                        if(pData=dbdata(m_dbproc, 1))
                                m_txn.OrderStatus.c_id = (*(DBINT *)
pData);
                        if(pData=dbdata(m_dbproc, 2))
                                UtilStrCpy(m_txn.OrderStatus.c_last, pData,
dbdatlen(m_dbproc,2));
                        if(pData=dbdata(m_dbproc, 3))
                                UtilStrCpy(m_txn.OrderStatus.c_first,
pData, dbdatlen(m_dbproc,3));
                        if(pData=dbdata(m_dbproc, 4))
                                UtilStrCpy(m_txn.OrderStatus.c_middle,
pData, dbdatlen(m_dbproc, 4));
                        if(pData=dbdata(m_dbproc, 5))
                        {
                                datetime = *((DBDATETIME *) pData);
                                dbdatecrack(m_dbproc, &daterec, &datetime);
                                m_txn.OrderStatus.o_entry_d.year   =
daterec.year;
                                m_txn.OrderStatus.o_entry_d.month  =
daterec.month;
                                m_txn.OrderStatus.o_entry_d.day    =
daterec.day;
                                m_txn.OrderStatus.o_entry_d.hour   =
daterec.hour;
                                m_txn.OrderStatus.o_entry_d.minute =
daterec.minute;
                                m_txn.OrderStatus.o_entry_d.second =
daterec.second;
                        }
                        if(pData=dbdata(m_dbproc, 6))
                                m_txn.OrderStatus.o_carrier_id =
(*(DBSMALLINT *) pData);
                        if(pData=dbdata(m_dbproc, 7))
                                dbconvert(m_dbproc, SQLNUMERIC, pData,
dbdatlen(m_dbproc,7),
                                                SQLFLT8, (BYTE
*)&m_txn.OrderStatus.c_balance, 8);
                        if(pData=dbdata(m_dbproc, 8))
                                m_txn.OrderStatus.o_id = (*(DBINT *)
pData);

                        DiscardNextRows(0);
                        DiscardNextResults(0);


                        if (m_txn.OrderStatus.o_ol_cnt == 0)
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                                throw new CTPCC_DBLIB_ERR(
CTPCC_DBLIB_ERR::ERR_NO_SUCH_ORDER );
                        else if (m_txn.OrderStatus.c_id == 0 &&
m_txn.OrderStatus.c_last[0] == 0)
                                throw new CTPCC_DBLIB_ERR(
CTPCC_DBLIB_ERR::ERR_INVALID_CUST );
                        else
                                m_txn.OrderStatus.exec_status_code = eOK;

                        return;
                }
                catch (CSQLERR *e)
                {
                        if ((e->m_msgno != 1205) || (++iTryCount >
iMaxRetries))
                                throw;

                        // hit deadlock; backoff for increasingly longer
period
                        delete e;
                        Sleep(10 * iTryCount);
                }
        }       // while (TRUE)
}


void CTPCC_DBLIB::Delivery()
{
        int                     i;
        int                     iTryCount = 0;
        const BYTE      *pData;

        ResetError();

        while (TRUE)
        {
                try
                {
                        dbrpcinit(m_dbproc, "tpcc_delivery", 0);

                        dbrpcparam(m_dbproc, NULL, 0, SQLINT2, -1, -1,
(BYTE *) &m_txn.Delivery.w_id);
                        dbrpcparam(m_dbproc, NULL, 0, SQLINT1, -1, -1,
(BYTE *) &m_txn.Delivery.o_carrier_id);

                        if (dbrpcexec(m_dbproc) == FAIL)
                                ThrowError(CDBLIBERR::eDbRpcExec);

                        if (dbresults(m_dbproc) != SUCCEED)
                                ThrowError(CDBLIBERR::eDbResults);

                        if (dbnextrow(m_dbproc) != REG_ROW)
                                ThrowError(CDBLIBERR::eDbNextRow);

                        if (dbnumcols(m_dbproc) != 10)
                                ThrowError(CDBLIBERR::eWrongNumCols);

                        for (i=0; i<10; i++)
                        {
                                if (pData = dbdata(m_dbproc, i+1))
                                        m_txn.Delivery.o_id[i] = *((DBINT
*)pData);
                        }

                        DiscardNextRows(0);
                        DiscardNextResults(0);

                        m_txn.Delivery.exec_status_code = eOK;
                        return;
                }
                catch (CSQLERR *e)
                {
                        if ((e->m_msgno != 1205) || (++iTryCount >
iMaxRetries))
                                throw;

                        // hit deadlock; backoff for increasingly longer
period
                        delete e;
                        Sleep(10 * iTryCount);
                }
        }       // while (TRUE)
}

void CTPCC_DBLIB::ResetError()
{
        if (m_DbLibErr != NULL)
        {
                delete m_DbLibErr;
                m_DbLibErr = (CDBLIBERR*)NULL;
        }

        if (m_SqlErr != NULL)
        {
                delete m_SqlErr;
                m_SqlErr = (CSQLERR*)NULL;
        }
        return;
}

/*      FILE:           TPCC_DBLIB.H
 *                              Microsoft TPC-C Kit Ver. 4.20.000
 *                              Copyright Microsoft, 1999
 *              All Rights Reserved
 *
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
 *                           Version 4.10.000 audited by Richard Gimarc,
Performance Metrics, 3/17/99
 *
 *      PURPOSE:       Header file for TPC-C txn class implementation.
 *
 *  Change history:
 *             4.20.000 - updated rev number to match kit
 */
#pragma once

#ifndef PDBPROCESS
#define DBPROCESS void   // dbprocess structure type
typedef DBPROCESS * PDBPROCESS;
#endif

// need to declare functions for import, unless define has already been
created
// by the DLL's .cpp module for export.
#ifndef DllDecl
#define DllDecl  __declspec( dllimport )
#endif

class CSQLERR : public CBaseErr
{
       public:

              CSQLERR(void)
              {
                     m_msgno = 0;
                     m_msgstate = 0;
                     m_severity = 0;
                     m_msgtext = NULL;
              };

              ~CSQLERR()
              {
                     delete [] m_msgtext;
              };

              int            m_msgno;
              int            m_msgstate;
              int            m_severity;
              char   *m_msgtext;

              int ErrorType() {return ERR_TYPE_SQL;};
              int ErrorNum() {return m_msgno;};
              char *ErrorText() {return m_msgtext;};

};

class CDBLIBERR : public CBaseErr
{
       public:

              enum ACTION
              {
                     eNone,
                     eUnknown,
                     eLogin,                          // error from dblogin
                     eDbOpen,                         // error from dbopen
                     eDbUse,                          // error from dbuse
                     eDbSqlExec,                      // error from
dbsqlexec
                     eDbSet,                          // error from one of
the dbset* routines
                     eDbNextRow,                      // error from
dbnextrow
                     eWrongRowCount,                  // more or less rows
returned than expected
                     eWrongNumCols,         // more or less columns
returned than expected
                     eDbResults,                      // error from
dbresults
                     eDbRpcExec,                      // error from
dbrpcexec
                     eDbSetMaxProcs,                  // error from
dbsetmaxprocs
                     eDbProcHandler         // error from either
dbprocerrhandle or dbprocmsghandle
              };

              CDBLIBERR(ACTION eAction, int severity = 0, int dberror =
0, int oserr = 0)
              {
                     m_eAction = eAction;
                     m_severity = severity;
                     m_dberror = dberror;
                     m_oserr = oserr;

                     m_dberrstr = NULL;
                     m_oserrstr = NULL;
              };

              ~CDBLIBERR()
              {
                     delete [] m_dberrstr;
                     delete [] m_oserrstr;
              };

              ACTION  m_eAction;
              int            m_severity;
              int            m_dberror;
              int            m_oserr;
              char   *m_dberrstr;
              char   *m_oserrstr;

              int ErrorType() {return ERR_TYPE_DBLIB;};
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
		int ErrorNum() {return m_dberror;};
		char *ErrorText() {return m_dberrstr;};

};

class CTPCC_DBLIB_ERR : public CBaseErr
{
	public:
		enum CTPCC_DBLIB_ERRS
		{
			ERR_WRONG_SP_VERSION = 1,	// "Wrong version of
stored procs on database server"
			ERR_INVALID_CUST,			// "Invalid
Customer id,name."
			ERR_NO_SUCH_ORDER			// "No orders
found for customer."
		};

		CTPCC_DBLIB_ERR( int iErr ) { m_errno = iErr; };

		int		m_errno;

		int ErrorType() {return ERR_TYPE_TPCC_DBLIB;};
		int ErrorNum() {return m_errno;};

		char *ErrorText();
};

class DllDecl CTPCC_DBLIB : public CTPCC_BASE
{
	private:
		// declare variables and private functions here...
		PDBPROCESS	m_dbproc;
		CDBLIBERR	*m_DbLibErr;		// not allocated
until needed (maybe never)
		CSQLERR		*m_SqlErr;			// not
allocated until needed (maybe never)
		int			m_MaxRetries;		// retry
count on deadlock

		void DiscardNextRows(int iExpectedCount);
		void DiscardNextResults(int iExpectedCount);
		void ThrowError( CDBLIBERR::ACTION eAction );
		void ResetError();

		union
		{
			NEW_ORDER_DATA		NewOrder;
			PAYMENT_DATA		Payment;
			DELIVERY_DATA		Delivery;
			STOCK_LEVEL_DATA	StockLevel;
			ORDER_STATUS_DATA	OrderStatus;
		}			m_txn;

	public:
		CTPCC_DBLIB(LPCSTR szServer, LPCSTR szUser, LPCSTR
szPassword, LPCSTR szHost, LPCSTR szDatabase );
		~CTPCC_DBLIB(void);

		inline PNEW_ORDER_DATA		BuffAddr_NewOrder()
	{ return &m_txn.NewOrder;	};
		inline PPAYMENT_DATA		BuffAddr_Payment()
	{ return &m_txn.Payment;	};
		inline PDELIVERY_DATA		BuffAddr_Delivery()
	{ return &m_txn.Delivery;	};
		inline PSTOCK_LEVEL_DATA	BuffAddr_StockLevel()	{
return &m_txn.StockLevel;	};
		inline PORDER_STATUS_DATA	BuffAddr_OrderStatus()	{
return &m_txn.OrderStatus; };

		void NewOrder		();
		void Payment		();
		void Delivery		();
		void StockLevel			();
		void OrderStatus	();


		// these are public because they must be called from the
dblib err_handler and msg_hangler
		// outside of the class
		void SetDbLibError(int severity, int dberr, int oserr,
LPCSTR dberrstr, LPCSTR oserrstr);
		void SetSqlError( int msgno, int msgstate, int severity,
LPCSTR msgtext );

};

extern "C" DllDecl CTPCC_DBLIB* CTPCC_DBLIB_new
	( LPCSTR szServer, LPCSTR szUser, LPCSTR szPassword, LPCSTR
szHost, LPCSTR szDatabase );

typedef CTPCC_DBLIB* (TYPE_CTPCC_DBLIB)(LPCSTR, LPCSTR, LPCSTR, LPCSTR,
LPCSTR);

/*	FILE:		TPCC_COM.CPP
 *				Microsoft TPC-C Kit Ver. 4.20.000
 *				Copyright Microsoft, 1999
 *		All Rights Reserved
 *
 *				not yet audited
 *
 *	PURPOSE:	Source file for TPC-C COM+ class implementation.
 *	Contact:	Charles Levine (clevine@microsoft.com)
 *
 *  Change history:
 *		4.20.000 - first version
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
 */

// needed for CoinitializeEx
#define _WIN32_WINNT 0x0400

#include <windows.h>

// need to declare functions for export
#define DllDecl  __declspec( dllexport )

#include "..\..\common\src\trans.h"          //tpckit transaction header
contains definations of structures specific to TPC-C
#include "..\..\common\src\error.h"
#include "..\..\common\src\txn_base.h"
#include "tpcc_com.h"

#include "..\..\tpcc_com_ps\src\tpcc_com_ps_i.c"
#include "..\..\tpcc_com_all\src\tpcc_com_all_i.c"

// wrapper routine for class constructor
__declspec(dllexport) CTPCC_COM* CTPCC_COM_new(BOOL bSinglePool)
{
        return new CTPCC_COM(bSinglePool);
}

CTPCC_COM::CTPCC_COM(BOOL bSinglePool)
{
        HRESULT hr = NULL;
        long lRet = 0;

        m_bSinglePool = bSinglePool;

        m_pNewOrder            = NULL;
        m_pPayment             = NULL;
        m_pStockLevel  = NULL;
        m_pOrderStatus = NULL;

        m_pTxn = (COM_DATA*)CoTaskMemAlloc(sizeof(COM_DATA));
        if (!m_pTxn)
                throw new CCOMERR( E_FAIL );

        hr = CoInitializeEx(NULL, COINIT_MULTITHREADED);
        if (FAILED(hr))
        {
                throw new CCOMERR( hr );
        }

        // create components
        if (m_bSinglePool)
        {
                hr = CoCreateInstance(CLSID_TPCC, NULL, CLSCTX_SERVER,
IID_ITPCC, (void **)&m_pNewOrder);
                if (FAILED(hr))
                        throw new CCOMERR(hr);

                // all txns will use same component
                m_pPayment = m_pNewOrder;
                m_pStockLevel = m_pNewOrder;
                m_pOrderStatus = m_pNewOrder;
        }
        else
        {
                // use different components for each txn

                hr = CoCreateInstance(CLSID_NewOrder, NULL, CLSCTX_SERVER,
IID_ITPCC, (void **)&m_pNewOrder);
                if (FAILED(hr))
                        throw new CCOMERR(hr);

                hr = CoCreateInstance(CLSID_Payment, NULL, CLSCTX_SERVER,
IID_ITPCC, (void **)&m_pPayment);
                if (FAILED(hr))
                        throw new CCOMERR(hr);

                hr = CoCreateInstance(CLSID_StockLevel, NULL,
CLSCTX_SERVER, IID_ITPCC, (void **)&m_pStockLevel);
                if (FAILED(hr))
                        throw new CCOMERR(hr);

                hr = CoCreateInstance(CLSID_OrderStatus, NULL,
CLSCTX_SERVER, IID_ITPCC, (void **)&m_pOrderStatus);
                if (FAILED(hr))
                        throw new CCOMERR(hr);
        }

        // call setcomplete to release each component back into pool
        hr = m_pNewOrder->CallSetComplete();
        if (FAILED(hr))
                throw new CCOMERR(hr);

        if (!m_bSinglePool)
        {
                hr = m_pPayment->CallSetComplete();
                if (FAILED(hr))
                        throw new CCOMERR(hr);

                hr = m_pStockLevel->CallSetComplete();
                if (FAILED(hr))
                        throw new CCOMERR(hr);

                hr = m_pOrderStatus->CallSetComplete();
                if (FAILED(hr))
                        throw new CCOMERR(hr);
        }
}
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
CTPCC_COM::~CTPCC_COM()
{
	if (m_pTxn)
		CoTaskMemFree(m_pTxn);

	ReleaseInterface(m_pNewOrder);
	if (!m_bSinglePool)
	{
		ReleaseInterface(m_pPayment);
		ReleaseInterface(m_pStockLevel);
		ReleaseInterface(m_pOrderStatus);
	}
	CoUninitialize();
}

void CTPCC_COM::NewOrder()
{
	int iSize = sizeof(COM_DATA);

	HRESULT hr = m_pNewOrder->NewOrder(&iSize, (unsigned
char**)&m_pTxn);
	if (FAILED(hr))
		throw new CCOMERR( hr );

	if ( m_pTxn->ErrorType != ERR_SUCCESS )
		throw new CCOMERR( m_pTxn->ErrorType, m_pTxn->error );
}

void CTPCC_COM::Payment()
{
	int iSize = sizeof(COM_DATA);

	HRESULT hr = m_pPayment->Payment(&iSize, (unsigned
char**)&m_pTxn);
	if (FAILED(hr))
		throw new CCOMERR( hr );

	if ( m_pTxn->ErrorType != ERR_SUCCESS )
		throw new CCOMERR( m_pTxn->ErrorType, m_pTxn->error );
}

void CTPCC_COM::StockLevel()
{
	int iSize = sizeof(COM_DATA);

	HRESULT hr = m_pStockLevel->StockLevel(&iSize, (unsigned
char**)&m_pTxn);
	if (FAILED(hr))
		throw new CCOMERR( hr );

	if ( m_pTxn->ErrorType != ERR_SUCCESS )
		throw new CCOMERR( m_pTxn->ErrorType, m_pTxn->error );
}

void CTPCC_COM::OrderStatus()
{
	int iSize = sizeof(COM_DATA);

	HRESULT hr = m_pOrderStatus->OrderStatus(&iSize, (unsigned
char**)&m_pTxn);
	if (FAILED(hr))
		throw new CCOMERR( hr );

	if ( m_pTxn->ErrorType != ERR_SUCCESS )
		throw new CCOMERR( m_pTxn->ErrorType, m_pTxn->error );
}

/*	FILE:		TPCC_COM.H
 *				Microsoft TPC-C Kit Ver. 4.20.000
 *				Copyright Microsoft, 1999
 *		All Rights Reserved
 *
 *				not yet audited
 *
 *	PURPOSE:	Header file for TPC-C COM+ class implementation.
 *
 *  Change history:
 *		4.20.000 - first version
 */

#pragma once

#include <stdio.h>
#include "..\..\tpcc_com_ps\src\tpcc_com_ps.h"

// need to declare functions for import, unless define has already been
created
// by the DLL's .cpp module for export.
#ifndef DllDecl
#define DllDecl  __declspec( dllimport )
#endif

class CCOMERR : public CBaseErr
{
	private:
		char m_szErrorText[64];

	public:
		// use this interface for genuine COM errors
		CCOMERR( HRESULT hr )
		{
			m_hr = hr;
			m_iErrorType = 0;
			m_iError = 0;
		}
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                // use this interface to impersonate a non-COM error type
                CCOMERR( int iErrorType, int iError )
                {
                        m_iErrorType = iErrorType;
                        m_iError = iError;
                        m_hr = S_OK;
                }

                int             m_hr;
                int             m_iErrorType;
                int             m_iError;

                // A CCOMERR class can impersonate another class, which
happens if the error
                // was not actually a COM Services error, but was simply
transmitted back via COM.
                int ErrorType()
                {
                        if (m_iErrorType == 0)
                                return ERR_TYPE_COM;
                        else
                                return m_iErrorType;
                }

                int ErrorNum() {return m_hr;}

                char *ErrorText()
                {
                        if (m_hr == S_OK)
                                sprintf( m_szErrorText, "Error: Class %d,
error # %d", m_iErrorType, m_iError );
                        else
                                sprintf( m_szErrorText, "Error: COM HRESULT
%x", m_hr );
                        return m_szErrorText;
                }
};

class DllDecl CTPCC_COM : public CTPCC_BASE
{
        private:
                BOOL m_bSinglePool;

                // COM Interface pointers
                ITPCC*                  m_pNewOrder;
                ITPCC*                  m_pPayment;
                ITPCC*                  m_pStockLevel;
                ITPCC*                  m_pOrderStatus;

                struct COM_DATA
                {
                        int ErrorType;
                        int error;
                        union
                        {
                                NEW_ORDER_DATA          NewOrder;
                                PAYMENT_DATA            Payment;
                                DELIVERY_DATA           Delivery;
                                STOCK_LEVEL_DATA        StockLevel;
                                ORDER_STATUS_DATA       OrderStatus;
                        } u;
                } *m_pTxn;

        public:
                CTPCC_COM(BOOL bSinglePool);
                ~CTPCC_COM(void);

                inline PNEW_ORDER_DATA          BuffAddr_NewOrder()
        { return &m_pTxn->u.NewOrder;    };
                inline PPAYMENT_DATA            BuffAddr_Payment()
        { return &m_pTxn->u.Payment;     };
                inline PDELIVERY_DATA           BuffAddr_Delivery()
        { return &m_pTxn->u.Delivery;    };
                inline PSTOCK_LEVEL_DATA        BuffAddr_StockLevel()  {
return &m_pTxn->u.StockLevel;  };
                inline PORDER_STATUS_DATA       BuffAddr_OrderStatus() {
return &m_pTxn->u.OrderStatus; };

                void NewOrder           ();
                void Payment            ();
                void StockLevel                 ();
                void OrderStatus        ();
                void Delivery           () { throw new CCOMERR(E_NOTIMPL); }
// not supported
};


inline void ReleaseInterface(IUnknown *pUnk)
{
        if (pUnk)
        {
                pUnk->Release();
                pUnk = NULL;
        }
}


// wrapper routine for class constructor
extern "C" __declspec(dllexport) CTPCC_COM* CTPCC_COM_new(BOOL);

typedef CTPCC_COM* (TYPE_CTPCC_COM)(BOOL);

/*      FILE:           METHODS.H
 *                              Microsoft TPC-C Kit Ver. 4.20.000
 *                              Copyright Microsoft, 1999
 *              All Rights Reserved
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
 *
 *                              not yet audited
 *
 *      PURPOSE:        Header file for COM components.
 *
 *  Change history:
 *              4.20.000 - first version
 */


enum COMPONENT_ERROR
{
        ERR_MISSING_REGISTRY_ENTRIES = 1,
        ERR_LOADDLL_FAILED,
        ERR_GETPROCADDR_FAILED,
        ERR_UNKNOWN_DB_PROTOCOL
};

class CCOMPONENT_ERR : public CBaseErr
{
        public:
                CCOMPONENT_ERR(COMPONENT_ERROR Err)
                {
                        m_Error = Err;
                        m_szTextDetail = NULL;
                        m_SystemErr = 0;
                        m_szErrorText = NULL;
                };

                CCOMPONENT_ERR(COMPONENT_ERROR Err, char *szTextDetail,
DWORD dwSystemErr)
                {
                        m_Error = Err;
                        m_szTextDetail = new char[strlen(szTextDetail)+1];
                        strcpy( m_szTextDetail, szTextDetail );
                        m_SystemErr = dwSystemErr;
                        m_szErrorText = NULL;
                };

                ~CCOMPONENT_ERR()
                {
                        if (m_szTextDetail != NULL)
                                delete [] m_szTextDetail;
                        if (m_szErrorText != NULL)
                                delete [] m_szErrorText;
                };

                COMPONENT_ERROR         m_Error;
                char                    *m_szTextDetail;
                char                    *m_szErrorText;
                DWORD                   m_SystemErr;

                int ErrorType() {return ERR_TYPE_COMPONENT;};
                int ErrorNum() {return m_Error;};
                char *ErrorText();
};


static void WriteMessageToEventLog(LPTSTR lpszMsg);


/////////////////////////////////////////////////////////////////////////
////
// CTPCC_Common
class CTPCC_Common :
        public ITPCC,
        public IObjectControl,
        public IObjectConstruct,
        public CComObjectRootEx<CComSingleThreadModel>
{
public:
BEGIN_COM_MAP(CTPCC_Common)
        COM_INTERFACE_ENTRY(ITPCC)
        COM_INTERFACE_ENTRY(IObjectControl)
        COM_INTERFACE_ENTRY(IObjectConstruct)
END_COM_MAP()

        CTPCC_Common();
        ~CTPCC_Common();

// ITPCC
public:
        HRESULT __stdcall NewOrder(           int* iSize, UCHAR** txn);
        HRESULT __stdcall Payment(            int* iSize, UCHAR** txn);
        HRESULT __stdcall Delivery(           int* iSize, UCHAR** txn)
{return E_NOTIMPL;}
        HRESULT __stdcall StockLevel( int* iSize, UCHAR** txn);
        HRESULT __stdcall OrderStatus(        int* iSize, UCHAR** txn);

        HRESULT __stdcall CallSetComplete();

// IObjectControl
        STDMETHODIMP_(BOOL) CanBePooled() { return m_bCanBePooled; }
        STDMETHODIMP Activate() { return S_OK; }     // we don't support
COM Services transactions (no enlistment)
        STDMETHODIMP_(void) Deactivate() { /* nothing to do */ }

// IObjectConstruct
        STDMETHODIMP Construct(IDispatch * pUnk);

        // helper methods
private:
        BOOL            m_bCanBePooled;
        CTPCC_BASE      *m_pTxn;

        struct COM_DATA
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
	{
		int retval;
		int error;
		union
		{
			NEW_ORDER_DATA		NewOrder;
			PAYMENT_DATA		Payment;
			DELIVERY_DATA		Delivery;
			STOCK_LEVEL_DATA	StockLevel;
			ORDER_STATUS_DATA	OrderStatus;
		} u;
	};

};

/////////////////////////////////////////////////////////////////////////
////
// CTPCC
class CTPCC :
	public CTPCC_Common,
	public CComCoClass<CTPCC, &CLSID_TPCC>
{
public:
DECLARE_REGISTRY_RESOURCEID(IDR_TPCC)

BEGIN_COM_MAP(CTPCC)
	COM_INTERFACE_ENTRY2(IUnknown, CComObjectRootEx)
	COM_INTERFACE_ENTRY_CHAIN(CTPCC_Common)
END_COM_MAP()

};


/////////////////////////////////////////////////////////////////////////
////
// CNewOrder
class CNewOrder :
	public CTPCC_Common,
	public CComCoClass<CNewOrder, &CLSID_NewOrder>
{
public:
DECLARE_REGISTRY_RESOURCEID(IDR_NEWORDER)

BEGIN_COM_MAP(CNewOrder)
	COM_INTERFACE_ENTRY2(IUnknown, CComObjectRootEx)
	COM_INTERFACE_ENTRY_CHAIN(CTPCC_Common)
END_COM_MAP()

// ITPCC
public:
//	HRESULT __stdcall NewOrder(		int* iSize, UCHAR** txn)
{return E_NOTIMPL;}
	HRESULT __stdcall Payment(		int* iSize, UCHAR** txn)
{return E_NOTIMPL;}
	HRESULT __stdcall StockLevel( int* iSize, UCHAR** txn) {return
E_NOTIMPL;}
	HRESULT __stdcall OrderStatus(	int* iSize, UCHAR** txn)
{return E_NOTIMPL;}
};


	/////////////////////////////////////////////////////////////////////
///////////
// COrderStatus
class COrderStatus :
	public CTPCC_Common,
	public CComCoClass<COrderStatus, &CLSID_OrderStatus>
{
public:
DECLARE_REGISTRY_RESOURCEID(IDR_ORDERSTATUS)

BEGIN_COM_MAP(COrderStatus)
	COM_INTERFACE_ENTRY2(IUnknown, CComObjectRootEx)
	COM_INTERFACE_ENTRY_CHAIN(CTPCC_Common)
END_COM_MAP()

// ITPCC
public:
	HRESULT __stdcall NewOrder(		int* iSize, UCHAR** txn)
{return E_NOTIMPL;}
	HRESULT __stdcall Payment(		int* iSize, UCHAR** txn)
{return E_NOTIMPL;}
	HRESULT __stdcall StockLevel( int* iSize, UCHAR** txn) {return
E_NOTIMPL;}
//	HRESULT __stdcall OrderStatus(	int* iSize, UCHAR** txn)
{return E_NOTIMPL;}
};


/////////////////////////////////////////////////////////////////////////
////
// CPayment
class CPayment :
	public CTPCC_Common,
	public CComCoClass<CPayment, &CLSID_Payment>
{
public:
DECLARE_REGISTRY_RESOURCEID(IDR_PAYMENT)

BEGIN_COM_MAP(CPayment)
	COM_INTERFACE_ENTRY2(IUnknown, CComObjectRootEx)
	COM_INTERFACE_ENTRY_CHAIN(CTPCC_Common)
END_COM_MAP()

// ITPCC
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
public:
        HRESULT __stdcall NewOrder(           int* iSize, UCHAR** txn)
{return E_NOTIMPL;}
//      HRESULT __stdcall Payment(            int* iSize, UCHAR** txn)
{return E_NOTIMPL;}
        HRESULT __stdcall StockLevel( int* iSize, UCHAR** txn) {return
E_NOTIMPL;}
        HRESULT __stdcall OrderStatus(        int* iSize, UCHAR** txn)
{return E_NOTIMPL;}
};


/////////////////////////////////////////////////////////////////////////
////
// CStockLevel
class CStockLevel :
        public CTPCC_Common,
        public CComCoClass<CStockLevel, &CLSID_StockLevel>
{
public:
DECLARE_REGISTRY_RESOURCEID(IDR_STOCKLEVEL)

BEGIN_COM_MAP(CStockLevel)
        COM_INTERFACE_ENTRY2(IUnknown, CComObjectRootEx)
        COM_INTERFACE_ENTRY_CHAIN(CTPCC_Common)
END_COM_MAP()

// ITPCC
public:
        HRESULT __stdcall NewOrder(           int* iSize, UCHAR** txn)
{return E_NOTIMPL;}
        HRESULT __stdcall Payment(            int* iSize, UCHAR** txn)
{return E_NOTIMPL;}
//      HRESULT __stdcall StockLevel( int* iSize, UCHAR** txn) {return
E_NOTIMPL;}
        HRESULT __stdcall OrderStatus(        int* iSize, UCHAR** txn)
{return E_NOTIMPL;}
};


//{{NO_DEPENDENCIES}}
// Microsoft Developer Studio generated include file.
// Used by tpcc_com_all.rc
//
#define IDS_PROJNAME                    100
#define IDR_TPCC                        101
#define IDR_NEWORDER                    102
#define IDR_ORDERSTATUS                 103
#define IDR_PAYMENT                     104
#define IDR_STOCKLEVEL                  105

// Next default values for new objects
//
```

```
#ifdef APSTUDIO_INVOKED
#ifndef APSTUDIO_READONLY_SYMBOLS
#define _APS_NEXT_RESOURCE_VALUE        202
#define _APS_NEXT_COMMAND_VALUE         32768
#define _APS_NEXT_CONTROL_VALUE         201
#define _APS_NEXT_SYMED_VALUE           106
#endif
#endif

/*      FILE:           TPCC_COM_ALL.CPP
 *                              Microsoft TPC-C Kit Ver. 4.20.000
 *                              Copyright Microsoft, 1999
 *              All Rights Reserved
 *
 *                              Version 4.10.000 audited by Richard Gimarc,
Performance Metrics, 3/17/99
 *
 *      PURPOSE:        Implementation for TPC-C Tuxedo class.
 *      Contact:        Charles Levine (clevine@microsoft.com)
 *
 *  Change history:
 *              4.20.000 - updated rev number to match kit
 */

#define STRICT
#define _WIN32_WINNT 0x0400
#define _ATL_APARTMENT_THREADED

#include <stdio.h>
#include <atlbase.h>
//You may derive a class from CComModule and use it if you want to
override
//something, but do not change the name of _Module
extern CComModule _Module;

#include <atlcom.h>
#include <initguid.h>
#include <transact.h>
#include <atlimpl.cpp>
#include <comsvcs.h>

#include <sqltypes.h>
#include <sql.h>
#include <sqlext.h>

#include "tpcc_com_ps.h"
#include "..\..\common\src\trans.h"
        //tpckit transaction header contains definations of structures
specific to TPC-C
#include "..\..\common\src\txn_base.h"
#include "..\..\common\src\error.h"
#include "..\..\common\src\ReadRegistry.h"
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
#include "..\..\db_dblib_dll\src\tpcc_dblib.h"              // DBLIB
implementation of TPC-C txns
#include "..\..\db_odbc_dll\src\tpcc_odbc.h"         // ODBC
implementation of TPC-C txns


#include "resource.h"
#include "tpcc_com_all.h"
#include "tpcc_com_all_i.c"
#include "Methods.h"
#include "..\..\tpcc_com_ps\src\tpcc_com_ps_i.c"
#include "..\..\common\src\ReadRegistry.cpp"


CComModule _Module;

BEGIN_OBJECT_MAP(ObjectMap)
        OBJECT_ENTRY(CLSID_TPCC, CTPCC)
        OBJECT_ENTRY(CLSID_NewOrder, CNewOrder)
        OBJECT_ENTRY(CLSID_OrderStatus, COrderStatus)
        OBJECT_ENTRY(CLSID_Payment, CPayment)
        OBJECT_ENTRY(CLSID_StockLevel, CStockLevel)
END_OBJECT_MAP()


// configuration settings from registry
TPCCREGISTRYDATA       Reg;
char
        szMyComputerName[MAX_COMPUTERNAME_LENGTH+1];

static  HINSTANCE hLibInstanceDb = NULL;

TYPE_CTPCC_DBLIB       *pCTPCC_DBLIB_new;
TYPE_CTPCC_ODBC                *pCTPCC_ODBC_new;


////////////////////////////////////////////////////////////////////////////
////
// DLL Entry Point

extern "C"
BOOL WINAPI DllMain(HINSTANCE hInstance, DWORD dwReason, LPVOID
/*lpReserved*/)
{
        char szDllName[128];

        try
        {
                if (dwReason == DLL_PROCESS_ATTACH)
                {
                        _Module.Init(ObjectMap, hInstance);
                        DisableThreadLibraryCalls(hInstance);
                        DWORD dwSize = MAX_COMPUTERNAME_LENGTH+1;
                        GetComputerName(szMyComputerName, &dwSize);
                        szMyComputerName[dwSize] = 0;

                        if ( ReadTPCCRegistrySettings( &Reg ) )
                                throw new CCOMPONENT_ERR(
ERR_MISSING_REGISTRY_ENTRIES );

                        if (Reg.eDB_Protocol == DBLIB)
                        {
                                strcpy( szDllName, Reg.szPath );
                                strcat( szDllName, "tpcc_dblib.dll");
                                hLibInstanceDb = LoadLibrary( szDllName );
                                if (hLibInstanceDb == NULL)
                                        throw new CCOMPONENT_ERR(
ERR_LOADDLL_FAILED, szDllName, GetLastError() );

                                // get function pointer to wrapper for
class constructor
                                pCTPCC_DBLIB_new = (TYPE_CTPCC_DBLIB*)
GetProcAddress(hLibInstanceDb,"CTPCC_DBLIB_new");
                                if (pCTPCC_DBLIB_new == NULL)
                                        throw new CCOMPONENT_ERR(
ERR_GETPROCADDR_FAILED, szDllName, GetLastError() );
                        }
                        else if (Reg.eDB_Protocol == ODBC)
                        {
                                strcpy( szDllName, Reg.szPath );
                                strcat( szDllName, "tpcc_odbc.dll");
                                hLibInstanceDb = LoadLibrary( szDllName );
                                if (hLibInstanceDb == NULL)
                                        throw new CCOMPONENT_ERR(
ERR_LOADDLL_FAILED, szDllName, GetLastError() );

                                // get function pointer to wrapper for
class constructor
                                pCTPCC_ODBC_new = (TYPE_CTPCC_ODBC*)
GetProcAddress(hLibInstanceDb,"CTPCC_ODBC_new");
                                if (pCTPCC_ODBC_new == NULL)
                                        throw new CCOMPONENT_ERR(
ERR_GETPROCADDR_FAILED, szDllName, GetLastError() );
                        }
                        else
                                throw new CCOMPONENT_ERR(
ERR_UNKNOWN_DB_PROTOCOL );
                }
                else if (dwReason == DLL_PROCESS_DETACH)
                        _Module.Term();

        }
        catch (CBaseErr *e)
        {
                WriteMessageToEventLog(e->ErrorText());
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                delete e;
                return FALSE;
        }
        catch (...)
        {
                WriteMessageToEventLog(TEXT("Unhandled exception in object
DllMain"));
                return FALSE;
        }

        return TRUE;   // OK
}

////////////////////////////////////////////////////////////////////////////
////
// Used to determine whether the DLL can be unloaded by OLE

STDAPI DllCanUnloadNow(void)
{
        return (_Module.GetLockCount()==0) ? S_OK : S_FALSE;
}

////////////////////////////////////////////////////////////////////////////
////
// Returns a class factory to create an object of the requested type

STDAPI DllGetClassObject(REFCLSID rclsid, REFIID riid, LPVOID* ppv)
{
        return _Module.GetClassObject(rclsid, riid, ppv);
}

////////////////////////////////////////////////////////////////////////////
////
// DllRegisterServer - Adds entries to the system registry

STDAPI DllRegisterServer(void)
{
        // registers object, typelib and all interfaces in typelib
        return _Module.RegisterServer(TRUE);
}

////////////////////////////////////////////////////////////////////////////
////
// DllUnregisterServer - Removes entries from the system registry

STDAPI DllUnregisterServer(void)
{
        _Module.UnregisterServer();
        return S_OK;
}


static void WriteMessageToEventLog(LPTSTR lpszMsg)
{
    TCHAR   szMsg[256];
    HANDLE  hEventSource;
    LPTSTR  lpszStrings[2];

    // Use event logging to log the error.
    //
    hEventSource = RegisterEventSource(NULL, TEXT("tpcc_com_all.dll"));

    _stprintf(szMsg, TEXT("Error in COM+ TPC-C Component: "));
    lpszStrings[0] = szMsg;
    lpszStrings[1] = lpszMsg;

    if (hEventSource != NULL)
        {
        ReportEvent(hEventSource, // handle of event source
            EVENTLOG_ERROR_TYPE,  // event type
            0,                    // event category
            0,                    // event ID
            NULL,                 // current user's SID
            2,                    // strings in lpszStrings
            0,                    // no bytes of raw data
            (LPCTSTR *)lpszStrings,    // array of error strings
            NULL);                // no raw data

        (VOID) DeregisterEventSource(hEventSource);
    }
}

inline void ReleaseInterface(IUnknown *pUnk)
{
        if (pUnk)
        {
                pUnk->Release();
                pUnk = NULL;
        }
}

/* FUNCTION: CCOMPONENT_ERR::ErrorText
 *
 */

char* CCOMPONENT_ERR::ErrorText(void)
{
        static SERRORMSG errorMsgs[] =
        {
                { ERR_MISSING_REGISTRY_ENTRIES,       "Required entries
missing from registry."                               },
                { ERR_LOADDLL_FAILED,                 "Load of DLL failed.
DLL="                                                       },
                { ERR_GETPROCADDR_FAILED,             "Could not map proc
in DLL.  GetProcAddr error.  DLL="    },
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                { ERR_UNKNOWN_DB_PROTOCOL,              "Unknown database
protocol specified in registry."              },
                { 0,                                                    ""

                                        }
        };

        char szTmp[256];
        int i = 0;
        while (TRUE)
        {
                if (errorMsgs[i].szMsg[0] == 0)
                {
                        strcpy( szTmp, "Unknown error number." );
                        break;
                }
                if (m_Error == errorMsgs[i].iError)
                {
                        strcpy( szTmp, errorMsgs[i].szMsg );
                        break;
                }
                i++;
        }

        if (m_szTextDetail)
                strcat( szTmp, m_szTextDetail );
        if (m_SystemErr)
                wsprintf( szTmp+strlen(szTmp), "  Error=%d", m_SystemErr
);

        m_szErrorText = new char[strlen(szTmp)+1];
        strcpy( m_szErrorText, szTmp );
        return m_szErrorText;
}


CTPCC_Common::CTPCC_Common()
{
        m_pTxn = NULL;
        m_bCanBePooled = TRUE;
}

CTPCC_Common::~CTPCC_Common()
{
        if (m_pTxn)
                delete m_pTxn;
}

HRESULT CTPCC_Common::CallSetComplete()
{
        IObjectContext* pObjectContext = NULL;

        // get our object context
        HRESULT hr = CoGetObjectContext( IID_IObjectContext, (void
**)&pObjectContext );
        pObjectContext->SetComplete();
        ReleaseInterface(pObjectContext);
        return hr;
}

//
//      called by the ctor activator
//
STDMETHODIMP CTPCC_Common::Construct(IDispatch * pUnk)
{
        // Code to access construction string, if needed later...
        //      if (!pUnk)
        //              return E_UNEXPECTED;
        //      IObjectConstructString * pString = NULL;
        //      HRESULT hr = pUnk-
>QueryInterface(IID_IObjectConstructString, (void **)&pString);
        //      pString->Release();

        try
        {
                if (Reg.eDB_Protocol == ODBC)
                        m_pTxn = pCTPCC_ODBC_new( Reg.szDbServer,
Reg.szDbUser, Reg.szDbPassword, szMyComputerName, Reg.szDbName );
                else if (Reg.eDB_Protocol == DBLIB)
                        m_pTxn = pCTPCC_DBLIB_new( Reg.szDbServer,
Reg.szDbUser, Reg.szDbPassword, szMyComputerName, Reg.szDbName );
        }
        catch (CBaseErr *e)
        {
                WriteMessageToEventLog(e->ErrorText());
                delete e;
                return E_FAIL;
        }
        catch (...)
        {
                WriteMessageToEventLog(TEXT("Unhandled exception in object
::Construct"));
                return E_FAIL;
        }

        return S_OK;
}

HRESULT CTPCC_Common::NewOrder(int* iSize, UCHAR **txn)
{
        PNEW_ORDER_DATA         pNewOrder;
        COM_DATA                *pData;

        try
        {
                pData = (COM_DATA*)*txn;
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                pNewOrder = m_pTxn->BuffAddr_NewOrder();

                memcpy(pNewOrder, &pData->u.NewOrder,
sizeof(NEW_ORDER_DATA));
                m_pTxn->NewOrder();
                memcpy( &pData->u.NewOrder, pNewOrder,
sizeof(NEW_ORDER_DATA));

                pData->retval = ERR_SUCCESS;
                pData->error = 0;
                return S_OK;
        }
        catch (CBaseErr *e)
        {
                // check for lost database connection; if yes, component
is toast
                if ( ((e->ErrorType() == ERR_TYPE_DBLIB) && (e->ErrorNum()
== 10005)) ||
                        ((e->ErrorType() == ERR_TYPE_ODBC) && (e-
>ErrorNum() == 10054)) )
                       m_bCanBePooled = FALSE;

                pData->retval = e->ErrorType();
                pData->error = e->ErrorNum();
                delete e;
                return E_FAIL;
        }
        catch (...)
        {
                WriteMessageToEventLog(TEXT("Unhandled exception."));
                pData->retval = ERR_TYPE_LOGIC;
                pData->error = 0;
                m_bCanBePooled = FALSE;
                return E_FAIL;
        }
}

HRESULT CTPCC_Common::Payment(int* iSize, UCHAR** txn)
{
        PPAYMENT_DATA  pPayment;
        COM_DATA               *pData;

        try
        {
                pData = (COM_DATA*)*txn;
                pPayment = m_pTxn->BuffAddr_Payment();

                memcpy(pPayment, &pData->u.Payment, sizeof(PAYMENT_DATA)
);
                m_pTxn->Payment();
                memcpy( &pData->u.Payment, pPayment, sizeof(PAYMENT_DATA)
);
                pData->retval = ERR_SUCCESS;
                pData->error = 0;
                return S_OK;
        }
        catch (CBaseErr *e)
        {
                // check for lost database connection; if yes, component
is toast
                if ( ((e->ErrorType() == ERR_TYPE_DBLIB) && (e->ErrorNum()
== 10005)) ||
                        ((e->ErrorType() == ERR_TYPE_ODBC) && (e-
>ErrorNum() == 10054)) )
                       m_bCanBePooled = FALSE;

                pData->retval = e->ErrorType();
                pData->error = e->ErrorNum();
                delete e;
                return E_FAIL;
        }
        catch (...)
        {
                WriteMessageToEventLog(TEXT("Unhandled exception."));
                pData->retval = ERR_TYPE_LOGIC;
                pData->error = 0;
                m_bCanBePooled = FALSE;
                return E_FAIL;
        }
}

HRESULT CTPCC_Common::StockLevel(int* iSize, UCHAR** txn)
{
        PSTOCK_LEVEL_DATA      pStockLevel;
        COM_DATA                       *pData;

        try
        {
                pData = (COM_DATA*)*txn;
                pStockLevel = m_pTxn->BuffAddr_StockLevel();

                memcpy(pStockLevel, &pData->u.StockLevel,
sizeof(STOCK_LEVEL_DATA) );
                m_pTxn->StockLevel();
                memcpy( &pData->u.StockLevel, pStockLevel,
sizeof(STOCK_LEVEL_DATA) );

                pData->retval = ERR_SUCCESS;
                pData->error = 0;
                return S_OK;
        }
        catch (CBaseErr *e)
        {
                // check for lost database connection; if yes, component
is toast
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                if ( ((e->ErrorType() == ERR_TYPE_DBLIB) && (e->ErrorNum()
== 10005)) ||
                        ((e->ErrorType() == ERR_TYPE_ODBC) && (e-
>ErrorNum() == 10054)) )
                        m_bCanBePooled = FALSE;

                pData->retval = e->ErrorType();
                pData->error = e->ErrorNum();
                delete e;
                return E_FAIL;
        }
        catch (...)
        {
                WriteMessageToEventLog(TEXT("Unhandled exception."));
                pData->retval = ERR_TYPE_LOGIC;
                pData->error = 0;
                m_bCanBePooled = FALSE;
                return E_FAIL;
        }
}

HRESULT CTPCC_Common::OrderStatus(int* iSize, UCHAR** txn)
{
        PORDER_STATUS_DATA    pOrderStatus;
        COM_DATA                        *pData;

        try
        {
                pData = (COM_DATA*)*txn;
                pOrderStatus = m_pTxn->BuffAddr_OrderStatus();

                memcpy(pOrderStatus, &pData->u.OrderStatus,
sizeof(ORDER_STATUS_DATA) );
                m_pTxn->OrderStatus();
                memcpy( &pData->u.OrderStatus, pOrderStatus,
sizeof(ORDER_STATUS_DATA) );

                pData->retval = ERR_SUCCESS;
                pData->error = 0;
                return S_OK;
        }
        catch (CBaseErr *e)
        {
                // check for lost database connection; if yes, component
is toast
                if ( ((e->ErrorType() == ERR_TYPE_DBLIB) && (e->ErrorNum()
== 10005)) ||
                        ((e->ErrorType() == ERR_TYPE_ODBC) && (e-
>ErrorNum() == 10054)) )
                        m_bCanBePooled = FALSE;

                pData->retval = e->ErrorType();
                pData->error = e->ErrorNum();
                delete e;
                return E_FAIL;
        }
        catch (...)
        {
                WriteMessageToEventLog(TEXT("Unhandled exception."));
                pData->retval = ERR_TYPE_LOGIC;
                pData->error = 0;
                m_bCanBePooled = FALSE;
                return E_FAIL;
        }
}

; tpcc_com_all.def : Declares the module parameters.

LIBRARY      "tpcc_com_all.dll"

EXPORTS
        DllCanUnloadNow     @1 PRIVATE
        DllGetClassObject   @2 PRIVATE
        DllRegisterServer   @3 PRIVATE
        DllUnregisterServer    @4 PRIVATE


#pragma warning( disable: 4049 )  /* more than 64k source lines */

/* this ALWAYS GENERATED file contains the definitions for the interfaces
*/


 /* File created by MIDL compiler version 5.02.0235 */
/* at Fri Aug 13 18:56:24 1999
 */
/* Compiler settings for .\src\tpcc_com_all.idl:
    Oicf (OptLev=i2), W1, Zp8, env=Win32 (32b run), ms_ext, c_ext
    error checks: allocation ref bounds_check enum stub_data
    VC __declspec() decoration level:
         __declspec(uuid()), __declspec(selectany), __declspec(novtable)
         DECLSPEC_UUID(), MIDL_INTERFACE()
*/
//@@MIDL_FILE_HEADING(  )


/* verify that the <rpcndr.h> version is high enough to compile this
file*/
#ifndef __REQUIRED_RPCNDR_H_VERSION__
#define __REQUIRED_RPCNDR_H_VERSION__ 440
#endif

#include "rpc.h"
#include "rpcndr.h"

#ifndef __tpcc_com_all_h__
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
#define __tpcc_com_all_h__

/* Forward Declarations */

#ifndef __TPCC_FWD_DEFINED__
#define __TPCC_FWD_DEFINED__

#ifdef __cplusplus
typedef class TPCC TPCC;
#else
typedef struct TPCC TPCC;
#endif /* __cplusplus */

#endif  /* __TPCC_FWD_DEFINED__ */


#ifndef __NewOrder_FWD_DEFINED__
#define __NewOrder_FWD_DEFINED__

#ifdef __cplusplus
typedef class NewOrder NewOrder;
#else
typedef struct NewOrder NewOrder;
#endif /* __cplusplus */

#endif  /* __NewOrder_FWD_DEFINED__ */


#ifndef __OrderStatus_FWD_DEFINED__
#define __OrderStatus_FWD_DEFINED__

#ifdef __cplusplus
typedef class OrderStatus OrderStatus;
#else
typedef struct OrderStatus OrderStatus;
#endif /* __cplusplus */

#endif  /* __OrderStatus_FWD_DEFINED__ */


#ifndef __Payment_FWD_DEFINED__
#define __Payment_FWD_DEFINED__

#ifdef __cplusplus
typedef class Payment Payment;
#else
typedef struct Payment Payment;
#endif /* __cplusplus */

#endif  /* __Payment_FWD_DEFINED__ */


#ifndef __StockLevel_FWD_DEFINED__
#define __StockLevel_FWD_DEFINED__

#ifdef __cplusplus
typedef class StockLevel StockLevel;
#else
typedef struct StockLevel StockLevel;
#endif /* __cplusplus */

#endif  /* __StockLevel_FWD_DEFINED__ */


/* header files for imported files */
#include "oaidl.h"
#include "ocidl.h"
#include "tpcc_com_ps.h"

#ifdef __cplusplus
extern "C"{
#endif

void __RPC_FAR * __RPC_USER MIDL_user_allocate(size_t);
void __RPC_USER MIDL_user_free( void __RPC_FAR * );

/* interface __MIDL_itf_tpcc_com_all_0000 */
/* [local] */




extern RPC_IF_HANDLE __MIDL_itf_tpcc_com_all_0000_v0_0_c_ifspec;
extern RPC_IF_HANDLE __MIDL_itf_tpcc_com_all_0000_v0_0_s_ifspec;


#ifndef __TPCCLib_LIBRARY_DEFINED__
#define __TPCCLib_LIBRARY_DEFINED__

/* library TPCCLib */
/* [helpstring][version][uuid] */


EXTERN_C const IID LIBID_TPCCLib;

EXTERN_C const CLSID CLSID_TPCC;

#ifdef __cplusplus

class DECLSPEC_UUID("122A3128-2520-11D3-BA71-00C04FBFE08B")
TPCC;
#endif
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
EXTERN_C const CLSID CLSID_NewOrder;

#ifdef __cplusplus

class DECLSPEC_UUID("975BAABF-84A7-11D2-BA47-00C04FBFE08B")
NewOrder;
#endif

EXTERN_C const CLSID CLSID_OrderStatus;

#ifdef __cplusplus

class DECLSPEC_UUID("266836AD-A50D-11D2-BA4E-00C04FBFE08B")
OrderStatus;
#endif

EXTERN_C const CLSID CLSID_Payment;

#ifdef __cplusplus

class DECLSPEC_UUID("CD02F7EF-A4FA-11D2-BA4E-00C04FBFE08B")
Payment;
#endif

EXTERN_C const CLSID CLSID_StockLevel;

#ifdef __cplusplus

class DECLSPEC_UUID("2668369E-A50D-11D2-BA4E-00C04FBFE08B")
StockLevel;
#endif
#endif /* __TPCCLib_LIBRARY_DEFINED__ */

/* Additional Prototypes for ALL interfaces */

/* end of Additional Prototypes */

#ifdef __cplusplus
}
#endif

#endif



/*      FILE:           TPCC.IDL
 *                              Microsoft TPC-C Kit Ver. 4.20.000
 *                              Copyright Microsoft, 1999
 *              All Rights Reserved
 *
 *                              not yet audited
 *
 *      PURPOSE:        IDL source for TPCC.dll.  This file is processed by
the MIDL tool to
 *                              produce the type library (TPCC.tlb) and
marshalling code.
 *
 *  Change history:
 *              4.20.000 - first version
 */

interface TPCC;
interface NewOrder;
interface OrderStatus;
interface Payment;
interface StockLevel;


import "oaidl.idl";
import "ocidl.idl";
import "..\tpcc_com_ps\src\tpcc_com_ps.idl";

[
        uuid(122A3117-2520-11D3-BA71-00C04FBFE08B),
        version(1.0),
        helpstring("TPC-C 1.0 Type Library")
]
library TPCCLib
{
        importlib("stdole32.tlb");
        importlib("stdole2.tlb");

        [
                uuid(122A3128-2520-11D3-BA71-00C04FBFE08B),
                helpstring("All Txns Class")
        ]
        coclass TPCC
        {
                [default] interface ITPCC;
        };


        [
                uuid(975BAABF-84A7-11D2-BA47-00C04FBFE08B),
                helpstring("NewOrder Class")
        ]
        coclass NewOrder
        {
                [default] interface ITPCC;
        };


        [
                uuid(266836AD-A50D-11D2-BA4E-00C04FBFE08B),
                helpstring("OrderStatus Class")
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
	]
	coclass OrderStatus
	{
		[default] interface ITPCC;
	};


	[
		uuid(CD02F7EF-A4FA-11D2-BA4E-00C04FBFE08B),
		helpstring("Payment Class")
	]
	coclass Payment
	{
		[default] interface ITPCC;
	};


	[
		uuid(2668369E-A50D-11D2-BA4E-00C04FBFE08B),
		helpstring("StockLevel Class")
	]
	coclass StockLevel
	{
		[default] interface ITPCC;
	};

};

//Microsoft Developer Studio generated resource script.
//
#include "resource.h"

#define APSTUDIO_READONLY_SYMBOLS
/////////////////////////////////////////////////////////////////////////////
////
//
// Generated from the TEXTINCLUDE 2 resource.
//
#include "winres.h"

/////////////////////////////////////////////////////////////////////////////
////
#undef APSTUDIO_READONLY_SYMBOLS

/////////////////////////////////////////////////////////////////////////////
////
// English (U.S.) resources

#if !defined(AFX_RESOURCE_DLL) || defined(AFX_TARG_ENU)
#ifdef _WIN32
LANGUAGE LANG_ENGLISH, SUBLANG_ENGLISH_US
#pragma code_page(1252)
#endif //_WIN32


#ifdef APSTUDIO_INVOKED
/////////////////////////////////////////////////////////////////////////////
////
//
// TEXTINCLUDE
//

1 TEXTINCLUDE DISCARDABLE
BEGIN
    "resource.h\0"
END

2 TEXTINCLUDE DISCARDABLE
BEGIN
    "#include ""winres.h""\r\n"
    "\0"
END

3 TEXTINCLUDE DISCARDABLE
BEGIN
    "1 TYPELIB ""tpcc_com_all.tlb""\r\n"
    "\0"
END

#endif    // APSTUDIO_INVOKED


#ifndef _MAC
/////////////////////////////////////////////////////////////////////////////
////
//
// Version
//

VS_VERSION_INFO VERSIONINFO
 FILEVERSION 1,0,0,1
 PRODUCTVERSION 1,0,0,1
 FILEFLAGSMASK 0x3fL
#ifdef _DEBUG
 FILEFLAGS 0x1L
#else
 FILEFLAGS 0x0L
#endif
 FILEOS 0x4L
 FILETYPE 0x2L
 FILESUBTYPE 0x0L
BEGIN
    BLOCK "StringFileInfo"
    BEGIN
        BLOCK "040904B0"
        BEGIN
            VALUE "CompanyName", "\0"
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
            VALUE "FileDescription", "tpcc_com_all Module\0"
            VALUE "FileVersion", "1, 0, 0, 1\0"
            VALUE "InternalName", "TPCCNEWORDER\0"
            VALUE "LegalCopyright", "Copyright 1997\0"
            VALUE "OriginalFilename", "tpcc_com_all.DLL\0"
            VALUE "ProductName", "tpcc_com_all Module\0"
            VALUE "ProductVersion", "1, 0, 0, 1\0"
            VALUE "OLESelfRegister", "\0"
        END
    END
    BLOCK "VarFileInfo"
    BEGIN
        VALUE "Translation", 0x409, 1200
    END
END

#endif    // !_MAC


/////////////////////////////////////////////////////////////////////////////
////
//
// REGISTRY
//

IDR_TPCC                REGISTRY DISCARDABLE    "tpcc_com_all.rgs"
IDR_NEWORDER            REGISTRY DISCARDABLE    "tpcc_com_no.rgs"
IDR_ORDERSTATUS         REGISTRY DISCARDABLE    "tpcc_com_os.rgs"
IDR_PAYMENT             REGISTRY DISCARDABLE    "tpcc_com_pay.rgs"
IDR_STOCKLEVEL          REGISTRY DISCARDABLE    "tpcc_com_sl.rgs"

/////////////////////////////////////////////////////////////////////////////
////
//
// String Table
//

STRINGTABLE DISCARDABLE
BEGIN
    IDS_PROJNAME            "tpcc_com_all"
END

#endif    // English (U.S.) resources
/////////////////////////////////////////////////////////////////////////////
////



#ifndef APSTUDIO_INVOKED
/////////////////////////////////////////////////////////////////////////////
////
//
// Generated from the TEXTINCLUDE 3 resource.
```

```
//
1 TYPELIB "tpcc_com_all.tlb"

/////////////////////////////////////////////////////////////////////////////
////
#endif    // not APSTUDIO_INVOKED


HKCR
{
        TPCC.AllTxns.1 = s 'All Txns Class'
        {
                CLSID = s '{122A3128-2520-11D3-BA71-00C04FBFE08B}'
        }
        TPCC.AllTxns = s 'TPCC Class'
        {
                CurVer = s 'TPCC.AllTxns.1'
        }
        NoRemove CLSID
        {
                ForceRemove {122A3128-2520-11D3-BA71-00C04FBFE08B} = s
'TPCC Class'
                {
                        ProgID = s 'TPCC.AllTxns.1'
                        VersionIndependentProgID = s 'TPCC.AllTxns'
                        InprocServer32 = s '%MODULE%'
                        {
                                val ThreadingModel = s 'Both'
                        }
                }
        }
}


#pragma warning( disable: 4049 )  /* more than 64k source lines */

/* this ALWAYS GENERATED file contains the IIDs and CLSIDs */

/* link this file in with the server and any clients */


 /* File created by MIDL compiler version 5.02.0235 */
/* at Fri Aug 13 18:56:24 1999
 */
/* Compiler settings for .\src\tpcc_com_all.idl:
    Oicf (OptLev=i2), W1, Zp8, env=Win32 (32b run), ms_ext, c_ext
    error checks: allocation ref bounds_check enum stub_data
    VC __declspec() decoration level:
         __declspec(uuid()), __declspec(selectany), __declspec(novtable)
         DECLSPEC_UUID(), MIDL_INTERFACE()
*/
//@@MIDL_FILE_HEADING(  )
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
#if !defined(_M_IA64) && !defined(_M_AXP64)

#ifdef __cplusplus
extern "C"{
#endif


#include <rpc.h>
#include <rpcndr.h>

#ifdef _MIDL_USE_GUIDDEF_

#ifndef INITGUID
#define INITGUID
#include <guiddef.h>
#undef INITGUID
#else
#include <guiddef.h>
#endif

#define MIDL_DEFINE_GUID(type,name,l,w1,w2,b1,b2,b3,b4,b5,b6,b7,b8) \
        DEFINE_GUID(name,l,w1,w2,b1,b2,b3,b4,b5,b6,b7,b8)

#else // !_MIDL_USE_GUIDDEF_

#ifndef __IID_DEFINED__
#define __IID_DEFINED__

typedef struct _IID
{
    unsigned long x;
    unsigned short s1;
    unsigned short s2;
    unsigned char  c[8];
} IID;

#endif // __IID_DEFINED__

#ifndef CLSID_DEFINED
#define CLSID_DEFINED
typedef IID CLSID;
#endif // CLSID_DEFINED

#define MIDL_DEFINE_GUID(type,name,l,w1,w2,b1,b2,b3,b4,b5,b6,b7,b8) \
        const type name = {l,w1,w2,{b1,b2,b3,b4,b5,b6,b7,b8}}

#endif !_MIDL_USE_GUIDDEF_

MIDL_DEFINE_GUID(IID,
LIBID_TPCCLib,0x122A3117,0x2520,0x11D3,0xBA,0x71,0x00,0xC0,0x4F,0xBF,0xE0
,0x8B);

MIDL_DEFINE_GUID(CLSID,
CLSID_TPCC,0x122A3128,0x2520,0x11D3,0xBA,0x71,0x00,0xC0,0x4F,0xBF,0xE0,0x
8B);


MIDL_DEFINE_GUID(CLSID,
CLSID_NewOrder,0x975BAABF,0x84A7,0x11D2,0xBA,0x47,0x00,0xC0,0x4F,0xBF,0xE
0,0x8B);


MIDL_DEFINE_GUID(CLSID,
CLSID_OrderStatus,0x266836AD,0xA50D,0x11D2,0xBA,0x4E,0x00,0xC0,0x4F,0xBF,
0xE0,0x8B);


MIDL_DEFINE_GUID(CLSID,
CLSID_Payment,0xCD02F7EF,0xA4FA,0x11D2,0xBA,0x4E,0x00,0xC0,0x4F,0xBF,0xE0
,0x8B);


MIDL_DEFINE_GUID(CLSID,
CLSID_StockLevel,0x2668369E,0xA50D,0x11D2,0xBA,0x4E,0x00,0xC0,0x4F,0xBF,0
xE0,0x8B);

#undef MIDL_DEFINE_GUID

#ifdef __cplusplus
}
#endif



#endif /* !defined(_M_IA64) && !defined(_M_AXP64)*/


#pragma warning( disable: 4049 )  /* more than 64k source lines */

/* this ALWAYS GENERATED file contains the IIDs and CLSIDs */

/* link this file in with the server and any clients */


 /* File created by MIDL compiler version 5.02.0235 */
/* at Fri Aug 13 18:56:25 1999
 */
/* Compiler settings for .\src\tpcc_com_all.idl:
    Oicf (OptLev=i2), W1, Zp8, env=Win64 (32b run,appending), ms_ext,
c_ext
    error checks: allocation ref bounds_check enum stub_data
    VC __declspec() decoration level:
         __declspec(uuid()), __declspec(selectany), __declspec(novtable)
         DECLSPEC_UUID(), MIDL_INTERFACE()
*/
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
//@@MIDL_FILE_HEADING(  )

#if defined(_M_IA64) || defined(_M_AXP64)

#ifdef __cplusplus
extern "C"{
#endif


#include <rpc.h>
#include <rpcndr.h>

#ifdef _MIDL_USE_GUIDDEF_

#ifndef INITGUID
#define INITGUID
#include <guiddef.h>
#undef INITGUID
#else
#include <guiddef.h>
#endif

#define MIDL_DEFINE_GUID(type,name,l,w1,w2,b1,b2,b3,b4,b5,b6,b7,b8) \
        DEFINE_GUID(name,l,w1,w2,b1,b2,b3,b4,b5,b6,b7,b8)

#else // !_MIDL_USE_GUIDDEF_

#ifndef __IID_DEFINED__
#define __IID_DEFINED__

typedef struct _IID
{
    unsigned long x;
    unsigned short s1;
    unsigned short s2;
    unsigned char  c[8];
} IID;

#endif // __IID_DEFINED__

#ifndef CLSID_DEFINED
#define CLSID_DEFINED
typedef IID CLSID;
#endif // CLSID_DEFINED

#define MIDL_DEFINE_GUID(type,name,l,w1,w2,b1,b2,b3,b4,b5,b6,b7,b8) \
        const type name = {l,w1,w2,{b1,b2,b3,b4,b5,b6,b7,b8}}

#endif !_MIDL_USE_GUIDDEF_

MIDL_DEFINE_GUID(IID,
LIBID_TPCCLib,0x122A3117,0x2520,0x11D3,0xBA,0x71,0x00,0xC0,0x4F,0xBF,0xE0
,0x8B);


MIDL_DEFINE_GUID(CLSID,
CLSID_TPCC,0x122A3128,0x2520,0x11D3,0xBA,0x71,0x00,0xC0,0x4F,0xBF,0xE0,0x
8B);


MIDL_DEFINE_GUID(CLSID,
CLSID_NewOrder,0x975BAABF,0x84A7,0x11D2,0xBA,0x47,0x00,0xC0,0x4F,0xBF,0xE
0,0x8B);


MIDL_DEFINE_GUID(CLSID,
CLSID_OrderStatus,0x266836AD,0xA50D,0x11D2,0xBA,0x4E,0x00,0xC0,0x4F,0xBF,
0xE0,0x8B);


MIDL_DEFINE_GUID(CLSID,
CLSID_Payment,0xCD02F7EF,0xA4FA,0x11D2,0xBA,0x4E,0x00,0xC0,0x4F,0xBF,0xE0
,0x8B);


MIDL_DEFINE_GUID(CLSID,
CLSID_StockLevel,0x2668369E,0xA50D,0x11D2,0xBA,0x4E,0x00,0xC0,0x4F,0xBF,0
xE0,0x8B);

#undef MIDL_DEFINE_GUID

#ifdef __cplusplus
}
#endif



#endif /* defined(_M_IA64) || defined(_M_AXP64)*/


HKCR
{
        TPCC.NewOrder.1 = s 'NewOrder Class'
        {
                CLSID = s '{975BAABF-84A7-11D2-BA47-00C04FBFE08B}'
        }
        TPCC.NewOrder = s 'NewOrder Class'
        {
                CurVer = s 'TPCC.NewOrder.1'
        }
        NoRemove CLSID
        {
                ForceRemove {975BAABF-84A7-11D2-BA47-00C04FBFE08B} = s
'NewOrder Class'
                {
                        ProgID = s 'TPCC.NewOrder.1'
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                        VersionIndependentProgID = s 'TPCC.NewOrder'
                        InprocServer32 = s '%MODULE%'
                        {
                                val ThreadingModel = s 'Both'
                        }
                }
        }
}

HKCR
{
        TPCC.OrderStatus.1 = s 'OrderStatus Class'
        {
                CLSID = s '{266836AD-A50D-11D2-BA4E-00C04FBFE08B}'
        }
        TPCC.OrderStatus = s 'OrderStatus Class'
        {
                CurVer = s 'TPCC.OrderStatus.1'
        }
        NoRemove CLSID
        {
                ForceRemove {266836AD-A50D-11D2-BA4E-00C04FBFE08B} = s
'OrderStatus Class'
                {
                        ProgID = s 'TPCC.OrderStatus.1'
                        VersionIndependentProgID = s 'TPCC.OrderStatus'
                        InprocServer32 = s '%MODULE%'
                        {
                                val ThreadingModel = s 'Both'
                        }
                }
        }
}

HKCR
{
        TPCC.Payment.1 = s 'Payment Class'
        {
                CLSID = s '{CD02F7EF-A4FA-11D2-BA4E-00C04FBFE08B}'
        }
        TPCC.Payment = s 'Payment Class'
        {
                CurVer = s 'TPCC.Payment.1'
        }
        NoRemove CLSID
        {
                ForceRemove {CD02F7EF-A4FA-11D2-BA4E-00C04FBFE08B} = s
'Payment Class'
                {
                        ProgID = s 'TPCC.Payment.1'
                        VersionIndependentProgID = s 'TPCC.Payment'
                        InprocServer32 = s '%MODULE%'
                        {
                                val ThreadingModel = s 'Both'
                        }
                }
        }
}


#pragma warning( disable: 4049 )  /* more than 64k source lines */

/* this ALWAYS GENERATED file contains the definitions for the interfaces
*/


 /* File created by MIDL compiler version 5.02.0235 */
/* at Fri Aug 13 18:56:17 1999
 */
/* Compiler settings for .\src\tpcc_com_ps.idl:
    Oicf (OptLev=i2), W1, Zp8, env=Win32 (32b run), ms_ext, c_ext
    error checks: allocation ref bounds_check enum stub_data
    VC __declspec() decoration level:
         __declspec(uuid()), __declspec(selectany), __declspec(novtable)
         DECLSPEC_UUID(), MIDL_INTERFACE()
*/
//@@MIDL_FILE_HEADING(  )


/* verify that the <rpcndr.h> version is high enough to compile this
file*/
#ifndef __REQUIRED_RPCNDR_H_VERSION__
#define __REQUIRED_RPCNDR_H_VERSION__ 440
#endif

#include "rpc.h"
#include "rpcndr.h"

#ifndef __RPCNDR_H_VERSION__
#error this stub requires an updated version of <rpcndr.h>
#endif // __RPCNDR_H_VERSION__

#ifndef COM_NO_WINDOWS_H
#include "windows.h"
#include "ole2.h"
#endif /*COM_NO_WINDOWS_H*/

#ifndef __tpcc_com_ps_h__
#define __tpcc_com_ps_h__

/* Forward Declarations */

#ifndef __ITPCC_FWD_DEFINED__
#define __ITPCC_FWD_DEFINED__
typedef interface ITPCC ITPCC;
#endif  /* __ITPCC_FWD_DEFINED__ */
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
/* header files for imported files */
#include "oaidl.h"
#include "ocidl.h"

#ifdef __cplusplus
extern "C"{
#endif

void __RPC_FAR * __RPC_USER MIDL_user_allocate(size_t);
void __RPC_USER MIDL_user_free( void __RPC_FAR * );

#ifndef __ITPCC_INTERFACE_DEFINED__
#define __ITPCC_INTERFACE_DEFINED__

/* interface ITPCC */
/* [unique][helpstring][uuid][object] */


EXTERN_C const IID IID_ITPCC;

#if defined(__cplusplus) && !defined(CINTERFACE)

    MIDL_INTERFACE("FEEE6AA2-84B1-11d2-BA47-00C04FBFE08B")
    ITPCC : public IUnknown
    {
    public:
        virtual HRESULT __stdcall NewOrder(
            /* [out][in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn) = 0;

        virtual HRESULT __stdcall Payment(
            /* [out][in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn) = 0;

        virtual HRESULT __stdcall Delivery(
            /* [in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn) = 0;

        virtual HRESULT __stdcall StockLevel(
            /* [out][in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn) = 0;

        virtual HRESULT __stdcall OrderStatus(
            /* [out][in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn) = 0;

        virtual HRESULT __stdcall CallSetComplete( void) = 0;

    };

#else   /* C style interface */

    typedef struct ITPCCVtbl
    {
        BEGIN_INTERFACE

        HRESULT ( STDMETHODCALLTYPE __RPC_FAR *QueryInterface )(
            ITPCC __RPC_FAR * This,
            /* [in] */ REFIID riid,
            /* [iid_is][out] */ void __RPC_FAR *__RPC_FAR *ppvObject);

        ULONG ( STDMETHODCALLTYPE __RPC_FAR *AddRef )(
            ITPCC __RPC_FAR * This);

        ULONG ( STDMETHODCALLTYPE __RPC_FAR *Release )(
            ITPCC __RPC_FAR * This);

        HRESULT ( __stdcall __RPC_FAR *NewOrder )(
            ITPCC __RPC_FAR * This,
            /* [out][in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn);

        HRESULT ( __stdcall __RPC_FAR *Payment )(
            ITPCC __RPC_FAR * This,
            /* [out][in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn);

        HRESULT ( __stdcall __RPC_FAR *Delivery )(
            ITPCC __RPC_FAR * This,
            /* [in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn);

        HRESULT ( __stdcall __RPC_FAR *StockLevel )(
            ITPCC __RPC_FAR * This,
            /* [out][in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn);

        HRESULT ( __stdcall __RPC_FAR *OrderStatus )(
            ITPCC __RPC_FAR * This,
            /* [out][in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn);

        HRESULT ( __stdcall __RPC_FAR *CallSetComplete )(
            ITPCC __RPC_FAR * This);
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        END_INTERFACE
    } ITPCCVtbl;

    interface ITPCC
    {
        CONST_VTBL struct ITPCCVtbl __RPC_FAR *lpVtbl;
    };



#ifdef COBJMACROS


#define ITPCC_QueryInterface(This,riid,ppvObject)    \
    (This)->lpVtbl -> QueryInterface(This,riid,ppvObject)

#define ITPCC_AddRef(This)    \
    (This)->lpVtbl -> AddRef(This)

#define ITPCC_Release(This)   \
    (This)->lpVtbl -> Release(This)


#define ITPCC_NewOrder(This,iSize,txn)       \
    (This)->lpVtbl -> NewOrder(This,iSize,txn)

#define ITPCC_Payment(This,iSize,txn) \
    (This)->lpVtbl -> Payment(This,iSize,txn)

#define ITPCC_Delivery(This,iSize,txn)       \
    (This)->lpVtbl -> Delivery(This,iSize,txn)

#define ITPCC_StockLevel(This,iSize,txn)     \
    (This)->lpVtbl -> StockLevel(This,iSize,txn)

#define ITPCC_OrderStatus(This,iSize,txn)    \
    (This)->lpVtbl -> OrderStatus(This,iSize,txn)

#define ITPCC_CallSetComplete(This)   \
    (This)->lpVtbl -> CallSetComplete(This)

#endif /* COBJMACROS */


#endif  /* C style interface */



HRESULT __stdcall ITPCC_NewOrder_Proxy(
    ITPCC __RPC_FAR * This,
    /* [out][in] */ int __RPC_FAR *iSize,
    /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR *__RPC_FAR
*txn);


void __RPC_STUB ITPCC_NewOrder_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);


HRESULT __stdcall ITPCC_Payment_Proxy(
    ITPCC __RPC_FAR * This,
    /* [out][in] */ int __RPC_FAR *iSize,
    /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR *__RPC_FAR
*txn);


void __RPC_STUB ITPCC_Payment_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);


HRESULT __stdcall ITPCC_Delivery_Proxy(
    ITPCC __RPC_FAR * This,
    /* [in] */ int __RPC_FAR *iSize,
    /* [size_is][size_is][in] */ unsigned char __RPC_FAR *__RPC_FAR
*txn);


void __RPC_STUB ITPCC_Delivery_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);


HRESULT __stdcall ITPCC_StockLevel_Proxy(
    ITPCC __RPC_FAR * This,
    /* [out][in] */ int __RPC_FAR *iSize,
    /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR *__RPC_FAR
*txn);


void __RPC_STUB ITPCC_StockLevel_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
HRESULT __stdcall ITPCC_OrderStatus_Proxy(
    ITPCC __RPC_FAR * This,
    /* [out][in] */ int __RPC_FAR *iSize,
    /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR *__RPC_FAR
*txn);


void __RPC_STUB ITPCC_OrderStatus_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);


HRESULT __stdcall ITPCC_CallSetComplete_Proxy(
    ITPCC __RPC_FAR * This);


void __RPC_STUB ITPCC_CallSetComplete_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);



#endif  /* __ITPCC_INTERFACE_DEFINED__ */


/* Additional Prototypes for ALL interfaces */

/* end of Additional Prototypes */

#ifdef __cplusplus
}
#endif

#endif



HKCR
{
       TPCC.StockLevel.1 = s 'StockLevel Class'
       {
              CLSID = s '{2668369E-A50D-11D2-BA4E-00C04FBFE08B}'
       }
       TPCC.StockLevel = s 'StockLevel Class'
       {
              CurVer = s 'TPCC.StockLevel.1'
       }
       NoRemove CLSID
       {
              ForceRemove {2668369E-A50D-11D2-BA4E-00C04FBFE08B} = s
'StockLevel Class'
              {
                     ProgID = s 'TPCC.StockLevel.1'
                     VersionIndependentProgID = s 'TPCC.StockLevel'
                     InprocServer32 = s '%MODULE%'
                     {
                            val ThreadingModel = s 'Both'
                     }
              }
       }
}

/****************************************************
   DllData file -- generated by MIDL compiler

        DO NOT ALTER THIS FILE

   This file is regenerated by MIDL on every IDL file compile.

   To completely reconstruct this file, delete it and rerun MIDL
   on all the IDL files in this DLL, specifying this file for the
   /dlldata command line option

****************************************************/


#include <rpcproxy.h>

#ifdef __cplusplus
extern "C"   {
#endif

EXTERN_PROXY_FILE( tpcc_com_ps )


PROXYFILE_LIST_START
/* Start of list */
  REFERENCE_PROXY_FILE( tpcc_com_ps ),
/* End of list */
PROXYFILE_LIST_END


DLLDATA_ROUTINES( aProxyFileList, GET_DLL_CLSID )

#ifdef __cplusplus
}  /*extern "C" */
#endif

/* end of generated dlldata file */


LIBRARY      "tpcc_com_ps"
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
DESCRIPTION  'Proxy/Stub DLL'

EXPORTS
       DllGetClassObject       @1    PRIVATE
       DllCanUnloadNow         @2    PRIVATE
       GetProxyDllInfo         @3    PRIVATE
       DllRegisterServer             @4      PRIVATE
       DllUnregisterServer           @5      PRIVATE


#pragma warning( disable: 4049 )  /* more than 64k source lines */

/* this ALWAYS GENERATED file contains the definitions for the interfaces
*/


 /* File created by MIDL compiler version 5.02.0235 */
/* at Fri Aug 13 18:56:17 1999
 */
/* Compiler settings for .\src\tpcc_com_ps.idl:
    Oicf (OptLev=i2), W1, Zp8, env=Win32 (32b run), ms_ext, c_ext
    error checks: allocation ref bounds_check enum stub_data
    VC __declspec() decoration level:
         __declspec(uuid()), __declspec(selectany), __declspec(novtable)
         DECLSPEC_UUID(), MIDL_INTERFACE()
*/
//@@MIDL_FILE_HEADING(  )


/* verify that the <rpcndr.h> version is high enough to compile this
file*/
#ifndef __REQUIRED_RPCNDR_H_VERSION__
#define __REQUIRED_RPCNDR_H_VERSION__ 440
#endif

#include "rpc.h"
#include "rpcndr.h"

#ifndef __RPCNDR_H_VERSION__
#error this stub requires an updated version of <rpcndr.h>
#endif // __RPCNDR_H_VERSION__

#ifndef COM_NO_WINDOWS_H
#include "windows.h"
#include "ole2.h"
#endif /*COM_NO_WINDOWS_H*/

#ifndef __tpcc_com_ps_h__
#define __tpcc_com_ps_h__

/* Forward Declarations */

#ifndef __ITPCC_FWD_DEFINED__
#define __ITPCC_FWD_DEFINED__
typedef interface ITPCC ITPCC;
#endif  /* __ITPCC_FWD_DEFINED__ */


/* header files for imported files */
#include "oaidl.h"
#include "ocidl.h"

#ifdef __cplusplus
extern "C"{
#endif

void __RPC_FAR * __RPC_USER MIDL_user_allocate(size_t);
void __RPC_USER MIDL_user_free( void __RPC_FAR * );

#ifndef __ITPCC_INTERFACE_DEFINED__
#define __ITPCC_INTERFACE_DEFINED__

/* interface ITPCC */
/* [unique][helpstring][uuid][object] */


EXTERN_C const IID IID_ITPCC;

#if defined(__cplusplus) && !defined(CINTERFACE)

    MIDL_INTERFACE("FEEE6AA2-84B1-11d2-BA47-00C04FBFE08B")
    ITPCC : public IUnknown
    {
    public:
        virtual HRESULT __stdcall NewOrder(
            /* [out][in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn) = 0;

        virtual HRESULT __stdcall Payment(
            /* [out][in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn) = 0;

        virtual HRESULT __stdcall Delivery(
            /* [in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn) = 0;

        virtual HRESULT __stdcall StockLevel(
            /* [out][in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn) = 0;

        virtual HRESULT __stdcall OrderStatus(
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
            /* [out][in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn) = 0;

        virtual HRESULT __stdcall CallSetComplete( void) = 0;

    };

#else   /* C style interface */

    typedef struct ITPCCVtbl
    {
        BEGIN_INTERFACE

        HRESULT ( STDMETHODCALLTYPE __RPC_FAR *QueryInterface )(
            ITPCC __RPC_FAR * This,
            /* [in] */ REFIID riid,
            /* [iid_is][out] */ void __RPC_FAR *__RPC_FAR *ppvObject);

        ULONG ( STDMETHODCALLTYPE __RPC_FAR *AddRef )(
            ITPCC __RPC_FAR * This);

        ULONG ( STDMETHODCALLTYPE __RPC_FAR *Release )(
            ITPCC __RPC_FAR * This);

        HRESULT ( __stdcall __RPC_FAR *NewOrder )(
            ITPCC __RPC_FAR * This,
            /* [out][in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn);

        HRESULT ( __stdcall __RPC_FAR *Payment )(
            ITPCC __RPC_FAR * This,
            /* [out][in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn);

        HRESULT ( __stdcall __RPC_FAR *Delivery )(
            ITPCC __RPC_FAR * This,
            /* [in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn);

        HRESULT ( __stdcall __RPC_FAR *StockLevel )(
            ITPCC __RPC_FAR * This,
            /* [out][in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn);

        HRESULT ( __stdcall __RPC_FAR *OrderStatus )(
            ITPCC __RPC_FAR * This,
            /* [out][in] */ int __RPC_FAR *iSize,
            /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR
*__RPC_FAR *txn);

        HRESULT ( __stdcall __RPC_FAR *CallSetComplete )(
            ITPCC __RPC_FAR * This);

        END_INTERFACE
    } ITPCCVtbl;

    interface ITPCC
    {
        CONST_VTBL struct ITPCCVtbl __RPC_FAR *lpVtbl;
    };


#ifdef COBJMACROS


#define ITPCC_QueryInterface(This,riid,ppvObject)    \
    (This)->lpVtbl -> QueryInterface(This,riid,ppvObject)

#define ITPCC_AddRef(This)    \
    (This)->lpVtbl -> AddRef(This)

#define ITPCC_Release(This)   \
    (This)->lpVtbl -> Release(This)


#define ITPCC_NewOrder(This,iSize,txn)       \
    (This)->lpVtbl -> NewOrder(This,iSize,txn)

#define ITPCC_Payment(This,iSize,txn) \
    (This)->lpVtbl -> Payment(This,iSize,txn)

#define ITPCC_Delivery(This,iSize,txn)       \
    (This)->lpVtbl -> Delivery(This,iSize,txn)

#define ITPCC_StockLevel(This,iSize,txn)     \
    (This)->lpVtbl -> StockLevel(This,iSize,txn)

#define ITPCC_OrderStatus(This,iSize,txn)    \
    (This)->lpVtbl -> OrderStatus(This,iSize,txn)

#define ITPCC_CallSetComplete(This)   \
    (This)->lpVtbl -> CallSetComplete(This)

#endif /* COBJMACROS */


#endif  /* C style interface */
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
HRESULT __stdcall ITPCC_NewOrder_Proxy(
    ITPCC __RPC_FAR * This,
    /* [out][in] */ int __RPC_FAR *iSize,
    /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR *__RPC_FAR
*txn);


void __RPC_STUB ITPCC_NewOrder_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);


HRESULT __stdcall ITPCC_Payment_Proxy(
    ITPCC __RPC_FAR * This,
    /* [out][in] */ int __RPC_FAR *iSize,
    /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR *__RPC_FAR
*txn);


void __RPC_STUB ITPCC_Payment_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);


HRESULT __stdcall ITPCC_Delivery_Proxy(
    ITPCC __RPC_FAR * This,
    /* [in] */ int __RPC_FAR *iSize,
    /* [size_is][size_is][in] */ unsigned char __RPC_FAR *__RPC_FAR
*txn);


void __RPC_STUB ITPCC_Delivery_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);


HRESULT __stdcall ITPCC_StockLevel_Proxy(
    ITPCC __RPC_FAR * This,
    /* [out][in] */ int __RPC_FAR *iSize,
    /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR *__RPC_FAR
*txn);


void __RPC_STUB ITPCC_StockLevel_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);


HRESULT __stdcall ITPCC_OrderStatus_Proxy(
    ITPCC __RPC_FAR * This,
    /* [out][in] */ int __RPC_FAR *iSize,
    /* [size_is][size_is][out][in] */ unsigned char __RPC_FAR *__RPC_FAR
*txn);


void __RPC_STUB ITPCC_OrderStatus_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);


HRESULT __stdcall ITPCC_CallSetComplete_Proxy(
    ITPCC __RPC_FAR * This);


void __RPC_STUB ITPCC_CallSetComplete_Stub(
    IRpcStubBuffer *This,
    IRpcChannelBuffer *_pRpcChannelBuffer,
    PRPC_MESSAGE _pRpcMessage,
    DWORD *_pdwStubPhase);



#endif  /* __ITPCC_INTERFACE_DEFINED__ */


/* Additional Prototypes for ALL interfaces */

/* end of Additional Prototypes */

#ifdef __cplusplus
}
#endif

#endif



/*      FILE:           ITPCC.IDL
 *                              Microsoft TPC-C Kit Ver. 4.20.000
 *                              Copyright Microsoft, 1999
 *              All Rights Reserved
 *
 *                              not yet audited
 *
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
 *     PURPOSE:       Defines the interface used by TPCC. This interface
can be implemented by C++ components.
 *
 *  Change history:
 *             4.20.000 - first version
 */

// Forward declare all types defined
//interface ITPCC;
import "oaidl.idl";
import "ocidl.idl";

       [
              object,
              uuid(FEEE6AA2-84B1-11d2-BA47-00C04FBFE08B),
              helpstring("ITPCC Interface"),
              pointer_default(unique)
       ]
       interface ITPCC : IUnknown
       {

       HRESULT _stdcall NewOrder
                                          (
                                          [in, out] int* iSize,
                                          [in, out, size_is( ,
*iSize)] char** txn
                                          );

       HRESULT _stdcall Payment
                                          (
                                          [in, out] int* iSize,
                                          [in, out, size_is( ,
*iSize)] char** txn
                                          );

       HRESULT _stdcall Delivery
                                          (
                                          [in] int* iSize,
                                          [in, size_is( , *iSize)]
char** txn
                                          );

       HRESULT _stdcall StockLevel
                                          (
                                          [in, out] int* iSize,
                                          [in, out, size_is( ,
*iSize)] char** txn
                                          );

       HRESULT _stdcall OrderStatus
                                          (
                                          [in, out] int* iSize,
                                          [in, out, size_is( ,
*iSize)] char** txn
                                          );

       HRESULT _stdcall CallSetComplete
                                          (
                                          );

    }; // interface ITPCC



#pragma warning( disable: 4049 )  /* more than 64k source lines */

/* this ALWAYS GENERATED file contains the IIDs and CLSIDs */

/* link this file in with the server and any clients */


 /* File created by MIDL compiler version 5.02.0235 */
/* at Fri Aug 13 18:56:17 1999
 */
/* Compiler settings for .\src\tpcc_com_ps.idl:
    Oicf (OptLev=i2), W1, Zp8, env=Win32 (32b run), ms_ext, c_ext
    error checks: allocation ref bounds_check enum stub_data
    VC __declspec() decoration level:
         __declspec(uuid()), __declspec(selectany), __declspec(novtable)
         DECLSPEC_UUID(), MIDL_INTERFACE()
*/
//@@MIDL_FILE_HEADING(  )

#if !defined(_M_IA64) && !defined(_M_AXP64)

#ifdef __cplusplus
extern "C"{
#endif


#include <rpc.h>
#include <rpcndr.h>

#ifdef _MIDL_USE_GUIDDEF_

#ifndef INITGUID
#define INITGUID
#include <guiddef.h>
#undef INITGUID
#else
#include <guiddef.h>
#endif

#define MIDL_DEFINE_GUID(type,name,l,w1,w2,b1,b2,b3,b4,b5,b6,b7,b8) \
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        DEFINE_GUID(name,l,w1,w2,b1,b2,b3,b4,b5,b6,b7,b8)

#else // !_MIDL_USE_GUIDDEF_

#ifndef __IID_DEFINED__
#define __IID_DEFINED__

typedef struct _IID
{
    unsigned long x;
    unsigned short s1;
    unsigned short s2;
    unsigned char  c[8];
} IID;

#endif // __IID_DEFINED__

#ifndef CLSID_DEFINED
#define CLSID_DEFINED
typedef IID CLSID;
#endif // CLSID_DEFINED

#define MIDL_DEFINE_GUID(type,name,l,w1,w2,b1,b2,b3,b4,b5,b6,b7,b8) \
        const type name = {l,w1,w2,{b1,b2,b3,b4,b5,b6,b7,b8}}

#endif !_MIDL_USE_GUIDDEF_

MIDL_DEFINE_GUID(IID,
IID_ITPCC,0xFEEE6AA2,0x84B1,0x11d2,0xBA,0x47,0x00,0xC0,0x4F,0xBF,0xE0,0x8
B);

#undef MIDL_DEFINE_GUID

#ifdef __cplusplus
}
#endif



#endif /* !defined(_M_IA64) && !defined(_M_AXP64)*/


#pragma warning( disable: 4049 )  /* more than 64k source lines */

/* this ALWAYS GENERATED file contains the IIDs and CLSIDs */

/* link this file in with the server and any clients */


 /* File created by MIDL compiler version 5.02.0235 */
/* at Fri Aug 13 18:56:18 1999
 */
/* Compiler settings for .\src\tpcc_com_ps.idl:
    Oicf (OptLev=i2), W1, Zp8, env=Win64 (32b run,appending), ms_ext,
c_ext
    error checks: allocation ref bounds_check enum stub_data
    VC __declspec() decoration level:
         __declspec(uuid()), __declspec(selectany), __declspec(novtable)
         DECLSPEC_UUID(), MIDL_INTERFACE()
*/
//@@MIDL_FILE_HEADING(  )

#if defined(_M_IA64) || defined(_M_AXP64)

#ifdef __cplusplus
extern "C"{
#endif


#include <rpc.h>
#include <rpcndr.h>

#ifdef _MIDL_USE_GUIDDEF_

#ifndef INITGUID
#define INITGUID
#include <guiddef.h>
#undef INITGUID
#else
#include <guiddef.h>
#endif

#define MIDL_DEFINE_GUID(type,name,l,w1,w2,b1,b2,b3,b4,b5,b6,b7,b8) \
        DEFINE_GUID(name,l,w1,w2,b1,b2,b3,b4,b5,b6,b7,b8)

#else // !_MIDL_USE_GUIDDEF_

#ifndef __IID_DEFINED__
#define __IID_DEFINED__

typedef struct _IID
{
    unsigned long x;
    unsigned short s1;
    unsigned short s2;
    unsigned char  c[8];
} IID;

#endif // __IID_DEFINED__

#ifndef CLSID_DEFINED
#define CLSID_DEFINED
typedef IID CLSID;
#endif // CLSID_DEFINED

#define MIDL_DEFINE_GUID(type,name,l,w1,w2,b1,b2,b3,b4,b5,b6,b7,b8) \
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        const type name = {l,w1,w2,{b1,b2,b3,b4,b5,b6,b7,b8}}

#endif !_MIDL_USE_GUIDDEF_

MIDL_DEFINE_GUID(IID,
IID_ITPCC,0xFEEE6AA2,0x84B1,0x11d2,0xBA,0x47,0x00,0xC0,0x4F,0xBF,0xE0,0x8
B);

#undef MIDL_DEFINE_GUID

#ifdef __cplusplus
}
#endif



#endif /* defined(_M_IA64) || defined(_M_AXP64)*/



#pragma warning( disable: 4049 )  /* more than 64k source lines */

/* this ALWAYS GENERATED file contains the proxy stub code */


 /* File created by MIDL compiler version 5.02.0235 */
/* at Fri Aug 13 18:56:17 1999
 */
/* Compiler settings for .\src\tpcc_com_ps.idl:
    Oicf (OptLev=i2), W1, Zp8, env=Win32 (32b run), ms_ext, c_ext
    error checks: allocation ref bounds_check enum stub_data
    VC __declspec() decoration level:
         __declspec(uuid()), __declspec(selectany), __declspec(novtable)
         DECLSPEC_UUID(), MIDL_INTERFACE()
*/
//@@MIDL_FILE_HEADING(  )

#if !defined(_M_IA64) && !defined(_M_AXP64)
#define USE_STUBLESS_PROXY


/* verify that the <rpcproxy.h> version is high enough to compile this
file*/
#ifndef __REDQ_RPCPROXY_H_VERSION__
#define __REQUIRED_RPCPROXY_H_VERSION__ 440
#endif


#include "rpcproxy.h"
#ifndef __RPCPROXY_H_VERSION__
#error this stub requires an updated version of <rpcproxy.h>
#endif // __RPCPROXY_H_VERSION__


#include "tpcc_com_ps.h"

#define TYPE_FORMAT_STRING_SIZE   33
#define PROC_FORMAT_STRING_SIZE   193
#define TRANSMIT_AS_TABLE_SIZE    0
#define WIRE_MARSHAL_TABLE_SIZE   0

typedef struct _MIDL_TYPE_FORMAT_STRING
    {
    short          Pad;
    unsigned char  Format[ TYPE_FORMAT_STRING_SIZE ];
    } MIDL_TYPE_FORMAT_STRING;

typedef struct _MIDL_PROC_FORMAT_STRING
    {
    short          Pad;
    unsigned char  Format[ PROC_FORMAT_STRING_SIZE ];
    } MIDL_PROC_FORMAT_STRING;


extern const MIDL_TYPE_FORMAT_STRING __MIDL_TypeFormatString;
extern const MIDL_PROC_FORMAT_STRING __MIDL_ProcFormatString;


/* Object interface: IUnknown, ver. 0.0,

GUID={0x00000000,0x0000,0x0000,{0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46}}
*/


/* Object interface: ITPCC, ver. 0.0,

GUID={0xFEEE6AA2,0x84B1,0x11d2,{0xBA,0x47,0x00,0xC0,0x4F,0xBF,0xE0,0x8B}}
*/


extern const MIDL_STUB_DESC Object_StubDesc;


extern const MIDL_SERVER_INFO ITPCC_ServerInfo;

#pragma code_seg(".orpc")
static const unsigned short ITPCC_FormatStringOffsetTable[] =
    {
    0,
    34,
    68,
    102,
    136,
    170
    };
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
static const MIDL_SERVER_INFO ITPCC_ServerInfo =
    {
    &Object_StubDesc,
    0,
    __MIDL_ProcFormatString.Format,
    &ITPCC_FormatStringOffsetTable[-3],
    0,
    0,
    0,
    0
    };

static const MIDL_STUBLESS_PROXY_INFO ITPCC_ProxyInfo =
    {
    &Object_StubDesc,
    __MIDL_ProcFormatString.Format,
    &ITPCC_FormatStringOffsetTable[-3],
    0,
    0,
    0
    };

CINTERFACE_PROXY_VTABLE(9) _ITPCCProxyVtbl =
{
    &ITPCC_ProxyInfo,
    &IID_ITPCC,
    IUnknown_QueryInterface_Proxy,
    IUnknown_AddRef_Proxy,
    IUnknown_Release_Proxy ,
    (void *)-1 /* ITPCC::NewOrder */ ,
    (void *)-1 /* ITPCC::Payment */ ,
    (void *)-1 /* ITPCC::Delivery */ ,
    (void *)-1 /* ITPCC::StockLevel */ ,
    (void *)-1 /* ITPCC::OrderStatus */ ,
    (void *)-1 /* ITPCC::CallSetComplete */
};

const CInterfaceStubVtbl _ITPCCStubVtbl =
{
    &IID_ITPCC,
    &ITPCC_ServerInfo,
    9,
    0, /* pure interpreted */
    CStdStubBuffer_METHODS
};


static const MIDL_STUB_DESC Object_StubDesc =
    {
    0,
    NdrOleAllocate,
    NdrOleFree,
    0,
    0,
    0,
    0,
    __MIDL_TypeFormatString.Format,
    1, /* -error bounds_check flag */
    0x20000, /* Ndr library version */
    0,
    0x50200eb, /* MIDL Version 5.2.235 */
    0,
    0,
    0,  /* notify & notify_flag routine table */
    1,  /* Flags */
    0,  /* Reserved3 */
    0,  /* Reserved4 */
    0   /* Reserved5 */
    };

#pragma data_seg(".rdata")

#if !defined(__RPC_WIN32__)
#error  Invalid build platform for this stub.
#endif

#if !(TARGET_IS_NT40_OR_LATER)
#error You need a Windows NT 4.0 or later to run this stub because it
uses these features:
#error   -Oif or -Oicf.
#error However, your C/C++ compilation flags indicate you intend to run
this app on earlier systems.
#error This app will die there with the RPC_X_WRONG_STUB_VERSION error.
#endif


static const MIDL_PROC_FORMAT_STRING __MIDL_ProcFormatString =
    {
        0,
        {

       /* Procedure NewOrder */

                    0x33,           /* FC_AUTO_HANDLE */
                    0x6c,           /* Old Flags:  object, Oi2 */
/*  2 */       NdrFcLong( 0x0 ),      /* 0 */
/*  6 */       NdrFcShort( 0x3 ),     /* 3 */
#ifndef _ALPHA_
/*  8 */       NdrFcShort( 0x10 ),    /* x86, MIPS, PPC Stack size/offset
= 16 */
#else
                    NdrFcShort( 0x20 ),    /* Alpha Stack size/offset =
32 */
#endif
/* 10 */       NdrFcShort( 0x8 ),     /* 8 */
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
/* 12 */       NdrFcShort( 0x10 ),    /* 16 */
/* 14 */       0x7,           /* Oi2 Flags:  srv must size, clt must
size, has return, */
                       0x3,           /* 3 */

       /* Parameter iSize */

/* 16 */       NdrFcShort( 0x158 ),   /* Flags:  in, out, base type,
simple ref, */
#ifndef _ALPHA_
/* 18 */       NdrFcShort( 0x4 ),     /* x86, MIPS, PPC Stack size/offset
= 4 */
#else
                       NdrFcShort( 0x8 ),     /* Alpha Stack size/offset =
8 */
#endif
/* 20 */       0x8,           /* FC_LONG */
                       0x0,           /* 0 */

       /* Parameter txn */

/* 22 */       NdrFcShort( 0x201b ),  /* Flags:  must size, must free, in,
out, srv alloc size=8 */
#ifndef _ALPHA_
/* 24 */       NdrFcShort( 0x8 ),     /* x86, MIPS, PPC Stack size/offset
= 8 */
#else
                       NdrFcShort( 0x10 ),    /* Alpha Stack size/offset =
16 */
#endif
/* 26 */       NdrFcShort( 0x6 ),     /* Type Offset=6 */

       /* Return value */

/* 28 */       NdrFcShort( 0x70 ),    /* Flags:  out, return, base type,
*/
#ifndef _ALPHA_
/* 30 */       NdrFcShort( 0xc ),     /* x86, MIPS, PPC Stack size/offset
= 12 */
#else
                       NdrFcShort( 0x18 ),    /* Alpha Stack size/offset =
24 */
#endif
/* 32 */       0x8,           /* FC_LONG */
                       0x0,           /* 0 */

       /* Procedure Payment */

/* 34 */       0x33,          /* FC_AUTO_HANDLE */
                       0x6c,          /* Old Flags:  object, Oi2 */
/* 36 */       NdrFcLong( 0x0 ),      /* 0 */
/* 40 */       NdrFcShort( 0x4 ),     /* 4 */
#ifndef _ALPHA_
/* 42 */       NdrFcShort( 0x10 ),    /* x86, MIPS, PPC Stack size/offset
= 16 */
#else
                       NdrFcShort( 0x20 ),    /* Alpha Stack size/offset =
32 */
#endif
/* 44 */       NdrFcShort( 0x8 ),     /* 8 */
/* 46 */       NdrFcShort( 0x10 ),    /* 16 */
/* 48 */       0x7,           /* Oi2 Flags:  srv must size, clt must
size, has return, */
                       0x3,           /* 3 */

       /* Parameter iSize */

/* 50 */       NdrFcShort( 0x158 ),   /* Flags:  in, out, base type,
simple ref, */
#ifndef _ALPHA_
/* 52 */       NdrFcShort( 0x4 ),     /* x86, MIPS, PPC Stack size/offset
= 4 */
#else
                       NdrFcShort( 0x8 ),     /* Alpha Stack size/offset =
8 */
#endif
/* 54 */       0x8,           /* FC_LONG */
                       0x0,           /* 0 */

       /* Parameter txn */

/* 56 */       NdrFcShort( 0x201b ),  /* Flags:  must size, must free, in,
out, srv alloc size=8 */
#ifndef _ALPHA_
/* 58 */       NdrFcShort( 0x8 ),     /* x86, MIPS, PPC Stack size/offset
= 8 */
#else
                       NdrFcShort( 0x10 ),    /* Alpha Stack size/offset =
16 */
#endif
/* 60 */       NdrFcShort( 0x6 ),     /* Type Offset=6 */

       /* Return value */

/* 62 */       NdrFcShort( 0x70 ),    /* Flags:  out, return, base type,
*/
#ifndef _ALPHA_
/* 64 */       NdrFcShort( 0xc ),     /* x86, MIPS, PPC Stack size/offset
= 12 */
#else
                       NdrFcShort( 0x18 ),    /* Alpha Stack size/offset =
24 */
#endif
/* 66 */       0x8,           /* FC_LONG */
                       0x0,           /* 0 */
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        /* Procedure Delivery */

/* 68 */       0x33,           /* FC_AUTO_HANDLE */
                   0x6c,           /* Old Flags:  object, Oi2 */
/* 70 */       NdrFcLong( 0x0 ),      /* 0 */
/* 74 */       NdrFcShort( 0x5 ),     /* 5 */
#ifndef _ALPHA_
/* 76 */       NdrFcShort( 0x10 ),    /* x86, MIPS, PPC Stack size/offset
= 16 */
#else
                   NdrFcShort( 0x20 ),    /* Alpha Stack size/offset =
32 */
#endif
/* 78 */       NdrFcShort( 0x8 ),     /* 8 */
/* 80 */       NdrFcShort( 0x8 ),     /* 8 */
/* 82 */       0x6,           /* Oi2 Flags:  clt must size, has return,
*/
                   0x3,           /* 3 */

        /* Parameter iSize */

/* 84 */       NdrFcShort( 0x148 ),   /* Flags:  in, base type, simple
ref, */
#ifndef _ALPHA_
/* 86 */       NdrFcShort( 0x4 ),     /* x86, MIPS, PPC Stack size/offset
= 4 */
#else
                   NdrFcShort( 0x8 ),     /* Alpha Stack size/offset =
8 */
#endif
/* 88 */       0x8,           /* FC_LONG */
                   0x0,           /* 0 */

        /* Parameter txn */

/* 90 */       NdrFcShort( 0x200b ),  /* Flags:  must size, must free, in,
srv alloc size=8 */
#ifndef _ALPHA_
/* 92 */       NdrFcShort( 0x8 ),     /* x86, MIPS, PPC Stack size/offset
= 8 */
#else
                   NdrFcShort( 0x10 ),    /* Alpha Stack size/offset =
16 */
#endif
/* 94 */       NdrFcShort( 0x18 ),    /* Type Offset=24 */

        /* Return value */

/* 96 */       NdrFcShort( 0x70 ),    /* Flags:  out, return, base type,
*/
#ifndef _ALPHA_
/* 98 */       NdrFcShort( 0xc ),     /* x86, MIPS, PPC Stack size/offset
= 12 */
#else
                   NdrFcShort( 0x18 ),    /* Alpha Stack size/offset =
24 */
#endif
/* 100 */      0x8,           /* FC_LONG */
                   0x0,           /* 0 */

        /* Procedure StockLevel */

/* 102 */      0x33,          /* FC_AUTO_HANDLE */
                   0x6c,          /* Old Flags:  object, Oi2 */
/* 104 */      NdrFcLong( 0x0 ),      /* 0 */
/* 108 */      NdrFcShort( 0x6 ),     /* 6 */
#ifndef _ALPHA_
/* 110 */      NdrFcShort( 0x10 ),    /* x86, MIPS, PPC Stack size/offset
= 16 */
#else
                   NdrFcShort( 0x20 ),    /* Alpha Stack size/offset =
32 */
#endif
/* 112 */      NdrFcShort( 0x8 ),     /* 8 */
/* 114 */      NdrFcShort( 0x10 ),    /* 16 */
/* 116 */      0x7,           /* Oi2 Flags:  srv must size, clt must
size, has return, */
                   0x3,           /* 3 */

        /* Parameter iSize */

/* 118 */      NdrFcShort( 0x158 ),   /* Flags:  in, out, base type,
simple ref, */
#ifndef _ALPHA_
/* 120 */      NdrFcShort( 0x4 ),     /* x86, MIPS, PPC Stack size/offset
= 4 */
#else
                   NdrFcShort( 0x8 ),     /* Alpha Stack size/offset =
8 */
#endif
/* 122 */      0x8,           /* FC_LONG */
                   0x0,           /* 0 */

        /* Parameter txn */

/* 124 */      NdrFcShort( 0x201b ),  /* Flags:  must size, must free, in,
out, srv alloc size=8 */
#ifndef _ALPHA_
/* 126 */      NdrFcShort( 0x8 ),     /* x86, MIPS, PPC Stack size/offset
= 8 */
#else
                   NdrFcShort( 0x10 ),    /* Alpha Stack size/offset =
16 */
#endif
/* 128 */      NdrFcShort( 0x6 ),     /* Type Offset=6 */
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        /* Return value */

/* 130 */      NdrFcShort( 0x70 ),    /* Flags:  out, return, base type,
*/
#ifndef _ALPHA_
/* 132 */      NdrFcShort( 0xc ),     /* x86, MIPS, PPC Stack size/offset
= 12 */
#else
                       NdrFcShort( 0x18 ),    /* Alpha Stack size/offset =
24 */
#endif
/* 134 */      0x8,           /* FC_LONG */
                       0x0,           /* 0 */

        /* Procedure OrderStatus */

/* 136 */      0x33,          /* FC_AUTO_HANDLE */
                       0x6c,          /* Old Flags:  object, Oi2 */
/* 138 */      NdrFcLong( 0x0 ),      /* 0 */
/* 142 */      NdrFcShort( 0x7 ),     /* 7 */
#ifndef _ALPHA_
/* 144 */      NdrFcShort( 0x10 ),    /* x86, MIPS, PPC Stack size/offset
= 16 */
#else
                       NdrFcShort( 0x20 ),    /* Alpha Stack size/offset =
32 */
#endif
/* 146 */      NdrFcShort( 0x8 ),     /* 8 */
/* 148 */      NdrFcShort( 0x10 ),    /* 16 */
/* 150 */      0x7,           /* Oi2 Flags:  srv must size, clt must
size, has return, */
                       0x3,           /* 3 */

        /* Parameter iSize */

/* 152 */      NdrFcShort( 0x158 ),   /* Flags:  in, out, base type,
simple ref, */
#ifndef _ALPHA_
/* 154 */      NdrFcShort( 0x4 ),     /* x86, MIPS, PPC Stack size/offset
= 4 */
#else
                       NdrFcShort( 0x8 ),     /* Alpha Stack size/offset =
8 */
#endif
/* 156 */      0x8,           /* FC_LONG */
                       0x0,           /* 0 */

        /* Parameter txn */

/* 158 */      NdrFcShort( 0x201b ),  /* Flags:  must size, must free, in,
out, srv alloc size=8 */
#ifndef _ALPHA_
/* 160 */      NdrFcShort( 0x8 ),     /* x86, MIPS, PPC Stack size/offset
= 8 */
#else
                       NdrFcShort( 0x10 ),    /* Alpha Stack size/offset =
16 */
#endif
/* 162 */      NdrFcShort( 0x6 ),     /* Type Offset=6 */

        /* Return value */

/* 164 */      NdrFcShort( 0x70 ),    /* Flags:  out, return, base type,
*/
#ifndef _ALPHA_
/* 166 */      NdrFcShort( 0xc ),     /* x86, MIPS, PPC Stack size/offset
= 12 */
#else
                       NdrFcShort( 0x18 ),    /* Alpha Stack size/offset =
24 */
#endif
/* 168 */      0x8,           /* FC_LONG */
                       0x0,           /* 0 */

        /* Procedure CallSetComplete */

/* 170 */      0x33,          /* FC_AUTO_HANDLE */
                       0x6c,          /* Old Flags:  object, Oi2 */
/* 172 */      NdrFcLong( 0x0 ),      /* 0 */
/* 176 */      NdrFcShort( 0x8 ),     /* 8 */
#ifndef _ALPHA_
/* 178 */      NdrFcShort( 0x8 ),     /* x86, MIPS, PPC Stack size/offset
= 8 */
#else
                       NdrFcShort( 0x10 ),    /* Alpha Stack size/offset =
16 */
#endif
/* 180 */      NdrFcShort( 0x0 ),     /* 0 */
/* 182 */      NdrFcShort( 0x8 ),     /* 8 */
/* 184 */      0x4,           /* Oi2 Flags:  has return, */
                       0x1,           /* 1 */

        /* Return value */

/* 186 */      NdrFcShort( 0x70 ),    /* Flags:  out, return, base type,
*/
#ifndef _ALPHA_
/* 188 */      NdrFcShort( 0x4 ),     /* x86, MIPS, PPC Stack size/offset
= 4 */
#else
                       NdrFcShort( 0x8 ),     /* Alpha Stack size/offset =
8 */
#endif
/* 190 */      0x8,           /* FC_LONG */
                       0x0,           /* 0 */
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                        0x0
        }
    };

static const MIDL_TYPE_FORMAT_STRING __MIDL_TypeFormatString =
    {
        0,
        {
                        NdrFcShort( 0x0 ),     /* 0 */
/*  2 */
                        0x11, 0x8,     /* FC_RP [simple_pointer] */
/*  4 */       0x8,           /* FC_LONG */
                        0x5c,          /* FC_PAD */
/*  6 */
                        0x11, 0x14,    /* FC_RP [alloced_on_stack]
[pointer_deref] */
/*  8 */       NdrFcShort( 0x2 ),     /* Offset= 2 (10) */
/* 10 */
                        0x13, 0x0,     /* FC_OP */
/* 12 */       NdrFcShort( 0x2 ),     /* Offset= 2 (14) */
/* 14 */
                        0x1b,          /* FC_CARRAY */
                        0x0,           /* 0 */
/* 16 */       NdrFcShort( 0x1 ),     /* 1 */
/* 18 */       0x28,          /* Corr desc:  parameter, FC_LONG */
                        0x54,          /* FC_DEREFERENCE */
#ifndef _ALPHA_
/* 20 */       NdrFcShort( 0x4 ),     /* x86, MIPS, PPC Stack size/offset
= 4 */
#else
                        NdrFcShort( 0x8 ),     /* Alpha Stack size/offset =
8 */
#endif
/* 22 */       0x2,           /* FC_CHAR */
                        0x5b,          /* FC_END */
/* 24 */
                        0x11, 0x14,    /* FC_RP [alloced_on_stack]
[pointer_deref] */
/* 26 */       NdrFcShort( 0x2 ),     /* Offset= 2 (28) */
/* 28 */
                        0x12, 0x0,     /* FC_UP */
/* 30 */       NdrFcShort( 0xfffffff0 ),      /* Offset= -16 (14) */

                        0x0
        }
    };

const CInterfaceProxyVtbl * _tpcc_com_ps_ProxyVtblList[] =
{
    ( CInterfaceProxyVtbl *) &_ITPCCProxyVtbl,
    0
};

const CInterfaceStubVtbl * _tpcc_com_ps_StubVtblList[] =
{
    ( CInterfaceStubVtbl *) &_ITPCCStubVtbl,
    0
};

PCInterfaceName const _tpcc_com_ps_InterfaceNamesList[] =
{
    "ITPCC",
    0
};


#define _tpcc_com_ps_CHECK_IID(n)     IID_GENERIC_CHECK_IID( _tpcc_com_ps,
pIID, n)

int __stdcall _tpcc_com_ps_IID_Lookup( const IID * pIID, int * pIndex )
{

    if(!_tpcc_com_ps_CHECK_IID(0))
        {
        *pIndex = 0;
        return 1;
        }

    return 0;
}

const ExtendedProxyFileInfo tpcc_com_ps_ProxyFileInfo =
{
    (PCInterfaceProxyVtblList *) & _tpcc_com_ps_ProxyVtblList,
    (PCInterfaceStubVtblList *) & _tpcc_com_ps_StubVtblList,
    (const PCInterfaceName * ) & _tpcc_com_ps_InterfaceNamesList,
    0, // no delegation
    & _tpcc_com_ps_IID_Lookup,
    1,
    2,
    0, /* table of [async_uuid] interfaces */
    0, /* Filler1 */
    0, /* Filler2 */
    0  /* Filler3 */
};


#endif /* !defined(_M_IA64) && !defined(_M_AXP64)*/


#pragma warning( disable: 4049 )  /* more than 64k source lines */

/* this ALWAYS GENERATED file contains the proxy stub code */
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
 /* File created by MIDL compiler version 5.02.0235 */
/* at Fri Aug 13 18:56:18 1999
 */
/* Compiler settings for .\src\tpcc_com_ps.idl:
    Oicf (OptLev=i2), W1, Zp8, env=Win64 (32b run,appending), ms_ext,
c_ext
    error checks: allocation ref bounds_check enum stub_data
    VC __declspec() decoration level:
         __declspec(uuid()), __declspec(selectany), __declspec(novtable)
         DECLSPEC_UUID(), MIDL_INTERFACE()
*/
//@@MIDL_FILE_HEADING(  )

#if defined(_M_IA64) || defined(_M_AXP64)
#define USE_STUBLESS_PROXY


/* verify that the <rpcproxy.h> version is high enough to compile this
file*/
#ifndef __REDQ_RPCPROXY_H_VERSION__
#define __REQUIRED_RPCPROXY_H_VERSION__ 440
#endif


#include "rpcproxy.h"
#ifndef __RPCPROXY_H_VERSION__
#error this stub requires an updated version of <rpcproxy.h>
#endif // __RPCPROXY_H_VERSION__


#include "tpcc_com_ps.h"

#define TYPE_FORMAT_STRING_SIZE   33
#define PROC_FORMAT_STRING_SIZE   193
#define TRANSMIT_AS_TABLE_SIZE    0
#define WIRE_MARSHAL_TABLE_SIZE   0

typedef struct _MIDL_TYPE_FORMAT_STRING
    {
    short          Pad;
    unsigned char  Format[ TYPE_FORMAT_STRING_SIZE ];
    } MIDL_TYPE_FORMAT_STRING;

typedef struct _MIDL_PROC_FORMAT_STRING
    {
    short          Pad;
    unsigned char  Format[ PROC_FORMAT_STRING_SIZE ];
    } MIDL_PROC_FORMAT_STRING;


extern const MIDL_TYPE_FORMAT_STRING __MIDL_TypeFormatString;
extern const MIDL_PROC_FORMAT_STRING __MIDL_ProcFormatString;


/* Object interface: IUnknown, ver. 0.0,

GUID={0x00000000,0x0000,0x0000,{0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46}}
*/


/* Object interface: ITPCC, ver. 0.0,

GUID={0xFEEE6AA2,0x84B1,0x11d2,{0xBA,0x47,0x00,0xC0,0x4F,0xBF,0xE0,0x8B}}
*/


extern const MIDL_STUB_DESC Object_StubDesc;


extern const MIDL_SERVER_INFO ITPCC_ServerInfo;

#pragma code_seg(".orpc")
static const unsigned short ITPCC_FormatStringOffsetTable[] =
    {
    0,
    34,
    68,
    102,
    136,
    170
    };

static const MIDL_SERVER_INFO ITPCC_ServerInfo =
    {
    &Object_StubDesc,
    0,
    __MIDL_ProcFormatString.Format,
    &ITPCC_FormatStringOffsetTable[-3],
    0,
    0,
    0,
    0
    };

static const MIDL_STUBLESS_PROXY_INFO ITPCC_ProxyInfo =
    {
    &Object_StubDesc,
    __MIDL_ProcFormatString.Format,
    &ITPCC_FormatStringOffsetTable[-3],
    0,
    0,
    0
    };

CINTERFACE_PROXY_VTABLE(9) _ITPCCProxyVtbl =
{
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
    &ITPCC_ProxyInfo,
    &IID_ITPCC,
    IUnknown_QueryInterface_Proxy,
    IUnknown_AddRef_Proxy,
    IUnknown_Release_Proxy ,
    (void *)-1 /* ITPCC::NewOrder */ ,
    (void *)-1 /* ITPCC::Payment */ ,
    (void *)-1 /* ITPCC::Delivery */ ,
    (void *)-1 /* ITPCC::StockLevel */ ,
    (void *)-1 /* ITPCC::OrderStatus */ ,
    (void *)-1 /* ITPCC::CallSetComplete */
};

const CInterfaceStubVtbl _ITPCCStubVtbl =
{
    &IID_ITPCC,
    &ITPCC_ServerInfo,
    9,
    0, /* pure interpreted */
    CStdStubBuffer_METHODS
};


static const MIDL_STUB_DESC Object_StubDesc =
    {
    0,
    NdrOleAllocate,
    NdrOleFree,
    0,
    0,
    0,
    0,
    0,
    __MIDL_TypeFormatString.Format,
    1, /* -error bounds_check flag */
    0x20000, /* Ndr library version */
    0,
    0x50200eb, /* MIDL Version 5.2.235 */
    0,
    0,
    0,  /* notify & notify_flag routine table */
    1,  /* Flags */
    0,  /* Reserved3 */
    0,  /* Reserved4 */
    0   /* Reserved5 */
    };

#pragma data_seg(".rdata")

#if !defined(__RPC_WIN64__)
#error  Invalid build platform for this stub.
#endif
```

```
static const MIDL_PROC_FORMAT_STRING __MIDL_ProcFormatString =
    {
        0,
        {

       /* Procedure NewOrder */

                        0x33,           /* FC_AUTO_HANDLE */
                        0x6c,           /* Old Flags:  object, Oi2 */
/*  2 */       NdrFcLong( 0x0 ),       /* 0 */
/*  6 */       NdrFcShort( 0x3 ),      /* 3 */
/*  8 */       NdrFcShort( 0x20 ),     /* ia64, axp64 Stack size/offset =
32 */
/* 10 */       NdrFcShort( 0x8 ),      /* 8 */
/* 12 */       NdrFcShort( 0x10 ),     /* 16 */
/* 14 */       0x7,            /* Oi2 Flags:  srv must size, clt must
size, has return, */
                        0x3,            /* 3 */

       /* Parameter iSize */

/* 16 */       NdrFcShort( 0x158 ),    /* Flags:  in, out, base type,
simple ref, */
/* 18 */       NdrFcShort( 0x8 ),      /* ia64, axp64 Stack size/offset = 8
*/
/* 20 */       0x8,            /* FC_LONG */
                        0x0,            /* 0 */

       /* Parameter txn */

/* 22 */       NdrFcShort( 0x201b ),  /* Flags:  must size, must free, in,
out, srv alloc size=8 */
/* 24 */       NdrFcShort( 0x10 ),     /* ia64, axp64 Stack size/offset =
16 */
/* 26 */       NdrFcShort( 0x6 ),      /* Type Offset=6 */

       /* Return value */

/* 28 */       NdrFcShort( 0x70 ),     /* Flags:  out, return, base type,
*/
/* 30 */       NdrFcShort( 0x18 ),     /* ia64, axp64 Stack size/offset =
24 */
/* 32 */       0x8,            /* FC_LONG */
                        0x0,            /* 0 */

       /* Procedure Payment */

/* 34 */       0x33,           /* FC_AUTO_HANDLE */
                        0x6c,           /* Old Flags:  object, Oi2 */
/* 36 */       NdrFcLong( 0x0 ),       /* 0 */
/* 40 */       NdrFcShort( 0x4 ),      /* 4 */
/* 42 */       NdrFcShort( 0x20 ),     /* ia64, axp64 Stack size/offset =
32 */
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
/* 44 */       NdrFcShort( 0x8 ),     /* 8 */
/* 46 */       NdrFcShort( 0x10 ),    /* 16 */
/* 48 */       0x7,           /* Oi2 Flags:  srv must size, clt must
size, has return, */
                       0x3,           /* 3 */

       /* Parameter iSize */

/* 50 */       NdrFcShort( 0x158 ),   /* Flags:  in, out, base type,
simple ref, */
/* 52 */       NdrFcShort( 0x8 ),     /* ia64, axp64 Stack size/offset = 8
*/
/* 54 */       0x8,           /* FC_LONG */
                       0x0,           /* 0 */

       /* Parameter txn */

/* 56 */       NdrFcShort( 0x201b ),  /* Flags:  must size, must free, in,
out, srv alloc size=8 */
/* 58 */       NdrFcShort( 0x10 ),    /* ia64, axp64 Stack size/offset =
16 */
/* 60 */       NdrFcShort( 0x6 ),     /* Type Offset=6 */

       /* Return value */

/* 62 */       NdrFcShort( 0x70 ),    /* Flags:  out, return, base type,
*/
/* 64 */       NdrFcShort( 0x18 ),    /* ia64, axp64 Stack size/offset =
24 */
/* 66 */       0x8,           /* FC_LONG */
                       0x0,           /* 0 */

       /* Procedure Delivery */

/* 68 */       0x33,          /* FC_AUTO_HANDLE */
                       0x6c,          /* Old Flags:  object, Oi2 */
/* 70 */       NdrFcLong( 0x0 ),      /* 0 */
/* 74 */       NdrFcShort( 0x5 ),     /* 5 */
/* 76 */       NdrFcShort( 0x20 ),    /* ia64, axp64 Stack size/offset =
32 */
/* 78 */       NdrFcShort( 0x8 ),     /* 8 */
/* 80 */       NdrFcShort( 0x8 ),     /* 8 */
/* 82 */       0x6,           /* Oi2 Flags:  clt must size, has return,
*/
                       0x3,           /* 3 */

       /* Parameter iSize */

/* 84 */       NdrFcShort( 0x148 ),   /* Flags:  in, base type, simple
ref, */
/* 86 */       NdrFcShort( 0x8 ),     /* ia64, axp64 Stack size/offset = 8
*/
/* 88 */       0x8,           /* FC_LONG */
                       0x0,           /* 0 */

       /* Parameter txn */

/* 90 */       NdrFcShort( 0x200b ),  /* Flags:  must size, must free, in,
srv alloc size=8 */
/* 92 */       NdrFcShort( 0x10 ),    /* ia64, axp64 Stack size/offset =
16 */
/* 94 */       NdrFcShort( 0x18 ),    /* Type Offset=24 */

       /* Return value */

/* 96 */       NdrFcShort( 0x70 ),    /* Flags:  out, return, base type,
*/
/* 98 */       NdrFcShort( 0x18 ),    /* ia64, axp64 Stack size/offset =
24 */
/* 100 */      0x8,           /* FC_LONG */
                       0x0,           /* 0 */

       /* Procedure StockLevel */

/* 102 */      0x33,          /* FC_AUTO_HANDLE */
                       0x6c,          /* Old Flags:  object, Oi2 */
/* 104 */      NdrFcLong( 0x0 ),      /* 0 */
/* 108 */      NdrFcShort( 0x6 ),     /* 6 */
/* 110 */      NdrFcShort( 0x20 ),    /* ia64, axp64 Stack size/offset =
32 */
/* 112 */      NdrFcShort( 0x8 ),     /* 8 */
/* 114 */      NdrFcShort( 0x10 ),    /* 16 */
/* 116 */      0x7,           /* Oi2 Flags:  srv must size, clt must
size, has return, */
                       0x3,           /* 3 */

       /* Parameter iSize */

/* 118 */      NdrFcShort( 0x158 ),   /* Flags:  in, out, base type,
simple ref, */
/* 120 */      NdrFcShort( 0x8 ),     /* ia64, axp64 Stack size/offset = 8
*/
/* 122 */      0x8,           /* FC_LONG */
                       0x0,           /* 0 */

       /* Parameter txn */

/* 124 */      NdrFcShort( 0x201b ),  /* Flags:  must size, must free, in,
out, srv alloc size=8 */
/* 126 */      NdrFcShort( 0x10 ),    /* ia64, axp64 Stack size/offset =
16 */
/* 128 */      NdrFcShort( 0x6 ),     /* Type Offset=6 */

       /* Return value */
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
/* 130 */      NdrFcShort( 0x70 ),    /* Flags:  out, return, base type,
*/
/* 132 */      NdrFcShort( 0x18 ),    /* ia64, axp64 Stack size/offset =
24 */
/* 134 */      0x8,           /* FC_LONG */
               0x0,           /* 0 */

       /* Procedure OrderStatus */

/* 136 */      0x33,          /* FC_AUTO_HANDLE */
               0x6c,          /* Old Flags:  object, Oi2 */
/* 138 */      NdrFcLong( 0x0 ),      /* 0 */
/* 142 */      NdrFcShort( 0x7 ),     /* 7 */
/* 144 */      NdrFcShort( 0x20 ),    /* ia64, axp64 Stack size/offset =
32 */
/* 146 */      NdrFcShort( 0x8 ),     /* 8 */
/* 148 */      NdrFcShort( 0x10 ),    /* 16 */
/* 150 */      0x7,           /* Oi2 Flags:  srv must size, clt must
size, has return, */
               0x3,           /* 3 */

       /* Parameter iSize */

/* 152 */      NdrFcShort( 0x158 ),   /* Flags:  in, out, base type,
simple ref, */
/* 154 */      NdrFcShort( 0x8 ),     /* ia64, axp64 Stack size/offset = 8
*/
/* 156 */      0x8,           /* FC_LONG */
               0x0,           /* 0 */

       /* Parameter txn */

/* 158 */      NdrFcShort( 0x201b ),  /* Flags:  must size, must free, in,
out, srv alloc size=8 */
/* 160 */      NdrFcShort( 0x10 ),    /* ia64, axp64 Stack size/offset =
16 */
/* 162 */      NdrFcShort( 0x6 ),     /* Type Offset=6 */

       /* Return value */

/* 164 */      NdrFcShort( 0x70 ),    /* Flags:  out, return, base type,
*/
/* 166 */      NdrFcShort( 0x18 ),    /* ia64, axp64 Stack size/offset =
24 */
/* 168 */      0x8,           /* FC_LONG */
               0x0,           /* 0 */

       /* Procedure CallSetComplete */

/* 170 */      0x33,          /* FC_AUTO_HANDLE */
               0x6c,          /* Old Flags:  object, Oi2 */
/* 172 */      NdrFcLong( 0x0 ),      /* 0 */
/* 176 */      NdrFcShort( 0x8 ),     /* 8 */
/* 178 */      NdrFcShort( 0x10 ),    /* ia64, axp64 Stack size/offset =
16 */
/* 180 */      NdrFcShort( 0x0 ),     /* 0 */
/* 182 */      NdrFcShort( 0x8 ),     /* 8 */
/* 184 */      0x4,           /* Oi2 Flags:  has return, */
               0x1,           /* 1 */

       /* Return value */

/* 186 */      NdrFcShort( 0x70 ),    /* Flags:  out, return, base type,
*/
/* 188 */      NdrFcShort( 0x8 ),     /* ia64, axp64 Stack size/offset = 8
*/
/* 190 */      0x8,           /* FC_LONG */
               0x0,           /* 0 */

               0x0
       }
    };

static const MIDL_TYPE_FORMAT_STRING __MIDL_TypeFormatString =
    {
        0,
        {
               NdrFcShort( 0x0 ),     /* 0 */
/*  2 */
               0x11, 0x8,     /* FC_RP [simple_pointer] */
/*  4 */       0x8,           /* FC_LONG */
               0x5c,          /* FC_PAD */
/*  6 */
               0x11, 0x14,    /* FC_RP [alloced_on_stack]
[pointer_deref] */
/*  8 */       NdrFcShort( 0x2 ),     /* Offset= 2 (10) */
/* 10 */
               0x13, 0x0,     /* FC_OP */
/* 12 */       NdrFcShort( 0x2 ),     /* Offset= 2 (14) */
/* 14 */
               0x1b,          /* FC_CARRAY */
               0x0,           /* 0 */
/* 16 */       NdrFcShort( 0x1 ),     /* 1 */
/* 18 */       0x28,          /* Corr desc:  parameter, FC_LONG */
               0x54,          /* FC_DEREFERENCE */
/* 20 */       NdrFcShort( 0x8 ),     /* ia64, axp64 Stack size/offset = 8
*/
/* 22 */       0x2,           /* FC_CHAR */
               0x5b,          /* FC_END */
/* 24 */
               0x11, 0x14,    /* FC_RP [alloced_on_stack]
[pointer_deref] */
/* 26 */       NdrFcShort( 0x2 ),     /* Offset= 2 (28) */
/* 28 */
               0x12, 0x0,     /* FC_UP */
/* 30 */       NdrFcShort( 0xfffffff0 ),      /* Offset= -16 (14) */
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                        0x0
        }
    };

const CInterfaceProxyVtbl * _tpcc_com_ps_ProxyVtblList[] =
{
    ( CInterfaceProxyVtbl *) &_ITPCCProxyVtbl,
    0
};

const CInterfaceStubVtbl * _tpcc_com_ps_StubVtblList[] =
{
    ( CInterfaceStubVtbl *) &_ITPCCStubVtbl,
    0
};

PCInterfaceName const _tpcc_com_ps_InterfaceNamesList[] =
{
    "ITPCC",
    0
};


#define _tpcc_com_ps_CHECK_IID(n)     IID_GENERIC_CHECK_IID( _tpcc_com_ps,
pIID, n)

int __stdcall _tpcc_com_ps_IID_Lookup( const IID * pIID, int * pIndex )
{

    if(!_tpcc_com_ps_CHECK_IID(0))
        {
        *pIndex = 0;
        return 1;
        }

    return 0;
}

const ExtendedProxyFileInfo tpcc_com_ps_ProxyFileInfo =
{
    (PCInterfaceProxyVtblList *) & _tpcc_com_ps_ProxyVtblList,
    (PCInterfaceStubVtblList *) & _tpcc_com_ps_StubVtblList,
    (const PCInterfaceName * ) & _tpcc_com_ps_InterfaceNamesList,
    0, // no delegation
    & _tpcc_com_ps_IID_Lookup,
    1,
    2,
    0, /* table of [async_uuid] interfaces */
    0, /* Filler1 */
    0, /* Filler2 */
    0  /* Filler3 */
};

#endif /* defined(_M_IA64) || defined(_M_AXP64)*/
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

Appendix B - Database Details

BACKUP.SQL

```
--  File:     BACKUP.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.20
--            Copyright Microsoft, 1999
--  Purpose:  Creates backup of tpcc database


declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

dump database tpcc to tpccback1, tpccback2 with init, stats = 1

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go
```

BACKUPDEV.SQL

```
--  File:     BACKUPDEVB.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.20
--            Copyright Microsoft, 1999
--  Purpose:  Creates tpcc database Backup Devices


use master
go

--  create backup devices

exec sp_addumpdevice 'disk','tpccback1','X:\tpccback1.dmp'
exec sp_addumpdevice 'disk','tpccback2','Y:\tpccback2.dmp'
go
```

CREATEDB.SQL

```
--  File:     CREATEDB.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.20
--            Copyright Microsoft, 1999
--  Purpose:  Creates tpcc database and backup files


use master
go

--      Create temporary table for timing

if exists ( select name from sysobjects where name = 'tpcc_timer' )
        drop table tpcc_timer
go

create table tpcc_timer
(
        start_date                              char(30),
        end_date                                char(30)
)

insert  into tpcc_timer values (0,0)
go

--      Store starting time

update  tpcc_timer
set     start_date      = (select convert(char(30), getdate(),9))
go

--  create main database files

CREATE DATABASE tpcc
ON PRIMARY
(       NAME            = MSSQL70_tpcc_root,
        FILENAME        = "C:\tpcc_root.mdf",
        SIZE            = 50MB,
        FILEGROWTH      =0),
FILEGROUP       MSSQL70_cs_fg
(       NAME            = MSSQL70_cs1,
        FILENAME        = "E:",
        SIZE            = 31000MB,
        FILEGROWTH      = 0),
(       NAME            = MSSQL70_cs2,
        FILENAME        = "F:",
        SIZE            = 31000MB,
        FILEGROWTH      = 0),
(       NAME            = MSSQL70_cs3,
        FILENAME        = "G:",
        SIZE            = 31000MB,
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        FILEGROWTH      = 0),
(       NAME            = MSSQL70_cs4,
        FILENAME        = "H:",
        SIZE            = 31000MB,
        FILEGROWTH      = 0),
(       NAME            = MSSQL70_cs5,
        FILENAME        = "I:",
        SIZE            = 31000MB,
        FILEGROWTH      = 0),
FILEGROUP       MSSQL70_misc_fg
(       NAME            = MSSQL70_misc1,
        FILENAME        = "N:",
        SIZE            = 17500MB,
        FILEGROWTH      = 0),
(       NAME            = MSSQL70_misc2,
        FILENAME        = "O:",
        SIZE            = 17500MB,
        FILEGROWTH      = 0),
(       NAME            = MSSQL70_misc3,
        FILENAME        = "P:",
        SIZE            = 17500MB,
        FILEGROWTH      = 0),
(       NAME            = MSSQL70_misc4,
        FILENAME        = "Q:",
        SIZE            = 17500MB,
        FILEGROWTH      = 0),
(       NAME            = MSSQL70_misc5,
        FILENAME        = "R:",
        SIZE            = 17500MB,
        FILEGROWTH      = 0)
LOG ON
(       NAME            =MSSQL70_tpcc_log,
        FILENAME        ="L:",
        SIZE            =80000MB,
        FILEGROWTH      =0)
go

-- Store ending time
update  tpcc_timer
set     end_date        = (select convert(char(30), getdate(),9))
go

select "Elapsed time (in seconds): ", datediff(second,(select start_date
from tpcc_timer),(select end_date from tpcc_timer))

--      remove temporary table

if exists ( select name from sysobjects where name = 'tpcc_timer' )
        drop table tpcc_timer
go
```

DBOPT1.SQL

```
--  File:     DBOPT1.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.21
--            Copyright Microsoft, 1999, 2000
--  Purpose:  Sets database options for data load


use master
go

exec sp_dboption tpcc,'select into/bulkcopy',true
exec sp_dboption tpcc,'trunc. log on chkpt.',true
go

use tpcc
go

checkpoint
go
```

DBOPT2.SQL

```
--  File:     DBOPT2.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.21
--            Copyright Microsoft, 1999, 2000
--  Purpose:  Resets database options after data load

sp_dboption tpcc,'select into/bulkcopy',FALSE
GO

sp_dboption tpcc,'trunc. log on chkpt.',FALSE
GO

USE tpcc
GO

CHECKPOINT
GO

sp_configure 'allow updates',1
GO

RECONFIGURE WITH OVERRIDE
GO

DECLARE @msg            varchar(50)
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
IF (SELECT (SUBSTRING((SELECT @@version),1,26))) = 'Microsoft SQL Server
2000'
       BEGIN
              --                                                --
              --        OPTIONS FOR SQL SERVER 8.0              --
              -- Set option values for user-defined indexes --
              --                                                --

              SET    @msg    = ' '
              PRINT  @msg                                     --
              SET    @msg    = 'Setting SQL Server 8.0 indexoptions'
              PRINT  @msg
              SET    @msg    = ' '
              PRINT  @msg                                     --

              EXEC sp_indexoption   'customer',    'DisAllowPageLocks',
       TRUE
              EXEC sp_indexoption   'district',    'DisAllowPageLocks',
       TRUE
              EXEC sp_indexoption   'warehouse',   'DisAllowPageLocks',
       TRUE
              EXEC sp_indexoption   'stock',       'DisAllowPageLocks',
       TRUE
              EXEC sp_indexoption   'order_line',  'DisAllowRowLocks',
       TRUE
              EXEC sp_indexoption   'orders',      'DisAllowRowLocks',
       TRUE
              EXEC sp_indexoption   'new_order',   'DisAllowRowLocks',
       TRUE
              EXEC sp_indexoption   'item',        'DisAllowRowLocks',
       TRUE
              EXEC sp_indexoption   'item',        'DisAllowPageLocks',
       TRUE
       END
ELSE
       BEGIN
              --                                                --
              --        OPTIONS FOR SQL SERVER 7.0              --
              -- Set option values for user-defined indexes --
              --

              SET    @msg    = ' '
              PRINT  @msg                                     --
              SET    @msg    = 'Setting SQL Server 7.0 indexoptions'
              PRINT  @msg
              SET    @msg    = ' '
              PRINT  @msg                                     --

              EXEC sp_indexoption   'customer',    'AllowPageLocks',
       FALSE
              EXEC sp_indexoption   'district',    'AllowPageLocks',
       FALSE
              EXEC sp_indexoption   'warehouse',   'AllowPageLocks',
       FALSE
              EXEC sp_indexoption   'stock',       'AllowPageLocks',
       FALSE
              EXEC sp_indexoption   'order_line',  'AllowRowLocks',
       FALSE
              EXEC sp_indexoption   'orders',      'AllowRowLocks',
       FALSE
              EXEC sp_indexoption   'new_order',   'AllowRowLocks',
       FALSE
              EXEC sp_indexoption   'item',        'AllowRowLocks',
       FALSE
              EXEC sp_indexoption   'item',        'AllowPageLocks',
       FALSE

       END
GO

Print ' '
Print '******************'
Print 'Pre-specified Locking Hierarchy:'
Print '   Lockflag = 0 ==> No pre-specified hierarchy'
Print '   Lockflag = 1 ==> Lock at Page-level then Table-level'
Print '   Lockflag = 2 ==> Lock at Row-level then Table-level'
Print '   Lockflag = 3 ==> Lock at Table-level'
Print ' '

SELECT  name,lockflags
FROM    sysindexes
WHERE   object_id('warehouse') = id OR
        object_id('district')  = id OR
        object_id('customer')  = id OR
        object_id('stock')     = id OR
        object_id('orders')    = id OR
        object_id('order_line')       = id OR
        object_id('history')   = id OR
        object_id('new_order') = id OR
        object_id('item')      = id
ORDER   BY lockflags asc
GO

sp_configure 'allow updates',0
GO

RECONFIGURE WITH OVERRIDE
GO

EXEC sp_dboption tpcc, 'auto update statistics',     FALSE
EXEC sp_dboption tpcc, 'auto create statistics',     FALSE
GO
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
EXEC sp_tableoption    'district',    'pintable',true
EXEC sp_tableoption    'warehouse',   'pintable',true
EXEC sp_tableoption    'new_order',   'pintable',true
EXEC sp_tableoption    'item',        'pintable',true
GO
```

REMOVEDB.SQL

```
--  File:     REMOVEDB.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.20
--            Copyright Microsoft, 1999
--  Purpose:  Removes tpcc database and backup files


use master
go

--  remove any existing database and backup files

exec sp_dbremove tpcc, dropdev
go

exec sp_dropdevice 'tpccback1'
exec sp_dropdevice 'tpccback2'
go
```

RESTORE.SQL

```
--  File:     RESTORE.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.20
--            Copyright Microsoft, 1999
--  Purpose:  Loads database backup from backup files


declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

load database tpcc from tpccback1, tpccback2, with stats = 1

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go
```

VERIFYTPCCLOAD.SQL

```
--  File:     VERIFYTPCCLOAD.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.21
--            Copyright Microsoft, 1999, 2000
--  Purpose:  Performs series of TPCC database checks to verify
--            that database load completed correctly

print   " "
select  convert(char(30), getdate(),9)
print   " "

use tpcc
go

--      ***********************************
--
--      Check rows per table from SYSINDEXES
--
--      ***********************************

print   'WAREHOUSE TABLE'

select  rows
from    sysindexes
where   id      = object_id("warehouse")
go

print   'DISTRICT TABLE = (10 * No of warehouses)'

select  rows
from    sysindexes
where   id      =object_id("district")
go

print   'ITEM TABLE = 100,000'

select  rows
from    sysindexes
where   id      =object_id("item")
go

print   'CUSTOMER TABLE = (30,000 * No of warehouses)'

select  rows
from    sysindexes
where   id      =object_id("customer")
go

print 'ORDERS TABLE = (30,000 * No of warehouses)'

select  rows
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
from    sysindexes
where   id      =object_id("orders")
go

print   'HISTORY TABLE = (30,000 * No of warehouses)'

select  rows
from    sysindexes
where   id      =object_id("history")
go

print   'STOCK TABLE = (100,000 * No of warehouses)'

select  rows
from    sysindexes
where   id      =object_id("stock")
go

print   'ORDER_LINE TABLE = (300,000 * No of warehouses + some change)'

select  rows
from    sysindexes
where   id      =object_id("order_line")
go

print   'NEW_ORDER TABLE = (9000 * No of warehouses)'

select  rows
from    sysindexes
where   id      =object_id("new_order")
go

--      *************
--
--      Check indices
--
--      *************

print '***************Index Check*****************'

use tpcc
go

sp_helpindex   customer
go

sp_helpindex   stock
go

sp_helpindex   district
go

sp_helpindex   item
go

sp_helpindex   new_order
go

sp_helpindex   orders
go

sp_helpindex   order_line
go

sp_helpindex   warehouse
go
```

IDXCUSCL.SQL

```
--  File:     IDXCUSCL.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.20
--            Copyright Microsoft, 1999
--  Purpose:  Creates clustered index on customer table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where name = 'customer_c1' )
       drop index customer.customer_c1

create unique clustered index customer_c1 on customer(c_w_id, c_d_id,
c_id)
       on MSSQL70_cs_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go
```

IDXCUSNC.SQL

```
--  File:     IDXCUSNC.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.20
--            Copyright Microsoft, 1999
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
--  Purpose:  Creates non-clustered index on customer table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where  name = 'customer_nc1' )
       drop index customer.customer_nc1

create unique nonclustered index customer_nc1 on customer(c_w_id, c_d_id,
c_last, c_first, c_id)
       on MSSQL70_cs_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go
```

IDXDISCL.SQL

```
--  File:     IDXDISCL.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.20
--            Copyright Microsoft, 1999
--  Purpose:  Creates clustered index on district table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where name = 'district_c1' )
       drop index district.district_c1

create unique clustered index  district_c1 on district(d_w_id, d_id)
       with fillfactor=100 on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go
```

IDXITMCL.SQL

```
--  File:     IDXITMCL.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.20
--            Copyright Microsoft, 1999
--  Purpose:  Creates clustered index on item table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where  name = 'item_c1' )
       drop index item.item_c1

create unique clustered index item_c1 on item(i_id)
       on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go
```

IDXNODCL.SQL

```
--  File:     IDXNODCL.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.20
--            Copyright Microsoft, 1999
--  Purpose:  Creates clustered index on new_order table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
if exists ( select name from sysindexes where  name = 'new_order_c1' )
       drop index new_order.new_order_c1

create unique clustered index new_order_c1 on new_order(no_w_id, no_d_id,
no_o_id)
       on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go
```

IDXODLCL.SQL

```
--  File:     IDXODLCL.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.20
--            Copyright Microsoft, 1999
--  Purpose:  Creates clustered index on order_line table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where  name = 'order_line_c1' )
       drop index order_line.order_line_c1

create unique clustered index order_line_c1 on order_line(ol_w_id,
ol_d_id, ol_o_id, ol_number)
       on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go
```

IDXORDCL.SQL

```
--  File:     IDXORDCL.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.20
--            Copyright Microsoft, 1999
--  Purpose:  Creates clustered index on orders table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where  name = 'orders_c1' )
       drop index orders.orders_c1

create unique clustered index orders_c1 on orders(o_w_id, o_d_id, o_id)
       on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go
```

IDXORDNC.SQL

```
--  File:     IDXORDNC.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.20
--            Copyright Microsoft, 1999
--  Purpose:  Creates non-clustered index on orders table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where  name = 'orders_nc1' )
       drop index orders.orders_nc1

create index orders_nc1 on orders(o_w_id, o_d_id, o_c_id, o_id)
       on MSSQL70_misc_fg
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go
```

IDXSTKCL.SQL

```
--  File:     IDXSTKCL.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.20
--            Copyright Microsoft, 1999
--  Purpose:  Creates clustered index on stock table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where  name = 'stock_c1' )
       drop index stock.stock_c1

create unique clustered index stock_c1 on stock(s_i_id, s_w_id)
       on MSSQL70_cs_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go
```

IDXWARCL.SQL

```
--  File:     IDXWARCL.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.20
--            Copyright Microsoft, 1999
--  Purpose:  Creates clustered index on warehouse table


use tpcc
go

declare @startdate datetime
declare @enddate datetime
select @startdate = getdate()
select "Start date:", convert(varchar(30),@startdate,9)

if exists ( select name from sysindexes where name = 'warehouse_c1' )
       drop index warehouse.warehouse_c1

create unique clustered index warehouse_c1 on warehouse(w_id)
       with fillfactor=100 on MSSQL70_misc_fg

select @enddate = getdate()
select "End date: ", convert(varchar(30),@enddate,9)
select "Elapsed time (in seconds): ", datediff(second, @startdate,
@enddate)

go
```

TABLES.SQL

```
--  File:     TABLES.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.20
--            Copyright Microsoft, 1999
--  Purpose:  Creates TPC-C tables


use tpcc
go

--
--  Remove all existing TPC-C tables
--

if exists ( select name from sysobjects where name = 'warehouse' )
      drop table warehouse
go
if exists ( select name from sysobjects where name = 'district' )
      drop table district
go
if exists ( select name from sysobjects where name = 'customer' )
      drop table customer
go
if exists ( select name from sysobjects where name = 'history' )
      drop table history
go
if exists ( select name from sysobjects where name = 'new_order' )
      drop table new_order
go
if exists ( select name from sysobjects where name = 'orders' )
      drop table orders
go
if exists ( select name from sysobjects where name = 'order_line' )
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
	drop table order_line
go
if exists ( select name from sysobjects where name = 'item' )
	drop table item
go
if exists ( select name from sysobjects where name = 'stock' )
	drop table stock
go

--
--  Create new tables
--

create table warehouse
(
	w_id				smallint,
	w_name				char(10),
	w_street_1			char(20),
	w_street_2			char(20),
	w_city				char(20),
	w_state				char(2),
	w_zip				char(9),
	w_tax				numeric(4,4),
	w_ytd				numeric(12,2)
) on MSSQL70_misc_fg
go

create table district
(
	d_id				tinyint,
	d_w_id				smallint,
	d_name				char(10),
	d_street_1			char(20),
	d_street_2			char(20),
	d_city				char(20),
	d_state				char(2),
	d_zip				char(9),
	d_tax				numeric(4,4),
	d_ytd				numeric(12,2),
	d_next_o_id			int
) on MSSQL70_misc_fg
go

create table customer
(
	c_id				int,
	c_d_id				tinyint,
	c_w_id				smallint,
	c_first				char(16),
	c_middle			char(2),
	c_last				char(16),
	c_street_1			char(20),
	c_street_2			char(20),
	c_city				char(20),
	c_state				char(2),
	c_zip				char(9),
	c_phone				char(16),
	c_since				datetime,
	c_credit			char(2),
	c_credit_lim		numeric(12,2),
	c_discount			numeric(4,4),
	c_balance			numeric(12,2),
	c_ytd_payment		numeric(12,2),
	c_payment_cnt		smallint,
	c_delivery_cnt		smallint,
	c_data				char(500)
) on MSSQL70_cs_fg
go

create table history
(
	h_c_id				int,
	h_c_d_id			tinyint,
	h_c_w_id			smallint,
	h_d_id				tinyint,
	h_w_id				smallint,
	h_date				datetime,
	h_amount			numeric(6,2),
	h_data				char(24)
) on MSSQL70_misc_fg
go

create table new_order
(
	no_o_id				int,
	no_d_id				tinyint,
	no_w_id				smallint
) on MSSQL70_misc_fg
go

create table orders
(
	o_id				int,
	o_d_id				tinyint,
	o_w_id				smallint,
	o_c_id				int,
	o_entry_d			datetime,
	o_carrier_id		tinyint,
	o_ol_cnt			tinyint,
	o_all_local			tinyint
) on MSSQL70_misc_fg
go

create table order_line
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
(
        ol_o_id                         int,
        ol_d_id                         tinyint,
        ol_w_id                         smallint,
        ol_number                       tinyint,
        ol_i_id                         int,
        ol_supply_w_id          smallint,
        ol_delivery_d           datetime,
        ol_quantity                     smallint,
        ol_amount                       numeric(6,2),
        ol_dist_info            char(24)
) on MSSQL70_misc_fg
go

create table item
(
        i_id                            int,
        i_im_id                         int,
        i_name                          char(24),
        i_price                         numeric(5,2),
        i_data                          char(50)
) on MSSQL70_misc_fg
go

create table stock
(
        s_i_id                          int,
        s_w_id                          smallint,
        s_quantity                      smallint,
        s_dist_01                       char(24),
        s_dist_02                       char(24),
        s_dist_03                       char(24),
        s_dist_04                       char(24),
        s_dist_05                       char(24),
        s_dist_06                       char(24),
        s_dist_07                       char(24),
        s_dist_08                       char(24),
        s_dist_09                       char(24),
        s_dist_10                       char(24),
        s_ytd                           int,
        s_order_cnt                     smallint,
        s_remote_cnt            smallint,
        s_data                          char(50)
) on MSSQL70_cs_fg
go
```

DELIVERY.SQL

```
--  File:     DELIVERY.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.21.000
--            Copyright Microsoft, 1999, 2000
--  Purpose:  Creates delivery transaction stored procedure
--
--      Interface Level: 4.10.000

use tpcc
go

if exists (select name from sysobjects where name = "tpcc_delivery" )
        drop procedure tpcc_delivery
go

create proc tpcc_delivery     @w_id           smallint,

                              @o_carrier_id   smallint
as

declare @d_id  tinyint,
        @o_id  int,
        @c_id  int,
        @total numeric(12,2),
        @oid1  int,
        @oid2  int,
        @oid3  int,
        @oid4  int,
        @oid5  int,
        @oid6  int,
        @oid7  int,
        @oid8  int,
        @oid9  int,
        @oid10 int

select @d_id = 0

begin tran d

        while (@d_id < 10)
        begin

                select @d_id  = @d_id + 1,
                       @total = 0,
                       @o_id  = 0

               select  top 1
                       @o_id   = no_o_id
               from    new_order (serializable updlock)
               where   no_w_id = @w_id and
                       no_d_id = @d_id
               order   by no_o_id asc

                if (@@rowcount <> 0)
               begin
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
--  claim the order for this district

                    delete  new_order
                    where   no_w_id = @w_id and
                            no_d_id = @d_id and
                            no_o_id = @o_id

--  set carrier_id on this order (and get customer id)

                    update  orders
                    set     o_carrier_id    = @o_carrier_id,
                            @c_id           = o_c_id
                    where   o_w_id          = @w_id and
                            o_d_id          = @d_id and
                            o_id            = @o_id

--  set date in all lineitems for this order (and sum amounts)

                    update  order_line
                    set     ol_delivery_d   = getdate(),
                            @total          = @total + ol_amount
                    where   ol_w_id         = @w_id and
                            ol_d_id         = @d_id and
                            ol_o_id         = @o_id

--  accummulate lineitem amounts for this order into customer

                    update  customer
                    set     c_balance       = c_balance + @total,
                            c_delivery_cnt  = c_delivery_cnt + 1

                    where   c_w_id          = @w_id and
                            c_d_id          = @d_id and
                            c_id            = @c_id

            end

        select @oid1 = case @d_id when  1  then @o_id else @oid1 end,
               @oid2 = case @d_id when  2  then @o_id else @oid2 end,
               @oid3 = case @d_id when  3  then @o_id else @oid3 end,
               @oid4 = case @d_id when  4  then @o_id else @oid4 end,
               @oid5 = case @d_id when  5  then @o_id else @oid5 end,
               @oid6 = case @d_id when  6  then @o_id else @oid6 end,
               @oid7 = case @d_id when  7  then @o_id else @oid7 end,
               @oid8 = case @d_id when  8  then @o_id else @oid8 end,
               @oid9 = case @d_id when  9  then @o_id else @oid9 end,
               @oid10 = case @d_id when 10 then @o_id else @oid10 end

    end

commit tran d
```

```
-- return delivery data to client

select @oid1,
       @oid2,
       @oid3,
       @oid4,
       @oid5,
       @oid6,
       @oid7,
       @oid8,
       @oid9,
       @oid10

go
```

NEWORD.SQL

```
--  File:     NEWORD.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.21.000
--            Copyright Microsoft, 1999, 2000
--  Purpose:  Creates new order transaction stored procedure
--
--      Interface Level: 4.10.000

use tpcc
go

if exists ( select name from sysobjects where name = "tpcc_neworder" )
       drop procedure tpcc_neworder
go

create proc tpcc_neworder
                                    @w_id           smallint,
                                    @d_id           tinyint,
                                    @c_id           int,
                                    @o_ol_cnt       tinyint,
                                    @o_all_local    tinyint,
                                    @i_id1  int = 0, @s_w_id1  smallint
= 0, @ol_qty1  smallint = 0,
                                    @i_id2  int = 0, @s_w_id2  smallint
= 0, @ol_qty2  smallint = 0,
                                    @i_id3  int = 0, @s_w_id3  smallint
= 0, @ol_qty3  smallint = 0,
                                    @i_id4  int = 0, @s_w_id4  smallint
= 0, @ol_qty4  smallint = 0,
                                    @i_id5  int = 0, @s_w_id5  smallint
= 0, @ol_qty5  smallint = 0,
                                    @i_id6  int = 0, @s_w_id6  smallint
= 0, @ol_qty6  smallint = 0,
                                    @i_id7  int = 0, @s_w_id7  smallint
= 0, @ol_qty7  smallint = 0,
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                                        @i_id8  int = 0, @s_w_id8  smallint
= 0, @ol_qty8  smallint = 0,
                                        @i_id9  int = 0, @s_w_id9  smallint
= 0, @ol_qty9  smallint = 0,
                                        @i_id10 int = 0, @s_w_id10 smallint
= 0, @ol_qty10 smallint = 0,
                                        @i_id11 int = 0, @s_w_id11 smallint
= 0, @ol_qty11 smallint = 0,
                                        @i_id12 int = 0, @s_w_id12 smallint
= 0, @ol_qty12 smallint = 0,
                                        @i_id13 int = 0, @s_w_id13 smallint
= 0, @ol_qty13 smallint = 0,
                                        @i_id14 int = 0, @s_w_id14 smallint
= 0, @ol_qty14 smallint = 0,
                                        @i_id15 int = 0, @s_w_id15 smallint
= 0, @ol_qty15 smallint = 0


as
declare @w_tax          numeric(4,4),
        @d_tax          numeric(4,4),
        @c_last         char(16),
        @c_credit       char(2),
        @c_discount     numeric(4,4),
        @i_price        numeric(5,2),
        @i_name         char(24),
        @i_data         char(50),
        @o_entry_d      datetime,
        @remote_flag    int,
        @s_quantity     smallint,
        @s_data         char(50),
        @s_dist         char(24),
        @li_no          int,
        @o_id           int,
        @commit_flag    tinyint,
        @li_id          int,
        @li_s_w_id      smallint,
        @li_qty         smallint,
        @ol_number      int,
        @c_id_local     int

begin

begin transaction n

--  get district tax and next availible order id and update
--  plus initialize local variables

        update  district
        set     @d_tax       = d_tax,
                @o_id        = d_next_o_id,
                d_next_o_id  = d_next_o_id + 1,
                @o_entry_d   = getdate(),
                @li_no       = 0,
                @commit_flag = 1
        where   d_w_id       = @w_id and
                d_id         = @d_id

--  process orderlines

        while (@li_no < @o_ol_cnt)
        begin

                select @li_no = @li_no + 1

--  set i_id, s_w_id, and qty for this lineitem

                select  @li_id = case @li_no
                                 when 1 then @i_id1
                                 when 2 then @i_id2
                                 when 3 then @i_id3
                                 when 4 then @i_id4
                                 when 5 then @i_id5
                                 when 6 then @i_id6
                                 when 7 then @i_id7
                                 when 8 then @i_id8
                                 when 9 then @i_id9
                                 when 10 then @i_id10
                                 when 11 then @i_id11
                                 when 12 then @i_id12
                                 when 13 then @i_id13
                                 when 14 then @i_id14
                                 when 15 then @i_id15
                                 end,

                        @li_s_w_id = case @li_no
                                     when 1 then @s_w_id1
                                     when 2 then @s_w_id2
                                     when 3 then @s_w_id3
                                     when 4 then @s_w_id4
                                     when 5 then @s_w_id5
                                     when 6 then @s_w_id6
                                     when 7 then @s_w_id7
                                     when 8 then @s_w_id8
                                     when 9 then @s_w_id9
                                     when 10 then @s_w_id10
                                     when 11 then @s_w_id11
                                     when 12 then @s_w_id12
                                     when 13 then @s_w_id13
                                     when 14 then @s_w_id14
                                     when 15 then @s_w_id15
                                     end,

                        @li_qty = case  @li_no
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                                when 1 then @ol_qty1
                                when 2 then @ol_qty2
                                when 3 then @ol_qty3
                                when 4 then @ol_qty4
                                when 5 then @ol_qty5
                                when 6 then @ol_qty6
                                when 7 then @ol_qty7
                                when 8 then @ol_qty8
                                when 9 then @ol_qty9
                                when 10 then @ol_qty10
                                when 11 then @ol_qty11
                                when 12 then @ol_qty12
                                when 13 then @ol_qty13
                                when 14 then @ol_qty14
                                when 15 then @ol_qty15
                                end

--  get item data (no one updates item)

               select  @i_price = i_price,
                       @i_name  = i_name,
                       @i_data  = i_data
               from    item (tablock repeatableread)
               where   i_id = @li_id

--  update stock values

               update  stock
               set     s_ytd          = s_ytd + @li_qty,
                       @s_quantity    = s_quantity   = s_quantity -
@li_qty +
                                                  case when (s_quantity -
@li_qty < 10) then 91 else 0 end,
                       s_order_cnt    = s_order_cnt + 1,
                       s_remote_cnt   = s_remote_cnt + case when
(@li_s_w_id = @w_id) then 0 else 1 end,
                       @s_data        = s_data,
                       @s_dist        = case @d_id
                                          when 1  then s_dist_01
                                          when 2  then s_dist_02
                                          when 3  then s_dist_03
                                          when 4  then s_dist_04
                                          when 5  then s_dist_05
                                          when 6  then s_dist_06
                                          when 7  then s_dist_07
                                          when 8  then s_dist_08
                                          when 9  then s_dist_09
                                          when 10 then s_dist_10
                                          end
               where   s_i_id         = @li_id and
                       s_w_id         = @li_s_w_id
```

```
--  if there actually is a stock (and item) with these ids, go to work

               if (@@rowcount > 0)
               begin

--  insert order_line data (using data from item and stock)

                       insert into order_line values(@o_id,
                                                     @d_id,
                                                     @w_id,
                                                     @li_no,
                                                     @li_id,
                                                     @li_s_w_id,
                                                     "dec 31, 1899",
                                                     @li_qty,
                                                     @i_price * @li_qty,
                                                     @s_dist)

--  send line-item data to client

                       select  @i_name,
                               @s_quantity,
                               b_g = case when (
(patindex("%ORIGINAL%",@i_data) > 0) and

(patindex("%ORIGINAL%",@s_data) > 0) )
                                        then "B" else "G" end,
                               @i_price,
                               @i_price * @li_qty

               end
               else
               begin

--  no item (or stock) found - triggers rollback condition


                       select "",0,"",0,0
                       select @commit_flag = 0

               end
       end

--  get customer last name, discount, and credit rating

       select  @c_last     = c_last,
               @c_discount = c_discount,
               @c_credit   = c_credit,
               @c_id_local = c_id
       from    customer (repeatableread)
       where   c_id           = @c_id and
               c_w_id         = @w_id and
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                c_d_id          = @d_id

--  insert fresh row into orders table

        insert into orders values (   @o_id,
                                      @d_id,
                                      @w_id,
                                      @c_id_local,
                                      @o_entry_d,
                                      0,
                                      @o_ol_cnt,
                                      @o_all_local)

--  insert corresponding row into new-order table

        insert into new_order values (        @o_id,
                                      @d_id,
                                      @w_id)

--  select warehouse tax

        select  @w_tax  = w_tax
        from    warehouse (repeatableread)
        where   w_id    = @w_id

        if (@commit_flag = 1)
                commit transaction n
        else

--  all that work for nuthin!!!

                rollback transaction n

--  return order data to client

        select  @w_tax,
                @d_tax,
                @o_id,
                @c_last,
                @c_discount,
                @c_credit,
                @o_entry_d,
                @commit_flag

end

go
```

ORDSTAT.SQL

```
--  File:     ORDSTAT.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.21.000
--            Copyright Microsoft, 1999, 2000
--  Purpose:  Creates order status transaction stored procedure
--
--      Interface Level: 4.10.000

use tpcc
go

if exists ( select name from sysobjects where name = "tpcc_orderstatus" )
        drop procedure   tpcc_orderstatus
go

create proc tpcc_orderstatus  @w_id   smallint,
                              @d_id   tinyint,

                              @c_id   int,
                              @c_last char(16) = ""

as

declare @c_balance            numeric(12,2),
        @c_first              char(16),
        @c_middle             char(2),
        @o_id                 int,
        @o_entry_d            datetime,
        @o_carrier_id         smallint,
        @cnt                  smallint

begin tran o

if (@c_id = 0)
        begin

--  get customer id and info using last name

                select  @cnt    = (count(*)+1)/2
                from    customer (repeatableread)
                where   c_last  = @c_last and
                        c_w_id  = @w_id and
                        c_d_id  = @d_id

                set     rowcount @cnt

                select  @c_id           = c_id,
                        @c_balance      = c_balance,
                           @c_first     = c_first,
                           @c_last      = c_last,
                           @c_middle    = c_middle
                from    customer (repeatableread)
                where   c_last          = @c_last and
                        c_w_id          = @w_id and
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                c_d_id          = @d_id
        order   by c_w_id, c_d_id, c_last, c_first

        set     rowcount 0
    end

    else

    begin

--  get customer info if by id

        select  @c_balance      = c_balance,
                @c_first        = c_first,
                @c_middle       = c_middle,
                @c_last         = c_last
        from    customer (repeatableread)
        where   c_id            = @c_id and
                c_d_id          = @d_id and
                c_w_id          = @w_id

        select  @cnt    = @@rowcount

    end

--  if no such customer

    if (@cnt = 0)
    begin
        raiserror("Customer not found",18,1)
        goto custnotfound
    end

--  get order info

    select  @o_id           = o_id,
            @o_entry_d      = o_entry_d,
            @o_carrier_id   = o_carrier_id
    from    orders (serializable)
    where   o_c_id          = @c_id and
            o_d_id          = @d_id and
            o_w_id          = @w_id
    order   by o_id asc

--  select order lines for the current order

    select  ol_supply_w_id,
            ol_i_id,
            ol_quantity,
            ol_amount,
            ol_delivery_d
    from    order_line (repeatableread)
    where   ol_o_id = @o_id and
            ol_d_id = @d_id and
            ol_w_id = @w_id

custnotfound:

commit tran o

--  return data to client

select  @c_id,
        @c_last,
        @c_first,
        @c_middle,
        @o_entry_d,
        @o_carrier_id,
        @c_balance,
        @o_id

go
```

PAYMENT.SQL

```
--  File:     PAYMENT.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.21.000
--            Copyright Microsoft, 1999, 2000
--  Purpose:  Creates payment transaction stored procedure
--
--      Interface Level: 4.10.000

use tpcc
go

if exists (select name from sysobjects where name = "tpcc_payment" )
        drop procedure tpcc_payment
go

create proc tpcc_payment        @w_id           smallint,
                                @c_w_id         smallint,
                                @h_amount       numeric(6,2),
                                @d_id           tinyint,
                                @c_d_id         tinyint,
                                @c_id           int,
                                @c_last         char(16) = ""


as
declare  @w_street_1     char(20),
         @w_street_2     char(20),
         @w_city         char(20),
         @w_state        char(2),
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        @w_zip          char(9),
        @w_name         char(10),
        @d_street_1     char(20),
        @d_street_2     char(20),
        @d_city         char(20),
        @d_state        char(2),
        @d_zip          char(9),
        @d_name         char(10),
        @c_first        char(16),
        @c_middle       char(2),
        @c_street_1     char(20),
        @c_street_2     char(20),
        @c_city         char(20),
        @c_state        char(2),
        @c_zip          char(9),
        @c_phone        char(16),
        @c_since        datetime,
        @c_credit       char(2),
        @c_credit_lim   numeric(12,2),
        @c_balance      numeric(12,2),
        @c_discount     numeric(4,4),
        @data           char(500),
        @c_data         char(500),
        @datetime       datetime,
        @w_ytd          numeric(12,2),
        @d_ytd          numeric(12,2),
        @cnt            smallint,
        @val            smallint,
        @screen_data   char(200),
       @d_id_local    tinyint,
       @w_id_local    smallint,
       @c_id_local    int

select @screen_data = ""

begin tran p

--  get payment date

       select  @datetime = getdate()

       if (@c_id = 0)
       begin

--  get customer id and info using last name

              select  @cnt    = count(*)
              from    customer (repeatableread)
              where   c_last  = @c_last and
                      c_w_id  = @c_w_id and
                      c_d_id  = @c_d_id

              select  @val = (@cnt + 1) / 2
              set     rowcount @val

              select  @c_id   = c_id
              from    customer (repeatableread)
              where   c_last  = @c_last and
                      c_w_id  = @c_w_id and
                      c_d_id  = @c_d_id
              order   by c_last, c_first

              set     rowcount 0
       end

--  get customer info and update balances

       update  customer
       set     @c_balance      = c_balance    = c_balance - @h_amount,
               c_payment_cnt   = c_payment_cnt + 1,
               c_ytd_payment   = c_ytd_payment + @h_amount,
               @c_first        = c_first,
               @c_middle       = c_middle,
               @c_last         = c_last,
               @c_street_1     = c_street_1,
               @c_street_2     = c_street_2,
               @c_city         = c_city,
               @c_state        = c_state,
               @c_zip          = c_zip,
               @c_phone        = c_phone,
               @c_credit       = c_credit,
               @c_credit_lim   = c_credit_lim,
               @c_discount     = c_discount,
               @c_since        = c_since,
               @data           = c_data,
               @c_id_local     = c_id
       where   c_id            = @c_id and
               c_w_id          = @c_w_id and
               c_d_id          = @c_d_id

--  if customer has bad credit get some more info

       if (@c_credit = "BC")
       begin

--     compute new info

              select @c_data = convert(char(5),@c_id) +
                               convert(char(4),@c_d_id) +
                               convert(char(5),@c_w_id) +
                               convert(char(4),@d_id) +
                               convert(char(5),@w_id) +
                               convert(char(19),@h_amount) +
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                                substring(@data, 1, 458)

--  update customer info

                update  customer
                set     c_data  = @c_data
                where   c_id    = @c_id and
                        c_w_id  = @c_w_id and
                        c_d_id  = @c_d_id

                select  @screen_data = substring (@c_data,1,200)
        end

--  get district data and update year-to-date

        update  district
        set     d_ytd           = d_ytd + @h_amount,
                @d_street_1     = d_street_1,
                @d_street_2     = d_street_2,
                @d_city         = d_city,
                @d_state        = d_state,
                @d_zip          = d_zip,
                @d_name         = d_name,
                @d_id_local     = d_id
        where   d_w_id          = @w_id and
                d_id            = @d_id

--  get warehouse data and update year-to-date

        update  warehouse
        set     w_ytd           = w_ytd + @h_amount,
                @w_street_1     = w_street_1,
                @w_street_2     = w_street_2,
                @w_city         = w_city,
                @w_state        = w_state,
                @w_zip          = w_zip,
                @w_name         = w_name,
                @w_id_local     = w_id
        where   w_id            = @w_id

--  create history record

        insert into history values (  @c_id_local,
                                      @c_d_id,
                                      @c_w_id,
                                      @d_id_local,
                                      @w_id_local,
                                      @datetime,
                                      @h_amount,
                                      @w_name + "    " + @d_name)
commit tran p
```

```
--  return data to client

select  @c_id,
        @c_last,
        @datetime,
        @w_street_1,
        @w_street_2,
        @w_city,
        @w_state,
        @w_zip,
        @d_street_1,
        @d_street_2,
        @d_city,
        @d_state,
        @d_zip,
        @c_first,
        @c_middle,
        @c_street_1,
        @c_street_2,
        @c_city,
        @c_state,
        @c_zip,
        @c_phone,
        @c_since,
        @c_credit,
        @c_credit_lim,
        @c_discount,
        @c_balance,
        @screen_data

go
```

STOCKLEV.SQL

```
--  File:     STOCKLEV.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.21.000
--            Copyright Microsoft, 1999, 2000
--  Purpose:  Creates stock level transaction stored procedure
--
--      Interface Level: 4.10.000

use tpcc
go

if exists (select name from sysobjects where name = "tpcc_stocklevel" )
        drop procedure tpcc_stocklevel
go

create proc tpcc_stocklevel   @w_id          smallint,
                              @d_id          tinyint,
                              @threshhold    smallint
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
as

declare @o_id_low int,
        @o_id_high int

select  @o_id_low       = (d_next_o_id - 20),
        @o_id_high      = (d_next_o_id - 1)
from    district
where   d_w_id          = @w_id and
        d_id            = @d_id

select  count(distinct(s_i_id))
from    stock, order_line
where   ol_w_id         =  @w_id and
        ol_d_id         =  @d_id and
        ol_o_id            between @o_id_low and
                           @o_id_high and
        s_w_id          =  ol_w_id and
        s_i_id          =  ol_i_id and
        s_quantity      <  @threshhold

go
```

VERSION.SQL

```
--  File:     VERSION.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.21.000
--            Copyright Microsoft, 1999, 2000
--  Purpose:  Returns version level of TPC-C stored procs
--  Note:     Always update the return value of this proc for
--            any interface changes or "must have" bug fixes.
--
--  The value returned by this SP defines the "interface level",
--  which must match between the stored procs and the client code.
--  The interface level may be down rev from the current kit.  This
--  indicates that the interface hasn't changed since that version.

use tpcc
go

if exists ( select name from sysobjects where name = "tpcc_version" )
        drop procedure tpcc_version
go

create proc tpcc_version
as
declare @version        char(8)

begin
        select @version = "4.10.000"
        select @version as "Version"
end

go
```

GETARGS.C

```
//      File:           GETARGS.C
//                              Microsoft TPC-C Kit Ver. 4.20
//                              Copyright Microsoft, 1996, 1997, 1998, 1999
//      Purpose:        Source file for command line processing


// Includes
#include "tpcc.h"

//===========================================================================
//
// Function name: GetArgsLoader
//
//===========================================================================

void GetArgsLoader(int argc, char **argv, TPCCLDR_ARGS *pargs)
{
    int         i;
    char        *ptr;

#ifdef DEBUG
    printf("[%ld]DBG: Entering GetArgsLoader()\n", (int)
GetCurrentThreadId());
#endif

    /* init args struct with some useful values */
    pargs->server                       =  SERVER;
    pargs->user                         =  USER;
    pargs->password                     =  PASSWORD;
    pargs->database                     =  DATABASE;
    pargs->batch                        =  BATCH;
    pargs->num_warehouses               =  UNDEF;
        pargs->tables_all                       =  TRUE;
        pargs->table_item                       =  FALSE;
        pargs->table_warehouse          =  FALSE;
        pargs->table_customer           =  FALSE;
        pargs->table_orders                     =  FALSE;
        pargs->loader_res_file                  =  LOADER_RES_FILE;
        pargs->pack_size                        =  DEFLDPACKSIZE;
        pargs->starting_warehouse       =  DEF_STARTING_WAREHOUSE;
        pargs->build_index                      =  BUILD_INDEX;
        pargs->index_order                      =  INDEX_ORDER;
        pargs->index_script_path        =  INDEX_SCRIPT_PATH;
        pargs->scale_down                       =  SCALE_DOWN;
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
/* check for zero command line args */
if ( argc == 1 )
           GetArgsLoaderUsage();

for (i = 1; i < argc; ++i)
{
           if (argv[i][0] != '-' && argv[i][0] != '/')
           {
           printf("\nUnrecognized command");
           GetArgsLoaderUsage();
           exit(1);
           }

           ptr = argv[i];

           switch (ptr[1])
           {
           case 'h':     /* Fall throught */
           case 'H':
                          GetArgsLoaderUsage();
                          break;

           case 'D':
                          pargs->database = ptr+2;
                          break;

           case 'P':
                          pargs->password = ptr+2;
                          break;

           case 'S':
                          pargs->server = ptr+2;
                          break;

           case 'U':
                          pargs->user = ptr+2;
                          break;

           case 'b':
                          pargs->batch = atol(ptr+2);
                          break;

           case 'W':
                          pargs->num_warehouses = atol(ptr+2);
                          break;

           case 's':
                          pargs->starting_warehouse = atol(ptr+2);
                          break;

           case 't':
                          {
                                 pargs->tables_all = FALSE;
                                 if (strcmp(ptr+2,"item") == 0)
                                        pargs->table_item = TRUE;
                                 else if (strcmp(ptr+2,"warehouse")
== 0)
                                        pargs->table_warehouse =
TRUE;
                                 else if (strcmp(ptr+2,"customer") ==
0)
                                        pargs->table_customer = TRUE;
                                 else if (strcmp(ptr+2,"orders") ==
0)
                                        pargs->table_orders = TRUE;
                                 else
                                 {
                                 printf("\nUnrecognized command");
                                 GetArgsLoaderUsage();
                                 exit(1);
                                 }

                                 break;
                          }
                   case 'f':
                          pargs->loader_res_file = ptr+2;
                          break;

                   case 'p':
                          pargs->pack_size = atol(ptr+2);
                          break;

                   case 'i':
                          pargs->build_index = atol(ptr+2);
                          break;

                   case 'o':
                          pargs->index_order = atol(ptr+2);
                          break;

                   case 'c':
                          pargs->scale_down = atol(ptr+2);
                          break;

            case 'd':
                          pargs->index_script_path = ptr+2;
                          break;

            default:
                   GetArgsLoaderUsage();
                   exit(-1);
                   break;
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
            }
    }

    /* check for required args */
    if (pargs->num_warehouses == UNDEF )
    {
            printf("Number of Warehouses is required\n");
            exit(-2);
    }

    return;
}


//==============================================================================
//
// Function name: GetArgsLoaderUsage
//
//==============================================================================

void GetArgsLoaderUsage()
{

#ifdef DEBUG
    printf("[%ld]DBG: Entering GetArgsLoaderUsage()\n", (int)
GetCurrentThreadId());
#endif


    printf("TPCCLDR:\n\n");
        printf("Parameter
Default\n");
        printf("-------------------------------------------------------------
----\n");
    printf("-W Number of Warehouses to Load                          Required
\n");
    printf("-S Server                                                %s\n",
SERVER);
    printf("-U Username                                              %s\n",
USER);
    printf("-P Password                                              %s\n",
PASSWORD);
    printf("-D Database                                              %s\n",
DATABASE);
        printf("-b Batch Size
%ld\n", (long) BATCH);
        printf("-p TDS packet size
%ld\n", (long) DEFLDPACKSIZE);
        printf("-f Loader Results Output Filename
%s\n", LOADER_RES_FILE);
        printf("-s Starting Warehouse
%ld\n", (long) DEF_STARTING_WAREHOUSE);
        printf("-i Build Option (data = 0, data and index = 1)
%ld\n", (long) BUILD_INDEX);
        printf("-o Cluster Index Build Order (before = 1, after = 0)
%ld\n", (long) INDEX_ORDER);
        printf("-c Build Scaled Database (normal = 0, tiny = 1)
%ld\n", (long) SCALE_DOWN);
        printf("-d Index Script Path
%s\n", INDEX_SCRIPT_PATH);
        printf("-t Table to Load                                         all
tables \n");
    printf("   [item|warehouse|customer|orders]\n");
    printf("   Notes: \n");
    printf("     - the '-t' parameter may be included multiple times to
\n");
    printf("       specify multiple tables to be loaded \n");
    printf("     - 'item' loads ITEM table \n");
    printf("     - 'warehouse' loads WAREHOUSE, DISTRICT, and STOCK tables
\n");
    printf("     - 'customer' loads CUSTOMER and HISTORY tables \n");
    printf("     - 'orders' load NEW-ORDER, ORDERS, ORDER-LINE tables
\n");

        printf("\nNote:  Command line switches are case sensitive.\n");

}
```

RANDOM.C

```
//      File:           RANDOM.C
//                              Microsoft TPC-C Kit Ver. 4.20
//                              Copyright Microsoft, 1996, 1997, 1998, 1999
//      Purpose:        Random number generation routines for database
loader


// Includes
#include "tpcc.h"
#include "math.h"

// Defines
#define A             16807
#define M        2147483647
#define Q            127773      /* M div A */
#define R              2836      /* M mod A */
#define Thread          __declspec(thread)

// Globals
long    Thread Seed = 0;        /* thread local seed */
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
/************************************************************************
*****
*
*
* random -
*
*       Implements a GOOD pseudo random number generator.  This generator
*
*       will/should? run the complete period before repeating.
*
*
*
* Copied from:
*
*       Random Numbers Generators: Good Ones Are Hard to Find.
*
*       Communications of the ACM - October 1988 Volume 31 Number 10
*
*
*
* Machine Dependencies:
*
*       long must be 2 ^ 31 - 1 or greater.
*
*
*
************************************************************************
****/

/************************************************************************
*****
* seed - load the Seed value used in irand and drand.  Should be used
before  *
*       first call to irand or drand.
*
************************************************************************
****/

void seed(long val)
{

#ifdef DEBUG
    printf("[%ld]DBG: Entering seed()...\n", (int) GetCurrentThreadId());
        printf("Old Seed %ld New Seed %ld\n",Seed, val);
#endif

    if ( val < 0 )
           val = abs(val);

    Seed = val;

}
```

```
/************************************************************************
****
*
*
* irand - returns a 32 bit integer pseudo random number with a period of
*
*       1 to 2 ^ 32 - 1.
*
*
*
* parameters:
*
*       none.
*
*
*
* returns:
*
*       32 bit integer - defined as long ( see above ).
*
*
*
* side effects:
*
*       seed get recomputed.
*
************************************************************************
***/

long irand()
{
    register long    s;      /* copy of seed */
    register long    test;   /* test flag */
    register long    hi;     /* tmp value for speed */
    register long    lo;     /* tmp value for speed */

#ifdef DEBUG
    printf("[%ld]DBG: Entering irand()...\n", (int) GetCurrentThreadId());
#endif

    s = Seed;
    hi = s / Q;
    lo = s % Q;

    test = A * lo - R * hi;
    if ( test > 0 )
               Seed = test;
    else
               Seed = test + M;
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
    return( Seed );
}


/*****************************************************************************
*
*
* drand - returns a double pseudo random number between 0.0 and 1.0.
*
*       See irand.
*
*****************************************************************************/
double drand()
{

#ifdef DEBUG
    printf("[%ld]DBG: Entering drand()...\n", (int) GetCurrentThreadId());
#endif

    return( (double)irand() / 2147483647.0);
}


//============================================================================
// Function   : RandomNumber
//
// Description:
//============================================================================
long RandomNumber(long lower, long upper)
{
    long rand_num;

#ifdef DEBUG
    printf("[%ld]DBG: Entering RandomNumber()...\n", (int)
GetCurrentThreadId());
#endif

        if ( upper == lower )  /* pgd 08-13-96 perf enhancement */
                return lower;

        upper++;

    if ( upper <= lower )
                rand_num = upper;
    else
                rand_num = lower + irand() % (upper - lower); /* pgd 08-13-
96 perf enhancement */


#ifdef DEBUG
    printf("[%ld]DBG: RandomNumber between %ld & %ld ==> %ld\n",
                                (int) GetCurrentThreadId(), lower, upper,
rand_num);
#endif


    return rand_num;
}


#if 0

//Orginal code pgd 08/13/96

long RandomNumber(long lower,
                                  long upper)
{
    long rand_num;

#ifdef DEBUG
    printf("[%ld]DBG: Entering RandomNumber()...\n", (int)
GetCurrentThreadId());
#endif

        upper++;

    if ((upper <= lower))
                rand_num = upper;
    else
                rand_num = lower + irand() % ((upper > lower) ? upper -
lower : upper);


#ifdef DEBUG
    printf("[%ld]DBG: RandomNumber between %ld & %ld ==> %ld\n",
                                (int) GetCurrentThreadId(), lower, upper,
rand_num);
#endif


    return rand_num;
}
#endif

//============================================================================
// Function   : NURand
//
// Description:
//============================================================================
long NURand(int iConst,
            long x,
            long y,
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
           long C)
{
    long rand_num;

#ifdef DEBUG
    printf("[%ld]DBG: Entering NURand()...\n", (int)
GetCurrentThreadId());
#endif

    rand_num = (((RandomNumber(0,iConst) | RandomNumber(x,y)) + C) % (y-
x+1))+x;

#ifdef DEBUG
    printf("[%ld]DBG: NURand: num = %d\n", (int) GetCurrentThreadId(),
rand_num);
#endif

    return rand_num;
}
```

STRINGS.C

```
//      File:           STRINGS.C
//                              Microsoft TPC-C Kit Ver. 4.20
//                              Copyright Microsoft, 1996, 1997, 1998, 1999
//      Purpose:        Source file for database loader string functions


// Includes
#include "tpcc.h"
#include <string.h>
#include <ctype.h>

//==============================================================================
//
// Function name: MakeAddress
//
//==============================================================================

void MakeAddress(char *street_1,
                                char *street_2,
                                char *city,
                                char *state,
                                char *zip)
{

#ifdef DEBUG
    printf("[%ld]DBG: Entering MakeAddress()\n", (int)
GetCurrentThreadId());
#endif

    MakeAlphaString (10, 20, ADDRESS_LEN, street_1);
    MakeAlphaString (10, 20, ADDRESS_LEN, street_2);
    MakeAlphaString (10, 20, ADDRESS_LEN, city);
    MakeAlphaString ( 2,  2, STATE_LEN, state);
    MakeZipNumberString( 9,  9, ZIP_LEN, zip);

#ifdef DEBUG
    printf("[%ld]DBG: MakeAddress: street_1: %s, street_2: %s, city: %s,
state: %s, zip: %s\n",
                        (int) GetCurrentThreadId(), street_1, street_2,
city, state, zip);
#endif

    return;

}


//==============================================================================
//
// Function name: LastName
//
//==============================================================================

void LastName(int  num,
                        char *name)
{
    static char *n[] =
    {
                "BAR" , "OUGHT", "ABLE" , "PRI"  , "PRES",
                "ESE" , "ANTI" , "CALLY", "ATION", "EING"
    };

#ifdef DEBUG
    printf("[%ld]DBG: Entering LastName()\n", (int) GetCurrentThreadId());
#endif

    if ((num >= 0) && (num < 1000))
    {
                strcpy(name, n[(num/100)%10]);
                strcat(name, n[(num/10)%10]);
                strcat(name, n[(num/1)%10]);

                if (strlen(name) < LAST_NAME_LEN)
                {
                        PaddString(LAST_NAME_LEN, name);
                }

    }
    else
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
    {
                printf("\nError in LastName()... num <%ld> out of range
(0,999)\n", num);
                exit(-1);
    }


#ifdef DEBUG
    printf("[%ld]DBG: LastName: num = [%d] ==> [%d][%d][%d]\n",
                        (int) GetCurrentThreadId(), num, num/100,
(num/10)%10, num%10);
        printf("[%ld]DBG: LastName: String = %s\n", (int)
GetCurrentThreadId(), name);
#endif

    return;

}


//============================================================================
//
// Function name: MakeAlphaString
//
//============================================================================

//philipdu 08/13/96 Changed MakeAlphaString to use A-Z, a-z, and 0-9 in
//accordance with spec see below:
//The spec says:
//4.3.2.2       The notation random a-string [x .. y]
//(respectively, n-string [x .. y]) represents a string of random
alphanumeric
//(respectively, numeric) characters of a random length of minimum x,
maximum y,
//and mean (y+x)/2. Alphanumerics are A..Z, a..z, and 0..9.  The only
other
//requirement is that the character set used "must be able to represent a
minimum
//of 128 different characters".  We are using 8-bit chars, so this is a
non issue.
//It is completely unreasonable to stuff non-printing chars into the text
fields.
//-CLevine 08/13/96

int MakeAlphaString( int  x, int  y, int  z, char *str)
{
        int             len;
        int             i;
    char    cc = 'a';
        static  char chArray[] =
"0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz";
        static  int     chArrayMax = 61;

#ifdef DEBUG
    printf("[%ld]DBG: Entering MakeAlphaString()\n", (int)
GetCurrentThreadId());
#endif

        len= RandomNumber(x, y);

        for (i=0; i<len; i++)
        {
                cc = chArray[RandomNumber(0, chArrayMax)];
         str[i] = cc;
    }
        if ( len < z )
                memset(str+len, ' ', z - len);
        str[len] = 0;

    return len;
}


//============================================================================
//
// Function name: MakeOriginalAlphaString
//
//============================================================================

int MakeOriginalAlphaString(int  x,
                                                        int  y,
                                                           int  z,
                                                        char *str,
                                                    int  percent)
{
    int         len;
    int         val;
    int         start;

#ifdef DEBUG
    printf("[%ld]DBG: Entering MakeOriginalAlphaString()\n", (int)
GetCurrentThreadId());
#endif

    // verify prercentage is valid
    if ((percent < 0) || (percent > 100))
    {
                printf("MakeOrigianlAlphaString: Invalid percentage: %d\n",
percent);
                exit(-1);
    }

    // verify string is at least 8 chars in length
    if ((x + y) <= 8)
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
    {
                printf("MakeOriginalAlphaString: string length must be >=
8\n");
                exit(-1);
    }

    // Make Alpha String
    len = MakeAlphaString(x,y, z, str);

    val = RandomNumber(1,100);
    if (val <= percent)
    {
                start = RandomNumber(0, len - 8);
                strncpy(str + start, "ORIGINAL", 8);
    }


#ifdef DEBUG
    printf("[%ld]DBG: MakeOriginalAlphaString: : %s\n",
                        (int) GetCurrentThreadId(), str);
#endif

    return strlen(str);
}


//============================================================================
//
// Function name: MakeNumberString
//
//============================================================================
int MakeNumberString(int  x, int  y, int  z, char *str)
{
       char tmp[16];

       //MakeNumberString is always called MakeZipNumberString(16, 16,
16, string)

       memset(str, '0', 16);
       itoa(RandomNumber(0, 99999999), tmp, 10);
       memcpy(str, tmp, strlen(tmp));

       itoa(RandomNumber(0, 99999999), tmp, 10);
       memcpy(str+8, tmp, strlen(tmp));

       str[16] = 0;

    return 16;
}


//============================================================================
//
// Function name: MakeZipNumberString
//
//============================================================================
int MakeZipNumberString(int  x, int  y, int  z, char *str)
{
       char tmp[16];

       //MakeZipNumberString is always called MakeZipNumberString(9, 9,
9, string)

       strcpy(str, "000011111");

       itoa(RandomNumber(0, 9999), tmp, 10);
       memcpy(str, tmp, strlen(tmp));

    return 9;
}


//============================================================================
//
// Function name: InitString
//
//============================================================================
void InitString(char *str, int len)
{
#ifdef DEBUG
    printf("[%ld]DBG: Entering InitString()\n", (int)
GetCurrentThreadId());
#endif

       memset(str, ' ', len);
    str[len] = 0;
}


//============================================================================
// Function name: InitAddress
//
// Description:
//
//============================================================================

void InitAddress(char *street_1, char *street_2, char *city, char *state,
char *zip)
{
       memset(street_1, ' ', ADDRESS_LEN+1);
       memset(street_2, ' ', ADDRESS_LEN+1);
       memset(city, ' ', ADDRESS_LEN+1);

    street_1[ADDRESS_LEN+1] = 0;
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
    street_2[ADDRESS_LEN+1] = 0;
    city[ADDRESS_LEN+1]     = 0;

        memset(state, ' ', STATE_LEN+1);
    state[STATE_LEN+1] = 0;

        memset(zip, ' ', ZIP_LEN+1);
    zip[ZIP_LEN+1] = 0;
}


//================================================================
//
// Function name: PaddString
//
//================================================================

void PaddString(int max, char *name)
{
        int             len;

        len = strlen(name);
        if ( len < max )
                memset(name+len, ' ', max - len);
        name[max] = 0;

        return;
}
```

TIME.C

```
//      File:           TIME.C
//                              Microsoft TPC-C Kit Ver. 4.20
//                              Copyright Microsoft, 1996, 1997, 1998, 1999
//      Purpose:        Source file for time functions


// Includes
#include "tpcc.h"

// Globals
static long start_sec;


//================================================================
//
// Function name: TimeNow
//
//================================================================

long TimeNow()
{
   long         time_now;
   struct       _timeb el_time;

#ifdef DEBUG
    printf("[%ld]DBG: Entering TimeNow()\n", (int) GetCurrentThreadId());
#endif

   _ftime(&el_time);

   time_now = ((el_time.time - start_sec) * 1000) + el_time.millitm;

   return time_now;
}
```

TPCC.H

```
//      File:           TPCC.H
//                              Microsoft TPC-C Kit Ver. 4.20
//                              Copyright Microsoft, 1996, 1997, 1998, 1999
//      Purpose:        Header file for TPC-C database loader


// Build number of TPC Benchmark Kit
#define TPCKIT_VER      "4.20"

// General headers
#include <windows.h>
#include <winbase.h>
#include <stdlib.h>
#include <stdio.h>
#include <process.h>
#include <stddef.h>
#include <stdarg.h>
#include <string.h>
#include <time.h>
#include <sys\timeb.h>
#include <sys\types.h>

// ODBC headers
#include <sql.h>
#include <sqlext.h>
#include <odbcss.h>

// General constants
#define MILLI                           1000
#define FALSE                                0
#define TRUE                                 1
#define UNDEF                                   -1
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
#define MINPRINTASCII                    32
#define MAXPRINTASCII                   126

// Default environment constants
#define SERVER                              ""
#define DATABASE                            "tpcc"
#define USER                                "sa"
#define PASSWORD                          ""

// Default loader arguments
#define BATCH                          10000
#define DEFLDPACKSIZE                  32768
#define LOADER_RES_FILE                  "logs\\load.out"
#define LOADER_NURAND_C                         123
#define DEF_STARTING_WAREHOUSE                1
#define BUILD_INDEX                                 1       // build both
data and indexes
#define INDEX_ORDER                                 1       // build
indexes before load
#define SCALE_DOWN                                  0   // build a normal
scale database
#define INDEX_SCRIPT_PATH     "scripts"

typedef struct
{
    char                            *server;
    char                            *database;
    char                            *user;
    char                            *password;
       BOOL                         tables_all;                 // set
if loading all tables
       BOOL                         table_item;                 // set
if loading ITEM table specifically
       BOOL                         table_warehouse;       // set if
loading WAREHOUSE, DISTRICT, and STOCK
       BOOL                         table_customer;             // set
if loading CUSTOMER and HISTORY
       BOOL                         table_orders;          // set if
loading NEW-ORDER, ORDERS, ORDER-LINE
    long                            num_warehouses;
    long                            batch;
    long                            verbose;
       long                         pack_size;
       char                         *loader_res_file;
       char                         *synch_servername;
       long                         case_sensitivity;
       long                         starting_warehouse;
       long                         build_index;
       long                         index_order;
       long                         scale_down;
       char                         *index_script_path;
} TPCCLDR_ARGS;
```

```
// String length constants
       #define SERVER_NAME_LEN            20
       #define DATABASE_NAME_LEN          20
       #define USER_NAME_LEN              20
       #define PASSWORD_LEN               20
       #define TABLE_NAME_LEN             20
       #define I_DATA_LEN                 50
       #define I_NAME_LEN                 24
       #define BRAND_LEN                   1
       #define LAST_NAME_LEN              16
       #define W_NAME_LEN                 10
       #define ADDRESS_LEN                20
       #define STATE_LEN                   2
       #define ZIP_LEN                     9
       #define S_DIST_LEN                 24
       #define S_DATA_LEN                 50
       #define D_NAME_LEN                 10
       #define FIRST_NAME_LEN             16
       #define MIDDLE_NAME_LEN             2
       #define PHONE_LEN                  16
       #define CREDIT_LEN                  2
       #define C_DATA_LEN                500
       #define H_DATA_LEN                 24
       #define DIST_INFO_LEN              24
       #define MAX_OL_NEW_ORDER_ITEMS     15
       #define MAX_OL_ORDER_STATUS_ITEMS  15
       #define STATUS_LEN                 25
       #define OL_DIST_INFO_LEN                  24
       #define C_SINCE_LEN                              23
       #define H_DATE_LEN                               23
       #define OL_DELIVERY_D_LEN                 23
       #define O_ENTRY_D_LEN              23


// Functions in random.c
void    seed();
long    irand();
double  drand();
void    WUCreate();
short   WURand();
long    RandomNumber(long lower, long upper);

// Functions in getargs.c;
void    GetArgsLoader();
void    GetArgsLoaderUsage();

// Functions in time.c
long    TimeNow();

// Functions in strings.c
void    MakeAddress();
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
void    LastName();
int     MakeAlphaString();
int     MakeOriginalAlphaString();
int     MakeNumberString();
int     MakeZipNumberString();
void    InitString();
void    InitAddress();
void    PaddString();
```

TPCCLDR.C

```
//      File:           TPCCLDR.C
//                              Microsoft TPC-C Kit Ver. 4.20
//                              Copyright Microsoft, 1996, 1997, 1998, 1999
//      Purpose:        Source file for TPC-C database loader


// Includes
#include "tpcc.h"
#include "search.h"

// Defines
#define MAXITEMS                100000
#define MAXITEMS_SCALE_DOWN             100
#define CUSTOMERS_PER_DISTRICT  3000
#define CUSTOMERS_SCALE_DOWN  30
#define DISTRICT_PER_WAREHOUSE  10
#define ORDERS_PER_DISTRICT     3000
#define ORDERS_SCALE_DOWN       30
#define MAX_CUSTOMER_THREADS    2
#define MAX_ORDER_THREADS       3
#define MAX_MAIN_THREADS        4

// Functions declarations

void HandleErrorDBC (SQLHDBC  hdbc1);

void CheckSQL();
void CheckDataBase();

long NURand();
void LoadItem();
void LoadWarehouse();

void Stock();
void District();

void LoadCustomer();
void CustomerBufInit();
void CustomerBufLoad();
void LoadCustomerTable();
void LoadHistoryTable();

void LoadOrders();
void OrdersBufInit();
void OrdersBufLoad();
void LoadOrdersTable();
void LoadNewOrderTable();
void LoadOrderLineTable();
void GetPermutation();
void CheckForCommit();
void OpenConnections();
void BuildIndex();
void FormatDate ();

// Shared memory structures

typedef struct
{
    long                ol;
    long                ol_i_id;
    short               ol_supply_w_id;
    short               ol_quantity;
    double              ol_amount;
    char                ol_dist_info[DIST_INFO_LEN+1];
       char                             ol_delivery_d[OL_DELIVERY_D_LEN+1];
} ORDER_LINE_STRUCT;

typedef struct
{
    long                o_id;
    short               o_d_id;
    short               o_w_id;
    long                o_c_id;
    short               o_carrier_id;
    short               o_ol_cnt;
    short               o_all_local;
       ORDER_LINE_STRUCT   o_ol[15];
} ORDERS_STRUCT;

typedef struct
{
    long                                c_id;
    short                               c_d_id;
    short                               c_w_id;
       char                             c_first[FIRST_NAME_LEN+1];
    char                                c_middle[MIDDLE_NAME_LEN+1];
    char                                c_last[LAST_NAME_LEN+1];
    char                                c_street_1[ADDRESS_LEN+1];
    char                                c_street_2[ADDRESS_LEN+1];
    char                                c_city[ADDRESS_LEN+1];
    char                                c_state[STATE_LEN+1];
    char                                c_zip[ZIP_LEN+1];
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
    char                            c_phone[PHONE_LEN+1];
    char                            c_credit[CREDIT_LEN+1];
    double                          c_credit_lim;
    double                          c_discount;
//  fix to avoid ODBC float to numeric conversion problem.
//  double                          c_balance;
       char                         c_balance[6];

    double                          c_ytd_payment;
       short                        c_payment_cnt;
       short                        c_delivery_cnt;
       char                         c_data[C_DATA_LEN+1];
       double                       h_amount;
    char                            h_data[H_DATA_LEN+1];
} CUSTOMER_STRUCT;

typedef struct
{
       char                         c_last[LAST_NAME_LEN+1];
       char                         c_first[FIRST_NAME_LEN+1];
    long                            c_id;
} CUSTOMER_SORT_STRUCT;

typedef struct
{
    long                time_start;
} LOADER_TIME_STRUCT;


// Global variables

char    szLastError[300];

HENV    henv;

HDBC    v_hdbc;                                 // for SQL Server
version verification
HDBC    i_hdbc1;                                // for ITEM table
HDBC    w_hdbc1;                                // for WAREHOUSE,
DISTRICT, STOCK
HDBC    c_hdbc1;                                // for CUSTOMER
HDBC    c_hdbc2;                                // for HISTORY
HDBC    o_hdbc1;                                // for ORDERS
HDBC    o_hdbc2;                                // for NEW-ORDER

HDBC    o_hdbc3;                                // for ORDER-LINE

HSTMT   v_hstmt;                                // for SQL Server
version verification
HSTMT   i_hstmt1;
HSTMT   w_hstmt1;
HSTMT   c_hstmt1, c_hstmt2;
HSTMT   o_hstmt1, o_hstmt2, o_hstmt3;

ORDERS_STRUCT   orders_buf[ORDERS_PER_DISTRICT];
CUSTOMER_STRUCT customer_buf[CUSTOMERS_PER_DISTRICT];
long            orders_rows_loaded;
long            new_order_rows_loaded;
long            order_line_rows_loaded;
long            history_rows_loaded;
long            customer_rows_loaded;
long            stock_rows_loaded;
long            district_rows_loaded;
long            item_rows_loaded;
long            warehouse_rows_loaded;
long            main_time_start;
long            main_time_end;
long                  max_items;
long                  customers_per_district;
long                  orders_per_district;
long                  first_new_order;
long                  last_new_order;


TPCCLDR_ARGS    *aptr, args;


//============================================================================
//
// Function name: main
//
//============================================================================

int main(int  argc, char **argv)
{

       DWORD           dwThreadID[MAX_MAIN_THREADS];
       HANDLE          hThread[MAX_MAIN_THREADS];
       FILE            *fLoader;
       char            buffer[255];
       int                           i;

       for (i=0; i<MAX_MAIN_THREADS; i++)
              hThread[i] = NULL;

    printf("\n*************************************************");
       printf("\n*                                                *");
       printf("\n*  Microsoft SQL Server                          *");
       printf("\n*                                                *");
       printf("\n*  TPC-C BENCHMARK KIT:  Database loader         *");
       printf("\n*  Version %s                                   *",
TPCKIT_VER);
       printf("\n*                                                *");
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        printf("\n*************************************************\n\n"
);

    // process command line arguments

    aptr = &args;
    GetArgsLoader(argc, argv, aptr);

        // verify correct SQL Server version in use
        // you must be using SQL Server 7.00.623 or better to load

        CheckSQL();

        // verify database and tables exist before attempting to load

        CheckDataBase();

        printf("Build interface is ODBC.\n");

        if (aptr->build_index == 0)
                printf("Data load only - no index creation.\n");
        else
                printf("Data load and index creation.\n");

        if (aptr->index_order == 0)
                printf("Clustered indexes will be created after bulk
load.\n");
        else
                printf("Clustered indexes will be created before bulk
load.\n");

        // set database scale values
        if (aptr->scale_down == 1)
        {
                printf("*** Scaled Down Database ***\n");
                max_items = MAXITEMS_SCALE_DOWN;
                customers_per_district = CUSTOMERS_SCALE_DOWN;
                orders_per_district = ORDERS_SCALE_DOWN;
                first_new_order = 0;
                last_new_order  = 30;
        }
        else
        {
                max_items = MAXITEMS;
                customers_per_district = CUSTOMERS_PER_DISTRICT;
                orders_per_district = ORDERS_PER_DISTRICT;
                first_new_order = 2100;
                last_new_order  = 3000;
        }


        // open connections to SQL Server
        OpenConnections();

        // open file for loader results
    fLoader = fopen(aptr->loader_res_file, "w");

    if (fLoader == NULL)
        {
                printf("Error, loader result file open failed.");
                exit(-1);
        }

    // start loading data

    sprintf(buffer,"TPC-C load started for %ld warehouses.\n",aptr-
>num_warehouses);

        printf("%s",buffer);
        fprintf(fLoader,"%s",buffer);

        main_time_start = (TimeNow() / MILLI);

        // start parallel load threads


        if (aptr->tables_all || aptr->table_item)
        {
                fprintf(fLoader, "\nStarting loader threads for: item\n");

                hThread[0] = CreateThread(NULL,
                                                                        0,
(LPTHREAD_START_ROUTINE) LoadItem,
                                                                        NULL,
                                                                        0,
&dwThreadID[0]);

                if (hThread[0] == NULL)
                {
                        printf("Error, failed in creating creating thread =
0.\n");
                        exit(-1);
                }
        }

        if (aptr->tables_all || aptr->table_warehouse)
        {
                fprintf(fLoader, "Starting loader threads for:
warehouse\n");

                hThread[1] = CreateThread(NULL,
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                                                        0,
(LPTHREAD_START_ROUTINE) LoadWarehouse,
                                                        NULL,
                                                        0,
&dwThreadID[1]);
                if (hThread[1] == NULL)
                {
                        printf("Error, failed in creating creating thread =
1.\n");
                        exit(-1);
                }
        }

        if (aptr->tables_all || aptr->table_customer)
        {
                fprintf(fLoader, "Starting loader threads for:
customer\n");
                hThread[2] = CreateThread(NULL,
                                                        0,
(LPTHREAD_START_ROUTINE) LoadCustomer,
                                                        NULL,
                                                        0,
&dwThreadID[2]);
                if (hThread[2] == NULL)
                {
                        printf("Error, failed in creating creating main
thread = 2.\n");
                        exit(-1);
                }
        }

        if (aptr->tables_all || aptr->table_orders)
        {
                fprintf(fLoader, "Starting loader threads for: orders\n");
                hThread[3] = CreateThread(NULL,
                                                        0,
(LPTHREAD_START_ROUTINE) LoadOrders,
                                                        NULL,
                                                        0,
&dwThreadID[3]);
                if (hThread[3] == NULL)
                {
                        printf("Error, failed in creating creating main
thread = 3.\n");
                        exit(-1);
                }
        }

        // Wait for threads to finish...
        for (i=0; i<MAX_MAIN_THREADS; i++)
        {
                if (hThread[i] != NULL)
                {
                        WaitForSingleObject( hThread[i], INFINITE );
                        CloseHandle(hThread[i]);
                        hThread[i] = NULL;
                }
        }

        main_time_end = (TimeNow() / MILLI);

    sprintf(buffer,"\nTPC-C load completed successfully in %ld
minutes.\n",
                        (main_time_end - main_time_start)/60);

        printf("%s",buffer);
        fprintf(fLoader, "%s", buffer);

        fclose(fLoader);

        SQLFreeEnv(henv);

    exit(0);

        return 0;
}


//============================================================================
//
// Function name: LoadItem
//
//============================================================================

void LoadItem()
{
    long                i_id;
        long            i_im_id;
    char                i_name[I_NAME_LEN+1];
    double              i_price;
    char                i_data[I_DATA_LEN+1];
        char            name[20];
        long            time_start;
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        RETCODE         rc;
        DBINT           rcint;
        char            bcphint[128];

        // Seed with unique number
        seed(1);

        printf("Loading item table...\n");

        // if build index before load
        if ((aptr->build_index == 1) && (aptr->index_order == 1))
                BuildIndex("idxitmcl");

        InitString(i_name, I_NAME_LEN+1);
        InitString(i_data, I_DATA_LEN+1);

        sprintf(name, "%s..%s", aptr->database, "item");

        rc = bcp_init(i_hdbc1, name, NULL, "logs\\item.err", DB_IN);
        if (rc != SUCCEED)
                HandleErrorDBC(i_hdbc1);

        if ((aptr->build_index == 1) && (aptr->index_order == 1))
        {
                sprintf(bcphint, "tablock, order (i_id), ROWS_PER_BATCH =
100000");
                rc = bcp_control(i_hdbc1, BCPHINTS, (void*) bcphint);
                if (rc != SUCCEED)
                        HandleErrorDBC(i_hdbc1);
        }

        rc = bcp_bind(i_hdbc1, (BYTE *) &i_id, 0, SQL_VARLEN_DATA, NULL,
0, SQLINT4, 1);
        if (rc != SUCCEED)
                HandleErrorDBC(i_hdbc1);

        rc = bcp_bind(i_hdbc1, (BYTE *) &i_im_id, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT4, 2);
        if (rc != SUCCEED)
                HandleErrorDBC(i_hdbc1);

        rc = bcp_bind(i_hdbc1, (BYTE *) i_name, 0, I_NAME_LEN, NULL, 0, 0,
3);
        if (rc != SUCCEED)
                HandleErrorDBC(i_hdbc1);

        rc = bcp_bind(i_hdbc1, (BYTE *) &i_price, 0, SQL_VARLEN_DATA,
NULL, 0, SQLFLT8, 4);
        if (rc != SUCCEED)
                HandleErrorDBC(i_hdbc1);

        rc = bcp_bind(i_hdbc1, (BYTE *) i_data, 0, I_DATA_LEN, NULL, 0, 0,
5);
        if (rc != SUCCEED)
                HandleErrorDBC(i_hdbc1);

        time_start = (TimeNow() / MILLI);

        item_rows_loaded = 0;

        for (i_id = 1; i_id <= max_items; i_id++)
        {
                i_im_id = RandomNumber(1L, 10000L);

                MakeAlphaString(14, 24, I_NAME_LEN, i_name);

                i_price = ((float) RandomNumber(100L, 10000L))/100.0;

                MakeOriginalAlphaString(26, 50, I_DATA_LEN, i_data, 10);

                rc = bcp_sendrow(i_hdbc1);
                if (rc != SUCCEED)
                        HandleErrorDBC(i_hdbc1);

                item_rows_loaded++;
                CheckForCommit(i_hdbc1, i_hstmt1, item_rows_loaded, "item",
&time_start);
        }

        rcint = bcp_done(i_hdbc1);
        if (rcint < 0)
                HandleErrorDBC(i_hdbc1);

        printf("Finished loading item table.\n");

        SQLFreeStmt(i_hstmt1, SQL_DROP);
        SQLDisconnect(i_hdbc1);
        SQLFreeConnect(i_hdbc1);

        // if build index after load
        if ((aptr->build_index == 1) && (aptr->index_order == 0))
                BuildIndex("idxitmcl");
}


//=============================================================================
=====
//
// Function   : LoadWarehouse
//
// Loads WAREHOUSE table and loads Stock and District as Warehouses are
created
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
//
//==============================================================================
=====

void LoadWarehouse()
{
    short      w_id;
    char       w_name[W_NAME_LEN+1];
    char       w_street_1[ADDRESS_LEN+1];
    char       w_street_2[ADDRESS_LEN+1];
    char       w_city[ADDRESS_LEN+1];
    char       w_state[STATE_LEN+1];
    char       w_zip[ZIP_LEN+1];
    double     w_tax;
    double     w_ytd;
       char    name[20];
       long    time_start;
       RETCODE rc;
       DBINT   rcint;
       char    bcphint[128];


       // Seed with unique number
       seed(2);

       printf("Loading warehouse table...\n");

       // if build index before load...
       if ((aptr->build_index == 1) && (aptr->index_order == 1))
              BuildIndex("idxwarcl");

       InitString(w_name, W_NAME_LEN+1);
       InitAddress(w_street_1, w_street_2, w_city, w_state, w_zip);

       sprintf(name, "%s..%s", aptr->database, "warehouse");

       rc = bcp_init(w_hdbc1, name, NULL, "logs\\whouse.err", DB_IN);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       if ((aptr->build_index == 1) && (aptr->index_order == 1))
       {
              sprintf(bcphint, "tablock, order (w_id), ROWS_PER_BATCH =
%d", aptr->num_warehouses);
              rc = bcp_control(w_hdbc1, BCPHINTS, (void*) bcphint);
              if (rc != SUCCEED)
                     HandleErrorDBC(w_hdbc1);
       }

       rc = bcp_bind(w_hdbc1, (BYTE *) &w_id, 0, SQL_VARLEN_DATA, NULL,
0, SQLINT2, 1);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) w_name, 0, W_NAME_LEN, NULL, 0, 0,
2);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) w_street_1, 0, ADDRESS_LEN, NULL,
0, 0, 3);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) w_street_2, 0, ADDRESS_LEN, NULL,
0, 0, 4);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) w_city, 0, ADDRESS_LEN, NULL, 0,
0, 5);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) w_state, 0, STATE_LEN, NULL, 0, 0,
6);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) w_zip, 0, ZIP_LEN, NULL, 0, 0, 7);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) &w_tax, 0, SQL_VARLEN_DATA, NULL,
0, SQLFLT8, 8);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) &w_ytd, 0, SQL_VARLEN_DATA, NULL,
0, SQLFLT8, 9);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       time_start = (TimeNow() / MILLI);

       warehouse_rows_loaded = 0;

       for (w_id = (short)aptr->starting_warehouse; w_id <= aptr-
>num_warehouses; w_id++)
       {
              MakeAlphaString(6,10, W_NAME_LEN, w_name);

              MakeAddress(w_street_1, w_street_2, w_city, w_state,
w_zip);
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                w_tax = ((float) RandomNumber(0L,2000L))/10000.00;

                w_ytd = 300000.00;

                rc = bcp_sendrow(w_hdbc1);
                if (rc != SUCCEED)
                        HandleErrorDBC(w_hdbc1);

                warehouse_rows_loaded++;
                CheckForCommit(w_hdbc1, i_hstmt1, warehouse_rows_loaded,
"warehouse", &time_start);
        }

        rcint = bcp_done(w_hdbc1);
        if (rcint < 0)
                HandleErrorDBC(w_hdbc1);

        printf("Finished loading warehouse table.\n");

        // if build index after load...
        if ((aptr->build_index == 1) && (aptr->index_order == 0))
                BuildIndex("idxwarcl");

        stock_rows_loaded = 0;
        district_rows_loaded = 0;

        District();
        Stock();

}


//============================================================================
//
// Function   : District
//
//============================================================================

void District()
{
    short       d_id;
    short       d_w_id;
    char        d_name[D_NAME_LEN+1];
    char        d_street_1[ADDRESS_LEN+1];
    char        d_street_2[ADDRESS_LEN+1];
    char        d_city[ADDRESS_LEN+1];
    char        d_state[STATE_LEN+1];
    char        d_zip[ZIP_LEN+1];
    double      d_tax;
    double      d_ytd;
        char    name[20];
    long        d_next_o_id;
        long    time_start;
        int             w_id;
        RETCODE rc;
        DBINT   rcint;
        char    bcphint[128];

        // Seed with unique number
        seed(4);

        printf("Loading district table...\n");

        // build index before load
        if ((aptr->build_index == 1) && (aptr->index_order == 1))
                BuildIndex("idxdiscl");

        InitString(d_name, D_NAME_LEN+1);
        InitAddress(d_street_1, d_street_2, d_city, d_state, d_zip);
        sprintf(name, "%s..%s", aptr->database, "district");

        rc = bcp_init(w_hdbc1, name, NULL, "logs\\district.err", DB_IN);
        if (rc != SUCCEED)
                HandleErrorDBC(w_hdbc1);

        if ((aptr->build_index == 1) && (aptr->index_order == 1))
        {
                sprintf(bcphint, "tablock, order (d_w_id, d_id),
ROWS_PER_BATCH = %u", (aptr->num_warehouses * 10));
                rc = bcp_control(w_hdbc1, BCPHINTS, (void*) bcphint);
                if (rc != SUCCEED)
                        HandleErrorDBC(w_hdbc1);
        }

        rc = bcp_bind(w_hdbc1, (BYTE *) &d_id, 0, SQL_VARLEN_DATA, NULL,
0, SQLINT2, 1);
        if (rc != SUCCEED)
                HandleErrorDBC(w_hdbc1);

        rc = bcp_bind(w_hdbc1, (BYTE *) &d_w_id, 0, SQL_VARLEN_DATA, NULL,
0, SQLINT2, 2);
        if (rc != SUCCEED)
                HandleErrorDBC(w_hdbc1);

        rc = bcp_bind(w_hdbc1, (BYTE *) d_name, 0, D_NAME_LEN, NULL, 0, 0,
3);
        if (rc != SUCCEED)
                HandleErrorDBC(w_hdbc1);

        rc = bcp_bind(w_hdbc1, (BYTE *) d_street_1, 0, ADDRESS_LEN, NULL,
0, 0, 4);
        if (rc != SUCCEED)
                HandleErrorDBC(w_hdbc1);
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        rc = bcp_bind(w_hdbc1, (BYTE *) d_street_2, 0, ADDRESS_LEN, NULL,
0, 0, 5);
        if (rc != SUCCEED)
                HandleErrorDBC(w_hdbc1);

        rc = bcp_bind(w_hdbc1, (BYTE *) d_city, 0, ADDRESS_LEN, NULL, 0,
0, 6);
        if (rc != SUCCEED)
                HandleErrorDBC(w_hdbc1);

        rc = bcp_bind(w_hdbc1, (BYTE *) d_state, 0, STATE_LEN, NULL, 0, 0,
7);
        if (rc != SUCCEED)
                HandleErrorDBC(w_hdbc1);

        rc = bcp_bind(w_hdbc1, (BYTE *) d_zip, 0, ZIP_LEN, NULL, 0, 0, 8);
        if (rc != SUCCEED)
                HandleErrorDBC(w_hdbc1);

        rc = bcp_bind(w_hdbc1, (BYTE *) &d_tax, 0, SQL_VARLEN_DATA, NULL,
0, SQLFLT8, 9);
        if (rc != SUCCEED)
                HandleErrorDBC(w_hdbc1);

        rc = bcp_bind(w_hdbc1, (BYTE *) &d_ytd, 0, SQL_VARLEN_DATA, NULL,
0, SQLFLT8, 10);
        if (rc != SUCCEED)
                HandleErrorDBC(w_hdbc1);

        rc = bcp_bind(w_hdbc1, (BYTE *) &d_next_o_id, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT4, 11);
        if (rc != SUCCEED)
                HandleErrorDBC(w_hdbc1);

        d_ytd  = 30000.0;

        d_next_o_id = orders_per_district+1;

        time_start = (TimeNow() / MILLI);

        for (w_id = aptr->starting_warehouse; w_id <= aptr-
>num_warehouses; w_id++)
        {
                d_w_id = w_id;

                for (d_id = 1; d_id <= DISTRICT_PER_WAREHOUSE; d_id++)
                {
                        MakeAlphaString(6,10,D_NAME_LEN, d_name);

                        MakeAddress(d_street_1, d_street_2, d_city,
d_state, d_zip);

                        d_tax = ((float) RandomNumber(0L,2000L))/10000.00;

                        rc = bcp_sendrow(w_hdbc1);
                        if (rc != SUCCEED)
                                HandleErrorDBC(w_hdbc1);

                        district_rows_loaded++;
                        CheckForCommit(w_hdbc1, w_hstmt1,
district_rows_loaded, "district", &time_start);
                }
    }

        rcint = bcp_done(w_hdbc1);
        if (rcint < 0)
                HandleErrorDBC(w_hdbc1);

        printf("Finished loading district table.\n");

        // if build index after load...
        if ((aptr->build_index == 1) && (aptr->index_order == 0))
                BuildIndex("idxdiscl");

    return;
}


//============================================================================
//
// Function   : Stock
//
//============================================================================

void Stock()
{
    long       s_i_id;
    short      s_w_id;
    short      s_quantity;
    char       s_dist_01[S_DIST_LEN+1];
    char       s_dist_02[S_DIST_LEN+1];
    char       s_dist_03[S_DIST_LEN+1];
    char       s_dist_04[S_DIST_LEN+1];
    char       s_dist_05[S_DIST_LEN+1];
    char       s_dist_06[S_DIST_LEN+1];
    char       s_dist_07[S_DIST_LEN+1];
    char       s_dist_08[S_DIST_LEN+1];
    char       s_dist_09[S_DIST_LEN+1];
    char       s_dist_10[S_DIST_LEN+1];
    long       s_ytd;
    short      s_order_cnt;
    short      s_remote_cnt;
    char       s_data[S_DATA_LEN+1];
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
    short      len;
       char    name[20];
       long    time_start;
       RETCODE rc;
       DBINT   rcint;
       char    bcphint[128];

       // Seed with unique number
       seed(3);

       // if build index before load...
       if ((aptr->build_index == 1) && (aptr->index_order == 1))
              BuildIndex("idxstkcl");

       sprintf(name, "%s..%s", aptr->database, "stock");

       rc = bcp_init(w_hdbc1, name, NULL, "logs\\stock.err", DB_IN);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       if ((aptr->build_index == 1) && (aptr->index_order == 1))
       {
              sprintf(bcphint, "tablock, order (s_i_id, s_w_id),
ROWS_PER_BATCH = %u", (aptr->num_warehouses * 100000));
              rc = bcp_control(w_hdbc1, BCPHINTS, (void*) bcphint);
              if (rc != SUCCEED)
                     HandleErrorDBC(w_hdbc1);
       }

       rc = bcp_bind(w_hdbc1, (BYTE *) &s_i_id, 0, SQL_VARLEN_DATA, NULL,
0, SQLINT4, 1);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       bcp_bind(w_hdbc1, (BYTE *) &s_w_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 2);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) &s_quantity, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT2, 3);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_01, 0, S_DIST_LEN, NULL, 0,
0, 4);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_02, 0, S_DIST_LEN, NULL, 0,
0, 5);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_03, 0, S_DIST_LEN, NULL, 0,
0, 6);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_04, 0, S_DIST_LEN, NULL, 0,
0, 7);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_05, 0, S_DIST_LEN, NULL, 0,
0, 8);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_06, 0, S_DIST_LEN, NULL, 0,
0, 9);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_07, 0, S_DIST_LEN, NULL, 0,
0, 10);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_08, 0, S_DIST_LEN, NULL, 0,
0, 11);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_09, 0, S_DIST_LEN, NULL, 0,
0, 12);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) s_dist_10, 0, S_DIST_LEN, NULL, 0,
0, 13);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) &s_ytd, 0, SQL_VARLEN_DATA, NULL,
0, SQLINT4, 14);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);

       rc = bcp_bind(w_hdbc1, (BYTE *) &s_order_cnt, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT2, 15);
       if (rc != SUCCEED)
              HandleErrorDBC(w_hdbc1);
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        rc = bcp_bind(w_hdbc1, (BYTE *) &s_remote_cnt, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT2, 16);
        if (rc != SUCCEED)
                HandleErrorDBC(w_hdbc1);

        rc = bcp_bind(w_hdbc1, (BYTE *) s_data, 0, S_DATA_LEN, NULL, 0, 0,
17);
        if (rc != SUCCEED)
                HandleErrorDBC(w_hdbc1);

        s_ytd = s_order_cnt = s_remote_cnt = 0;

        time_start = (TimeNow() / MILLI);

        printf("...Loading stock table\n");

        for (s_i_id=1; s_i_id <= max_items; s_i_id++)
        {

                for (s_w_id = (short)aptr->starting_warehouse; s_w_id <=
aptr->num_warehouses; s_w_id++)
                {

                        s_quantity = (short)RandomNumber(10L,100L);
                        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_01);
                        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_02);
                        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_03);
                        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_04);
                        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_05);
                        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_06);
                        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_07);
                        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_08);
                        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_09);
                        len = MakeAlphaString(24,24,S_DIST_LEN, s_dist_10);

                        len = MakeOriginalAlphaString(26,50, S_DATA_LEN,
s_data,10);

                        rc = bcp_sendrow(w_hdbc1);
                        if (rc != SUCCEED)
                                HandleErrorDBC(w_hdbc1);

                        stock_rows_loaded++;
                        CheckForCommit(w_hdbc1, w_hstmt1,
stock_rows_loaded, "stock", &time_start);

                }
        }

        rcint = bcp_done(w_hdbc1);
        if (rcint < 0)
                HandleErrorDBC(w_hdbc1);

        printf("Finished loading stock table.\n");

        SQLFreeStmt(w_hstmt1, SQL_DROP);
        SQLDisconnect(w_hdbc1);
        SQLFreeConnect(w_hdbc1);

        // if build index after load...
        if ((aptr->build_index == 1) && (aptr->index_order == 0))
                BuildIndex("idxstkcl");

        return;
}


//============================================================================
//
// Function   : LoadCustomer
//
//============================================================================

void LoadCustomer()
{
        LOADER_TIME_STRUCT      customer_time_start;
        LOADER_TIME_STRUCT      history_time_start;
        short                           w_id;
    short                       d_id;
        DWORD                           dwThreadID[MAX_CUSTOMER_THREADS];
        HANDLE                          hThread[MAX_CUSTOMER_THREADS];
        char                            name[20];
        RETCODE                                 rc;
        DBINT                                   rcint;
        char                                    bcphint[128];
        char                                    cmd[256];
        // SQLRETURN                            rc_1;
        // SQLSMALLINT                          recnum, MsgLen;
        // SQLCHAR                                      SqlState[6],
Msg[SQL_MAX_MESSAGE_LENGTH];
        // SQLINTEGER                           NativeError;

        // Seed with unique number
        seed(5);

        printf("Loading customer and history tables...\n");

        // if build index before load...
        if ((aptr->build_index == 1) && (aptr->index_order == 1))
                BuildIndex("idxcuscl");

        // Initialize bulk copy
        sprintf(name, "%s..%s", aptr->database, "customer");
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
	rc = bcp_init(c_hdbc1, name, NULL, "logs\\customer.err", DB_IN);
	if (rc != SUCCEED)
		HandleErrorDBC(c_hdbc1);

	if ((aptr->build_index == 1) && (aptr->index_order == 1))
	{
		sprintf(bcphint, "tablock, order (c_w_id, c_d_id, c_id),
ROWS_PER_BATCH = %u", (aptr->num_warehouses * 30000));
		rc = bcp_control(c_hdbc1, BCPHINTS, (void*) bcphint);
		if (rc != SUCCEED)
			HandleErrorDBC(c_hdbc1);
	}

	sprintf(name, "%s..%s", aptr->database, "history");

	rc = bcp_init(c_hdbc2, name, NULL, "logs\\history.err", DB_IN);
	if (rc != SUCCEED)
		HandleErrorDBC(c_hdbc2);

	sprintf(bcphint, "tablock");
	rc = bcp_control(c_hdbc2, BCPHINTS, (void*) bcphint);
	if (rc != SUCCEED)
		HandleErrorDBC(c_hdbc2);

	customer_rows_loaded    = 0;
	history_rows_loaded     = 0;

	CustomerBufInit();

	customer_time_start.time_start = (TimeNow() / MILLI);
	history_time_start.time_start = (TimeNow() / MILLI);

	for (w_id = (short)aptr->starting_warehouse; w_id <= aptr-
>num_warehouses; w_id++)
	{
		for (d_id = 1; d_id <= DISTRICT_PER_WAREHOUSE; d_id++)
		{

			CustomerBufLoad(d_id, w_id);

			// Start parallel loading threads here...

			// Start customer table thread

			printf("...Loading customer table for: d_id = %d,
w_id = %d\n", d_id, w_id);

			hThread[0] = CreateThread(NULL,
                                                                0,

(LPTHREAD_START_ROUTINE) LoadCustomerTable,

&customer_time_start,
                                                                0,

&dwThreadID[0]);

			if (hThread[0] == NULL)
			{
				printf("Error, failed in creating creating
thread = 0.\n");
				exit(-1);
			}

			// Start History table thread

			printf("...Loading history table for: d_id = %d,
w_id = %d\n", d_id, w_id);

			hThread[1] = CreateThread(NULL,
                                                                0,

(LPTHREAD_START_ROUTINE) LoadHistoryTable,

&history_time_start,
                                                                0,

&dwThreadID[1]);

			if (hThread[1] == NULL)
			{
				printf("Error, failed in creating creating
thread = 1.\n");
				exit(-1);
			}

			WaitForSingleObject( hThread[0], INFINITE );
			WaitForSingleObject( hThread[1], INFINITE );

			if (CloseHandle(hThread[0]) == FALSE)
			{
				printf("Error, failed in closing customer
thread handle with errno: %d\n", GetLastError());
			}

			if (CloseHandle(hThread[1]) == FALSE)
			{
				printf("Error, failed in closing history
thread handle with errno: %d\n", GetLastError());
			}


		}
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        }

        // flush the bulk connection
        rcint = bcp_done(c_hdbc1);
        if (rcint < 0)
                HandleErrorDBC(c_hdbc1);

        rcint = bcp_done(c_hdbc2);
        if (rcint < 0)
                HandleErrorDBC(c_hdbc2);

        printf("Finished loading customer table.\n");

        // if build index after load...
        if ((aptr->build_index == 1) && (aptr->index_order == 0))
                BuildIndex("idxcuscl");

        // build non-clustered index
        if (aptr->build_index == 1)
                BuildIndex("idxcusnc");

        // Output the NURAND used for the loader into C_FIRST for C_ID =
1,
        // C_W_ID = 1, and C_D_ID = 1
        sprintf(cmd, "isql -S%s -U%s -P%s -d%s -e -Q\"update customer set
c_first = 'C_LOAD = %d' where c_id = 1 and c_w_id = 1 and c_d_id = 1\" >
logs\\nurand_load.log",
                                aptr->server,
                                aptr->user,
                                aptr->password,
                                aptr->database,
                                LOADER_NURAND_C);

        system(cmd);

        SQLFreeStmt(c_hstmt1, SQL_DROP);
        SQLDisconnect(c_hdbc1);
        SQLFreeConnect(c_hdbc1);

        SQLFreeStmt(c_hstmt2, SQL_DROP);
        SQLDisconnect(c_hdbc2);
        SQLFreeConnect(c_hdbc2);


    return;

}


//============================================================================
//
// Function   : CustomerBufInit
//
//============================================================================

void CustomerBufInit()
{
        int     i;

        for (i=0;i<customers_per_district;i++)
        {
                customer_buf[i].c_id = 0;
                customer_buf[i].c_d_id = 0;
                customer_buf[i].c_w_id = 0;

                strcpy(customer_buf[i].c_first,"");
                strcpy(customer_buf[i].c_middle,"");
                strcpy(customer_buf[i].c_last,"");
                strcpy(customer_buf[i].c_street_1,"");
                strcpy(customer_buf[i].c_street_2,"");
                strcpy(customer_buf[i].c_city,"");
                strcpy(customer_buf[i].c_state,"");
                strcpy(customer_buf[i].c_zip,"");
                strcpy(customer_buf[i].c_phone,"");
                strcpy(customer_buf[i].c_credit,"");

                customer_buf[i].c_credit_lim = 0;
                customer_buf[i].c_discount = (float) 0;

                //  fix to avoid ODBC float to numeric conversion problem.
                //      customer_buf[i].c_balance = 0;
                strcpy(customer_buf[i].c_balance,"");

                customer_buf[i].c_ytd_payment = 0;
                customer_buf[i].c_payment_cnt = 0;
                customer_buf[i].c_delivery_cnt = 0;

                strcpy(customer_buf[i].c_data,"");

                customer_buf[i].h_amount = 0;

                strcpy(customer_buf[i].h_data,"");

        }


}


//============================================================================
//
// Function   : CustomerBufLoad
//
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
// Fills shared buffer for HISTORY and CUSTOMER
//============================================================================

void CustomerBufLoad(int d_id, int w_id)
{
        long                                    i;
    CUSTOMER_SORT_STRUCT    c[CUSTOMERS_PER_DISTRICT];


        for (i=0;i<customers_per_district;i++)
        {
                if (i < 1000)
                        LastName(i, c[i].c_last);
                else
                        LastName(NURand(255,0,999,LOADER_NURAND_C),
c[i].c_last);

                MakeAlphaString(8,16,FIRST_NAME_LEN, c[i].c_first);

                c[i].c_id = i+1;

        }

        printf("...Loading customer buffer for: d_id = %d, w_id = %d\n",
                        d_id, w_id);

        for (i=0;i<customers_per_district;i++)
        {

                customer_buf[i].c_d_id = d_id;
                customer_buf[i].c_w_id = w_id;
                customer_buf[i].h_amount = 10.0;

                customer_buf[i].c_ytd_payment  = 10.0;

                customer_buf[i].c_payment_cnt  = 1;
                customer_buf[i].c_delivery_cnt = 0;

                // Generate CUSTOMER and HISTORY data

                customer_buf[i].c_id = c[i].c_id;

                strcpy(customer_buf[i].c_first, c[i].c_first);
                strcpy(customer_buf[i].c_last, c[i].c_last);

                customer_buf[i].c_middle[0] = 'O';
                customer_buf[i].c_middle[1] = 'E';

                MakeAddress(customer_buf[i].c_street_1,
                                customer_buf[i].c_street_2,
                                customer_buf[i].c_city,
                                customer_buf[i].c_state,
                                customer_buf[i].c_zip);

                MakeNumberString(16, 16, PHONE_LEN,
customer_buf[i].c_phone);

                if (RandomNumber(1L, 100L) > 10)
                        customer_buf[i].c_credit[0] = 'G';
                else
                        customer_buf[i].c_credit[0] = 'B';
                customer_buf[i].c_credit[1] = 'C';

                customer_buf[i].c_credit_lim = 50000.0;
                customer_buf[i].c_discount = ((float) RandomNumber(0L,
5000L)) / 10000.0;

                //  fix to avoid ODBC float to numeric conversion problem.

                //  customer_buf[i].c_balance = -10.0;
                strcpy(customer_buf[i].c_balance,"-10.0");

                MakeAlphaString(300, 500, C_DATA_LEN,
customer_buf[i].c_data);

                // Generate HISTORY data
                MakeAlphaString(12, 24, H_DATA_LEN,
customer_buf[i].h_data);

        }
}

//============================================================================
//
// Function   : LoadCustomerTable
//
//============================================================================

void LoadCustomerTable(LOADER_TIME_STRUCT *customer_time_start)
{
        int         i;
    long        c_id;
    short       c_d_id;
    short       c_w_id;
    char        c_first[FIRST_NAME_LEN+1];
    char        c_middle[MIDDLE_NAME_LEN+1];
    char        c_last[LAST_NAME_LEN+1];
    char        c_street_1[ADDRESS_LEN+1];
    char        c_street_2[ADDRESS_LEN+1];
    char        c_city[ADDRESS_LEN+1];
    char        c_state[STATE_LEN+1];
    char        c_zip[ZIP_LEN+1];
    char        c_phone[PHONE_LEN+1];
    char        c_credit[CREDIT_LEN+1];
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
    double      c_credit_lim;
    double      c_discount;

       //  fix to avoid ODBC float to numeric conversion problem.

       //  double            c_balance;
       char           c_balance[6];

    double      c_ytd_payment;
    short       c_payment_cnt;
    short       c_delivery_cnt;
    char        c_data[C_DATA_LEN+1];
       char           c_since[C_SINCE_LEN+1];
       RETCODE        rc;

    rc = bcp_bind(c_hdbc1, (BYTE *) &c_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT4, 1);
       if (rc != SUCCEED)
              HandleErrorDBC(c_hdbc1);

    rc = bcp_bind(c_hdbc1, (BYTE *) &c_d_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 2);
       if (rc != SUCCEED)
              HandleErrorDBC(c_hdbc1);

       rc = bcp_bind(c_hdbc1, (BYTE *) &c_w_id, 0, SQL_VARLEN_DATA, NULL,
0, SQLINT2, 3);
       if (rc != SUCCEED)
              HandleErrorDBC(c_hdbc1);

    rc = bcp_bind(c_hdbc1, (BYTE *) c_first, 0, FIRST_NAME_LEN, NULL, 0,
0, 4);
       if (rc != SUCCEED)
              HandleErrorDBC(c_hdbc1);

    rc = bcp_bind(c_hdbc1, (BYTE *) c_middle, 0, MIDDLE_NAME_LEN,NULL, 0,
0, 5);
       if (rc != SUCCEED)
              HandleErrorDBC(c_hdbc1);

    rc = bcp_bind(c_hdbc1, (BYTE *) c_last, 0, LAST_NAME_LEN, NULL, 0, 0,
6);
       if (rc != SUCCEED)
              HandleErrorDBC(c_hdbc1);

    rc = bcp_bind(c_hdbc1, (BYTE *) c_street_1, 0, ADDRESS_LEN, NULL, 0,
0, 7);
       if (rc != SUCCEED)
              HandleErrorDBC(c_hdbc1);

    rc = bcp_bind(c_hdbc1, (BYTE *) c_street_2, 0, ADDRESS_LEN, NULL,0,0,
8);
       if (rc != SUCCEED)
              HandleErrorDBC(c_hdbc1);

    rc = bcp_bind(c_hdbc1, (BYTE *) c_city, 0, ADDRESS_LEN, NULL, 0, 0,
9);
       if (rc != SUCCEED)
              HandleErrorDBC(c_hdbc1);

    rc = bcp_bind(c_hdbc1, (BYTE *) c_state, 0, STATE_LEN, NULL, 0, 0,
10);
       if (rc != SUCCEED)
              HandleErrorDBC(c_hdbc1);

    rc = bcp_bind(c_hdbc1, (BYTE *) c_zip, 0, ZIP_LEN, NULL, 0, 0, 11);
       if (rc != SUCCEED)
              HandleErrorDBC(c_hdbc1);

    rc = bcp_bind(c_hdbc1, (BYTE *) c_phone, 0, PHONE_LEN, NULL, 0, 0,
12);
       if (rc != SUCCEED)
              HandleErrorDBC(c_hdbc1);

       rc = bcp_bind(c_hdbc1, (BYTE *) &c_since, 0, C_SINCE_LEN, NULL, 0,
SQLCHARACTER, 13);
       if (rc != SUCCEED)
              HandleErrorDBC(c_hdbc1);

    rc = bcp_bind(c_hdbc1, (BYTE *) c_credit, 0, CREDIT_LEN, NULL, 0, 0,
14);
       if (rc != SUCCEED)
              HandleErrorDBC(c_hdbc1);

    rc = bcp_bind(c_hdbc1, (BYTE *) &c_credit_lim, 0, SQL_VARLEN_DATA,
NULL, 0, SQLFLT8, 15);
       if (rc != SUCCEED)
              HandleErrorDBC(c_hdbc1);

    rc = bcp_bind(c_hdbc1, (BYTE *) &c_discount, 0, SQL_VARLEN_DATA, NULL,
0, SQLFLT8, 16);
       if (rc != SUCCEED)
              HandleErrorDBC(c_hdbc1);


       //  fix to avoid ODBC float to numeric conversion problem.

    //  rc = bcp_bind(c_hdbc1, (BYTE *) &c_balance, 0, SQL_VARLEN_DATA,
NULL, 0, SQLFLT8, 17);
       //  if (rc != SUCCEED)
       //      HandleErrorDBC(c_hdbc1);
    rc = bcp_bind(c_hdbc1, (BYTE *) c_balance, 0, 5, NULL, 0,
SQLCHARACTER, 17);
       if (rc != SUCCEED)
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                HandleErrorDBC(c_hdbc1);


    rc = bcp_bind(c_hdbc1, (BYTE *) &c_ytd_payment, 0, SQL_VARLEN_DATA,
NULL, 0, SQLFLT8, 18);
        if (rc != SUCCEED)
                HandleErrorDBC(c_hdbc1);

    rc = bcp_bind(c_hdbc1, (BYTE *) &c_payment_cnt, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT2, 19);
        if (rc != SUCCEED)
                HandleErrorDBC(c_hdbc1);

    rc = bcp_bind(c_hdbc1, (BYTE *) &c_delivery_cnt,0, SQL_VARLEN_DATA,
NULL, 0, SQLINT2, 20);
        if (rc != SUCCEED)
                HandleErrorDBC(c_hdbc1);

    rc = bcp_bind(c_hdbc1, (BYTE *) c_data, 0, 500, NULL, 0, 0, 21);
        if (rc != SUCCEED)
                HandleErrorDBC(c_hdbc1);

    for (i = 0; i < customers_per_district; i++)
    {
                c_id = customer_buf[i].c_id;
                c_d_id = customer_buf[i].c_d_id;
                c_w_id = customer_buf[i].c_w_id;

                strcpy(c_first, customer_buf[i].c_first);
                strcpy(c_middle, customer_buf[i].c_middle);
                strcpy(c_last, customer_buf[i].c_last);
                strcpy(c_street_1, customer_buf[i].c_street_1);
                strcpy(c_street_2, customer_buf[i].c_street_2);
                strcpy(c_city, customer_buf[i].c_city);
                strcpy(c_state, customer_buf[i].c_state);
                strcpy(c_zip, customer_buf[i].c_zip);
                strcpy(c_phone, customer_buf[i].c_phone);
                strcpy(c_credit, customer_buf[i].c_credit);

                FormatDate(&c_since);

                c_credit_lim = customer_buf[i].c_credit_lim;
                c_discount = customer_buf[i].c_discount;

                //  fix to avoid ODBC float to numeric conversion problem.

                //  c_balance = customer_buf[i].c_balance;
                strcpy(c_balance, customer_buf[i].c_balance);

                c_ytd_payment = customer_buf[i].c_ytd_payment;
                c_payment_cnt = customer_buf[i].c_payment_cnt;
                c_delivery_cnt = customer_buf[i].c_delivery_cnt;
                strcpy(c_data, customer_buf[i].c_data);

                // Send data to server
                rc = bcp_sendrow(c_hdbc1);
                if (rc != SUCCEED)
                        HandleErrorDBC(c_hdbc1);

                customer_rows_loaded++;
                CheckForCommit(c_hdbc1, c_hstmt1, customer_rows_loaded,
"customer", &customer_time_start->time_start);
    }

}


//===========================================================================
//
// Function   : LoadHistoryTable
//
//===========================================================================

void LoadHistoryTable(LOADER_TIME_STRUCT *history_time_start)
{
        int         i;
    long        c_id;
    short       c_d_id;
    short       c_w_id;
        double          h_amount;
    char        h_data[H_DATA_LEN+1];
        char            h_date[H_DATE_LEN+1];
        RETCODE         rc;

    rc = bcp_bind(c_hdbc2, (BYTE *) &c_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT4, 1);
        if (rc != SUCCEED)
                HandleErrorDBC(c_hdbc2);

    rc = bcp_bind(c_hdbc2, (BYTE *) &c_d_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 2);
        if (rc != SUCCEED)
                HandleErrorDBC(c_hdbc2);

    rc = bcp_bind(c_hdbc2, (BYTE *) &c_w_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 3);
        if (rc != SUCCEED)
                HandleErrorDBC(c_hdbc2);

    rc = bcp_bind(c_hdbc2, (BYTE *) &c_d_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 4);
        if (rc != SUCCEED)
                HandleErrorDBC(c_hdbc2);
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
    rc = bcp_bind(c_hdbc2, (BYTE *) &c_w_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 5);
        if (rc != SUCCEED)
                HandleErrorDBC(c_hdbc2);

        rc = bcp_bind(c_hdbc2, (BYTE *) &h_date, 0, H_DATE_LEN, NULL, 0,
SQLCHARACTER, 6);
        if (rc != SUCCEED)
                HandleErrorDBC(c_hdbc2);

    rc = bcp_bind(c_hdbc2, (BYTE *) &h_amount, 0, SQL_VARLEN_DATA, NULL,
0, SQLFLT8, 7);
        if (rc != SUCCEED)
                HandleErrorDBC(c_hdbc2);

    rc = bcp_bind(c_hdbc2, (BYTE *) h_data, 0, H_DATA_LEN, NULL, 0, 0, 8);
        if (rc != SUCCEED)
                HandleErrorDBC(c_hdbc2);

    for (i = 0; i < customers_per_district; i++)
    {
                c_id = customer_buf[i].c_id;
                c_d_id = customer_buf[i].c_d_id;
                c_w_id = customer_buf[i].c_w_id;
                h_amount = customer_buf[i].h_amount;
                strcpy(h_data, customer_buf[i].h_data);

                FormatDate(&h_date);

                // send to server
                rc = bcp_sendrow(c_hdbc2);
                if (rc != SUCCEED)
                        HandleErrorDBC(c_hdbc2);

                history_rows_loaded++;
                CheckForCommit(c_hdbc2, c_hstmt2, history_rows_loaded,
"history", &history_time_start->time_start);
    }

}


//============================================================-
===============
//
// Function   : LoadOrders
//
//============================================================================
====

void LoadOrders()
{
        LOADER_TIME_STRUCT      orders_time_start;
        LOADER_TIME_STRUCT      new_order_time_start;
        LOADER_TIME_STRUCT      order_line_time_start;
        short                           w_id;
    short                       d_id;
        DWORD                           dwThreadID[MAX_ORDER_THREADS];
        HANDLE                          hThread[MAX_ORDER_THREADS];
        char                            name[20];
        RETCODE                                 rc;
        char                                    bcphint[128];

        // seed with unique number
        seed(6);

        printf("Loading orders...\n");

        // if build index before load...
        if ((aptr->build_index == 1) && (aptr->index_order == 1))
        {
                        BuildIndex("idxordcl");
                        BuildIndex("idxnodcl");
                        BuildIndex("idxodlcl");
        }

        // initialize bulk copy
        sprintf(name, "%s..%s", aptr->database, "orders");

        rc = bcp_init(o_hdbc1, name, NULL, "logs\\orders.err", DB_IN);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc1);

        if ((aptr->build_index == 1) && (aptr->index_order == 1))
        {
                sprintf(bcphint, "tablock, order (o_w_id, o_d_id, o_id),
ROWS_PER_BATCH = %u", (aptr->num_warehouses * 30000));
                rc = bcp_control(o_hdbc1, BCPHINTS, (void*) bcphint);
                if (rc != SUCCEED)
        }

        sprintf(name, "%s..%s", aptr->database, "new_order");

        rc = bcp_init(o_hdbc2, name, NULL, "logs\\neword.err", DB_IN);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc2);

        if ((aptr->build_index == 1) && (aptr->index_order == 1))
        {
                sprintf(bcphint, "tablock, order (no_w_id, no_d_id,
no_o_id), ROWS_PER_BATCH = %u", (aptr->num_warehouses * 9000));
                rc = bcp_control(o_hdbc2, BCPHINTS, (void*) bcphint);
                if (rc != SUCCEED)
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                        HandleErrorDBC(o_hdbc2);
        }

        sprintf(name, "%s..%s", aptr->database, "order_line");

        rc = bcp_init(o_hdbc3, name, NULL, "logs\\ordline.err", DB_IN);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc3);

    if ((aptr->build_index == 1) && (aptr->index_order == 1))
        {
                sprintf(bcphint, "tablock, order (ol_w_id, ol_d_id,
ol_o_id, ol_number), ROWS_PER_BATCH = %u", (aptr->num_warehouses *
300000));
                rc = bcp_control(o_hdbc3, BCPHINTS, (void*) bcphint);
                if (rc != SUCCEED)
                        HandleErrorDBC(o_hdbc3);
        }

        orders_rows_loaded      = 0;
        new_order_rows_loaded   = 0;
        order_line_rows_loaded  = 0;

        OrdersBufInit();

        orders_time_start.time_start = (TimeNow() / MILLI);
        new_order_time_start.time_start = (TimeNow() / MILLI);
        order_line_time_start.time_start = (TimeNow() / MILLI);

        for (w_id = (short)aptr->starting_warehouse; w_id <= aptr-
>num_warehouses; w_id++)
        {
                for (d_id = 1; d_id <= DISTRICT_PER_WAREHOUSE; d_id++)
                {

                        OrdersBufLoad(d_id, w_id);

                        // start parallel loading threads here...

                        // start Orders table thread

                        printf("...Loading Order Table for: d_id = %d, w_id
= %d\n", d_id, w_id);

                        hThread[0] = CreateThread(NULL,
                                                                        0,

(LPTHREAD_START_ROUTINE) LoadOrdersTable,

&orders_time_start,
                                                                        0,

&dwThreadID[0]);

                        if (hThread[0] == NULL)
                        {
                                printf("Error, failed in creating creating
thread = 0.\n");
                                exit(-1);
                        }

                        // start NewOrder table thread

                        printf("...Loading New-Order Table for: d_id = %d,
w_id = %d\n", d_id, w_id);

                        hThread[1] = CreateThread(NULL,
                                                                        0,

(LPTHREAD_START_ROUTINE) LoadNewOrderTable,

&new_order_time_start,
                                                                        0,

&dwThreadID[1]);

                        if (hThread[1] == NULL)
                        {
                                printf("Error, failed in creating creating
thread = 1.\n");
                                exit(-1);
                        }

                        // start Order-Line table thread

                        printf("...Loading Order-Line Table for: d_id = %d,
w_id = %d\n", d_id, w_id);

                        hThread[2] = CreateThread(NULL,
                                                                        0,

(LPTHREAD_START_ROUTINE) LoadOrderLineTable,

&order_line_time_start,
                                                                        0,

&dwThreadID[2]);

                        if (hThread[2] == NULL)
                        {
                                printf("Error, failed in creating creating
thread = 2.\n");
                                exit(-1);
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                        }

                        WaitForSingleObject( hThread[0], INFINITE );
                        WaitForSingleObject( hThread[1], INFINITE );
                        WaitForSingleObject( hThread[2], INFINITE );

                        if (CloseHandle(hThread[0]) == FALSE)
                        {
                                printf("Error, failed in closing Orders
thread handle with errno: %d\n", GetLastError());
                        }

                        if (CloseHandle(hThread[1]) == FALSE)
                        {
                                printf("Error, failed in closing NewOrder
thread handle with errno: %d\n", GetLastError());
                        }

                        if (CloseHandle(hThread[2]) == FALSE)
                        {
                                printf("Error, failed in closing OrderLine
thread handle with errno: %d\n", GetLastError());
                        }

                }
        }

        printf("Finished loading orders.\n");


    return;
}


//============================================================================
//
// Function   : OrdersBufInit
//
// Clears shared buffer for ORDERS, NEWORDER, and ORDERLINE
//
//============================================================================

void OrdersBufInit()
{
        int     i;
        int             j;

        for (i=0;i<orders_per_district;i++)
        {
                orders_buf[i].o_id = 0;
                orders_buf[i].o_d_id = 0;
                orders_buf[i].o_w_id = 0;
                orders_buf[i].o_c_id = 0;
                orders_buf[i].o_carrier_id = 0;
                orders_buf[i].o_ol_cnt = 0;
                orders_buf[i].o_all_local = 0;

                for (j=0;j<=14;j++)
                {
                        orders_buf[i].o_ol[j].ol = 0;
                        orders_buf[i].o_ol[j].ol_i_id = 0;
                        orders_buf[i].o_ol[j].ol_supply_w_id = 0;
                        orders_buf[i].o_ol[j].ol_quantity = 0;
                        orders_buf[i].o_ol[j].ol_amount = 0;
                        strcpy(orders_buf[i].o_ol[j].ol_dist_info,"");
                }
        }


}


//============================================================================
//
// Function   : OrdersBufLoad
//
// Fills shared buffer for ORDERS, NEWORDER, and ORDERLINE
//
//============================================================================

void OrdersBufLoad(int d_id, int w_id)
{

        int     cust[ORDERS_PER_DISTRICT+1];
        long    o_id;
        short   ol;

        printf("...Loading Order Buffer for: d_id = %d, w_id = %d\n",
                        d_id, w_id);

        GetPermutation(cust, orders_per_district);

        for (o_id=0;o_id<orders_per_district;o_id++)
        {

                // Generate ORDER and NEW-ORDER data

                orders_buf[o_id].o_d_id = d_id;
                orders_buf[o_id].o_w_id = w_id;
                orders_buf[o_id].o_id = o_id+1;
                orders_buf[o_id].o_c_id = cust[o_id+1];
                orders_buf[o_id].o_ol_cnt = (short)RandomNumber(5L, 15L);

                if (o_id < first_new_order)
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
		{
			orders_buf[o_id].o_carrier_id =
(short)RandomNumber(1L, 10L);
			orders_buf[o_id].o_all_local  = 1;
		}
		else
		{
			orders_buf[o_id].o_carrier_id = 0;
			orders_buf[o_id].o_all_local  = 1;
		}

		for (ol=0; ol<orders_buf[o_id].o_ol_cnt; ol++)
		{

			orders_buf[o_id].o_ol[ol].ol = ol+1;
			orders_buf[o_id].o_ol[ol].ol_i_id =
RandomNumber(1L, max_items);
			orders_buf[o_id].o_ol[ol].ol_supply_w_id = w_id;
			orders_buf[o_id].o_ol[ol].ol_quantity = 5;
			MakeAlphaString(24, 24, OL_DIST_INFO_LEN,
&orders_buf[o_id].o_ol[ol].ol_dist_info);

			// Generate ORDER-LINE data
			if (o_id < first_new_order)
			{
				orders_buf[o_id].o_ol[ol].ol_amount = 0;
				// Added to insure ol_delivery_d set
properly during load


	FormatDate(&orders_buf[o_id].o_ol[ol].ol_delivery_d);

			}
			else
			{
				orders_buf[o_id].o_ol[ol].ol_amount =
RandomNumber(1,999999)/100.0;
				// Added to insure ol_delivery_d set
properly during load

				// odbc datetime format

	strcpy(orders_buf[o_id].o_ol[ol].ol_delivery_d,"1899-12-31
00:00:00.000");

			}
		}
	}
}


//============================================================================
```

```
//
// Function   : LoadOrdersTable
//
//============================================================================

void LoadOrdersTable(LOADER_TIME_STRUCT *orders_time_start)
{
	int		i;
    long	o_id;
	short		o_d_id;
	short		o_w_id;
    long	o_c_id;
    short	o_carrier_id;
    short	o_ol_cnt;
    short	o_all_local;
	char		o_entry_d[O_ENTRY_D_LEN+1];
	RETCODE		rc;
	DBINT		rcint;

	// bind ORDER data
    rc = bcp_bind(o_hdbc1, (BYTE *) &o_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT4, 1);
	if (rc != SUCCEED)
		HandleErrorDBC(o_hdbc1);

    rc = bcp_bind(o_hdbc1, (BYTE *) &o_d_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 2);
	if (rc != SUCCEED)
		HandleErrorDBC(o_hdbc1);

    rc = bcp_bind(o_hdbc1, (BYTE *) &o_w_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 3);
	if (rc != SUCCEED)
		HandleErrorDBC(o_hdbc1);

    rc = bcp_bind(o_hdbc1, (BYTE *) &o_c_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT4, 4);
	if (rc != SUCCEED)
		HandleErrorDBC(o_hdbc1);

	rc = bcp_bind(o_hdbc1, (BYTE *) &o_entry_d, 0, O_ENTRY_D_LEN,
NULL, 0, SQLCHARACTER, 5);
	if (rc != SUCCEED)
		HandleErrorDBC(o_hdbc1);

    rc = bcp_bind(o_hdbc1, (BYTE *) &o_carrier_id, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT2, 6);
	if (rc != SUCCEED)
		HandleErrorDBC(o_hdbc1);

    rc = bcp_bind(o_hdbc1, (BYTE *) &o_ol_cnt, 0, SQL_VARLEN_DATA, NULL,
0, SQLINT2, 7);
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc1);

    rc = bcp_bind(o_hdbc1, (BYTE *) &o_all_local, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT2, 8);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc1);

    for (i = 0; i < orders_per_district; i++)
    {
                o_id         = orders_buf[i].o_id;
                o_d_id       = orders_buf[i].o_d_id;
                o_w_id       = orders_buf[i].o_w_id;
                o_c_id       = orders_buf[i].o_c_id;
                o_carrier_id = orders_buf[i].o_carrier_id;
                o_ol_cnt     = orders_buf[i].o_ol_cnt;
                o_all_local  = orders_buf[i].o_all_local;

                FormatDate(&o_entry_d);

                // send data to server
                rc = bcp_sendrow(o_hdbc1);
                if (rc != SUCCEED)
                        HandleErrorDBC(o_hdbc1);

                orders_rows_loaded++;
                CheckForCommit(o_hdbc1, o_hstmt1, orders_rows_loaded,
"orders", &orders_time_start->time_start);
    }

        // rcint = bcp_batch(o_hdbc1);
        // if (rcint < 0)
        //      HandleErrorDBC(o_hdbc1);

        if ((o_w_id == aptr->num_warehouses) && (o_d_id == 10))
        {
                rcint = bcp_done(o_hdbc1);
                if (rcint < 0)
                        HandleErrorDBC(o_hdbc1);

                SQLFreeStmt(o_hstmt1, SQL_DROP);
                SQLDisconnect(o_hdbc1);
                SQLFreeConnect(o_hdbc1);

                // if build index after load...
                if ((aptr->build_index == 1) && (aptr->index_order == 0))
                        BuildIndex("idxordcl");

                // build non-clustered index
                if (aptr->build_index == 1)
                        BuildIndex("idxordnc");
        }
}


//============================================================================
//
// Function   : LoadNewOrderTable
//
//============================================================================

void LoadNewOrderTable(LOADER_TIME_STRUCT *new_order_time_start)
{
    int         i;
    long        o_id;
    short       o_d_id;
    short       o_w_id;
        RETCODE         rc;
        DBINT           rcint;

        // Bind NEW-ORDER data

    rc = bcp_bind(o_hdbc2, (BYTE *) &o_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT4, 1);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc2);

    rc = bcp_bind(o_hdbc2, (BYTE *) &o_d_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 2);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc2);

    rc = bcp_bind(o_hdbc2, (BYTE *) &o_w_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 3);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc2);

        for (i = first_new_order; i < last_new_order; i++)
        {
                o_id    = orders_buf[i].o_id;
                o_d_id  = orders_buf[i].o_d_id;
                o_w_id  = orders_buf[i].o_w_id;

                rc = bcp_sendrow(o_hdbc2);
                if (rc != SUCCEED)
                        HandleErrorDBC(o_hdbc2);

                new_order_rows_loaded++;
                CheckForCommit(o_hdbc2, o_hstmt2, new_order_rows_loaded,
"new_order", &new_order_time_start->time_start);
        }

        // rcint = bcp_batch(o_hdbc2);
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        // if (rcint < 0)
        //      HandleErrorDBC(o_hdbc2);

        if ((o_w_id == aptr->num_warehouses) && (o_d_id == 10))
        {
                rcint = bcp_done(o_hdbc2);
                if (rcint < 0)
                        HandleErrorDBC(o_hdbc2);

                SQLFreeStmt(o_hstmt2, SQL_DROP);
                SQLDisconnect(o_hdbc2);
                SQLFreeConnect(o_hdbc2);

                // if build index after load...
                if ((aptr->build_index == 1) && (aptr->index_order == 0))
                        BuildIndex("idxnodcl");

        }

}


//================================================================================
//
// Function   : LoadOrderLineTable
//
//================================================================================

void LoadOrderLineTable(LOADER_TIME_STRUCT *order_line_time_start)
{
        int         i,j;
    long        o_id;
        short           o_d_id;
        short           o_w_id;
    long        ol;
        long            ol_i_id;
    short       ol_supply_w_id;
    short       ol_quantity;
    double      ol_amount;
    char        ol_dist_info[DIST_INFO_LEN+1];
        char            ol_delivery_d[OL_DELIVERY_D_LEN+1];
        RETCODE         rc;
        DBINT           rcint;

        // bind ORDER-LINE data
    rc = bcp_bind(o_hdbc3, (BYTE *) &o_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT4, 1);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc3);

    rc = bcp_bind(o_hdbc3, (BYTE *) &o_d_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 2);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc3);

    rc = bcp_bind(o_hdbc3, (BYTE *) &o_w_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT2, 3);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc3);

    rc = bcp_bind(o_hdbc3, (BYTE *) &ol, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT4, 4);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc3);

    rc = bcp_bind(o_hdbc3, (BYTE *) &ol_i_id, 0, SQL_VARLEN_DATA, NULL, 0,
SQLINT4, 5);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc3);

    rc = bcp_bind(o_hdbc3, (BYTE *) &ol_supply_w_id, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT2, 6);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc3);

        rc = bcp_bind(o_hdbc3, (BYTE *) &ol_delivery_d, 0,
OL_DELIVERY_D_LEN, NULL, 0, SQLCHARACTER, 7);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc3);

    rc = bcp_bind(o_hdbc3, (BYTE *) &ol_quantity, 0, SQL_VARLEN_DATA,
NULL, 0, SQLINT2, 8);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc3);

    rc = bcp_bind(o_hdbc3, (BYTE *) &ol_amount, 0, SQL_VARLEN_DATA, NULL,
0, SQLFLT8, 9);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc3);

    rc = bcp_bind(o_hdbc3, (BYTE *) ol_dist_info, 0, DIST_INFO_LEN, NULL,
0, 0, 10);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc3);

        for (i = 0; i < orders_per_district; i++)
    {
                o_id    = orders_buf[i].o_id;
                o_d_id  = orders_buf[i].o_d_id;
                o_w_id  = orders_buf[i].o_w_id;

                for (j=0; j < orders_buf[i].o_ol_cnt; j++)
                {
                        ol              = orders_buf[i].o_ol[j].ol;
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                        ol_i_id         = orders_buf[i].o_ol[j].ol_i_id;
                        ol_supply_w_id =
orders_buf[i].o_ol[j].ol_supply_w_id;
                        ol_quantity     = orders_buf[i].o_ol[j].ol_quantity;
                        ol_amount       = orders_buf[i].o_ol[j].ol_amount;


        strcpy(ol_delivery_d,orders_buf[i].o_ol[j].ol_delivery_d);


        strcpy(ol_dist_info,orders_buf[i].o_ol[j].ol_dist_info);

                        rc = bcp_sendrow(o_hdbc3);
                        if (rc != SUCCEED)
                                HandleErrorDBC(o_hdbc3);

                        order_line_rows_loaded++;
                        CheckForCommit(o_hdbc3, o_hstmt3,
order_line_rows_loaded, "order_line", &order_line_time_start->time_start);
                }

        }

        // rcint = bcp_batch(o_hdbc3);
        // if (rcint < 0)
        //      HandleErrorDBC(o_hdbc3);

        if ((o_w_id == aptr->num_warehouses) && (o_d_id == 10))
        {
                rcint = bcp_done(o_hdbc3);
                if (rcint < 0)
                        HandleErrorDBC(o_hdbc3);

                SQLFreeStmt(o_hstmt3, SQL_DROP);
                SQLDisconnect(o_hdbc3);
                SQLFreeConnect(o_hdbc3);

                // if build index after load...
                if ((aptr->build_index == 1) && (aptr->index_order == 0))
                        BuildIndex("idxodlcl");

        }

}


//============================================================================
//
// Function   : GetPermutation
//
//============================================================================
void GetPermutation(int perm[], int n)
{
        int i, r, t;

        for (i=1;i<=n;i++)
                perm[i] = i;

        for (i=1;i<=n;i++)
        {
                r = RandomNumber(i,n);
                t = perm[i];
                perm[i] = perm[r];
                perm[r] = t;
        }

}


//============================================================================
//
// Function   : CheckForCommit
//
//============================================================================

void CheckForCommit(HDBC hdbc,
                                        HSTMT hstmt,
                                int rows_loaded,
                                        char *table_name,
                                long *time_start)
{

    long       time_end, time_diff;
       // DBINT       rcint;

    if ( !(rows_loaded % aptr->batch) )
    {

                // rcint = bcp_batch(hdbc);
                // if (rcint < 0)
                //      HandleErrorDBC(hdbc);

                time_end = (TimeNow() / MILLI);
                time_diff = time_end - *time_start;

                printf("-> Loaded %ld rows into %s in %ld sec - Total = %d
(%.2f rps)\n",
                        aptr->batch,
                        table_name,
                                time_diff,
                        rows_loaded,
                                (float) aptr->batch / (time_diff ? time_diff
: 1L));
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                *time_start = time_end;
    }

    return;
}


//=============================================================================
//
// Function   : OpenConnections
//
//=============================================================================

void OpenConnections()
{

        RETCODE         rc;

        char                    szDriverString[300];
        char                    szDriverStringOut[1024];
        SQLSMALLINT             cbDriverStringOut;


        SQLAllocHandle(SQL_HANDLE_ENV, SQL_NULL_HANDLE, &henv );

        SQLSetEnvAttr(henv, SQL_ATTR_ODBC_VERSION, (void*)SQL_OV_ODBC3, 0
);

        SQLAllocHandle(SQL_HANDLE_DBC, henv , &i_hdbc1);
        SQLAllocHandle(SQL_HANDLE_DBC, henv , &w_hdbc1);
        SQLAllocHandle(SQL_HANDLE_DBC, henv , &c_hdbc1);
        SQLAllocHandle(SQL_HANDLE_DBC, henv , &c_hdbc2);
        SQLAllocHandle(SQL_HANDLE_DBC, henv , &o_hdbc1);
        SQLAllocHandle(SQL_HANDLE_DBC, henv , &o_hdbc2);
        SQLAllocHandle(SQL_HANDLE_DBC, henv , &o_hdbc3);

        SQLSetConnectAttr(i_hdbc1, SQL_COPT_SS_BCP, (void *)SQL_BCP_ON,
SQL_IS_INTEGER );
        SQLSetConnectAttr(w_hdbc1, SQL_COPT_SS_BCP, (void *)SQL_BCP_ON,
SQL_IS_INTEGER );
        SQLSetConnectAttr(c_hdbc1, SQL_COPT_SS_BCP, (void *)SQL_BCP_ON,
SQL_IS_INTEGER );
        SQLSetConnectAttr(c_hdbc2, SQL_COPT_SS_BCP, (void *)SQL_BCP_ON,
SQL_IS_INTEGER );
        SQLSetConnectAttr(o_hdbc1, SQL_COPT_SS_BCP, (void *)SQL_BCP_ON,
SQL_IS_INTEGER );
        SQLSetConnectAttr(o_hdbc2, SQL_COPT_SS_BCP, (void *)SQL_BCP_ON,
SQL_IS_INTEGER );
        SQLSetConnectAttr(o_hdbc3, SQL_COPT_SS_BCP, (void *)SQL_BCP_ON,
SQL_IS_INTEGER );

        // Open connections to SQL Server

        // Connection 1

        sprintf( szDriverString , "DRIVER={SQL
Server};SERVER=%s;UID=%s;PWD=%s;DATABASE=%s" ,
                                                aptr->server,
                                                aptr->user,
                                                aptr->password,
                                                aptr->database );

        rc = SQLSetConnectOption (i_hdbc1, SQL_PACKET_SIZE, aptr-
>pack_size);
        if (rc != SUCCEED)
                HandleErrorDBC(i_hdbc1);

        rc = SQLDriverConnect ( i_hdbc1,
                                            NULL,
                                                (SQLCHAR*)&szDriverString[0]
,
                                                SQL_NTS,

        (SQLCHAR*)&szDriverStringOut[0],
                                                sizeof(szDriverStringOut),
                                                &cbDriverStringOut,
                                                SQL_DRIVER_NOPROMPT );
        if (rc != SUCCEED)
                HandleErrorDBC(i_hdbc1);

        // Connection 2

        sprintf( szDriverString , "DRIVER={SQL
Server};SERVER=%s;UID=%s;PWD=%s;DATABASE=%s" ,
                                                aptr->server,
                                                aptr->user,
                                                aptr->password,
                                                aptr->database );

        rc = SQLSetConnectOption (w_hdbc1, SQL_PACKET_SIZE, aptr-
>pack_size);
        if (rc != SUCCEED)
                HandleErrorDBC(w_hdbc1);

        rc = SQLDriverConnect ( w_hdbc1,
                                                                NULL,

        (SQLCHAR*)&szDriverString[0] ,
                                                SQL_NTS,

        (SQLCHAR*)&szDriverStringOut[0],

        sizeof(szDriverStringOut),
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                                                &cbDriverStringOut,
                                                SQL_DRIVER_NOPROMPT
);
        if (rc != SUCCEED)
                HandleErrorDBC(w_hdbc1);

        // Connection 3

        sprintf( szDriverString , "DRIVER={SQL
Server};SERVER=%s;UID=%s;PWD=%s;DATABASE=%s" ,
                                                aptr->server,
                                                aptr->user,
                                                aptr->password,
                                                aptr->database );

        rc = SQLSetConnectOption (c_hdbc1, SQL_PACKET_SIZE, aptr-
>pack_size);
        if (rc != SUCCEED)
                HandleErrorDBC(c_hdbc1);

        rc = SQLDriverConnect ( c_hdbc1,
                                                NULL,

        (SQLCHAR*)&szDriverString[0] ,
                                                SQL_NTS,

        (SQLCHAR*)&szDriverStringOut[0],

        sizeof(szDriverStringOut),
                                                &cbDriverStringOut,
                                                SQL_DRIVER_NOPROMPT
);
        if (rc != SUCCEED)
                HandleErrorDBC(c_hdbc1);

        // Connection 4

        sprintf( szDriverString , "DRIVER={SQL
Server};SERVER=%s;UID=%s;PWD=%s;DATABASE=%s" ,
                                                aptr->server,
                                                aptr->user,
                                                aptr->password,
                                                aptr->database );

        rc = SQLSetConnectOption (c_hdbc2, SQL_PACKET_SIZE, aptr-
>pack_size);
        if (rc != SUCCEED)
                HandleErrorDBC(c_hdbc2);

        rc = SQLDriverConnect ( c_hdbc2,
                                                NULL,

        (SQLCHAR*)&szDriverString[0] ,
                                                SQL_NTS,

        (SQLCHAR*)&szDriverStringOut[0],

        sizeof(szDriverStringOut),
                                                &cbDriverStringOut,
                                                SQL_DRIVER_NOPROMPT
);
        if (rc != SUCCEED)
                HandleErrorDBC(c_hdbc2);

        // Connection 5

        sprintf( szDriverString , "DRIVER={SQL
Server};SERVER=%s;UID=%s;PWD=%s;DATABASE=%s" ,
                                                aptr->server,
                                                aptr->user,
                                                aptr->password,
                                                aptr->database );

        rc = SQLSetConnectOption (o_hdbc1, SQL_PACKET_SIZE, aptr-
>pack_size);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc1);

        rc = SQLDriverConnect ( o_hdbc1,
                                                NULL,

        (SQLCHAR*)&szDriverString[0] ,
                                                SQL_NTS,

        (SQLCHAR*)&szDriverStringOut[0],

        sizeof(szDriverStringOut),
                                                &cbDriverStringOut,
                                                SQL_DRIVER_NOPROMPT
);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc1);

        // Connection 6

        sprintf( szDriverString , "DRIVER={SQL
Server};SERVER=%s;UID=%s;PWD=%s;DATABASE=%s" ,
                                                aptr->server,
                                                aptr->user,
                                                aptr->password,
                                                aptr->database );
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        rc = SQLSetConnectOption (o_hdbc2, SQL_PACKET_SIZE, aptr-
>pack_size);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc2);

        rc = SQLDriverConnect ( o_hdbc2,
                                                NULL,

        (SQLCHAR*)&szDriverString[0] ,
                                                SQL_NTS,

        (SQLCHAR*)&szDriverStringOut[0],

        sizeof(szDriverStringOut),
                                                &cbDriverStringOut,
                                                SQL_DRIVER_NOPROMPT
);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc2);

        // Connection 7

        sprintf( szDriverString , "DRIVER={SQL
Server};SERVER=%s;UID=%s;PWD=%s;DATABASE=%s" ,
                                                aptr->server,
                                                aptr->user,
                                                aptr->password,
                                                aptr->database );

        rc = SQLSetConnectOption (o_hdbc3, SQL_PACKET_SIZE, aptr-
>pack_size);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc3);

        rc = SQLDriverConnect ( o_hdbc3,
                                                NULL,

        (SQLCHAR*)&szDriverString[0] ,
                                                SQL_NTS,

        (SQLCHAR*)&szDriverStringOut[0],

        sizeof(szDriverStringOut),
                                                &cbDriverStringOut,
                                                SQL_DRIVER_NOPROMPT
);
        if (rc != SUCCEED)
                HandleErrorDBC(o_hdbc3);

}

//============================================================================
```

```
//
// Function name: BuildIndex
//
//============================================================================

void BuildIndex(char   *index_script)
{
        char    cmd[256];

        printf("Starting index creation:  %s\n",index_script);

        sprintf(cmd, "isql -S%s -U%s -P%s -e -i%s\\%s.sql > logs\\%s.log",
                                aptr->server,
                                aptr->user,
                                aptr->password,
                                aptr->index_script_path,
                                index_script,
                                index_script);

        system(cmd);

        printf("Finished index creation:  %s\n",index_script);
}


void HandleErrorDBC (SQLHDBC  hdbc1)
{

        SQLCHAR         SqlState[6], Msg[SQL_MAX_MESSAGE_LENGTH];
        SQLINTEGER  NativeError;
        SQLSMALLINT i, MsgLen;
        SQLRETURN   rc2;
        char            timebuf[128];
        char            datebuf[128];
        FILE            *fp1;

        i = 1;
        while (( rc2 = SQLGetDiagRec(SQL_HANDLE_DBC , hdbc1, i, SqlState ,
&NativeError,
                                        Msg, sizeof(Msg) , &MsgLen )) !=
SQL_NO_DATA )
        {

                sprintf( szLastError , "%s" ,  Msg );

                _strtime(timebuf);
                _strdate(datebuf);

                printf( "[%s : %s] %s\n" , datebuf, timebuf, szLastError);

                fp1 = fopen("logs\\tpccldr.err","w");
                if (fp1 == NULL)
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                printf("ERROR:  Unable to open errorlog file.\n");
        else
        {
                fprintf(fp1, "[%s : %s] %s\n" , datebuf, timebuf,
szLastError);
                fclose(fp1);
        }

        i++;

    }
}

void HandleErrorSTMT (HSTMT  hstmt1)
{

    SQLCHAR        SqlState[6], Msg[SQL_MAX_MESSAGE_LENGTH];
    SQLINTEGER  NativeError;
    SQLSMALLINT i, MsgLen;
    SQLRETURN   rc2;
    char           timebuf[128];
    char           datebuf[128];
    FILE           *fp1;

    i = 1;
    while (( rc2 = SQLGetDiagRec(SQL_HANDLE_STMT , hstmt1, i, SqlState
, &NativeError,
                                Msg, sizeof(Msg) , &MsgLen )) !=
SQL_NO_DATA )
    {

        sprintf( szLastError , "%s" ,  Msg );

        _strtime(timebuf);
        _strdate(datebuf);

        printf( "[%s : %s] %s\n" , datebuf, timebuf, szLastError);

        fp1 = fopen("logs\\tpccldr.err","w");
        if (fp1 == NULL)
                printf("ERROR:  Unable to open errorlog file.\n");
        else
        {
                fprintf(fp1, "[%s : %s] %s\n" , datebuf, timebuf,
szLastError);
                fclose(fp1);
        }

        i++;

    }
}
```

```
void FormatDate ( char* szTimeCOutput )
{

    struct tm when;
    time_t now;

    time( &now );
    when  = *localtime( &now );

    mktime( &when );

    // odbc datetime format
    strftime( szTimeCOutput , 30 , "%Y-%m-%d %H:%M:%S.000", &when );

    return;

}


//============================================================================
//
// Function   : CheckSQL
//
//============================================================================

void CheckSQL()
{

    RETCODE         rc;

    char                    szDriverString[300];
    char                    szDriverStringOut[1024];
    int                             SQLBuildFlag;

    SQLSMALLINT             cbDriverStringOut;
    SQLCHAR                 SQLVersion[19];
    SQLINTEGER              SQLVersionInd;



    SQLAllocHandle(SQL_HANDLE_ENV, SQL_NULL_HANDLE, &henv );

    SQLSetEnvAttr(henv, SQL_ATTR_ODBC_VERSION, (void*)SQL_OV_ODBC3, 0
);

    SQLAllocHandle(SQL_HANDLE_DBC, henv , &v_hdbc);

    SQLSetConnectAttr(v_hdbc, SQL_COPT_SS_BCP, (void *)SQL_BCP_ON,
SQL_IS_INTEGER );
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
	// Open connection to SQL Server

	sprintf( szDriverString , "DRIVER={SQL
Server};SERVER=%s;UID=%s;PWD=%s" ,
                                                aptr->server,
                                                aptr->user,
                                                aptr->password );

	if ( SQLSetConnectAttr( v_hdbc, SQL_ATTR_PACKET_SIZE,
(SQLPOINTER)aptr->pack_size, SQL_IS_UINTEGER ) != SQL_SUCCESS )
		HandleErrorDBC(v_hdbc);

	rc = SQLDriverConnect ( v_hdbc,
                                        NULL,
                                            (SQLCHAR*)&szDriverString[0]
,
                                            SQL_NTS,

	(SQLCHAR*)&szDriverStringOut[0],
                                            sizeof(szDriverStringOut),
                                            &cbDriverStringOut,
                                            SQL_DRIVER_NOPROMPT );

	if ((rc != SQL_SUCCESS) && (rc != SQL_SUCCESS_WITH_INFO))
		HandleErrorDBC(v_hdbc);

	if ( SQLAllocHandle(SQL_HANDLE_STMT, v_hdbc , &v_hstmt) !=
SQL_SUCCESS )
		HandleErrorSTMT(v_hstmt);

	rc = SQLBindCol(v_hstmt, 4, SQL_C_CHAR, &SQLVersion,
sizeof(SQLVersion), &SQLVersionInd);

	// issue SQL Server extended stored procedure (xp_msver) to
determine installed version
	rc = SQLExecDirect(v_hstmt, "EXECUTE xp_msver ProductVersion",
SQL_NTS);

	if ((rc != SQL_SUCCESS) && (rc != SQL_SUCCESS_WITH_INFO))
		HandleErrorSTMT(v_hstmt);

	rc = SQLFetch(v_hstmt);

	if (rc != SQL_SUCCESS)
		HandleErrorDBC(v_hdbc);

	// Check build number to ensure 7.00.623 or higher

	SQLBuildFlag = 1;

	if ( SQLVersion[0] == 55 )
	{
		if ( SQLVersion[2] == 48 )
		{
			if ( SQLVersion[5] == 56 )
			{
				if ( (SQLVersion[6] >= 48) &  (SQLVersion[7]
>= 53) )
				{
					SQLBuildFlag = 0;
					printf("You are using SQL Server
version = %9s\n\n", SQLVersion);
				}
				else
				{
					SQLBuildFlag = 1;
				}
			}
			else
			{
				if ( SQLVersion[5] >= 54 )
				{
					if ( (SQLVersion[6] >= 50) &
(SQLVersion[7] >= 51) )
					{
						SQLBuildFlag = 0;
						printf("You are using SQL
Server version = %9s\n\n", SQLVersion);
					}
					else
					{
						SQLBuildFlag = 1;
					}
				}
				else
				{
					if ( SQLVersion[5] >= 55 )
					{
						if ( (SQLVersion[6] >= 48) &
(SQLVersion[7] >= 48) )
						{
							SQLBuildFlag = 0;
							printf("You are using
SQL Server version = %9s\n\n", SQLVersion);
						}
						else
						{
							SQLBuildFlag = 1;
						}
					}
				}

			}
		}
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                else
                {
                        if ( SQLVersion[5] >= 49 )
                        {
                                if ( (SQLVersion[6] >= 52) & (SQLVersion[7]
>= 48) )
                                {
                                        SQLBuildFlag = 0;
                                        printf("You are using SQL Server
version = %9s\n\n", SQLVersion);
                                }
                                else
                                {
                                        SQLBuildFlag = 1;
                                }
                        }
                        else
                        {
                                SQLBuildFlag = 1;
                        }
                }
        }
        else
        {
                SQLBuildFlag = 1;
        }


        if ( SQLBuildFlag == 1 )
        {
                printf("ERROR.  The SQL Server version you are using is not
supported\n");
                printf("for TPC-C benchmarking.  You currently have SQL
Server version %9s\n",SQLVersion);
                printf("installed.  Please upgrade to Microsoft SQL Server
7.00.623 or better.\n");
                printf("and re-run the SETUP program.\n\n");
                exit(1);
        }

        SQLFreeHandle(SQL_HANDLE_STMT, v_hstmt);
        SQLDisconnect(v_hdbc);
        SQLFreeHandle(SQL_HANDLE_DBC, v_hdbc);

        return;

}

//============================================================================
//
// Function   : CheckDataBase
//
//============================================================================

void CheckDataBase()
{

        RETCODE         rc;

        char                    szDriverString[300];
        char                    szDriverStringOut[1024];
        char                    TablesBitMap[9] = {"000000000"};
        int                             i, ExitFlag;

        SQLSMALLINT             cbDriverStringOut;
        SQLCHAR                 TabName[10];
        SQLINTEGER              TabNameInd, TabCount, TabCountInd;


        ExitFlag = 0;

        SQLAllocHandle(SQL_HANDLE_ENV, SQL_NULL_HANDLE, &henv );

        SQLSetEnvAttr(henv, SQL_ATTR_ODBC_VERSION, (void*)SQL_OV_ODBC3, 0
);

        SQLAllocHandle(SQL_HANDLE_DBC, henv , &v_hdbc);

        SQLSetConnectAttr(v_hdbc, SQL_COPT_SS_BCP, (void *)SQL_BCP_ON,
SQL_IS_INTEGER );

        // Open connection to SQL Server

        sprintf( szDriverString , "DRIVER={SQL
Server};SERVER=%s;UID=%s;PWD=%s;DATABASE=%s" ,
                                                aptr->server,
                                                aptr->user,
                                                aptr->password,
                                                aptr->database );

        rc = SQLSetConnectAttr( v_hdbc, SQL_ATTR_PACKET_SIZE,
(SQLPOINTER)aptr->pack_size, SQL_IS_UINTEGER );
        if (rc != SQL_SUCCESS)
                HandleErrorDBC(v_hdbc);

        rc = SQLDriverConnect ( v_hdbc,
                                                NULL,
                                                    (SQLCHAR*)&szDriverString[0]
,
                                                    SQL_NTS,

        (SQLCHAR*)&szDriverStringOut[0],
                                                    sizeof(szDriverStringOut),
                                                    &cbDriverStringOut,
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                                        SQL_DRIVER_NOPROMPT );

        // if the rc is SQL_ERROR, the the TPCC database probably does not
exist
        if (rc == SQL_ERROR)
        {
                printf("The database TPCC does not appear to exist!\n");
                printf("\nCheck LOGS\\ directory for database creation
errors.\n");

                // cleanup database connections and handles
                SQLFreeHandle(SQL_HANDLE_STMT, v_hstmt);
                SQLDisconnect(v_hdbc);
                SQLFreeHandle(SQL_HANDLE_DBC, v_hdbc);

                // since there is not a database, exit back to SETUP.CMD
                exit(1);
        }

        if ( SQLAllocHandle(SQL_HANDLE_STMT, v_hdbc , &v_hstmt) !=
SQL_SUCCESS )
                HandleErrorDBC(v_hdbc);

        if ( SQLBindCol(v_hstmt, 1, SQL_C_ULONG, &TabCount, 0,
&TabCountInd) != SQL_SUCCESS )
                HandleErrorSTMT(v_hstmt);

        // count the number of user tables from sysobjects
        rc = SQLExecDirect(v_hstmt, "select count(*) from sysobjects where
xtype = \'U\'", SQL_NTS);
        if ((rc != SQL_SUCCESS) && (rc != SQL_SUCCESS_WITH_INFO))
                HandleErrorSTMT(v_hstmt);

        if ( SQLFetch(v_hstmt) != SQL_SUCCESS )
                HandleErrorSTMT(v_hstmt);

        // if the number of tables is less than 9, select all the user
tables in TPCC
        if (TabCount != 9)
        {
                SQLFreeHandle(SQL_HANDLE_STMT, v_hstmt);

                SQLAllocHandle(SQL_HANDLE_STMT, v_hdbc , &v_hstmt);

                if ( SQLBindCol(v_hstmt, 1, SQL_C_CHAR, &TabName,
sizeof(TabName), &TabNameInd) != SQL_SUCCESS )
                        HandleErrorSTMT(v_hstmt);

                // select the list of user tables into a result set
                rc = SQLExecDirect(v_hstmt, "select * from sysobjects where
xtype = \'U\'", SQL_NTS);
                if ((rc != SQL_SUCCESS) && (rc != SQL_SUCCESS_WITH_INFO))
                        HandleErrorSTMT(v_hstmt);

                // go through the result set and set the bitmap for each
found table
                // set the bitmap to '1' if the table name is found

                while ((rc = SQLFetch(v_hstmt)) != SQL_NO_DATA)
                {
                        switch( TabName[0] )
                        {
                        case 'w':
                                TablesBitMap[0] = '1';
                                break;
                        case 'd':
                                TablesBitMap[1] = '1';
                                break;
                        case 'c':
                                TablesBitMap[2] = '1';
                                break;
                        case 'h':
                                TablesBitMap[3] = '1';
                                break;
                        case 'n':
                                TablesBitMap[4] = '1';
                                break;
                        case 'o':
                                if (TabName[5] = 's')
                                        TablesBitMap[5] = '1';
                                if (TabName[5] = '_')
                                        TablesBitMap[6] = '1';
                                break;
                        case 'i':
                                TablesBitMap[7] = '1';
                                break;
                        case 's':
                                TablesBitMap[8] = '1';
                                break;
                        }
                }

                // a '0' ExitFlag means do NOT exit the loader early, a '1'
means exit the loader early
                ExitFlag = 0;

                // interate through the bitmap to display which table(s) is
actually missing
                for (i = 0; i <= 8; i++)
                {
                        switch(i)
                        {
                        case 0:
                                if (TablesBitMap[i] == '0')
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                        {
                                printf("The Warehouse table is
missing or damaged.\n");
                                ExitFlag = 1;
                        }
                        break;
                case 1:
                        if (TablesBitMap[i] == '0')
                        {
                                printf("The District table is
missing or damaged.\n");
                                ExitFlag = 1;
                        }
                        break;
                case 2:
                        if (TablesBitMap[i] == '0')
                        {
                                printf("The Customer table is
missing or damaged.\n");
                                ExitFlag = 1;
                        }
                        break;
                case 3:
                        if (TablesBitMap[i] == '0')
                        {
                                printf("The History table is missing
or damaged.\n");
                                ExitFlag = 1;
                        }
                        break;
                case 4:
                        if (TablesBitMap[i] == '0')
                        {
                                printf("The New_Order table is
missing or damaged.\n");
                                ExitFlag = 1;
                        }
                        break;
                case 5:
                        if (TablesBitMap[i] == '0')
                        {
                                printf("The Orders table is missing
or damaged.\n");
                                ExitFlag = 1;
                        }
                        break;
                case 6:
                        if (TablesBitMap[i] == '0')
                        {
                                printf("The Order_Line table is
missing or damaged.\n");
                                ExitFlag = 1;
                        }
                        break;
                case 7:
                        if (TablesBitMap[i] == '0')
                        {
                                printf("The Item table is missing or
damaged.\n");
                                ExitFlag = 1;
                        }
                        break;
                case 8:
                        if (TablesBitMap[i] == '0')
                        {
                                printf("The Stock table is missing
or damaged.\n");
                                ExitFlag = 1;
                        }
                        break;
                }
        }

        // if one or more tables are missing, display message and
exit the loader
        if (ExitFlag = 1)
        {
                printf("\nExiting TPC-C Loader!\n");
                printf("\nCheck LOGS\\ directory for database\n");
                printf("or table creation errors.\n");

                // cleanup database connections and handles
                SQLFreeHandle(SQL_HANDLE_STMT, v_hstmt);
                SQLDisconnect(v_hdbc);
                SQLFreeHandle(SQL_HANDLE_DBC, v_hdbc);

                exit(1);
        }
    }

    // cleanup database connections and handles
    SQLFreeHandle(SQL_HANDLE_STMT, v_hstmt);
    SQLDisconnect(v_hdbc);
    SQLFreeHandle(SQL_HANDLE_DBC, v_hdbc);

    return;
}
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

Appendix C - Tunable Parameters and Options

This section discloses hardware information and the Windows 2000 Advanced Server registry parameters used on the PRIMERGY N400 server system.

```
System Information report written at: 08/29/2000 12:28:05
[System Summary]

Item    Value
OS Name Microsoft Windows 2000 Advanced Server
Version 5.0.2195 Service Pack 1 Build 2195
OS Manufacturer         Microsoft Corporation
System Name     H400
System Manufacturer     FSC
System Model    H400
System Type     X86-based PC
Processor       x86 Family 6 Model 10 Stepping 1 GenuineIntel ~700 Mhz
Processor       x86 Family 6 Model 10 Stepping 1 GenuineIntel ~700 Mhz
Processor       x86 Family 6 Model 10 Stepping 1 GenuineIntel ~700 Mhz
Processor       x86 Family 6 Model 10 Stepping 1 GenuineIntel ~700 Mhz
BIOS Version    PhoenixBIOS Version 4.06  Rev. 1.06.1173
Windows Directory       C:\WINNT
System Directory        C:\WINNT\System32
Boot Device     \Device\Harddisk0\Partition1
Locale  United States
User Name       H400\Administrator
Time Zone       W. Europe Daylight Time
Total Physical Memory  8,207,652 KB
Available Physical Memory      58,676 KB
Total Virtual Memory   18,308,324 KB
Available Virtual Memory       2,184,620 KB
Page File Space         10,100,672 KB
Page File       C:\pagefile.sys

[I/O]

Address Range   Device  Status
0x0000-0x03AF   PCI bus OK
0x0000-0x03AF   Direct memory access controller       OK
0x03B0-0x03DF   PCI bus OK
0x03B0-0x03DF   ATI Technologies Inc. RAGE XL PCI     OK
0x03E0-0x0CF7   PCI bus OK
0x0D00-0x0FFF   PCI bus OK
0x1000-0x3FFF   PCI bus OK
0x1000-0x3FFF   ATI Technologies Inc. RAGE XL PCI     OK
0x03C0-0x03DF   ATI Technologies Inc. RAGE XL PCI     OK
0x1450-0x1457   PCI Device      OK
0x2000-0x2FFF   DEC 21154 PCI to PCI bridge   OK
0x2000-0x2FFF   Mylex EXR2000 Disk Array Controller   OK
0x3000-0x3FFF   DEC 21154 PCI to PCI bridge   OK
0x3000-0x3FFF   Mylex EXR2000 Disk Array Controller   OK
0x0A79-0x0A79   ISAPNP Read Data Port  OK
0x0279-0x0279   ISAPNP Read Data Port  OK
0x02F4-0x02F7   ISAPNP Read Data Port  OK
0x0060-0x0060   Standard 101/102-Key or Microsoft Natural PS/2 Keyboard
        OK
0x0064-0x0064   Standard 101/102-Key or Microsoft Natural PS/2 Keyboard
        OK
0x0081-0x008F   Direct memory access controller       OK
0x00C0-0x00DF   Direct memory access controller       OK
0x0070-0x0071   System CMOS/real time clock   OK
0x0020-0x0021   Programmable interrupt controller     OK
0x00A0-0x00A1   Programmable interrupt controller     OK
0x00F0-0x00FF   Numeric data processor OK
0x0040-0x0043   System timer   OK
0x0061-0x0061   System speaker OK
0x0026-0x0027   Motherboard resources  OK
0x0080-0x0080   Motherboard resources  OK
0x0500-0x054F   Motherboard resources  OK
0x0580-0x058F   Motherboard resources  OK
0x040B-0x040B   Motherboard resources  OK
0x04D0-0x04D1   Motherboard resources  OK
0x04D6-0x04D6   Motherboard resources  OK
0x0C00-0x0C01   Motherboard resources  OK
0x0C14-0x0C14   Motherboard resources  OK
0x0C49-0x0C4A   Motherboard resources  OK
0x0C52-0x0C52   Motherboard resources  OK
0x0C6C-0x0C6C   Motherboard resources  OK
0x0C6F-0x0C6F   Motherboard resources  OK
0x0C90-0x0C97   Motherboard resources  OK
0x0CA0-0x0CBF   Motherboard resources  OK
0x0CD6-0x0CD7   Motherboard resources  OK
0x0F50-0x0F57   Motherboard resources  OK
0x03F0-0x03F5   Standard floppy disk controller       OK
0x03F7-0x03F7   Standard floppy disk controller       OK
0x3FF0-0x3FFF   Standard Dual Channel PCI IDE Controller      OK
0x01F0-0x01F7   Primary IDE Channel    OK
0x03F6-0x03F6   Primary IDE Channel    OK
0x0170-0x0177   Secondary IDE Channel  OK
0x0376-0x0376   Secondary IDE Channel  OK
0x4000-0x6FFF   PCI bus OK
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
0x4000-0x6FFF  Adaptec AIC-7899 Ultra160/m PCI SCSI Card    OK
0x4400-0x44FF  Adaptec AIC-7899 Ultra160/m PCI SCSI Card    OK
0x5000-0x5FFF  DEC 21154 PCI to PCI bridge   OK
0x5000-0x5FFF  Mylex EXR2000 Disk Array Controller  OK
0x6000-0x6FFF  DEC 21154 PCI to PCI bridge   OK
0x6000-0x6FFF  Mylex EXR2000 Disk Array Controller  OK
0x7000-0x8FFF  PCI bus OK
0x7000-0x8FFF  DEC 21154 PCI to PCI bridge   OK
0x7000-0x8FFF  Mylex EXR2000 Disk Array Controller  OK
0x8000-0x8FFF  DEC 21154 PCI to PCI bridge   OK
0x8000-0x8FFF  Mylex EXR2000 Disk Array Controller  OK

[IRQs]

IRQ Number     Device
9      Microsoft ACPI-Compliant System
9      PCI Device
17     ATI Technologies Inc. RAGE XL PCI
26     Mylex EXR2000 Disk Array Controller
24     Mylex EXR2000 Disk Array Controller
1      Standard 101/102-Key or Microsoft Natural PS/2 Keyboard
8      System CMOS/real time clock
13     Numeric data processor
12     PS/2 Compatible Mouse
6      Standard floppy disk controller
18     Adaptec AIC-7899 Ultra160/m PCI SCSI Card
19     Adaptec AIC-7899 Ultra160/m PCI SCSI Card
20     Mylex EXR2000 Disk Array Controller
22     Mylex EXR2000 Disk Array Controller
28     Mylex EXR2000 Disk Array Controller
29     Mylex EXR2000 Disk Array Controller
31     Alteon WebSystems PCI Gigabit Ethernet Adapter

[Memory]

Range  Device  Status
0xA0000-0xBFFFF       PCI bus OK
0xA0000-0xBFFFF       ATI Technologies Inc. RAGE XL PCI    OK
0xD0000-0xE7FFF       PCI bus OK
0xF5000000-0xF77FFFFF  PCI bus OK
0xF5000000-0xF77FFFFF  ATI Technologies Inc. RAGE XL PCI    OK
0xF7800000-0xF87FFFFF  PCI bus OK
0xF7800000-0xF87FFFFF  DEC 21154 PCI to PCI bridge   OK
0xF7800000-0xF87FFFFF  Mylex EXR2000 Disk Array Controller  OK
0xF6121000-0xF6121FFF  ATI Technologies Inc. RAGE XL PCI    OK
0xF6122000-0xF6122FFF  PCI Device     OK
0xF6000000-0xF60FFFFF  PCI Device     OK
0xF6800000-0xF6FFFFFF  DEC 21154 PCI to PCI bridge   OK
0xF6800000-0xF6FFFFFF  Mylex EXR2000 Disk Array Controller  OK
0xF7000000-0xF77FFFFF  DEC 21154 PCI to PCI bridge   OK
0xF7000000-0xF77FFFFF  Mylex EXR2000 Disk Array Controller  OK
0xF8000000-0xF87FFFFF  DEC 21154 PCI to PCI bridge   OK
0xF8000000-0xF87FFFFF  Mylex EXR2000 Disk Array Controller  OK
0xFEC00000-0xFEC0FFFF  Motherboard resources  OK
0xFEE00000-0xFEE00FFF  Motherboard resources  OK
0xF8800000-0xF9FFFFFF  PCI bus OK
0xF8800000-0xF9FFFFFF  Adaptec AIC-7899 Ultra160/m PCI SCSI Card    OK
0xFA000000-0xFAFFFFFF  PCI bus OK
0xFA000000-0xFAFFFFFF  DEC 21154 PCI to PCI bridge   OK
0xFA000000-0xFAFFFFFF  Mylex EXR2000 Disk Array Controller  OK
0xF8801000-0xF8801FFF  Adaptec AIC-7899 Ultra160/m PCI SCSI Card    OK
0xF9000000-0xF97FFFFF  DEC 21154 PCI to PCI bridge   OK
0xF9000000-0xF97FFFFF  Mylex EXR2000 Disk Array Controller  OK
0xF9800000-0xF9FFFFFF  DEC 21154 PCI to PCI bridge   OK
0xF9800000-0xF9FFFFFF  Mylex EXR2000 Disk Array Controller  OK
0xFA800000-0xFAFFFFFF  DEC 21154 PCI to PCI bridge   OK
0xFA800000-0xFAFFFFFF  Mylex EXR2000 Disk Array Controller  OK
0xFB000000-0xFC7FFFFF  PCI bus OK
0xFB000000-0xFC7FFFFF  Alteon WebSystems PCI Gigabit Ethernet Adapter
       OK
0xFC800000-0xFD7FFFFF  PCI bus OK
0xFC800000-0xFD7FFFFF  DEC 21154 PCI to PCI bridge   OK
0xFC800000-0xFD7FFFFF  Mylex EXR2000 Disk Array Controller  OK
0xFB800000-0xFBFFFFFF  DEC 21154 PCI to PCI bridge   OK
0xFB800000-0xFBFFFFFF  Mylex EXR2000 Disk Array Controller  OK
0xFC000000-0xFC7FFFFF  DEC 21154 PCI to PCI bridge   OK
0xFC000000-0xFC7FFFFF  Mylex EXR2000 Disk Array Controller  OK
0xFD000000-0xFD7FFFFF  DEC 21154 PCI to PCI bridge   OK
0xFD000000-0xFD7FFFFF  Mylex EXR2000 Disk Array Controller  OK

[Network]

[ Following are sub-categories of this main category ]

[Adapter]


Name    [00000007] Alteon WebSystems PCI Gigabit Ethernet Adapter
Adapter Type   Ethernet 802.3
Product Name   Alteon WebSystems PCI Gigabit Ethernet Adapter
Installed      True
PNP Device ID
       PCI\VEN_12AE&DEV_0001&SUBSYS_00000000&REV_01\3&12F48E42&1&58
Last Reset     8/29/2000 11:22:49
Index  7
Service Name   altnd5
IP Address     129.103.181.144
IP Subnet      255.255.255.0
Default IP Gateway     Not Available
DHCP Enabled   False
DHCP Server    Not Available
DHCP Lease Expires     Not Available
DHCP Lease Obtained    Not Available
MAC Address    00:60:CF:20:07:0D
Service Name   altnd5
IRQ Number     31
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Driver c:\winnt\system32\drivers\altnd5.sys (597776, 1.17.13)


[Storage]

[ Following are sub-categories of this main category ]

[Drives]

Item    Value
Drive   A:
Description     3 1/2 Inch Floppy Drive

Drive   C:
Description     Local Fixed Disk
Compressed      False
File System     NTFS
Size    8.50 GB (9,121,800,192 bytes)
Free Space      2.37 GB (2,546,360,320 bytes)
Volume Name
Volume Serial Number    40AF51BF
Partition       Disk #0, Partition #0
Partition Size 8.50 GB (9,121,803,264 bytes)
Starting Offset         32256 bytes
Drive Description       Disk drive
Drive Manufacturer      (Standard disk drives)
Drive Model     FUJITSU MAG3091LC SCSI Disk Device
Drive BytesPerSector    512
Drive MediaLoaded       True
Drive MediaType         Fixed hard disk media
Drive Partitions        1
Drive SCSIBus   0
Drive SCSILogicalUnit   0
Drive SCSIPort 2
Drive SCSITargetId      0
Drive SectorsPerTrack   63
Drive Size      9121835520 bytes
Drive TotalCylinders    1109
Drive TotalSectors      17816085
Drive TotalTracks       282795
Drive TracksPerCylinder         255

Drive   E:
Description     Local Fixed Disk
Compressed      Not Available
File System     Not Available
Size    Not Available
Free Space      Not Available
Volume Name     Not Available
Volume Serial Number    Not Available

Drive   F:
Description     Local Fixed Disk
Compressed      Not Available
File System     Not Available
Size    Not Available
Free Space      Not Available
Volume Name     Not Available
Volume Serial Number    Not Available

Drive   G:
Description     Local Fixed Disk
Compressed      Not Available
File System     Not Available
Size    Not Available
Free Space      Not Available
Volume Name     Not Available
Volume Serial Number    Not Available

Drive   H:
Description     Local Fixed Disk
Compressed      Not Available
File System     Not Available
Size    Not Available
Free Space      Not Available
Volume Name     Not Available
Volume Serial Number    Not Available

Drive   I:
Description     Local Fixed Disk
Compressed      Not Available
File System     Not Available
Size    Not Available
Free Space      Not Available
Volume Name     Not Available
Volume Serial Number    Not Available

Drive   L:
Description     Local Fixed Disk
Compressed      Not Available
File System     Not Available
Size    Not Available
Free Space      Not Available
Volume Name     Not Available
Volume Serial Number    Not Available

Drive   N:
Description     Local Fixed Disk
Compressed      Not Available
File System     Not Available
Size    Not Available
Free Space      Not Available
Volume Name     Not Available
Volume Serial Number    Not Available

Drive   O:
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Description    Local Fixed Disk
Compressed     Not Available
File System    Not Available
Size    Not Available
Free Space     Not Available
Volume Name    Not Available
Volume Serial Number   Not Available

Drive   P:
Description    Local Fixed Disk
Compressed     Not Available
File System    Not Available
Size    Not Available
Free Space     Not Available
Volume Name    Not Available
Volume Serial Number   Not Available

Drive   Q:
Description    Local Fixed Disk
Compressed     Not Available
File System    Not Available
Size    Not Available
Free Space     Not Available
Volume Name    Not Available
Volume Serial Number   Not Available

Drive   R:
Description    Local Fixed Disk
Compressed     Not Available
File System    Not Available
Size    Not Available
Free Space     Not Available
Volume Name    Not Available
Volume Serial Number   Not Available

Drive   X:
Description    Local Fixed Disk
Compressed     False
File System    NTFS
Size    195.31 GB (209,711,706,112 bytes)
Free Space     83.06 GB (89,182,175,232 bytes)
Volume Name    Volume X
Volume Serial Number   1406CB7E
Partition      Disk #6, Partition #0
Partition Size 804.36 GB (863,679,075,840 bytes)
Starting Offset        8225280 bytes
Drive Description      \\.\PHYSICALDRIVE6
Drive Manufacturer     Not Available
Drive Model    Not Available
Drive BytesPerSector   512
Drive MediaLoaded      True
Drive MediaType        Fixed hard disk media
Drive Partitions       3
Drive SCSIBus  4
Drive SCSILogicalUnit  0
Drive SCSIPort 9
Drive SCSITargetId     0
Drive SectorsPerTrack  63
Drive Size     863687301120 bytes
Drive TotalCylinders   105004
Drive TotalSectors     1686889260
Drive TotalTracks      26776020
Drive TracksPerCylinder        255

Drive   Y:
Description    Local Fixed Disk
Compressed     False
File System    NTFS
Size    195.31 GB (209,711,706,112 bytes)
Free Space     95.24 GB (102,265,864,192 bytes)
Volume Name    Volume Y
Volume Serial Number   80253162
Partition      Disk #2, Partition #0
Partition Size 804.36 GB (863,679,075,840 bytes)
Starting Offset        8225280 bytes
Drive Description      \\.\PHYSICALDRIVE2
Drive Manufacturer     Not Available
Drive Model    Not Available
Drive BytesPerSector   512
Drive MediaLoaded      True
Drive MediaType        Fixed hard disk media
Drive Partitions       3
Drive SCSIBus  4
Drive SCSILogicalUnit  0
Drive SCSIPort 5
Drive SCSITargetId     0
Drive SectorsPerTrack  63
Drive Size     863687301120 bytes
Drive TotalCylinders   105004
Drive TotalSectors     1686889260
Drive TotalTracks      26776020
Drive TracksPerCylinder        255


[SCSI]

Item    Value
Name    Mylex EXR2000 Disk Array Controller
Caption Mylex EXR2000 Disk Array Controller
Driver  dac2w2k
Status  OK
PNP Device ID
        PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&3EC16A1&0&4050
Device ID
        PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&3EC16A1&0&4050
Device Map     Not Available
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Index  Not Available
Max Number Controlled  Not Available
IRQ Number     26
I/O Port       0x2000-0x2FFF
Driver  c:\winnt\system32\drivers\dac2w2k.sys (185488, 6.00-03)

Name   Mylex EXR2000 Disk Array Controller
Caption Mylex EXR2000 Disk Array Controller
Driver  dac2w2k
Status  OK
PNP Device ID
       PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&BA2977F&0&4060
Device ID
       PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&BA2977F&0&4060
Device Map     Not Available
Index  Not Available
Max Number Controlled  Not Available
IRQ Number     24
I/O Port       0x3000-0x3FFF
Driver  c:\winnt\system32\drivers\dac2w2k.sys (185488, 6.00-03)

Name   Adaptec AIC-7899 Ultra160/m PCI SCSI Card
Caption Adaptec AIC-7899 Ultra160/m PCI SCSI Card
Driver  adpu160m
Status  OK
PNP Device ID  PCI\VEN_9005&DEV_00CF&SUBSYS_6618110A&REV_01\3&3ADD9D&0&30
Device ID      PCI\VEN_9005&DEV_00CF&SUBSYS_6618110A&REV_01\3&3ADD9D&0&30
Device Map     Not Available
Index  Not Available
Max Number Controlled  Not Available
IRQ Number     18
I/O Port       0x4000-0x6FFF
Driver  c:\winnt\system32\drivers\adpu160m.sys (64432, v3.10a)

Name   Adaptec AIC-7899 Ultra160/m PCI SCSI Card
Caption Adaptec AIC-7899 Ultra160/m PCI SCSI Card
Driver  adpu160m
Status  OK
PNP Device ID  PCI\VEN_9005&DEV_00CF&SUBSYS_6618110A&REV_01\3&3ADD9D&0&31
Device ID      PCI\VEN_9005&DEV_00CF&SUBSYS_6618110A&REV_01\3&3ADD9D&0&31
Device Map     Not Available
Index  Not Available
Max Number Controlled  Not Available
IRQ Number     19
I/O Port       0x4400-0x44FF
Driver  c:\winnt\system32\drivers\adpu160m.sys (64432, v3.10a)

Name   Mylex EXR2000 Disk Array Controller
Caption Mylex EXR2000 Disk Array Controller
Driver  dac2w2k
Status  OK
PNP Device ID
       PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&3B7BA8BE&0&4040
Device ID
       PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&3B7BA8BE&0&4040
Device Map     Not Available
Index  Not Available
Max Number Controlled  Not Available
IRQ Number     20
I/O Port       0x5000-0x5FFF
Driver  c:\winnt\system32\drivers\dac2w2k.sys (185488, 6.00-03)

Name   Mylex EXR2000 Disk Array Controller
Caption Mylex EXR2000 Disk Array Controller
Driver  dac2w2k
Status  OK
PNP Device ID
       PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&FADCF69&0&4050
Device ID
       PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&FADCF69&0&4050
Device Map     Not Available
Index  Not Available
Max Number Controlled  Not Available
IRQ Number     22
I/O Port       0x6000-0x6FFF
Driver  c:\winnt\system32\drivers\dac2w2k.sys (185488, 6.00-03)

Name   Mylex EXR2000 Disk Array Controller
Caption Mylex EXR2000 Disk Array Controller
Driver  dac2w2k
Status  OK
PNP Device ID
       PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&1EA4B82&0&4040
Device ID
       PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&1EA4B82&0&4040
Device Map     Not Available
Index  Not Available
Max Number Controlled  Not Available
IRQ Number     28
I/O Port       0x7000-0x8FFF
Driver  c:\winnt\system32\drivers\dac2w2k.sys (185488, 6.00-03)

Name   Mylex EXR2000 Disk Array Controller
Caption Mylex EXR2000 Disk Array Controller
Driver  dac2w2k
Status  OK
PNP Device ID
       PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&2CAD654F&0&4048
Device ID
       PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&2CAD654F&0&4048
Device Map     Not Available
Index  Not Available
Max Number Controlled  Not Available
IRQ Number     29
I/O Port       0x8000-0x8FFF
Driver  c:\winnt\system32\drivers\dac2w2k.sys (185488, 6.00-03)
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
========== disk configuration controller 0 .. 5 ==============

Begin
 BeginGroup
        PhysicalDevice0 = Channel=0, Target=0, Size=34712mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice1 = Channel=1, Target=0, Size=34712mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice2 = Channel=0, Target=1, Size=34712mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice3 = Channel=1, Target=1, Size=34712mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice4 = Channel=0, Target=2, Size=34712mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice5 = Channel=1, Target=2, Size=34712mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        IntermediateDevice0 = StripeSize=64kb, Raid=1, WriteThrough=1,
Size=34712mb,
                (PhysicalDevice0, StartAddress=0mb, Size=34712mb),
                (PhysicalDevice1, StartAddress=0mb, Size=34712mb);
        IntermediateDevice1 = StripeSize=64kb, Raid=1, WriteThrough=1,
Size=34712mb,
                (PhysicalDevice2, StartAddress=0mb, Size=34712mb),
                (PhysicalDevice3, StartAddress=0mb, Size=34712mb);
        IntermediateDevice2 = StripeSize=64kb, Raid=1, WriteThrough=1,
Size=34712mb,
                (PhysicalDevice4, StartAddress=0mb, Size=34712mb),
                (PhysicalDevice5, StartAddress=0mb, Size=34712mb);
        LogicalDevice0 = StripeSize=64kb, Raid=12, WriteThrough=1,
Size=104136mb, BIOSGeometry=2GB,
                (IntermediateDevice0, StartAddress=0mb, Size=69424mb),
                (IntermediateDevice1, StartAddress=0mb, Size=69424mb),
                (IntermediateDevice2, StartAddress=0mb, Size=69424mb),
 EndGroup
 BeginControllerParameter
        ControllerName = eXtremeRAID 2000;
        ControllerType = 28;
        FirmwareVersion = 5.60;
        CacheLineSize = 8KB;
        BackgroundTaskRate = 50;
        InitiatorID = 7;
        DiskStartupMode = AutoSpin;
        DevicesPerSpin = 2;
        InitialDelay = 6S;
        SequentialDelay = 0S;
        EnableDriveSizing = 1;
        EnableClustering = 0;
        EnableBGInit = 1;
        EnableReadAhead = 0;
        EnableBiosLoadDelay = 0;
        EnableForcedUnitAccess = 0;
        DisableBios = 1;
        EnableCDROMBoot = 0;
        EnableStorageWorks = 0;
        EnableSAFTE = 1;
        EnableSES = 1;
        EnableARM = 0;
        EnableOFM = 0;
        OEMCode = 0;
        StartupOption = 0;
 EndControllerParameter
End
egin
 BeginGroup
        PhysicalDevice0 = Channel=0, Target=0, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice1 = Channel=0, Target=1, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice2 = Channel=0, Target=2, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice3 = Channel=0, Target=3, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice4 = Channel=0, Target=4, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice5 = Channel=0, Target=5, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice6 = Channel=0, Target=10, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice7 = Channel=0, Target=11, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice8 = Channel=0, Target=12, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice9 = Channel=0, Target=13, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice10 = Channel=0, Target=14, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice11 = Channel=0, Target=15, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice12 = Channel=1, Target=0, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice13 = Channel=1, Target=1, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice14 = Channel=1, Target=2, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice15 = Channel=1, Target=3, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice16 = Channel=1, Target=4, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice17 = Channel=1, Target=5, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice18 = Channel=1, Target=10, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        PhysicalDevice19 = Channel=1, Target=11, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice20 = Channel=1, Target=12, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice21 = Channel=1, Target=13, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice22 = Channel=1, Target=14, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice23 = Channel=1, Target=15, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice24 = Channel=2, Target=0, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice25 = Channel=2, Target=1, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice26 = Channel=2, Target=2, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice27 = Channel=2, Target=3, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice28 = Channel=2, Target=4, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice29 = Channel=2, Target=5, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice30 = Channel=2, Target=10, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice31 = Channel=2, Target=11, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice32 = Channel=2, Target=12, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice33 = Channel=2, Target=13, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice34 = Channel=2, Target=14, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice35 = Channel=2, Target=15, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice36 = Channel=3, Target=0, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice37 = Channel=3, Target=1, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice38 = Channel=3, Target=2, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice39 = Channel=3, Target=3, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice40 = Channel=3, Target=4, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice41 = Channel=3, Target=5, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice42 = Channel=3, Target=10, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice43 = Channel=3, Target=11, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice44 = Channel=3, Target=12, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice45 = Channel=3, Target=13, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice46 = Channel=3, Target=14, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice47 = Channel=3, Target=15, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        IntermediateDevice0 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=100704mb,
                (PhysicalDevice0, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice1, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice2, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice3, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice4, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice5, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice6, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice7, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice8, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice9, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice10, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice11, StartAddress=0mb, Size=8392mb);
        IntermediateDevice1 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=100704mb,
                (PhysicalDevice12, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice13, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice14, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice15, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice16, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice17, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice18, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice19, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice20, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice21, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice22, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice23, StartAddress=0mb, Size=8392mb);
        IntermediateDevice2 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=100704mb,
                (PhysicalDevice24, StartAddress=0mb, Size=8392mb),
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
               (PhysicalDevice25, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice26, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice27, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice28, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice29, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice30, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice31, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice32, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice33, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice34, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice35, StartAddress=0mb, Size=8392mb);
       IntermediateDevice3 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=100704mb,
               (PhysicalDevice36, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice37, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice38, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice39, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice40, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice41, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice42, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice43, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice44, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice45, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice46, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice47, StartAddress=0mb, Size=8392mb);
       LogicalDevice0 = StripeSize=64kb, Raid=12, WriteThrough=1,
Size=402816mb, BIOSGeometry=2GB,
               (IntermediateDevice0, StartAddress=0mb, Size=100704mb),
               (IntermediateDevice1, StartAddress=0mb, Size=100704mb),
               (IntermediateDevice2, StartAddress=0mb, Size=100704mb),
               (IntermediateDevice3, StartAddress=0mb, Size=100704mb);
 EndGroup
 BeginControllerParameter
       ControllerName = eXtremeRAID 2000;
       ControllerType = 28;
       FirmwareVersion = 5.60;
       CacheLineSize = 8KB;
       BackgroundTaskRate = 50;
       InitiatorID = 7;
       DiskStartupMode = AutoSpin;
       DevicesPerSpin = 2;
       InitialDelay = 6S;
       SequentialDelay = 0S;
       EnableDriveSizing = 1;
       EnableClustering = 0;
       EnableBGInit = 1;
       EnableReadAhead = 0;
       EnableBiosLoadDelay = 0;
       EnableForcedUnitAccess = 0;
       DisableBios = 1;
       EnableCDROMBoot = 0;
       EnableStorageWorks = 0;
       EnableSAFTE = 1;
       EnableSES = 1;
       EnableARM = 0;
       EnableOFM = 0;
       OEMCode = 0;
       StartupOption = 0;
 EndControllerParameter
End
Begin
 BeginGroup
       PhysicalDevice0 = Channel=0, Target=0, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice1 = Channel=0, Target=1, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice2 = Channel=0, Target=2, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice3 = Channel=0, Target=3, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice4 = Channel=0, Target=4, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice5 = Channel=0, Target=5, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice6 = Channel=0, Target=10, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice7 = Channel=0, Target=11, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice8 = Channel=0, Target=12, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice9 = Channel=0, Target=13, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice10 = Channel=0, Target=14, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice11 = Channel=0, Target=15, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice12 = Channel=1, Target=0, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice13 = Channel=1, Target=1, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice14 = Channel=1, Target=2, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice15 = Channel=1, Target=3, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice16 = Channel=1, Target=4, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice17 = Channel=1, Target=5, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice18 = Channel=1, Target=10, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice19 = Channel=1, Target=11, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
       PhysicalDevice20 = Channel=1, Target=12, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice21 = Channel=1, Target=13, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice22 = Channel=1, Target=14, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice23 = Channel=1, Target=15, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice24 = Channel=2, Target=0, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice25 = Channel=2, Target=1, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice26 = Channel=2, Target=2, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice27 = Channel=2, Target=3, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice28 = Channel=2, Target=4, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice29 = Channel=2, Target=5, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice30 = Channel=2, Target=10, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice31 = Channel=2, Target=11, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice32 = Channel=2, Target=12, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice33 = Channel=2, Target=13, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice34 = Channel=2, Target=14, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice35 = Channel=2, Target=15, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice36 = Channel=3, Target=0, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice37 = Channel=3, Target=1, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice38 = Channel=3, Target=2, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice39 = Channel=3, Target=3, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice40 = Channel=3, Target=4, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice41 = Channel=3, Target=5, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice42 = Channel=3, Target=10, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice43 = Channel=3, Target=11, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice44 = Channel=3, Target=12, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice45 = Channel=3, Target=13, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice46 = Channel=3, Target=14, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice47 = Channel=3, Target=15, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       IntermediateDevice0 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=100704mb,
              (PhysicalDevice0, StartAddress=0mb, Size=8392mb),
       (PhysicalDevice1, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice2, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice3, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice4, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice5, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice6, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice7, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice8, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice9, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice10, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice11, StartAddress=0mb, Size=8392mb);
       IntermediateDevice1 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=100704mb,
              (PhysicalDevice12, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice13, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice14, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice15, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice16, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice17, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice18, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice19, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice20, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice21, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice22, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice23, StartAddress=0mb, Size=8392mb);
       IntermediateDevice2 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=100704mb,
              (PhysicalDevice24, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice25, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice26, StartAddress=0mb, Size=8392mb),
              (PhysicalDevice27, StartAddress=0mb, Size=8392mb),
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
               (PhysicalDevice28, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice29, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice30, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice31, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice32, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice33, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice34, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice35, StartAddress=0mb, Size=8392mb);
       IntermediateDevice3 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=100704mb,
               (PhysicalDevice36, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice37, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice38, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice39, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice40, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice41, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice42, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice43, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice44, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice45, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice46, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice47, StartAddress=0mb, Size=8392mb);
       LogicalDevice0 = StripeSize=64kb, Raid=12, WriteThrough=1,
Size=402816mb, BIOSGeometry=2GB,
               (IntermediateDevice0, StartAddress=0mb, Size=100704mb),
               (IntermediateDevice1, StartAddress=0mb, Size=100704mb),
               (IntermediateDevice2, StartAddress=0mb, Size=100704mb),
               (IntermediateDevice3, StartAddress=0mb, Size=100704mb);
 EndGroup
 BeginControllerParameter
       ControllerName = eXtremeRAID 2000;
       ControllerType = 28;
       FirmwareVersion = 5.60;
       CacheLineSize = 8KB;
       BackgroundTaskRate = 50;
       InitiatorID = 7;
       DiskStartupMode = AutoSpin;
       DevicesPerSpin = 2;
       InitialDelay = 6S;
       SequentialDelay = 0S;
       EnableDriveSizing = 1;
       EnableClustering = 0;
       EnableBGInit = 1;
       EnableReadAhead = 0;
       EnableBiosLoadDelay = 0;
       EnableForcedUnitAccess = 0;
       DisableBios = 1;
       EnableCDROMBoot = 0;
       EnableStorageWorks = 0;
       EnableSAFTE = 1;
       EnableSES = 1;
       EnableARM = 0;
       EnableOFM = 0;
       OEMCode = 0;
       StartupOption = 0;
 EndControllerParameter
End
Begin
 BeginGroup
       PhysicalDevice0 = Channel=0, Target=0, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice1 = Channel=0, Target=1, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice2 = Channel=0, Target=2, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice3 = Channel=0, Target=3, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice4 = Channel=0, Target=4, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice5 = Channel=0, Target=5, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice6 = Channel=0, Target=10, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice7 = Channel=0, Target=11, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice8 = Channel=0, Target=12, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice9 = Channel=0, Target=13, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice10 = Channel=0, Target=14, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice11 = Channel=0, Target=15, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice12 = Channel=1, Target=0, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice13 = Channel=1, Target=1, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice14 = Channel=1, Target=2, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice15 = Channel=1, Target=3, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice16 = Channel=1, Target=4, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice17 = Channel=1, Target=5, Size=8392mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice18 = Channel=1, Target=10, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice19 = Channel=1, Target=11, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice20 = Channel=1, Target=12, Size=8392mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        PhysicalDevice21 = Channel=1, Target=13, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice22 = Channel=1, Target=14, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice23 = Channel=1, Target=15, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice24 = Channel=2, Target=0, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice25 = Channel=2, Target=1, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice26 = Channel=2, Target=2, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice27 = Channel=2, Target=3, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice28 = Channel=2, Target=4, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice29 = Channel=2, Target=5, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice30 = Channel=2, Target=10, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice31 = Channel=2, Target=11, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice32 = Channel=2, Target=12, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice33 = Channel=2, Target=13, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice34 = Channel=2, Target=14, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice35 = Channel=2, Target=15, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice36 = Channel=3, Target=0, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice37 = Channel=3, Target=1, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice38 = Channel=3, Target=2, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice39 = Channel=3, Target=3, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice40 = Channel=3, Target=4, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice41 = Channel=3, Target=5, Size=8392mb, State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice42 = Channel=3, Target=10, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice43 = Channel=3, Target=11, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice44 = Channel=3, Target=12, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice45 = Channel=3, Target=13, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice46 = Channel=3, Target=14, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice47 = Channel=3, Target=15, Size=8392mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        IntermediateDevice0 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=100704mb,
                (PhysicalDevice0, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice1, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice2, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice3, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice4, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice5, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice6, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice7, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice8, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice9, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice10, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice11, StartAddress=0mb, Size=8392mb);
        IntermediateDevice1 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=100704mb,
                (PhysicalDevice12, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice13, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice14, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice15, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice16, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice17, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice18, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice19, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice20, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice21, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice22, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice23, StartAddress=0mb, Size=8392mb);
        IntermediateDevice2 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=100704mb,
                (PhysicalDevice24, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice25, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice26, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice27, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice28, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice29, StartAddress=0mb, Size=8392mb),
                (PhysicalDevice30, StartAddress=0mb, Size=8392mb),
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
               (PhysicalDevice31, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice32, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice33, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice34, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice35, StartAddress=0mb, Size=8392mb);
       IntermediateDevice3 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=100704mb,
               (PhysicalDevice36, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice37, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice38, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice39, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice40, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice41, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice42, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice43, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice44, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice45, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice46, StartAddress=0mb, Size=8392mb),
               (PhysicalDevice47, StartAddress=0mb, Size=8392mb);
       LogicalDevice0 = StripeSize=64kb, Raid=12, WriteThrough=1,
Size=402816mb, BIOSGeometry=2GB,
               (IntermediateDevice0, StartAddress=0mb, Size=100704mb),
               (IntermediateDevice1, StartAddress=0mb, Size=100704mb),
               (IntermediateDevice2, StartAddress=0mb, Size=100704mb),
               (IntermediateDevice3, StartAddress=0mb, Size=100704mb);
 EndGroup
 BeginControllerParameter
       ControllerName = eXtremeRAID 2000;
       ControllerType = 28;
       FirmwareVersion = 5.60;
       CacheLineSize = 8KB;
       BackgroundTaskRate = 50;
       InitiatorID = 7;
       DiskStartupMode = AutoSpin;
       DevicesPerSpin = 2;
       InitialDelay = 6S;
       SequentialDelay = 0S;
       EnableDriveSizing = 1;
       EnableClustering = 0;
       EnableBGInit = 1;
       EnableReadAhead = 0;
       EnableBiosLoadDelay = 0;
       EnableForcedUnitAccess = 0;
       DisableBios = 1;
       EnableCDROMBoot = 0;
       EnableStorageWorks = 0;
       EnableSAFTE = 1;
       EnableSES = 1;
       EnableARM = 0;
       EnableOFM = 0;
       OEMCode = 0;
       StartupOption = 0;
 EndControllerParameter
End
Begin
 BeginGroup
       PhysicalDevice0 = Channel=0, Target=0, Size=17160mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice1 = Channel=0, Target=1, Size=17160mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice2 = Channel=0, Target=2, Size=17160mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice3 = Channel=0, Target=3, Size=17160mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice4 = Channel=0, Target=4, Size=17160mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice5 = Channel=0, Target=5, Size=17160mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice6 = Channel=0, Target=10, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice7 = Channel=0, Target=11, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice8 = Channel=0, Target=12, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice9 = Channel=0, Target=13, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice10 = Channel=0, Target=14, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice11 = Channel=0, Target=15, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice12 = Channel=1, Target=0, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice13 = Channel=1, Target=1, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice14 = Channel=1, Target=2, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice15 = Channel=1, Target=3, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice16 = Channel=1, Target=4, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice17 = Channel=1, Target=5, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice18 = Channel=1, Target=10, Size=17160mb,
State=Online,
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice19 = Channel=1, Target=11, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice20 = Channel=1, Target=12, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice21 = Channel=1, Target=13, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice22 = Channel=1, Target=14, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice23 = Channel=1, Target=15, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice24 = Channel=2, Target=0, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice25 = Channel=2, Target=1, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice26 = Channel=2, Target=2, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice27 = Channel=2, Target=3, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice28 = Channel=2, Target=4, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice29 = Channel=2, Target=5, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice30 = Channel=2, Target=10, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice31 = Channel=2, Target=11, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice32 = Channel=2, Target=12, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice33 = Channel=2, Target=13, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice34 = Channel=2, Target=14, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice35 = Channel=2, Target=15, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice36 = Channel=3, Target=0, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice37 = Channel=3, Target=1, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice38 = Channel=3, Target=2, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice39 = Channel=3, Target=3, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice40 = Channel=3, Target=4, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice41 = Channel=3, Target=5, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice42 = Channel=3, Target=10, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice43 = Channel=3, Target=11, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice44 = Channel=3, Target=12, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice45 = Channel=3, Target=13, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice46 = Channel=3, Target=14, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice47 = Channel=3, Target=15, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        IntermediateDevice0 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=205920mb,
                (PhysicalDevice0, StartAddress=0mb, Size=17160mb),
                (PhysicalDevice1, StartAddress=0mb, Size=17160mb),
                (PhysicalDevice2, StartAddress=0mb, Size=17160mb),
                (PhysicalDevice3, StartAddress=0mb, Size=17160mb),
                (PhysicalDevice4, StartAddress=0mb, Size=17160mb),
                (PhysicalDevice5, StartAddress=0mb, Size=17160mb),
                (PhysicalDevice6, StartAddress=0mb, Size=17160mb),
                (PhysicalDevice7, StartAddress=0mb, Size=17160mb),
                (PhysicalDevice8, StartAddress=0mb, Size=17160mb),
                (PhysicalDevice9, StartAddress=0mb, Size=17160mb),
                (PhysicalDevice10, StartAddress=0mb, Size=17160mb),
                (PhysicalDevice11, StartAddress=0mb, Size=17160mb);
        IntermediateDevice1 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=205920mb,
                (PhysicalDevice12, StartAddress=0mb, Size=17160mb),
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
               (PhysicalDevice13, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice14, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice15, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice16, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice17, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice18, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice19, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice20, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice21, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice22, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice23, StartAddress=0mb, Size=17160mb);
       IntermediateDevice2 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=205920mb,
               (PhysicalDevice24, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice25, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice26, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice27, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice28, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice29, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice30, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice31, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice32, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice33, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice34, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice35, StartAddress=0mb, Size=17160mb);
       IntermediateDevice3 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=205920mb,
               (PhysicalDevice36, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice37, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice38, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice39, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice40, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice41, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice42, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice43, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice44, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice45, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice46, StartAddress=0mb, Size=17160mb),
               (PhysicalDevice47, StartAddress=0mb, Size=17160mb);
       LogicalDevice0 = StripeSize=64kb, Raid=12, WriteThrough=1,
Size=823680mb, BIOSGeometry=2GB,
               (IntermediateDevice0, StartAddress=0mb, Size=205920mb),
               (IntermediateDevice1, StartAddress=0mb, Size=205920mb),
               (IntermediateDevice2, StartAddress=0mb, Size=205920mb),
               (IntermediateDevice3, StartAddress=0mb, Size=205920mb);
 EndGroup
 BeginControllerParameter
       ControllerName = eXtremeRAID 2000;
       ControllerType = 28;
       FirmwareVersion = 5.60;
       CacheLineSize = 8KB;
       BackgroundTaskRate = 50;
       InitiatorID = 7;
       DiskStartupMode = AutoSpin;
       DevicesPerSpin = 2;
       InitialDelay = 6S;
       SequentialDelay = 0S;
       EnableDriveSizing = 1;
       EnableClustering = 0;
       EnableBGInit = 1;
       EnableReadAhead = 0;
       EnableBiosLoadDelay = 0;
       EnableForcedUnitAccess = 0;
       DisableBios = 1;
       EnableCDROMBoot = 0;
       EnableStorageWorks = 0;
       EnableSAFTE = 1;
       EnableSES = 1;
       EnableARM = 0;
       EnableOFM = 0;
       OEMCode = 0;
       StartupOption = 0;
 EndControllerParameter
End
Begin
 BeginGroup
       PhysicalDevice0 = Channel=0, Target=0, Size=17160mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice1 = Channel=0, Target=1, Size=17160mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice2 = Channel=0, Target=2, Size=17160mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice3 = Channel=0, Target=3, Size=17160mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice4 = Channel=0, Target=4, Size=17160mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice5 = Channel=0, Target=5, Size=17160mb, State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice6 = Channel=0, Target=10, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice7 = Channel=0, Target=11, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice8 = Channel=0, Target=12, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice9 = Channel=0, Target=13, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice10 = Channel=0, Target=14, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       PhysicalDevice11 = Channel=0, Target=15, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
        PhysicalDevice12 = Channel=1, Target=0, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice13 = Channel=1, Target=1, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice14 = Channel=1, Target=2, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice15 = Channel=1, Target=3, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice16 = Channel=1, Target=4, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice17 = Channel=1, Target=5, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice18 = Channel=1, Target=10, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice19 = Channel=1, Target=11, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice20 = Channel=1, Target=12, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice21 = Channel=1, Target=13, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice22 = Channel=1, Target=14, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice23 = Channel=1, Target=15, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice24 = Channel=2, Target=0, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice25 = Channel=2, Target=1, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice26 = Channel=2, Target=2, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice27 = Channel=2, Target=3, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice28 = Channel=2, Target=4, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice29 = Channel=2, Target=5, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice30 = Channel=2, Target=10, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice31 = Channel=2, Target=11, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice32 = Channel=2, Target=12, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice33 = Channel=2, Target=13, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice34 = Channel=2, Target=14, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice35 = Channel=2, Target=15, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice36 = Channel=3, Target=0, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice37 = Channel=3, Target=1, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice38 = Channel=3, Target=2, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice39 = Channel=3, Target=3, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice40 = Channel=3, Target=4, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice41 = Channel=3, Target=5, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice42 = Channel=3, Target=10, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice43 = Channel=3, Target=11, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice44 = Channel=3, Target=12, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice45 = Channel=3, Target=13, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
        PhysicalDevice46 = Channel=3, Target=14, Size=17160mb,
State=Online,
                TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
       PhysicalDevice47 = Channel=3, Target=15, Size=17160mb,
State=Online,
              TransferSpeed=40MHz, TransferWidth=16Bit, MaxTag=16;
       IntermediateDevice0 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=205920mb,
              (PhysicalDevice0, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice1, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice2, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice3, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice4, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice5, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice6, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice7, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice8, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice9, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice10, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice11, StartAddress=0mb, Size=17160mb);
       IntermediateDevice1 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=205920mb,
              (PhysicalDevice12, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice13, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice14, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice15, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice16, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice17, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice18, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice19, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice20, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice21, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice22, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice23, StartAddress=0mb, Size=17160mb);
       IntermediateDevice2 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=205920mb,
              (PhysicalDevice24, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice25, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice26, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice27, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice28, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice29, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice30, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice31, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice32, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice33, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice34, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice35, StartAddress=0mb, Size=17160mb);
       IntermediateDevice3 = StripeSize=64kb, Raid=0, WriteThrough=1,
Size=205920mb,
              (PhysicalDevice36, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice37, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice38, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice39, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice40, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice41, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice42, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice43, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice44, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice45, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice46, StartAddress=0mb, Size=17160mb),
              (PhysicalDevice47, StartAddress=0mb, Size=17160mb);
       LogicalDevice0 = StripeSize=64kb, Raid=12, WriteThrough=1,
Size=823680mb, BIOSGeometry=2GB,
              (IntermediateDevice0, StartAddress=0mb, Size=205920mb),
              (IntermediateDevice1, StartAddress=0mb, Size=205920mb),
              (IntermediateDevice2, StartAddress=0mb, Size=205920mb),
              (IntermediateDevice3, StartAddress=0mb, Size=205920mb);
 EndGroup
 BeginControllerParameter
       ControllerName = eXtremeRAID 2000;
       ControllerType = 28;
       FirmwareVersion = 5.60;
       CacheLineSize = 8KB;
       BackgroundTaskRate = 50;
       InitiatorID = 7;
       DiskStartupMode = AutoSpin;
       DevicesPerSpin = 2;
       InitialDelay = 6S;
       SequentialDelay = 0S;
       EnableDriveSizing = 1;
       EnableClustering = 0;
       EnableBGInit = 1;
       EnableReadAhead = 0;
       EnableBiosLoadDelay = 0;
       EnableForcedUnitAccess = 0;
       DisableBios = 1;
       EnableCDROMBoot = 0;
       EnableStorageWorks = 0;
       EnableSAFTE = 1;
       EnableSES = 1;
       EnableARM = 0;
       EnableOFM = 0;
       OEMCode = 0;
       StartupOption = 0;
 EndControllerParameter
End

Name    Description    Status         Startup Type   Log On As
Alerter Notifies selected users and computers of administrative alerts.
       Started Automatic      LocalSystem
Application Management Provides software installation services such as
Assign, Publish, and Remove.  Started Manual  LocalSystem
ClipBook       Supports ClipBook Viewer, which allows pages to be seen by
remote ClipBooks.             Manual  LocalSystem
COM+ Event System      Provides automatic distribution of events to
subscribing COM components.           Manual  LocalSystem
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Computer Browser       Maintains an up-to-date list of computers on your
network and supplies the list to programs that request it.
       Disabled       LocalSystem
DHCP Client    Manages network configuration by registering and updating
IP addresses and DNS names.          Disabled       LocalSystem
Distributed File System       Manages logical volumes distributed across
a local or wide area network.        Manual  LocalSystem
Distributed Link Tracking Client      Sends notifications of files moving
between NTFS volumes in a network domain.          Disabled
       LocalSystem
Distributed Link Tracking Server      Stores information so that files
moved between volumes can be tracked for each volume in the domain.
       Manual  LocalSystem
Distributed Transaction Coordinator   Coordinates transactions that are
distributed across two or more databases, message queues, file systems,
or other transaction protected resource managers.          Manual
       LocalSystem
DNS Client     Resolves and caches Domain Name System (DNS) names.
       Disabled       LocalSystem
Event Log      Logs event messages issued by programs and Windows.  Event
Log reports contain information that can be useful in diagnosing
problems.  Reports are viewed in Event Viewer.       Started Automatic
       LocalSystem
Fax Service    Helps you send and receive faxes            Disabled
       LocalSystem
File Replication       Maintains file synchronization of file directory
contents among multiple servers.            Manual  LocalSystem
Indexing Service       Indexes contents and properties of files on local
and remote computers; provides rapid access to files through flexible
querying language.           Manual  LocalSystem
Internet Connection Sharing   Provides network address translation,
addressing, and name resolution services for all computers on your home
network through a dial-up connection.        Manual  LocalSystem
Intersite Messaging    Allows sending and receiving messages between
Windows Advanced Server sites.               Manual  LocalSystem
IPSEC Policy Agent     Manages IP security policy and starts the
ISAKMP/Oakley (IKE) and the IP security driver.            Manual
       LocalSystem
Kerberos Key Distribution Center      Generates session keys and grants
service tickets for mutual client/server authentication.
       Disabled       LocalSystem
License Logging Service                      Manual  LocalSystem
Logical Disk Manager   Logical Disk Manager Watchdog Service        Manual
       LocalSystem
Logical Disk Manager Administrative Service  Administrative service for
disk management requests             Manual  LocalSystem
Messenger      Sends and receives messages transmitted by administrators
or by the Alerter service.     Started Automatic      LocalSystem
Microsoft Search       Creates full-text indexes on content and properties
of structured and semi-structured data to allow fast linguistic searches
on this data           Manual  LocalSystem
MSSQLSERVER                  Manual  LocalSystem
MSSQLServerADHelper                  Manual  LocalSystem
Net Logon      Supports pass-through authentication of account logon
events for computers in a domain.            Manual  LocalSystem
NetMeeting Remote Desktop Sharing     Allows authorized people to remotely
access your Windows desktop using NetMeeting.              Manual
       LocalSystem
Network Connections    Manages objects in the Network and Dial-Up
Connections folder, in which you can view both local area network and
remote connections.    Started Manual  LocalSystem
Network DDE    Provides network transport and security for dynamic data
exchange (DDE).              Manual  LocalSystem
Network DDE DSDM       Manages shared dynamic data exchange and is used by
Network DDE            Manual  LocalSystem
NT LM Security Support Provider       Provides security to remote
procedure call (RPC) programs that use transports other than named pipes.
              Manual  LocalSystem
Performance Logs and Alerts   Configures performance logs and alerts.
              Manual  LocalSystem
Plug and Play  Manages device installation and configuration and notifies
programs of device changes.    Started Automatic      LocalSystem
Print Spooler  Loads files to memory for later printing.           Manual
       LocalSystem
Protected Storage      Provides protected storage for sensitive data, such
as private keys, to prevent access by unauthorized services, processes,
or users.              Manual  LocalSystem
QoS RSVP       Provides network signaling and local traffic control setup
functionality for QoS-aware programs and control applets.    Started Manual
       LocalSystem
Remote Access Auto Connection Manager Creates a connection to a remote
network whenever a program references a remote DNS or NetBIOS name or
address.               Manual  LocalSystem
Remote Access Connection Manager      Creates a network connection.
       Manual  LocalSystem
Remote Command Service               Disabled       LocalSystem
Remote Procedure Call (RPC)    Provides the endpoint mapper and other
miscellaneous RPC services.    Started Automatic      LocalSystem
Remote Procedure Call (RPC) Locator   Manages the RPC name service
database.              Manual  LocalSystem
Remote Registry Service       Allows remote registry manipulation.
       Manual  LocalSystem
Removable Storage      Manages removable media, drives, and libraries.
              Disabled       LocalSystem
Routing and Remote Access     Offers routing services to businesses in
local area and wide area network environments.             Manual
       LocalSystem
RunAs Service  Enables starting processes under alternate credentials
              Manual  LocalSystem
Security Accounts Manager     Stores security information for local user
accounts.              Manual  LocalSystem
Server  Provides RPC support and file, print, and named pipe sharing.
              Manual  LocalSystem
Smart Card     Manages and controls access to a smart card inserted into
a smart card reader attached to the computer.              Manual
       LocalSystem
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Smart Card Helper      Provides support for legacy smart card readers
attached to the computer.              Manual  LocalSystem
SQLSERVERAGENT                 Manual  LocalSystem
System Event Notification      Tracks system events such as Windows logon,
network, and power events.  Notifies COM+ Event System subscribers of
these events.          Automatic       LocalSystem
Task Scheduler Enables a program to run at a designated time.
       Manual  LocalSystem
TCP/IP NetBIOS Helper Service Enables support for NetBIOS over TCP/IP
(NetBT) service and NetBIOS name resolution. Started Automatic
       LocalSystem
Telephony      Provides Telephony API (TAPI) support for programs that
control telephony devices and IP based voice connections on the local
computer and, through the LAN, on servers that are also running the
service.               Disabled        LocalSystem
Telnet  Allows a remote user to log on to the system and run console
programs using the command line.               Manual  LocalSystem
Terminal Services      Provides a multisession environment that allows
client devices to access a virtual Windows 2000 Professional desktop
session and Windows-based programs running on the server.
       Disabled        LocalSystem
Uninterruptible Power Supply  Manages an uninterruptible power supply
(UPS) connected to the computer.               Manual  LocalSystem
Utility Manager        Starts and configures accessibility tools from one
window         Manual  LocalSystem
Windows Installer                      Manual  LocalSystem
Windows Management Instrumentation    Provides system management
information.    Started Automatic      LocalSystem
Windows Management Instrumentation Driver Extensions Provides systems
management information to and from drivers.  Started Manual  LocalSystem
Windows Time   Sets the computer clock.              Manual  LocalSystem
Workstation    Provides network connections and communications.
       Started Automatic       LocalSystem


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS
Class Name:        <NO CLASS>
Last Write Time:   3/15/2000 - 14:17
Value 0
  Name:            DisplayName
  Type:            REG_SZ
  Data:            NDIS System Driver

Value 1
  Name:            ErrorControl
  Type:            REG_DWORD
  Data:            0x1

Value 2
  Name:            Group
  Type:            REG_SZ
  Data:            NDIS Wrapper

Value 3
  Name:            Start
  Type:            REG_DWORD
  Data:            0

Value 4
  Name:            Type
  Type:            REG_DWORD
  Data:            0x1



Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\Parameters
Class Name:        <NO CLASS>
Last Write Time:   8/15/2000 - 14:35
Value 0
  Name:            ProcessorAffinityMask
  Type:            REG_DWORD
  Data:            0


Key Name:          SYSTEM\CurrentControlSet\Control\Session
Manager\Memory Management
Class Name:        <NO CLASS>
Last Write Time:   8/8/2000 - 13:22
Value 0
  Name:            ClearPageFileAtShutdown
  Type:            REG_DWORD
  Data:            0

Value 1
  Name:            DisablePagingExecutive
  Type:            REG_DWORD
  Data:            0

Value 2
  Name:            DontVerifyRandomDrivers
  Type:            REG_DWORD
  Data:            0x1

Value 3
  Name:            IoPageLockLimit
  Type:            REG_DWORD
  Data:            0

Value 4
  Name:            LargeSystemCache
  Type:            REG_DWORD
  Data:            0

Value 5
  Name:            NonPagedPoolQuota
  Type:            REG_DWORD
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
  Data:             0

Value 6
  Name:             NonPagedPoolSize
  Type:             REG_DWORD
  Data:             0

Value 7
  Name:             PagedPoolQuota
  Type:             REG_DWORD
  Data:             0

Value 8
  Name:             PagedPoolSize
  Type:             REG_DWORD
  Data:             0

Value 9
  Name:             PagingFiles
  Type:             REG_MULTI_SZ
  Data:             C:\pagefile.sys 2046 4092


Value 10
  Name:             PhysicalAddressExtension
  Type:             REG_DWORD
  Data:             0x1

Value 11
  Name:             SecondLevelDataCache
  Type:             REG_DWORD
  Data:             0

Value 12
  Name:             SystemPages
  Type:             REG_DWORD
  Data:             0


Key Name:           SYSTEM\CurrentControlSet\Control\Session Manager\I/O
System
Class Name:         <NO CLASS>
Last Write Time:    8/23/2000 - 11:22
Value 0
  Name:             CountOperations
  Type:             REG_DWORD
  Data:             0

Value 1
  Name:             LargeIrpStackLocations
  Type:             REG_DWORD
  Data:             0x9


Key Name:           SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-
11CE-BFC1-08002BE10318}\0007
Class Name:         <NO CLASS>
Last Write Time:    8/8/2000 - 13:37
Value 0
  Name:             BusType
  Type:             REG_SZ
  Data:             5

Value 1
  Name:             Characteristics
  Type:             REG_DWORD
  Data:             0x84

Value 2
  Name:             CksumOffload
  Type:             REG_SZ
  Data:             1

Value 3
  Name:             ComponentId
  Type:             REG_SZ
  Data:             pci\ven_12ae&dev_0001&subsys_00000000

Value 4
  Name:             DebugPci
  Type:             REG_SZ
  Data:             0

Value 5
  Name:             DriverDate
  Type:             REG_SZ
  Data:             10-19-1999

Value 6
  Name:             DriverDateData
  Type:             REG_BINARY
  Data:
00000000   00 c0 db e2 c4 19 bf 01 -                           .ÀÛâÄ.¿.

Value 7
  Name:             DriverDesc
  Type:             REG_SZ
  Data:             Alteon WebSystems PCI Gigabit Ethernet Adapter

Value 8
  Name:             DriverVersion
  Type:             REG_SZ
  Data:             1.16.2.0

Value 9
  Name:             FdrFilter
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
  Type:             REG_SZ
  Data:             0

Value 10
  Name:             Fix450GX
  Type:             REG_SZ
  Data:             0

Value 11
  Name:             HostTracing
  Type:             REG_SZ
  Data:             1

Value 12
  Name:             InfPath
  Type:             REG_SZ
  Data:             netalt.inf

Value 13
  Name:             InfSection
  Type:             REG_SZ
  Data:             acenic.ndi

Value 14
  Name:             InfSectionExt
  Type:             REG_SZ
  Data:             .NT

Value 15
  Name:             IntCount
  Type:             REG_SZ
  Data:             2000

Value 16
  Name:             JumboFrames
  Type:             REG_SZ
  Data:             0

Value 17
  Name:             JumboMtu
  Type:             REG_SZ
  Data:             1500

Value 18
  Name:             LinkNegotiation
  Type:             REG_SZ
  Data:             1

Value 19
  Name:             MatchingDeviceId
  Type:             REG_SZ
  Data:             pci\ven_12ae&dev_0001&subsys_00000000

Value 20
  Name:             NetCfgInstanceId
  Type:             REG_SZ
  Data:             {2158A453-5875-4347-9C36-52CC9CD2C2CF}

Value 21
  Name:             NicTracing
  Type:             REG_SZ
  Data:             0

Value 22
  Name:             PciLatencyTimer
  Type:             REG_SZ
  Data:             40

Value 23
  Name:             PciMemInvalidate
  Type:             REG_SZ
  Data:             1

Value 24
  Name:             PciReadMax
  Type:             REG_SZ
  Data:             ffffffff

Value 25
  Name:             PciWriteMax
  Type:             REG_SZ
  Data:             ffffffff

Value 26
  Name:             ProviderName
  Type:             REG_SZ
  Data:             Microsoft

Value 27
  Name:             RecvCoalMax
  Type:             REG_SZ
  Data:             80

Value 28
  Name:             RecvCoalTicks
  Type:             REG_SZ
  Data:             10000

Value 29
  Name:             RxFlowControl
  Type:             REG_SZ
  Data:             0

Value 30
  Name:             SendCoalMax
  Type:             REG_SZ
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
  Data:            80

Value 31
  Name:            SendCoalTicks
  Type:            REG_SZ
  Data:            10000

Value 32
  Name:            StatTicks
  Type:            REG_SZ
  Data:            100000

Value 33
  Name:            TxFlowControl
  Type:            REG_SZ
  Data:            0


Key Name:          SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-
11CE-BFC1-08002BE10318}\0007\Linkage
Class Name:        <NO CLASS>
Last Write Time:   8/7/2000 - 13:40
Value 0
  Name:            Export
  Type:            REG_MULTI_SZ
  Data:            \Device\{2158A453-5875-4347-9C36-52CC9CD2C2CF}


Value 1
  Name:            RootDevice
  Type:            REG_MULTI_SZ
  Data:            {2158A453-5875-4347-9C36-52CC9CD2C2CF}


Value 2
  Name:            UpperBind
  Type:            REG_MULTI_SZ
  Data:            Tcpip



Key Name:          SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-
11CE-BFC1-08002BE10318}\0007\Ndi
Class Name:        <NO CLASS>
Last Write Time:   8/7/2000 - 13:40
Value 0
  Name:            Service
  Type:            REG_SZ
  Data:            altnd5


Key Name:          SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-
11CE-BFC1-08002BE10318}\0007\Ndi\Interfaces
Class Name:        <NO CLASS>
Last Write Time:   8/7/2000 - 13:40
Value 0
  Name:            LowerRange
  Type:            REG_SZ
  Data:            ethernet

Value 1
  Name:            UpperRange
  Type:            REG_SZ
  Data:            ndis5


Key Name:          SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-
11CE-BFC1-08002BE10318}\0007\Ndi\params
Class Name:        <NO CLASS>
Last Write Time:   8/7/2000 - 13:40

Key Name:          SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-
11CE-BFC1-08002BE10318}\0007\Ndi\params\JumboFrames
Class Name:        <NO CLASS>
Last Write Time:   8/7/2000 - 13:40
Value 0
  Name:            default
  Type:            REG_SZ
  Data:            0

Value 1
  Name:            ParamDesc
  Type:            REG_SZ
  Data:            JumboFrames

Value 2
  Name:            type
  Type:            REG_SZ
  Data:            enum


Key Name:          SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-
11CE-BFC1-08002BE10318}\0007\Ndi\params\JumboFrames\enum
Class Name:        <NO CLASS>
Last Write Time:   8/7/2000 - 13:40
Value 0
  Name:            0
  Type:            REG_SZ
  Data:            Off

Value 1
  Name:            1
  Type:            REG_SZ
  Data:            On
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Key Name:          SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-
11CE-BFC1-08002BE10318}\0007\Ndi\params\JumboMtu
Class Name:        <NO CLASS>
Last Write Time:   8/7/2000 - 13:40
Value 0
  Name:            base
  Type:            REG_SZ
  Data:            10

Value 1
  Name:            default
  Type:            REG_SZ
  Data:            1500

Value 2
  Name:            max
  Type:            REG_SZ
  Data:            9000

Value 3
  Name:            min
  Type:            REG_SZ
  Data:            1500

Value 4
  Name:            ParamDesc
  Type:            REG_SZ
  Data:            JumboMtu

Value 5
  Name:            step
  Type:            REG_SZ
  Data:            100

Value 6
  Name:            type
  Type:            REG_SZ
  Data:            dword


Key Name:          SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-
11CE-BFC1-08002BE10318}\0007\Ndi\params\LinkNegotiation
Class Name:        <NO CLASS>
Last Write Time:   8/7/2000 - 13:40
Value 0
  Name:            default
  Type:            REG_SZ
  Data:            1

Value 1
  Name:            ParamDesc
  Type:            REG_SZ
  Data:            LinkNegotiation

Value 2
  Name:            type
  Type:            REG_SZ
  Data:            enum


Key Name:          SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-
11CE-BFC1-08002BE10318}\0007\Ndi\params\LinkNegotiation\enum
Class Name:        <NO CLASS>
Last Write Time:   8/7/2000 - 13:40
Value 0
  Name:            0
  Type:            REG_SZ
  Data:            Off

Value 1
  Name:            1
  Type:            REG_SZ
  Data:            On


Key Name:          SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-
11CE-BFC1-08002BE10318}\0007\Ndi\params\NetworkAddress
Class Name:        <NO CLASS>
Last Write Time:   8/7/2000 - 13:40
Value 0
  Name:            default
  Type:            REG_SZ
  Data:            0060CF000000

Value 1
  Name:            optional
  Type:            REG_SZ
  Data:            1

Value 2
  Name:            ParamDesc
  Type:            REG_SZ
  Data:            NetworkAddress

Value 3
  Name:            type
  Type:            REG_SZ
  Data:            edit


Key Name:          SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-
11CE-BFC1-08002BE10318}\0007\Ndi\params\RxFlowControl
Class Name:        <NO CLASS>
Last Write Time:   8/7/2000 - 13:40
Value 0
  Name:            default
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
  Type:            REG_SZ
  Data:            1

Value 1
  Name:            ParamDesc
  Type:            REG_SZ
  Data:            RxFlowControl

Value 2
  Name:            type
  Type:            REG_SZ
  Data:            enum


Key Name:          SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-
11CE-BFC1-08002BE10318}\0007\Ndi\params\RxFlowControl\enum
Class Name:        <NO CLASS>
Last Write Time:   8/7/2000 - 13:40
Value 0
  Name:            0
  Type:            REG_SZ
  Data:            Off

Value 1
  Name:            1
  Type:            REG_SZ
  Data:            On


Key Name:          SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-
11CE-BFC1-08002BE10318}\0007\Ndi\params\TxFlowControl
Class Name:        <NO CLASS>
Last Write Time:   8/7/2000 - 13:40
Value 0
  Name:            default
  Type:            REG_SZ
  Data:            0

Value 1
  Name:            ParamDesc
  Type:            REG_SZ
  Data:            TxFlowControl

Value 2
  Name:            type
  Type:            REG_SZ
  Data:            enum


Key Name:          SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-
11CE-BFC1-08002BE10318}\0007\Ndi\params\TxFlowControl\enum
Class Name:        <NO CLASS>
Last Write Time:   8/7/2000 - 13:40
Value 0
  Name:            0
  Type:            REG_SZ
  Data:            Off

Value 1
  Name:            1
  Type:            REG_SZ
  Data:            On


Key Name:          SYSTEM\CurrentControlSet\Services\dac2w2k
Class Name:        <NO CLASS>
Last Write Time:   3/15/2000 - 15:50
Value 0
  Name:            ErrorControl
  Type:            REG_DWORD
  Data:            0x1

Value 1
  Name:            Group
  Type:            REG_SZ
  Data:            SCSI Miniport

Value 2
  Name:            ImagePath
  Type:            REG_EXPAND_SZ
  Data:            System32\DRIVERS\dac2w2k.sys

Value 3
  Name:            Start
  Type:            REG_DWORD
  Data:            0

Value 4
  Name:            Tag
  Type:            REG_DWORD
  Data:            0x21

Value 5
  Name:            Type
  Type:            REG_DWORD
  Data:            0x1


Key Name:          SYSTEM\CurrentControlSet\Services\dac2w2k\Enum
Class Name:        <NO CLASS>
Last Write Time:   8/23/2000 - 13:42
Value 0
  Name:            0
  Type:            REG_SZ
  Data:
PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&3ec16a1&0&4050
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Value 1
  Name:            1
  Type:            REG_SZ
  Data:
PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&ba2977f&0&4060

Value 2
  Name:            2
  Type:            REG_SZ
  Data:
PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&3b7ba8be&0&4040

Value 3
  Name:            3
  Type:            REG_SZ
  Data:
PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&fadcf69&0&4050

Value 4
  Name:            4
  Type:            REG_SZ
  Data:
PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&1ea4b82&0&4040

Value 5
  Name:            5
  Type:            REG_SZ
  Data:
PCI\VEN_1069&DEV_BA56&SUBSYS_00401069&REV_00\4&2cad654f&0&4048

Value 6
  Name:            Count
  Type:            REG_DWORD
  Data:            0x6

Value 7
  Name:            NextInstance
  Type:            REG_DWORD
  Data:            0x6


Key Name:          SYSTEM\CurrentControlSet\Services\dac2w2k\Parameters
Class Name:        <NO CLASS>
Last Write Time:   3/15/2000 - 15:50

Key Name:
SYSTEM\CurrentControlSet\Services\dac2w2k\Parameters\Device
Class Name:        <NO CLASS>
Last Write Time:   8/16/2000 - 13:06
Value 0
  Name:            DriverParameter
  Type:            REG_SZ
  Data:            ConfigureSIR=16


Key Name:
SYSTEM\CurrentControlSet\Services\dac2w2k\Parameters\Device0
Class Name:        <NO CLASS>
Last Write Time:   4/3/2000 - 15:49
Value 0
  Name:            NumberOfRequests
  Type:            REG_DWORD
  Data:            0x60


Key Name:
SYSTEM\CurrentControlSet\Services\dac2w2k\Parameters\Device1
Class Name:        <NO CLASS>
Last Write Time:   4/3/2000 - 15:50
Value 0
  Name:            NumberOfRequests
  Type:            REG_DWORD
  Data:            0x60


Key Name:
SYSTEM\CurrentControlSet\Services\dac2w2k\Parameters\Device2
Class Name:        <NO CLASS>
Last Write Time:   4/3/2000 - 15:50
Value 0
  Name:            NumberOfRequests
  Type:            REG_DWORD
  Data:            0x60


Key Name:
SYSTEM\CurrentControlSet\Services\dac2w2k\Parameters\Device3
Class Name:        <NO CLASS>
Last Write Time:   4/3/2000 - 15:50
Value 0
  Name:            NumberOfRequests
  Type:            REG_DWORD
  Data:            0x60


Key Name:
SYSTEM\CurrentControlSet\Services\dac2w2k\Parameters\Device4
Class Name:        <NO CLASS>
Last Write Time:   4/3/2000 - 15:50
Value 0
  Name:            NumberOfRequests
  Type:            REG_DWORD
  Data:            0x60
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Key Name:
SYSTEM\CurrentControlSet\Services\dac2w2k\Parameters\Device5
Class Name:         <NO CLASS>
Last Write Time:    4/3/2000 - 15:50
Value 0
  Name:             NumberOfRequests
  Type:             REG_DWORD
  Data:             0x60


Key Name:
SYSTEM\CurrentControlSet\Services\dac2w2k\Parameters\Device6
Class Name:         <NO CLASS>
Last Write Time:    4/3/2000 - 15:50
Value 0
  Name:             NumberOfRequests
  Type:             REG_DWORD
  Data:             0x60


Key Name:
SYSTEM\CurrentControlSet\Services\dac2w2k\Parameters\PnpInterface
Class Name:         <NO CLASS>
Last Write Time:    3/15/2000 - 15:50
Value 0
  Name:             5
  Type:             REG_DWORD
  Data:             0x1


Key Name:           SYSTEM\CurrentControlSet\Services\dac2w2k\Security
Class Name:         <NO CLASS>
Last Write Time:    3/15/2000 - 15:50
Value 0
  Name:             Security
  Type:             REG_BINARY
  Data:
00000000   01 00 14 80 a0 00 00 00 - ac 00 00 00 14 00 00 00
.... ...¬.......
00000010   30 00 00 00 02 00 1c 00 - 01 00 00 00 02 80 14 00
0...............
00000020   ff 01 0f 00 01 01 00 00 - 00 00 00 01 00 00 00 00
ÿ...............
00000030   02 00 70 00 04 00 00 00 - 00 00 18 00 fd 01 02 00
..p.........ý...
00000040   01 01 00 00 00 00 00 05 - 12 00 00 00 74 00 69 00
............t.i.
00000050   00 00 1c 00 ff 01 0f 00 - 01 02 00 00 00 00 00 05
....ÿ...........
00000060   20 00 00 00 20 02 00 00 - 76 00 65 00 00 00 18 00    ...
...v.e.....
00000070   8d 01 02 00 01 01 00 00 - 00 00 00 05 0b 00 00 00
................
00000080   20 02 00 00 00 00 1c 00 - fd 01 02 00 01 02 00 00
.......ý........
00000090   00 00 00 05 20 00 00 00 - 23 02 00 00 76 00 65 00  ....
...#...v.e.
000000a0   01 01 00 00 00 00 00 05 - 12 00 00 00 01 01 00 00
................
000000b0   00 00 00 05 12 00 00 00 -                          ........


Key Name:           SYSTEM\CurrentControlSet\Services\NDIS
Class Name:         <NO CLASS>
Last Write Time:    6/7/2000 - 1:21 PM
Value 0
  Name:             DisplayName
  Type:             REG_SZ
  Data:             NDIS System Driver

Value 1
  Name:             ErrorControl
  Type:             REG_DWORD
  Data:             0x1

Value 2
  Name:             Group
  Type:             REG_SZ
  Data:             NDIS Wrapper

Value 3
  Name:             Start
  Type:             REG_DWORD
  Data:             0

Value 4
  Name:             Type
  Type:             REG_DWORD
  Data:             0x1


Key Name:           SYSTEM\CurrentControlSet\Services\NDIS\Enum
Class Name:         <NO CLASS>
Last Write Time:    6/19/2000 - 11:17 AM
Value 0
  Name:             0
  Type:             REG_SZ
  Data:             Root\LEGACY_NDIS\0000

Value 1
  Name:             Count
  Type:             REG_DWORD
  Data:             0x1

Value 2
  Name:             NextInstance
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
  Type:            REG_DWORD
  Data:            0x1


Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\MediaTypes
Class Name:        <NO CLASS>
Last Write Time:   6/7/2000 - 1:21 PM

Key Name:          SYSTEM\CurrentControlSet\Services\NDIS\Parameters
Class Name:        <NO CLASS>
Last Write Time:   6/7/2000 - 1:21 PM
Value 0
  Name:            ProcessorAffinityMask
  Type:            REG_DWORD
  Data:            0
```

This section discloses hardware information and the Windows 2000 registry parameters used on the PRIMERGY 170 client systems.

```
System Information report written at: 08/23/2000 02:58:13 PM
[System Summary]

Item    Value
OS Name Microsoft Windows 2000 Server
Version 5.0.2195 Service Pack 1 Build 2195
OS Manufacturer        Microsoft Corporation
System Name    WEINROT
System Manufacturer    FUJITSU SIEMENS
System Model   Pentium II
System Type    X86-based PC
Processor      x86 Family 6 Model 8 Stepping 1 GenuineIntel ~748 Mhz
BIOS Version   PhoenixBIOS Version 4.06  Rev. 1.13.1107
Windows Directory      C:\WINNT
System Directory       C:\WINNT\System32
Boot Device    \Device\Harddisk0\Partition1
Locale  United States
User Name      WEINROT\Administrator
Time Zone      W. Europe Daylight Time
Total Physical Memory  261,668 KB
Available Physical Memory     193,396 KB
Total Virtual Memory   894,632 KB
Available Virtual Memory      772,900 KB
Page File Space        632,964 KB
Page File      C:\pagefile.sys

System Information report written at: 08/23/2000 02:58:34 PM
[Hardware Resources]

[ Following are sub-categories of this main category ]

[Conflicts/Sharing]

Resource       Device
IRQ 9   Microsoft ACPI-Compliant System
IRQ 9   Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter #7
IRQ 9   Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter #8
IRQ 9   Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter #5
IRQ 9   Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter #6
IRQ 9   Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter
IRQ 9   Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter #2
IRQ 9   Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter #3
IRQ 9   Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter #4
IRQ 9   Symbios 895A-based, 8953U PCI SCSI Adapter

[DMA]

Channel Device  Status
4       Direct memory access controller       OK
2       Standard floppy disk controller       OK

[Forced Hardware]

Device  PNP Device ID
No Forced Hardware

[I/O]

Address Range  Device  Status
0x0000-0x0CF7  PCI bus OK
0x0000-0x0CF7  Direct memory access controller       OK
0x0D00-0xFFFF  PCI bus OK
0x03B0-0x03BB  Intel 82443BX Pentium(r) II Processor to AGP Controller
       OK
0x03B0-0x03BB  Matrox Graphics MGA-G100 AGP  OK
0x03C0-0x03DF  Intel 82443BX Pentium(r) II Processor to AGP Controller
       OK
0x03C0-0x03DF  Matrox Graphics MGA-G100 AGP  OK
0x0A79-0x0A79  ISAPNP Read Data Port  OK
0x0279-0x0279  ISAPNP Read Data Port  OK
0x0274-0x0277  ISAPNP Read Data Port  OK
0x0010-0x001F  Motherboard resources  OK
0x0022-0x003F  Motherboard resources  OK
0x0050-0x0053  Motherboard resources  OK
0x0062-0x0063  Motherboard resources  OK
0x0065-0x006F  Motherboard resources  OK
0x0074-0x007F  Motherboard resources  OK
0x0090-0x0091  Motherboard resources  OK
0x0093-0x009F  Motherboard resources  OK
0x00A2-0x00B1  Motherboard resources  OK
0x00B4-0x00BF  Motherboard resources  OK
0x00E0-0x00EF  Motherboard resources  OK
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
0x0072-0x0073  Motherboard resources  OK
0x0370-0x0371  Motherboard resources  OK
0x04D0-0x04D1  Motherboard resources  OK
0xF0B0-0xF0BF  Motherboard resources  OK
0xF0C0-0xF0CF  Motherboard resources  OK
0xF0D0-0xF0FF  Motherboard resources  OK
0x0080-0x008F  Direct memory access controller       OK
0x00C0-0x00DF  Direct memory access controller       OK
0x0020-0x0021  Programmable interrupt controller     OK
0x00A0-0x00A1  Programmable interrupt controller     OK
0x0070-0x0071  System CMOS/real time clock   OK
0x0040-0x0043  System timer   OK
0x00F0-0x00FE  Numeric data processor OK
0x0061-0x0061  System speaker OK
0x0060-0x0060  Standard 101/102-Key or Microsoft Natural PS/2 Keyboard
       OK
0x0064-0x0064  Standard 101/102-Key or Microsoft Natural PS/2 Keyboard
       OK
0x03F0-0x03F5  Standard floppy disk controller       OK
0x03F7-0x03F7  Standard floppy disk controller       OK
0xFCF0-0xFCFF  Intel(r) 82371AB/EB PCI Bus Master IDE Controller    OK
0x01F0-0x01F7  Primary IDE Channel    OK
0x03F6-0x03F6  Primary IDE Channel    OK
0x0170-0x0177  Secondary IDE Channel  OK
0x0376-0x0376  Secondary IDE Channel  OK
0xE000-0xEFFF  DEC 21154 PCI to PCI bridge   OK
0xEC00-0xECFF  Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter
#7     OK
0xE800-0xE8FF  Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter
#8     OK
0xD000-0xDFFF  DEC 21154 PCI to PCI bridge   OK
0xDC00-0xDCFF  Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter
#5     OK
0xD800-0xD8FF  Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter
#6     OK
0xC000-0xCFFF  DEC 21154 PCI to PCI bridge   OK
0xCC00-0xCCFF  Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter
       OK
0xC800-0xC8FF  Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter
#2     OK
0xB000-0xBFFF  DEC 21154 PCI to PCI bridge   OK
0xBC00-0xBCFF  Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter
#3     OK
0xB800-0xB8FF  Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter
#4     OK
0xF400-0xF4FF  Symbios 895A-based, 8953U PCI SCSI Adapter   OK

[IRQs]

IRQ Number     Device
9      Microsoft ACPI-Compliant System
9      Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter #7
9      Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter #8
9      Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter #5
9      Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter #6
9      Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter
9      Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter #2
9      Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter #3
9      Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter #4
9      Symbios 895A-based, 8953U PCI SCSI Adapter
8      System CMOS/real time clock
13     Numeric data processor
1      Standard 101/102-Key or Microsoft Natural PS/2 Keyboard
12     PS/2 Compatible Mouse
6      Standard floppy disk controller
14     Primary IDE Channel
15     Secondary IDE Channel

[Memory]

Range   Device  Status
0xA0000-0xBFFFF        PCI bus OK
0xA0000-0xBFFFF        Intel 82443BX Pentium(r) II Processor to AGP
Controller     OK
0xA0000-0xBFFFF        Matrox Graphics MGA-G100 AGP  OK
0xC8000-0xDFFFF        PCI bus OK
0x10000000-0xFFFDFFFF  PCI bus OK
0xFE000000-0xFECFFFFF  Intel 82443BX Pentium(r) II Processor to AGP
Controller     OK
0xFE000000-0xFECFFFFF  Matrox Graphics MGA-G100 AGP  OK
0xF6000000-0xF6FFFFFF  Intel 82443BX Pentium(r) II Processor to AGP
Controller     OK
0xF6000000-0xF6FFFFFF  Matrox Graphics MGA-G100 AGP  OK
0xF8000000-0xFBFFFFFF  Intel 82443BX Pentium(r) II Processor to AGP
Controller     OK
0xFECF8000-0xFECFBFFF  Matrox Graphics MGA-G100 AGP  OK
0xFFC00000-0xFFDFFFFF  DEC 21154 PCI to PCI bridge   OK
0xFFD80000-0xFFDFFFFF  Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet
Adapter #7     OK
0xFFD00000-0xFFD7FFFF  Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet
Adapter #8     OK
0xFFA00000-0xFFBFFFFF  DEC 21154 PCI to PCI bridge   OK
0xFFB80000-0xFFBFFFFF  Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet
Adapter #5     OK
0xFFB00000-0xFFB7FFFF  Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet
Adapter #6     OK
0xFF800000-0xFF9FFFFF  DEC 21154 PCI to PCI bridge   OK
0xFF980000-0xFF9FFFFF  Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet
Adapter OK
0xFF900000-0xFF97FFFF  Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet
Adapter #2     OK
0xFF600000-0xFF7FFFFF  DEC 21154 PCI to PCI bridge   OK
0xFF780000-0xFF7FFFFF  Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet
Adapter #3     OK
0xFF700000-0xFF77FFFF  Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet
Adapter #4     OK
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
0xFEDFDC00-0xFEDFDFFF  Symbios 895A-based, 8953U PCI SCSI Adapter   OK
0xFEDFE000-0xFEDFFFFF  Symbios 895A-based, 8953U PCI SCSI Adapter   OK

System Information report written at: 08/23/2000 02:59:01 PM
[Storage]

[ Following are sub-categories of this main category ]

[Drives]

Item    Value
Drive   A:
Description     3 1/2 Inch Floppy Drive

Drive   C:
Description     Local Fixed Disk
Compressed      False
File System     NTFS
Size    8.50 GB (9,121,800,192 bytes)
Free Space      6.74 GB (7,240,564,736 bytes)
Volume Name
Volume Serial Number   2C01718B
Partition       Disk #0, Partition #0
Partition Size 8.50 GB (9,121,803,264 bytes)
Starting Offset         32256 bytes
Drive Description       Disk drive
Drive Manufacturer      (Standard disk drives)
Drive Model     FUJITSU MAE3091LC SCSI Disk Device
Drive BytesPerSector   512
Drive MediaLoaded       True
Drive MediaType         Fixed hard disk media
Drive Partitions        1
Drive SCSIBus   0
Drive SCSILogicalUnit  0
Drive SCSIPort 2
Drive SCSITargetId      0
Drive SectorsPerTrack  63
Drive Size      9121835520 bytes
Drive TotalCylinders   1109
Drive TotalSectors      17816085
Drive TotalTracks       282795
Drive TracksPerCylinder        255


[SCSI]

Item    Value
Name    Symbios 895A-based, 8953U PCI SCSI Adapter
Caption Symbios 895A-based, 8953U PCI SCSI Adapter
Driver  Sym_895a
Status  OK
PNP Device ID
        PCI\VEN_1000&DEV_0012&SUBSYS_6020110A&REV_01\3&61AAA01&0&90
Device ID
        PCI\VEN_1000&DEV_0012&SUBSYS_6020110A&REV_01\3&61AAA01&0&90
Device Map      Not Available
Index   Not Available
Max Number Controlled  Not Available
IRQ Number      9
I/O Port        0xF400-0xF4FF
Driver  c:\winnt\system32\drivers\sym_895a.sys (22448, SYM_895A-4.14.00)

Name    Symbios 895A-based, 8953U PCI SCSI Adapter
Caption Symbios 895A-based, 8953U PCI SCSI Adapter
Driver  Sym_895a
Status  Error
PNP Device ID  ROOT\SCSIADAPTER\0000
Device ID       ROOT\SCSIADAPTER\0000
Device Map      Not Available
Index   Not Available
Max Number Controlled  Not Available
Driver  c:\winnt\system32\drivers\sym_895a.sys (22448, SYM_895A-4.14.00)

System Information report written at: 08/23/2000 03:01:57 PM
[Adapter]

Item    Value
Name    [00000000] Intel 8255x-based PCI Ethernet Adapter (10/100)
Adapter Type    Not Available
Product Name    Intel 8255x-based PCI Ethernet Adapter (10/100)
Installed       True
PNP Device ID  Not Available
Last Reset      8/23/2000 3:34:16 PM
Index   0
Service Name    E100B
IP Address      Not Available
IP Subnet       Not Available
Default IP Gateway     Not Available
DHCP Enabled    True
DHCP Server     Not Available
DHCP Lease Expires     Not Available
DHCP Lease Obtained    Not Available
MAC Address     Not Available
Service Name    Not Available

Name    [00000001] RAS Async Adapter
Adapter Type    Not Available
Product Name    RAS Async Adapter
Installed       True
PNP Device ID  Not Available
Last Reset      8/23/2000 3:34:16 PM
Index   1
Service Name    AsyncMac
IP Address      Not Available
IP Subnet       Not Available
Default IP Gateway     Not Available
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
DHCP Enabled   False
DHCP Server    Not Available
DHCP Lease Expires     Not Available
DHCP Lease Obtained    Not Available
MAC Address    Not Available
Service Name   Not Available

Name    [00000002] WAN Miniport (L2TP)
Adapter Type   Not Available
Product Name   WAN Miniport (L2TP)
Installed      True
PNP Device ID  ROOT\MS_L2TPMINIPORT\0000
Last Reset     8/23/2000 3:34:16 PM
Index  2
Service Name   Rasl2tp
IP Address     Not Available
IP Subnet      Not Available
Default IP Gateway     Not Available
DHCP Enabled   False
DHCP Server    Not Available
DHCP Lease Expires     Not Available
DHCP Lease Obtained    Not Available
MAC Address    Not Available
Service Name   Rasl2tp
Driver  c:\winnt\system32\drivers\rasl2tp.sys (50800, 5.00.2179.1)

Name    [00000003] WAN Miniport (PPTP)
Adapter Type   Wide Area Network (WAN)
Product Name   WAN Miniport (PPTP)
Installed      True
PNP Device ID  ROOT\MS_PPTPMINIPORT\0000
Last Reset     8/23/2000 3:34:16 PM
Index  3
Service Name   PptpMiniport
IP Address     Not Available
IP Subnet      Not Available
Default IP Gateway     Not Available
DHCP Enabled   False
DHCP Server    Not Available
DHCP Lease Expires     Not Available
DHCP Lease Obtained    Not Available
MAC Address    50:50:54:50:30:30
Service Name   PptpMiniport
Driver  c:\winnt\system32\drivers\raspptp.sys (47856, 5.00.2160.1)

Name    [00000004] Direct Parallel
Adapter Type   Not Available
Product Name   Direct Parallel
Installed      True
PNP Device ID  ROOT\MS_PTIMINIPORT\0000
Last Reset     8/23/2000 3:34:16 PM
Index  4
Service Name   Raspti
IP Address     Not Available
IP Subnet      Not Available
Default IP Gateway     Not Available
DHCP Enabled   False
DHCP Server    Not Available
DHCP Lease Expires     Not Available
DHCP Lease Obtained    Not Available
MAC Address    Not Available
Service Name   Raspti
Driver  c:\winnt\system32\drivers\raspti.sys (16880, 5.00.2146.1)

Name    [00000005] WAN Miniport (IP)
Adapter Type   Not Available
Product Name   WAN Miniport (IP)
Installed      True
PNP Device ID  ROOT\MS_NDISWANIP\0000
Last Reset     8/23/2000 3:34:16 PM
Index  5
Service Name   NdisWan
IP Address     Not Available
IP Subnet      Not Available
Default IP Gateway     Not Available
DHCP Enabled   False
DHCP Server    Not Available
DHCP Lease Expires     Not Available
DHCP Lease Obtained    Not Available
MAC Address    Not Available
Service Name   NdisWan
Driver  c:\winnt\system32\drivers\ndiswan.sys (90768, 5.00.2184.1)

Name    [00000006] Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet
Adapter
Adapter Type   Ethernet 802.3
Product Name   Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter
Installed      True
PNP Device ID
       PCI\VEN_9004&DEV_6915&SUBSYS_00109004&REV_03\4&171F2C55&0&2070
Last Reset     8/23/2000 3:34:16 PM
Index  6
Service Name   ADPTSF
IP Address     129.103.181.131
IP Subnet      255.255.255.0
Default IP Gateway     Not Available
DHCP Enabled   False
DHCP Server    Not Available
DHCP Lease Expires     Not Available
DHCP Lease Obtained    Not Available
MAC Address    00:00:D1:D9:AC:C7
Service Name   ADPTSF
IRQ Number     9
I/O Port       0xCC00-0xCCFF
Driver  c:\winnt\system32\drivers\adptsf50.sys (49120, V5.10.06)
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Name    [00000007] Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet
Adapter
Adapter Type    Ethernet 802.3
Product Name    Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter
Installed       True
PNP Device ID
        PCI\VEN_9004&DEV_6915&SUBSYS_00109004&REV_03\4&171F2C55&0&2870
Last Reset      8/23/2000 3:34:16 PM
Index   7
Service Name    ADPTSF
IP Address      129.103.150.2
IP Subnet       255.255.255.0
Default IP Gateway      Not Available
DHCP Enabled    False
DHCP Server     Not Available
DHCP Lease Expires      Not Available
DHCP Lease Obtained     Not Available
MAC Address     00:00:D1:D9:AC:C8
Service Name    ADPTSF
IRQ Number      9
I/O Port        0xC800-0xC8FF
Driver  c:\winnt\system32\drivers\adptsf50.sys (49120, V5.10.06)

Name    [00000008] Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet
Adapter
Adapter Type    Ethernet 802.3
Product Name    Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter
Installed       True
PNP Device ID
        PCI\VEN_9004&DEV_6915&SUBSYS_00109004&REV_03\4&2681C776&0&2080
Last Reset      8/23/2000 3:34:16 PM
Index   8
Service Name    ADPTSF
IP Address      129.103.151.2
IP Subnet       255.255.255.0
Default IP Gateway      Not Available
DHCP Enabled    False
DHCP Server     Not Available
DHCP Lease Expires      Not Available
DHCP Lease Obtained     Not Available
MAC Address     00:00:D1:D9:AB:B5
Service Name    ADPTSF
IRQ Number      9
I/O Port        0xBC00-0xBCFF
Driver  c:\winnt\system32\drivers\adptsf50.sys (49120, V5.10.06)

Name    [00000009] Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet
Adapter
Adapter Type    Ethernet 802.3
Product Name    Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter
Installed       True
PNP Device ID
        PCI\VEN_9004&DEV_6915&SUBSYS_00109004&REV_03\4&2681C776&0&2880
Last Reset      8/23/2000 3:34:16 PM
Index   9
Service Name    ADPTSF
IP Address      129.103.152.2
IP Subnet       255.255.255.0
Default IP Gateway      Not Available
DHCP Enabled    False
DHCP Server     Not Available
DHCP Lease Expires      Not Available
DHCP Lease Obtained     Not Available
MAC Address     00:00:D1:D9:AB:B6
Service Name    ADPTSF
IRQ Number      9
I/O Port        0xB800-0xB8FF
Driver  c:\winnt\system32\drivers\adptsf50.sys (49120, V5.10.06)

Name    [00000010] Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet
Adapter
Adapter Type    Ethernet 802.3
Product Name    Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter
Installed       True
PNP Device ID
        PCI\VEN_9004&DEV_6915&SUBSYS_00109004&REV_03\4&7907E35&0&2060
Last Reset      8/23/2000 3:34:16 PM
Index   10
Service Name    ADPTSF
IP Address      129.103.153.2
IP Subnet       255.255.255.0
Default IP Gateway      Not Available
DHCP Enabled    False
DHCP Server     Not Available
DHCP Lease Expires      Not Available
DHCP Lease Obtained     Not Available
MAC Address     00:00:D1:D9:AE:01
Service Name    ADPTSF
IRQ Number      9
I/O Port        0xDC00-0xDCFF
Driver  c:\winnt\system32\drivers\adptsf50.sys (49120, V5.10.06)

Name    [00000011] Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet
Adapter
Adapter Type    Ethernet 802.3
Product Name    Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter
Installed       True
PNP Device ID
        PCI\VEN_9004&DEV_6915&SUBSYS_00109004&REV_03\4&7907E35&0&2860
Last Reset      8/23/2000 3:34:16 PM
Index   11
Service Name    ADPTSF
IP Address      129.103.154.2
IP Subnet       255.255.255.0
Default IP Gateway      Not Available
DHCP Enabled    False
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
DHCP Server    Not Available
DHCP Lease Expires    Not Available
DHCP Lease Obtained    Not Available
MAC Address    00:00:D1:D9:AE:02
Service Name   ADPTSF
IRQ Number     9
I/O Port       0xD800-0xD8FF
Driver c:\winnt\system32\drivers\adptsf50.sys (49120, V5.10.06)

Name    [00000012] Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet
Adapter
Adapter Type   Ethernet 802.3
Product Name   Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter
Installed      True
PNP Device ID
       PCI\VEN_9004&DEV_6915&SUBSYS_00109004&REV_03\4&7FE2FEB&0&2050
Last Reset     8/23/2000 3:34:16 PM
Index  12
Service Name   ADPTSF
IP Address     129.103.155.2
IP Subnet      255.255.255.0
Default IP Gateway    Not Available
DHCP Enabled   False
DHCP Server    Not Available
DHCP Lease Expires    Not Available
DHCP Lease Obtained   Not Available
MAC Address    00:00:D1:D9:AD:F5
Service Name   ADPTSF
IRQ Number     9
I/O Port       0xEC00-0xECFF
Driver c:\winnt\system32\drivers\adptsf50.sys (49120, V5.10.06)

Name    [00000013] Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet
Adapter
Adapter Type   Ethernet 802.3
Product Name   Adaptec ANA62022 64-bit 2 port PCI Fast Ethernet Adapter
Installed      True
PNP Device ID
       PCI\VEN_9004&DEV_6915&SUBSYS_00109004&REV_03\4&7FE2FEB&0&2850
Last Reset     8/23/2000 3:34:16 PM
Index  13
Service Name   ADPTSF
IP Address     129.103.156.2
IP Subnet      255.255.255.0
Default IP Gateway    Not Available
DHCP Enabled   False
DHCP Server    Not Available
DHCP Lease Expires    Not Available
DHCP Lease Obtained   Not Available
MAC Address    00:00:D1:D9:AD:F6
Service Name   ADPTSF
IRQ Number     9
I/O Port       0xE800-0xE8FF
Driver c:\winnt\system32\drivers\adptsf50.sys (49120, V5.10.06)

System Information report written at: 08/23/2000 03:02:10 PM
[Protocol]

Item    Value
Name    MSAFD Tcpip [TCP/IP]
ConnectionlessService  False
GuaranteesDelivery     True
GuaranteesSequencing   True
MaximumAddressSize     16 bytes
MaximumMessageSize     0 bytes
MessageOriented        False
MinimumAddressSize     16 bytes
PseudoStreamOriented   False
SupportsBroadcasting   False
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  True
SupportsGracefulClosing       True
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False

Name    MSAFD Tcpip [UDP/IP]
ConnectionlessService  True
GuaranteesDelivery     False
GuaranteesSequencing   False
MaximumAddressSize     16 bytes
MaximumMessageSize     65467 bytes
MessageOriented        True
MinimumAddressSize     16 bytes
PseudoStreamOriented   False
SupportsBroadcasting   True
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   True

Name    RSVP UDP Service Provider
ConnectionlessService  True
GuaranteesDelivery     False
GuaranteesSequencing   False
MaximumAddressSize     16 bytes
MaximumMessageSize     65467 bytes
MessageOriented        True
MinimumAddressSize     16 bytes
PseudoStreamOriented   False
SupportsBroadcasting   True
SupportsConnectData    False
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
SupportsDisconnectData False
SupportsEncryption     True
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting  True

Name    RSVP TCP Service Provider
ConnectionlessService  False
GuaranteesDelivery     True
GuaranteesSequencing   True
MaximumAddressSize     16 bytes
MaximumMessageSize     0 bytes
MessageOriented        False
MinimumAddressSize     16 bytes
PseudoStreamOriented   False
SupportsBroadcasting   False
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     True
SupportsExpeditedData  True
SupportsGracefulClosing       True
SupportsGuaranteedBandwidth   False
SupportsMulticasting  False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{8C0E0FA5-A0E9-44B8-9574-
56CB00B61622}] SEQPACKET 10
ConnectionlessService  False
GuaranteesDelivery     True
GuaranteesSequencing   True
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   False
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting  False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{8C0E0FA5-A0E9-44B8-9574-
56CB00B61622}] DATAGRAM 10
ConnectionlessService  True
GuaranteesDelivery     False
GuaranteesSequencing   False
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   True
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting  False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{7629D0B4-AE77-42EF-AA30-
02FE94958788}] SEQPACKET 9
ConnectionlessService  False
GuaranteesDelivery     True
GuaranteesSequencing   True
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   False
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting  False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{7629D0B4-AE77-42EF-AA30-
02FE94958788}] DATAGRAM 9
ConnectionlessService  True
GuaranteesDelivery     False
GuaranteesSequencing   False
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   True
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting  False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{22952CD6-289C-4091-AD4A-
EE1380D61F22}] SEQPACKET 8
ConnectionlessService  False
GuaranteesDelivery     True
GuaranteesSequencing   True
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   False
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{22952CD6-289C-4091-AD4A-
EE1380D61F22}] DATAGRAM 8
ConnectionlessService  True
GuaranteesDelivery     False
GuaranteesSequencing   False
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   True
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{AED5412A-08F2-459B-BC77-
DD710DC51898}] SEQPACKET 7
ConnectionlessService  False
GuaranteesDelivery     True
GuaranteesSequencing   True
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   False
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{AED5412A-08F2-459B-BC77-
DD710DC51898}] DATAGRAM 7
ConnectionlessService  True
GuaranteesDelivery     False
GuaranteesSequencing   False
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   True
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{AD95CD3F-D209-4F78-8AD3-
167D84FE9B25}] SEQPACKET 6
ConnectionlessService  False
GuaranteesDelivery     True
GuaranteesSequencing   True
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   False
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{AD95CD3F-D209-4F78-8AD3-
167D84FE9B25}] DATAGRAM 6
ConnectionlessService  True
GuaranteesDelivery     False
GuaranteesSequencing   False
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   True
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{FAEE3419-626D-4AB7-828D-
EE0929036888}] SEQPACKET 5
ConnectionlessService  False
GuaranteesDelivery     True
GuaranteesSequencing   True
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   False
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{FAEE3419-626D-4AB7-828D-
EE0929036888}] DATAGRAM 5
ConnectionlessService  True
GuaranteesDelivery     False
GuaranteesSequencing   False
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   True
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{83865ADC-A313-40AE-8435-
69B3878E6994}] SEQPACKET 4
ConnectionlessService  False
GuaranteesDelivery     True
GuaranteesSequencing   True
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   False
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{83865ADC-A313-40AE-8435-
69B3878E6994}] DATAGRAM 4
ConnectionlessService  True
GuaranteesDelivery     False
GuaranteesSequencing   False
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   True
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{9539A88F-6D41-446C-91D7-
3E5A18407CA6}] SEQPACKET 3
ConnectionlessService  False
GuaranteesDelivery     True
GuaranteesSequencing   True
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   False
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{9539A88F-6D41-446C-91D7-
3E5A18407CA6}] DATAGRAM 3
ConnectionlessService  True
GuaranteesDelivery     False
GuaranteesSequencing   False
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   True
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{24BF2DA1-6F4A-4EF8-82F7-
2C8DB7817F52}] SEQPACKET 0
ConnectionlessService  False
GuaranteesDelivery     True
GuaranteesSequencing   True
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   False
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{24BF2DA1-6F4A-4EF8-82F7-
2C8DB7817F52}] DATAGRAM 0
ConnectionlessService  True
GuaranteesDelivery     False
GuaranteesSequencing   False
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   True
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{43107499-2A1C-4CD0-ACA2-
0A98B1575075}] SEQPACKET 1
ConnectionlessService  False
GuaranteesDelivery     True
GuaranteesSequencing   True
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   False
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{43107499-2A1C-4CD0-ACA2-
0A98B1575075}] DATAGRAM 1
ConnectionlessService  True
GuaranteesDelivery     False
GuaranteesSequencing   False
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   True
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False

Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{3CBACB77-DB9E-4FBC-AF47-
B6534495A056}] SEQPACKET 2
ConnectionlessService  False
GuaranteesDelivery     True
GuaranteesSequencing   True
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   False
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Name    MSAFD NetBIOS [\Device\NetBT_Tcpip_{3CBACB77-DB9E-4FBC-AF47-
B6534495A056}] DATAGRAM 2
ConnectionlessService  True
GuaranteesDelivery     False
GuaranteesSequencing   False
MaximumAddressSize     20 bytes
MaximumMessageSize     64000 bytes
MessageOriented        True
MinimumAddressSize     20 bytes
PseudoStreamOriented   False
SupportsBroadcasting   True
SupportsConnectData    False
SupportsDisconnectData False
SupportsEncryption     False
SupportsExpeditedData  False
SupportsGracefulClosing       False
SupportsGuaranteedBandwidth   False
SupportsMulticasting   False


System Information report written at: 08/23/2000 02:58:13 PM
[System Summary]

Item    Value
OS Name Microsoft Windows 2000 Server
Version 5.0.2195 Service Pack 1 Build 2195
OS Manufacturer        Microsoft Corporation
System Name     WEINROT
System Manufacturer    FUJITSU SIEMENS
System Model    Pentium II
System Type     X86-based PC
Processor       x86 Family 6 Model 8 Stepping 1 GenuineIntel ~748 Mhz
BIOS Version    PhoenixBIOS Version 4.06  Rev. 1.13.1107
Windows Directory      C:\WINNT
System Directory       C:\WINNT\System32
Boot Device     \Device\Harddisk0\Partition1
Locale  United States
User Name       WEINROT\Administrator
Time Zone       W. Europe Daylight Time
Total Physical Memory  261,668 KB
Available Physical Memory     193,396 KB
Total Virtual Memory   894,632 KB
Available Virtual Memory      772,900 KB
Page File Space        632,964 KB
Page File      C:\pagefile.sys


System Information report written at: 08/23/2000 03:03:02 PM
[Environment Variables]

Variable       Value   User Name
ComSpec %SystemRoot%\system32\cmd.exe <SYSTEM>
Os2LibPath     %SystemRoot%\system32\os2\dll;        <SYSTEM>
Path
       %SystemRoot%\system32;%SystemRoot%;%SystemRoot%\system32\WBEM;C:\M
SSQL7\BINN;C:\Program Files\Microsoft SQL Server\80\Tools\BINN
       <SYSTEM>
windir  %SystemRoot%    <SYSTEM>
OS      Windows_NT      <SYSTEM>
PROCESSOR_ARCHITECTURE x86      <SYSTEM>
PROCESSOR_LEVEL        6       <SYSTEM>
PROCESSOR_IDENTIFIER   x86 Family 6 Model 8 Stepping 1, GenuineIntel
       <SYSTEM>
PROCESSOR_REVISION     0801    <SYSTEM>
NUMBER_OF_PROCESSORS   1       <SYSTEM>
PATHEXT .COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH     <SYSTEM>
TEMP    %SystemRoot%\TEMP      <SYSTEM>
TMP     %SystemRoot%\TEMP      <SYSTEM>
TEMP    %USERPROFILE%\Local Settings\Temp     WEINROT\Administrator
TMP     %USERPROFILE%\Local Settings\Temp     WEINROT\Administrator


System Information report written at: 08/23/2000 03:03:18 PM
[Services]

Display Name    Name    State   Start Mode      Service Type    Path    Error
Control Start Name      Tag ID
Alerter Alerter Running Auto    Share Process
       c:\winnt\system32\services.exe        Normal  LocalSystem     0
Application Management AppMgmt Stopped Manual  Share Process
       c:\winnt\system32\services.exe        Normal  LocalSystem     0
Computer Browser       Browser Stopped Manual  Share Process
       c:\winnt\system32\services.exe        Normal  LocalSystem     0
Indexing Service       cisvc   Stopped Manual  Share Process
       c:\winnt\system32\cisvc.exe   Normal  LocalSystem     0
ClipBook       ClipSrv Stopped Manual  Own Process
       c:\winnt\system32\clipsrv.exe Normal  LocalSystem     0
Distributed File System        Dfs     Stopped Manual  Own Process
       c:\winnt\system32\dfssvc.exe  Normal  LocalSystem     0
DHCP Client     Dhcp    Stopped Disabled        Share Process
       c:\winnt\system32\services.exe        Normal  LocalSystem     0
Logical Disk Manager Administrative Service  dmadmin Stopped Manual  Share
Process c:\winnt\system32\dmadmin.exe /com    Normal  LocalSystem     0
Logical Disk Manager   dmserver        Stopped Manual  Share Process
       c:\winnt\system32\services.exe        Normal  LocalSystem     0
DNS Client      Dnscache        Stopped Disabled        Share Process
       c:\winnt\system32\services.exe        Normal  LocalSystem     0
Event Log       Eventlog        Running Auto    Share Process
       c:\winnt\system32\services.exe        Normal  LocalSystem     0
COM+ Event System      EventSystem     Running Manual  Share Process
       c:\winnt\system32\svchost.exe -k netsvcs     Normal  LocalSystem
       0
Fax Service     Fax     Stopped Disabled        Own Process
       c:\winnt\system32\faxsvc.exe  Normal  LocalSystem     0
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
IIS Admin Service      IISADMIN       Running Auto    Share Process
       c:\winnt\system32\inetsrv\inetinfo.exe       Normal  LocalSystem
       0
Intersite Messaging    IsmServ Stopped Disabled       Own Process
       c:\winnt\system32\ismserv.exe Normal  LocalSystem    0
Kerberos Key Distribution Center      kdc     Stopped Disabled       Share
Process c:\winnt\system32\lsass.exe   Normal  LocalSystem    0
Server  lanmanserver   Running Auto    Share Process
       c:\winnt\system32\services.exe        Normal  LocalSystem    0
Workstation    lanmanworkstation      Running Auto    Share Process
       c:\winnt\system32\services.exe        Normal  LocalSystem    0
License Logging Service       LicenseService  Stopped Manual  Own Process
       c:\winnt\system32\llssrv.exe  Normal  LocalSystem    0
TCP/IP NetBIOS Helper Service LmHosts Running Auto    Share Process
       c:\winnt\system32\services.exe        Normal  LocalSystem    0
Messenger      Messenger      Running Auto    Share Process
       c:\winnt\system32\services.exe        Normal  LocalSystem    0
NetMeeting Remote Desktop Sharing     mnmsrvc Stopped Manual  Own Process
       c:\winnt\system32\mnmsrvc.exe Normal  LocalSystem    0
Distributed Transaction Coordinator   MSDTC   Stopped Manual  Own Process
       c:\winnt\system32\msdtc.exe   Normal  LocalSystem    1
Windows Installer      MSIServer      Stopped Manual  Share Process
       c:\winnt\system32\msiexec.exe /v      Normal  LocalSystem    0
Message Queuing        MSMQ    Stopped Disabled       Own Process
       c:\winnt\system32\mqsvc.exe   Normal  LocalSystem    0
Network DDE    NetDDE  Stopped Manual  Share Process
       c:\winnt\system32\netdde.exe  Normal  LocalSystem    0
Network DDE DSDM       NetDDEdsdm     Stopped Manual  Share Process
       c:\winnt\system32\netdde.exe  Normal  LocalSystem    0
Net Logon      Netlogon       Stopped Manual  Share Process
       c:\winnt\system32\lsass.exe   Normal  LocalSystem    0
Network Connections    Netman  Running Manual  Share Process
       c:\winnt\system32\svchost.exe -k netsvcs     Normal  LocalSystem
       0
File Replication       NtFrs   Stopped Manual  Own Process
       c:\winnt\system32\ntfrs.exe   Ignore  LocalSystem    0
NT LM Security Support Provider       NtLmSsp Stopped Manual  Share Process
       c:\winnt\system32\lsass.exe   Normal  LocalSystem    0
Removable Storage      NtmsSvc Stopped Disabled       Share Process
       c:\winnt\system32\svchost.exe -k netsvcs     Normal  LocalSystem
       0
Plug and Play  PlugPlay       Running Auto    Share Process
       c:\winnt\system32\services.exe        Normal  LocalSystem    0
IPSEC Policy Agent     PolicyAgent    Stopped Manual  Share Process
       c:\winnt\system32\lsass.exe   Normal  LocalSystem    0
Protected Storage      ProtectedStorage       Running Manual  Share Process
       c:\winnt\system32\services.exe        Normal  LocalSystem    0
Remote Access Auto Connection Manager RasAuto Stopped Manual  Share Process
       c:\winnt\system32\svchost.exe -k netsvcs     Normal  LocalSystem
       0
Remote Access Connection Manager      RasMan  Stopped Manual  Share Process
       c:\winnt\system32\svchost.exe -k netsvcs     Normal  LocalSystem
       0
Routing and Remote Access     RemoteAccess   Stopped Disabled       Share
Process c:\winnt\system32\svchost.exe -k netsvcs     Normal  LocalSystem
       0
Remote Registry Service       RemoteRegistry Stopped Manual  Own Process
       c:\winnt\system32\regsvc.exe  Normal  LocalSystem    0
Remote Command Service RMSYS   Stopped Manual  Own Process
       c:\benchcrf\rsys.exe   Normal  LocalSystem    0
Remote Procedure Call (RPC) Locator   RpcLocator     Stopped Manual  Own
Process c:\winnt\system32\locator.exe Normal  LocalSystem    0
Remote Procedure Call (RPC)   RpcSs   Running Auto    Share Process
       c:\winnt\system32\svchost -k rpcss    Normal  LocalSystem    0
QoS RSVP       RSVP    Running Manual  Own Process
       c:\winnt\system32\rsvp.exe -s Normal  LocalSystem    0
Security Accounts Manager     SamSs   Running Auto    Share Process
       c:\winnt\system32\lsass.exe   Normal  LocalSystem    0
Smart Card Helper      SCardDrv       Stopped Manual  Share Process
       c:\winnt\system32\scardsvr.exe        Ignore  LocalSystem    0
Smart Card     SCardSvr       Stopped Manual  Share Process
       c:\winnt\system32\scardsvr.exe        Ignore  LocalSystem    0
Task Scheduler Schedule       Stopped Manual  Share Process
       c:\winnt\system32\mstask.exe  Normal  LocalSystem    0
RunAs Service  seclogon       Stopped Manual  Share Process
       c:\winnt\system32\services.exe        Ignore  LocalSystem    0
System Event Notification     SENS    Running Auto    Share Process
       c:\winnt\system32\svchost.exe -k netsvcs     Normal  LocalSystem
       0
Internet Connection Sharing   SharedAccess   Stopped Manual  Share Process
       c:\winnt\system32\svchost.exe -k netsvcs     Normal  LocalSystem
       0
Simple Mail Transport Protocol (SMTP) SMTPSVC Stopped Disabled       Share
Process c:\winnt\system32\inetsrv\inetinfo.exe       Normal  LocalSystem
       0
Print Spooler  Spooler Stopped Disabled       Own Process
       c:\winnt\system32\spoolsv.exe Normal  LocalSystem    0
Performance Logs and Alerts   SysmonLog      Stopped Manual  Own Process
       c:\winnt\system32\smlogsvc.exe        Normal  LocalSystem    0
Telephony      TapiSrv Stopped Disabled       Share Process
       c:\winnt\system32\svchost.exe -k tapisrv     Normal  LocalSystem
       0
Terminal Services      TermService    Stopped Disabled       Own Process
       c:\winnt\system32\termsrv.exe Normal  LocalSystem    0
Telnet  TlntSvr Stopped Manual  Own Process
       c:\winnt\system32\tlntsvr.exe Normal  LocalSystem    0
Distributed Link Tracking Server      TrkSvr  Stopped Manual  Share Process
       c:\winnt\system32\services.exe        Normal  LocalSystem    0
Distributed Link Tracking Client      TrkWks  Stopped Manual  Share Process
       c:\winnt\system32\services.exe        Normal  LocalSystem    0
Uninterruptible Power Supply  UPS     Stopped Manual  Own Process
       c:\winnt\system32\ups.exe     Normal  LocalSystem    0
Utility Manager        UtilMan Stopped Manual  Own Process
       c:\winnt\system32\utilman.exe Normal  LocalSystem    0
Windows Time    W32Time Stopped Manual  Share Process
       c:\winnt\system32\services.exe        Normal  LocalSystem    0
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
World Wide Web Publishing Service     W3SVC   Running Auto    Share Process
       c:\winnt\system32\inetsrv\inetinfo.exe       Normal  LocalSystem
       0
Windows Management Instrumentation    WinMgmt Running Auto    Own Process
       c:\winnt\system32\wbem\winmgmt.exe    Ignore  LocalSystem    0
Windows Management Instrumentation Driver Extensions Wmi     Running Manual
       Share Process  c:\winnt\system32\services.exe       Normal
       LocalSystem    0


Key Name:          SYSTEM\CurrentControlSet\Services\ADPTSF
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM
Value 0
  Name:            DisplayName
  Type:            REG_SZ
  Data:            Adaptec DuraLAN PCI Ethernet/Fast Ethernet driver for
Windows NT

Value 1
  Name:            ErrorControl
  Type:            REG_DWORD
  Data:            0x1

Value 2
  Name:            Group
  Type:            REG_SZ
  Data:            NDIS

Value 3
  Name:            ImagePath
  Type:            REG_EXPAND_SZ
  Data:            System32\DRIVERS\adptsf50.sys

Value 4
  Name:            Start
  Type:            REG_DWORD
  Data:            0x3

Value 5
  Name:            Tag
  Type:            REG_DWORD
  Data:            0xd

Value 6
  Name:            Type
  Type:            REG_DWORD
  Data:            0x1


Key Name:          SYSTEM\CurrentControlSet\Services\ADPTSF\Enum
Class Name:        <NO CLASS>
Last Write Time:   8/23/2000 - 1:34 PM
Value 0
  Name:            0
  Type:            REG_SZ
  Data:
PCI\VEN_9004&DEV_6915&SUBSYS_00109004&REV_03\4&7fe2feb&0&2050

Value 1
  Name:            1
  Type:            REG_SZ
  Data:
PCI\VEN_9004&DEV_6915&SUBSYS_00109004&REV_03\4&7fe2feb&0&2850

Value 2
  Name:            2
  Type:            REG_SZ
  Data:
PCI\VEN_9004&DEV_6915&SUBSYS_00109004&REV_03\4&7907e35&0&2060

Value 3
  Name:            3
  Type:            REG_SZ
  Data:
PCI\VEN_9004&DEV_6915&SUBSYS_00109004&REV_03\4&7907e35&0&2860

Value 4
  Name:            4
  Type:            REG_SZ
  Data:
PCI\VEN_9004&DEV_6915&SUBSYS_00109004&REV_03\4&171f2c55&0&2070

Value 5
  Name:            5
  Type:            REG_SZ
  Data:
PCI\VEN_9004&DEV_6915&SUBSYS_00109004&REV_03\4&171f2c55&0&2870

Value 6
  Name:            6
  Type:            REG_SZ
  Data:
PCI\VEN_9004&DEV_6915&SUBSYS_00109004&REV_03\4&2681c776&0&2080

Value 7
  Name:            7
  Type:            REG_SZ
  Data:
PCI\VEN_9004&DEV_6915&SUBSYS_00109004&REV_03\4&2681c776&0&2880

Value 8
  Name:            Count
  Type:            REG_DWORD
  Data:            0x8
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Value 9
  Name:            NextInstance
  Type:            REG_DWORD
  Data:            0x8


Key Name:          SYSTEM\CurrentControlSet\Services\ADPTSF\Security
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM
Value 0
  Name:            Security
  Type:            REG_BINARY
  Data:
00000000   01 00 14 80 a0 00 00 00 - ac 00 00 00 14 00 00 00
.... ...¬.......
00000010   30 00 00 00 02 00 1c 00 - 01 00 00 00 02 80 14 00
0...............
00000020   ff 01 0f 00 01 01 00 00 - 00 00 00 01 00 00 00 00
ÿ...............
00000030   02 00 70 00 04 00 00 00 - 00 00 18 00 fd 01 02 00
..p.........ý...
00000040   01 01 00 00 00 00 00 05 - 12 00 00 00 74 00 6c 00
............t.l.
00000050   00 00 1c 00 ff 01 0f 00 - 01 02 00 00 00 00 00 05
....ÿ...........
00000060   20 00 00 00 20 02 00 00 - 00 00 00 00 00 00 18 00   ...
...........
00000070   8d 01 02 00 01 01 00 00 - 00 00 00 05 0b 00 00 00
................
00000080   20 02 00 00 00 00 1c 00 - fd 01 02 00 01 02 00 00
.......ý.......
00000090   00 00 00 05 20 00 00 00 - 23 02 00 00 00 00 00 00  ....
...#........
000000a0   01 01 00 00 00 00 00 05 - 12 00 00 00 01 01 00 00
................
000000b0   00 00 00 05 12 00 00 00 -                          ........


Key Name:          SOFTWARE\Microsoft\MSSQLServer
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM

Key Name:          SOFTWARE\Microsoft\MSSQLServer\Client
Class Name:        <NO CLASS>
Last Write Time:   8/18/2000 - 10:36 AM
Value 0
  Name:            SharedMemoryOn
  Type:            REG_DWORD
  Data:            0


Key Name:          SOFTWARE\Microsoft\MSSQLServer\Client\ConnectTo
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM
Value 0
  Name:            DSQUERY
  Type:            REG_SZ
  Data:            DBMSSOCN


Key Name:          SOFTWARE\Microsoft\MSSQLServer\Client\DB-Lib
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM
Value 0
  Name:            AutoAnsiToOem
  Type:            REG_SZ
  Data:            ON

Value 1
  Name:            UseIntlSettings
  Type:            REG_SZ
  Data:            ON


Key Name:
SOFTWARE\Microsoft\MSSQLServer\Client\SuperSocketNetLib
Class Name:        <NO CLASS>
Last Write Time:   8/18/2000 - 10:36 AM
Value 0
  Name:            Encrypt
  Type:            REG_DWORD
  Data:            0

Value 1
  Name:            ProtocolOrder
  Type:            REG_MULTI_SZ
  Data:            tcp
                   np


Key Name:
SOFTWARE\Microsoft\MSSQLServer\Client\SuperSocketNetLib\LastConnect
Class Name:        <NO CLASS>
Last Write Time:   8/11/2000 - 11:14 AM
Value 0
  Name:            h400
  Type:            REG_SZ
  Data:            -1862270968:tcp:h400,1433


Key Name:
SOFTWARE\Microsoft\MSSQLServer\Client\SuperSocketNetLib\Np
Class Name:        <NO CLASS>
Last Write Time:   5/19/2000 - 10:11 AM
Value 0
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
  Name:            DefaultPipe
  Type:            REG_SZ
  Data:            sql\query


Key Name:
SOFTWARE\Microsoft\MSSQLServer\Client\SuperSocketNetLib\Tcp
Class Name:        <NO CLASS>
Last Write Time:   5/19/2000 - 10:11 AM
Value 0
  Name:            DefaultPort
  Type:            REG_DWORD
  Data:            0x599


Key Name:
SOFTWARE\Microsoft\MSSQLServer\Client\SuperSocketNetLib\VIA
Class Name:        <NO CLASS>
Last Write Time:   6/8/2000 - 11:37 AM
Value 0
  Name:            DefaultClientNIC
  Type:            REG_SZ
  Data:            0

Value 1
  Name:            DefaultServerPort
  Type:            REG_SZ
  Data:            0:1433

Value 2
  Name:            RecognizedVendors
  Type:            REG_SZ
  Data:            Giganet, ServerNet II

Value 3
  Name:            Vendor
  Type:            REG_SZ
  Data:


Key Name:          SOFTWARE\Microsoft\MSSQLServer\MSSQLServer
Class Name:        <NO CLASS>
Last Write Time:   6/8/2000 - 11:38 AM
Value 0
  Name:            FullTextDefaultPath
  Type:            REG_SZ
  Data:            .\FTData


Key Name:          SOFTWARE\Microsoft\MSSQLServer\Replication
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM


Key Name:
SOFTWARE\Microsoft\MSSQLServer\Replication\MergeReplicationProvider
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM

Key Name:
SOFTWARE\Microsoft\MSSQLServer\Replication\MergeReplicationProvider\7.0
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM

Key Name:
SOFTWARE\Microsoft\MSSQLServer\Replication\MergeReplicationProvider\7.0\M
sJet
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM
Value 0
  Name:            <NO NAME>
  Type:            REG_SZ
  Data:            {f159cf30-0db4-11d1-b272-00aa00b8de95}


Key Name:          SOFTWARE\Microsoft\MSSQLServer\Setup
Class Name:        <NO CLASS>
Last Write Time:   5/19/2000 - 10:11 AM

Key Name:          SOFTWARE\Microsoft\MSSQLServer\Setup\Resume
Class Name:        <NO CLASS>
Last Write Time:   8/18/2000 - 10:32 AM
Value 0
  Name:            Marker
  Type:            REG_DWORD
  Data:            0x1


Key Name:          SOFTWARE\Microsoft\MSSQLServer\Tracking
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM
Value 0
  Name:            {E07FDDAA-5A21-11d2-9DAD-00C04F79D434}
  Type:            REG_SZ
  Data:

Value 1
  Name:            {E07FDDAC-5A21-11d2-9DAD-00C04F79D434}
  Type:            REG_SZ
  Data:

Value 2
  Name:            {E07FDDAD-5A21-11d2-9DAD-00C04F79D434}
  Type:            REG_SZ
  Data:

Key Name:          SYSTEM\CurrentControlSet\Services\InetInfo
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Class Name:         <NO CLASS>
Last Write Time:    2/25/2000 - 1:49 PM

Key Name:           SYSTEM\CurrentControlSet\Services\InetInfo\Parameters
Class Name:         <NO CLASS>
Last Write Time:    6/13/2000 - 11:34 AM
Value 0
  Name:             DispatchEntries
  Type:             REG_MULTI_SZ
  Data:             LDAPSVC
                    SMTPSVC


Value 1
  Name:             ListenBackLog
  Type:             REG_DWORD
  Data:             0x19

Value 2
  Name:             PoolThreadLimit
  Type:             REG_DWORD
  Data:             0x80

Value 3
  Name:             ThreadTimeout
  Type:             REG_DWORD
  Data:             0x15180


Key Name:           SYSTEM\CurrentControlSet\Services\InetInfo\Performance
Class Name:         <NO CLASS>
Last Write Time:    8/23/2000 - 1:35 PM
Value 0
  Name:             Close
  Type:             REG_SZ
  Data:             CloseINFOPerformanceData

Value 1
  Name:             Collect
  Type:             REG_SZ
  Data:             CollectINFOPerformanceData

Value 2
  Name:             FileSize
  Type:             REG_DWORD
  Data:             0x2510

Value 3
  Name:             FileTime
  Type:             REG_BINARY
  Data:
00000000   10 a5 ad 42 41 3b bf 01 -                          .¥-BA;¿.

Value 4
  Name:             First Counter
  Type:             REG_DWORD
  Data:             0xbb2

Value 5
  Name:             First Help
  Type:             REG_DWORD
  Data:             0xbb3

Value 6
  Name:             Last Counter
  Type:             REG_DWORD
  Data:             0xbf2

Value 7
  Name:             Last Help
  Type:             REG_DWORD
  Data:             0xbf3

Value 8
  Name:             Library
  Type:             REG_SZ
  Data:             infoctrs.dll

Value 9
  Name:             Library Validation Code
  Type:             REG_BINARY
  Data:
00000000   50 ee 8a 71 87 7f bf 01 - 10 25 00 00 00 00 00 00
Pî.q..¿..%......

Value 10
  Name:             Open
  Type:             REG_SZ
  Data:             OpenINFOPerformanceData

Value 11
  Name:             WbemAdapFileSize
  Type:             REG_DWORD
  Data:             0x2510

Value 12
  Name:             WbemAdapFileTime
  Type:             REG_BINARY
  Data:
00000000   00 fe e3 e4 0b f3 bf 01 -                          .þãä.ó¿.

Value 13
  Name:             WbemAdapStatus
  Type:             REG_DWORD
  Data:             0
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Key Name:          SYSTEM\CurrentControlSet\Services\Tcpip\Parameters
Class Name:        Class
Last Write Time:   4/12/2000 - 4:10 PM
Value 0
  Name:            AllowUnqualifiedQuery
  Type:            REG_DWORD
  Data:            0

Value 1
  Name:            DataBasePath
  Type:            REG_EXPAND_SZ
  Data:            %SystemRoot%\System32\drivers\etc

Value 2
  Name:            DeadGWDetectDefault
  Type:            REG_DWORD
  Data:            0x1

Value 3
  Name:            Domain
  Type:            REG_SZ
  Data:

Value 4
  Name:            DontAddDefaultGatewayDefault
  Type:            REG_DWORD
  Data:            0

Value 5
  Name:            EnableICMPRedirect
  Type:            REG_DWORD
  Data:            0x1

Value 6
  Name:            EnableSecurityFilters
  Type:            REG_DWORD
  Data:            0

Value 7
  Name:            ForwardBroadcasts
  Type:            REG_DWORD
  Data:            0

Value 8
  Name:            Hostname
  Type:            REG_SZ
  Data:            WEINROT

Value 9
  Name:            IPEnableRouter
  Type:            REG_DWORD
  Data:            0

Value 10
  Name:            MaxUserPort
  Type:            REG_DWORD
  Data:            0xfffe

Value 11
  Name:            NameServer
  Type:            REG_SZ
  Data:

Value 12
  Name:            NV Hostname
  Type:            REG_SZ
  Data:            WEINROT

Value 13
  Name:            PrioritizeRecordData
  Type:            REG_DWORD
  Data:            0x1

Value 14
  Name:            SearchList
  Type:            REG_SZ
  Data:

Value 15
  Name:            UseDomainNameDevolution
  Type:            REG_DWORD
  Data:            0


Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Adapters
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM

Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Adapters\NdisWanIp
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM
Value 0
  Name:            IpConfig
  Type:            REG_MULTI_SZ
  Data:            Tcpip\Parameters\Interfaces\{43107499-2A1C-4CD0-ACA2-
0A98B1575075}
                   Tcpip\Parameters\Interfaces\{3CBACB77-DB9E-4FBC-AF47-
B6534495A056}


Value 1
  Name:            IpInterfaces
  Type:            REG_BINARY
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
  Data:
00000000   99 74 10 43 1c 2a d0 4c - ac a2 0a 98 b1 57 50 75
.t.C.*ÐL¬¢..±WPu
00000010   77 cb ba 3c 9e db bc 4f - af 47 b6 53 44 95 a0 56
wË°<.Û¼O¯G¶SD. V

Value 2
  Name:            LLInterface
  Type:            REG_SZ
  Data:            WANARP

Value 3
  Name:            NumInterfaces
  Type:            REG_DWORD
  Data:            0x2


Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Adapters\{22952CD6-
289C-4091-AD4A-EE1380D61F22}
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM
Value 0
  Name:            IpConfig
  Type:            REG_MULTI_SZ
  Data:            Tcpip\Parameters\Interfaces\{22952CD6-289C-4091-AD4A-
EE1380D61F22}


Value 1
  Name:            LLInterface
  Type:            REG_SZ
  Data:


Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Adapters\{24BF2DA1-
6F4A-4EF8-82F7-2C8DB7817F52}
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM
Value 0
  Name:            IpConfig
  Type:            REG_MULTI_SZ
  Data:            Tcpip\Parameters\Interfaces\{24BF2DA1-6F4A-4EF8-82F7-
2C8DB7817F52}


Value 1
  Name:            LLInterface
  Type:            REG_SZ
  Data:


Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Adapters\{7629D0B4-
AE77-42EF-AA30-02FE94958788}
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM
Value 0
  Name:            IpConfig
  Type:            REG_MULTI_SZ
  Data:            Tcpip\Parameters\Interfaces\{7629D0B4-AE77-42EF-AA30-
02FE94958788}


Value 1
  Name:            LLInterface
  Type:            REG_SZ
  Data:


Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Adapters\{83865ADC-
A313-40AE-8435-69B3878E6994}
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM
Value 0
  Name:            IpConfig
  Type:            REG_MULTI_SZ
  Data:            Tcpip\Parameters\Interfaces\{83865ADC-A313-40AE-8435-
69B3878E6994}


Value 1
  Name:            LLInterface
  Type:            REG_SZ
  Data:


Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Adapters\{8C0E0FA5-
A0E9-44B8-9574-56CB00B61622}
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM
Value 0
  Name:            IpConfig
  Type:            REG_MULTI_SZ
  Data:            Tcpip\Parameters\Interfaces\{8C0E0FA5-A0E9-44B8-9574-
56CB00B61622}


Value 1
  Name:            LLInterface
  Type:            REG_SZ
  Data:
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Adapters\{9539A88F-
6D41-446C-91D7-3E5A18407CA6}
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM
Value 0
  Name:            IpConfig
  Type:            REG_MULTI_SZ
  Data:            Tcpip\Parameters\Interfaces\{9539A88F-6D41-446C-91D7-
3E5A18407CA6}


Value 1
  Name:            LLInterface
  Type:            REG_SZ
  Data:


Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Adapters\{AD95CD3F-
D209-4F78-8AD3-167D84FE9B25}
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM
Value 0
  Name:            IpConfig
  Type:            REG_MULTI_SZ
  Data:            Tcpip\Parameters\Interfaces\{AD95CD3F-D209-4F78-8AD3-
167D84FE9B25}


Value 1
  Name:            LLInterface
  Type:            REG_SZ
  Data:


Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Adapters\{AED5412A-
08F2-459B-BC77-DD710DC51898}
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM
Value 0
  Name:            IpConfig
  Type:            REG_MULTI_SZ
  Data:            Tcpip\Parameters\Interfaces\{AED5412A-08F2-459B-BC77-
DD710DC51898}


Value 1
  Name:            LLInterface
  Type:            REG_SZ
  Data:


Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Adapters\{FAEE3419-
626D-4AB7-828D-EE0929036888}
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM
Value 0
  Name:            IpConfig
  Type:            REG_MULTI_SZ
  Data:            Tcpip\Parameters\Interfaces\{FAEE3419-626D-4AB7-828D-
EE0929036888}


Value 1
  Name:            LLInterface
  Type:            REG_SZ
  Data:


Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\DNSRegisteredAdapters
Class Name:        DynDnsRootClass
Last Write Time:   2/25/2000 - 1:49 PM

Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM

Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\{22952CD6-
289C-4091-AD4A-EE1380D61F22}
Class Name:        <NO CLASS>
Last Write Time:   5/18/2000 - 4:20 PM
Value 0
  Name:            DefaultGateway
  Type:            REG_MULTI_SZ
  Data:

Value 1
  Name:            DefaultGatewayMetric
  Type:            REG_MULTI_SZ
  Data:

Value 2
  Name:            DisableDynamicUpdate
  Type:            REG_DWORD
  Data:            0x1

Value 3
  Name:            Domain
  Type:            REG_SZ
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
  Data:

Value 4
  Name:             EnableAdapterDomainNameRegistration
  Type:             REG_DWORD
  Data:             0

Value 5
  Name:             EnableDeadGWDetect
  Type:             REG_DWORD
  Data:             0x1

Value 6
  Name:             EnableDHCP
  Type:             REG_DWORD
  Data:             0

Value 7
  Name:             InterfaceMetric
  Type:             REG_DWORD
  Data:             0x1

Value 8
  Name:             IPAddress
  Type:             REG_MULTI_SZ
  Data:             129.103.154.2


Value 9
  Name:             NameServer
  Type:             REG_SZ
  Data:

Value 10
  Name:             NTEContextList
  Type:             REG_MULTI_SZ
  Data:             0x00000006


Value 11
  Name:             RawIPAllowedProtocols
  Type:             REG_MULTI_SZ
  Data:             0


Value 12
  Name:             SubnetMask
  Type:             REG_MULTI_SZ
  Data:             255.255.255.0


Value 13
  Name:             TCPAllowedPorts
  Type:             REG_MULTI_SZ
  Data:             0


Value 14
  Name:             UDPAllowedPorts
  Type:             REG_MULTI_SZ
  Data:             0


Value 15
  Name:             UseZeroBroadcast
  Type:             REG_DWORD
  Data:             0


Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\{24BF2DA1-
6F4A-4EF8-82F7-2C8DB7817F52}
Class Name:         <NO CLASS>
Last Write Time:    5/9/2000 - 1:59 PM
Value 0
  Name:             DefaultGateway
  Type:             REG_MULTI_SZ
  Data:

Value 1
  Name:             DefaultGatewayMetric
  Type:             REG_MULTI_SZ
  Data:

Value 2
  Name:             DhcpIPAddress
  Type:             REG_SZ
  Data:             0.0.0.0

Value 3
  Name:             DhcpSubnetMask
  Type:             REG_SZ
  Data:             255.0.0.0

Value 4
  Name:             DisableDynamicUpdate
  Type:             REG_DWORD
  Data:             0

Value 5
  Name:             Domain
  Type:             REG_SZ
  Data:

Value 6
  Name:             EnableAdapterDomainNameRegistration
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
  Type:            REG_DWORD
  Data:            0

Value 7
  Name:            EnableDeadGWDetect
  Type:            REG_DWORD
  Data:            0x1

Value 8
  Name:            EnableDHCP
  Type:            REG_DWORD
  Data:            0x1

Value 9
  Name:            InterfaceMetric
  Type:            REG_DWORD
  Data:            0x1

Value 10
  Name:            IPAddress
  Type:            REG_MULTI_SZ
  Data:            0.0.0.0


Value 11
  Name:            NameServer
  Type:            REG_SZ
  Data:

Value 12
  Name:            NTEContextList
  Type:            REG_MULTI_SZ
  Data:

Value 13
  Name:            RawIPAllowedProtocols
  Type:            REG_MULTI_SZ
  Data:            0


Value 14
  Name:            SubnetMask
  Type:            REG_MULTI_SZ
  Data:            0.0.0.0


Value 15
  Name:            TCPAllowedPorts
  Type:            REG_MULTI_SZ
  Data:            0


Value 16
  Name:            UDPAllowedPorts
  Type:            REG_MULTI_SZ
  Data:            0


Value 17
  Name:            UseZeroBroadcast
  Type:            REG_DWORD
  Data:            0


Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\{3CBACB77-
DB9E-4FBC-AF47-B6534495A056}
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:55 PM
Value 0
  Name:            DefaultGateway
  Type:            REG_MULTI_SZ
  Data:

Value 1
  Name:            DontAddDefaultGateway
  Type:            REG_DWORD
  Data:            0

Value 2
  Name:            EnableDeadGWDetect
  Type:            REG_DWORD
  Data:            0x1

Value 3
  Name:            EnableDHCP
  Type:            REG_DWORD
  Data:            0

Value 4
  Name:            IPAddress
  Type:            REG_MULTI_SZ
  Data:            0.0.0.0


Value 5
  Name:            SubnetMask
  Type:            REG_MULTI_SZ
  Data:            0.0.0.0


Value 6
  Name:            UseZeroBroadcast
  Type:            REG_DWORD
  Data:            0
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\{43107499-
2A1C-4CD0-ACA2-0A98B1575075}
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:55 PM
Value 0
  Name:            DefaultGateway
  Type:            REG_MULTI_SZ
  Data:

Value 1
  Name:            DontAddDefaultGateway
  Type:            REG_DWORD
  Data:            0

Value 2
  Name:            EnableDeadGWDetect
  Type:            REG_DWORD
  Data:            0x1

Value 3
  Name:            EnableDHCP
  Type:            REG_DWORD
  Data:            0

Value 4
  Name:            IPAddress
  Type:            REG_MULTI_SZ
  Data:            0.0.0.0


Value 5
  Name:            SubnetMask
  Type:            REG_MULTI_SZ
  Data:            0.0.0.0


Value 6
  Name:            UseZeroBroadcast
  Type:            REG_DWORD
  Data:            0


Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\{7629D0B4-
AE77-42EF-AA30-02FE94958788}
Class Name:        <NO CLASS>
Last Write Time:   5/18/2000 - 4:20 PM
Value 0
  Name:            DefaultGateway
  Type:            REG_MULTI_SZ
  Data:

Value 1
  Name:            DefaultGatewayMetric
  Type:            REG_MULTI_SZ
  Data:

Value 2
  Name:            DisableDynamicUpdate
  Type:            REG_DWORD
  Data:            0x1

Value 3
  Name:            Domain
  Type:            REG_SZ
  Data:

Value 4
  Name:            EnableAdapterDomainNameRegistration
  Type:            REG_DWORD
  Data:            0

Value 5
  Name:            EnableDeadGWDetect
  Type:            REG_DWORD
  Data:            0x1

Value 6
  Name:            EnableDHCP
  Type:            REG_DWORD
  Data:            0

Value 7
  Name:            InterfaceMetric
  Type:            REG_DWORD
  Data:            0x1

Value 8
  Name:            IPAddress
  Type:            REG_MULTI_SZ
  Data:            129.103.155.2


Value 9
  Name:            NameServer
  Type:            REG_SZ
  Data:

Value 10
  Name:            NTEContextList
  Type:            REG_MULTI_SZ
  Data:            0x00000009
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Value 11
  Name:            RawIPAllowedProtocols
  Type:            REG_MULTI_SZ
  Data:            0


Value 12
  Name:            SubnetMask
  Type:            REG_MULTI_SZ
  Data:            255.255.255.0


Value 13
  Name:            TCPAllowedPorts
  Type:            REG_MULTI_SZ
  Data:            0


Value 14
  Name:            UDPAllowedPorts
  Type:            REG_MULTI_SZ
  Data:            0


Value 15
  Name:            UseZeroBroadcast
  Type:            REG_DWORD
  Data:            0


Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\{83865ADC-
A313-40AE-8435-69B3878E6994}
Class Name:        <NO CLASS>
Last Write Time:   5/18/2000 - 4:20 PM
Value 0
  Name:            DefaultGateway
  Type:            REG_MULTI_SZ
  Data:

Value 1
  Name:            DefaultGatewayMetric
  Type:            REG_MULTI_SZ
  Data:

Value 2
  Name:            DisableDynamicUpdate
  Type:            REG_DWORD
  Data:            0x1

Value 3
  Name:            Domain
  Type:            REG_SZ
  Data:

Value 4
  Name:            EnableAdapterDomainNameRegistration
  Type:            REG_DWORD
  Data:            0

Value 5
  Name:            EnableDeadGWDetect
  Type:            REG_DWORD
  Data:            0x1

Value 6
  Name:            EnableDHCP
  Type:            REG_DWORD
  Data:            0

Value 7
  Name:            InterfaceMetric
  Type:            REG_DWORD
  Data:            0x1

Value 8
  Name:            IPAddress
  Type:            REG_MULTI_SZ
  Data:            129.103.150.2


Value 9
  Name:            NameServer
  Type:            REG_SZ
  Data:

Value 10
  Name:            NTEContextList
  Type:            REG_MULTI_SZ
  Data:            0x00000004


Value 11
  Name:            RawIPAllowedProtocols
  Type:            REG_MULTI_SZ
  Data:            0


Value 12
  Name:            SubnetMask
  Type:            REG_MULTI_SZ
  Data:            255.255.255.0


Value 13
  Name:            TCPAllowedPorts
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
  Type:            REG_MULTI_SZ
  Data:            0


Value 14
  Name:            UDPAllowedPorts
  Type:            REG_MULTI_SZ
  Data:            0


Value 15
  Name:            UseZeroBroadcast
  Type:            REG_DWORD
  Data:            0


Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\{8C0E0FA5-
A0E9-44B8-9574-56CB00B61622}
Class Name:        <NO CLASS>
Last Write Time:   5/18/2000 - 4:20 PM
Value 0
  Name:            DefaultGateway
  Type:            REG_MULTI_SZ
  Data:

Value 1
  Name:            DefaultGatewayMetric
  Type:            REG_MULTI_SZ
  Data:

Value 2
  Name:            DisableDynamicUpdate
  Type:            REG_DWORD
  Data:            0x1

Value 3
  Name:            Domain
  Type:            REG_SZ
  Data:

Value 4
  Name:            EnableAdapterDomainNameRegistration
  Type:            REG_DWORD
  Data:            0

Value 5
  Name:            EnableDeadGWDetect
  Type:            REG_DWORD
  Data:            0x1

Value 6
  Name:            EnableDHCP
  Type:            REG_DWORD
  Data:            0

Value 7
  Name:            InterfaceMetric
  Type:            REG_DWORD
  Data:            0x1

Value 8
  Name:            IPAddress
  Type:            REG_MULTI_SZ
  Data:            129.103.156.2


Value 9
  Name:            NameServer
  Type:            REG_SZ
  Data:

Value 10
  Name:            NTEContextList
  Type:            REG_MULTI_SZ
  Data:            0x00000008


Value 11
  Name:            RawIPAllowedProtocols
  Type:            REG_MULTI_SZ
  Data:            0


Value 12
  Name:            SubnetMask
  Type:            REG_MULTI_SZ
  Data:            255.255.255.0


Value 13
  Name:            TCPAllowedPorts
  Type:            REG_MULTI_SZ
  Data:            0


Value 14
  Name:            UDPAllowedPorts
  Type:            REG_MULTI_SZ
  Data:            0


Value 15
  Name:            UseZeroBroadcast
  Type:            REG_DWORD
  Data:            0
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\{9539A88F-
6D41-446C-91D7-3E5A18407CA6}
Class Name:         <NO CLASS>
Last Write Time:    5/18/2000 - 4:20 PM
Value 0
  Name:             DefaultGateway
  Type:             REG_MULTI_SZ
  Data:

Value 1
  Name:             DefaultGatewayMetric
  Type:             REG_MULTI_SZ
  Data:

Value 2
  Name:             DisableDynamicUpdate
  Type:             REG_DWORD
  Data:             0x1

Value 3
  Name:             Domain
  Type:             REG_SZ
  Data:

Value 4
  Name:             EnableAdapterDomainNameRegistration
  Type:             REG_DWORD
  Data:             0

Value 5
  Name:             EnableDeadGWDetect
  Type:             REG_DWORD
  Data:             0x1

Value 6
  Name:             EnableDHCP
  Type:             REG_DWORD
  Data:             0

Value 7
  Name:             InterfaceMetric
  Type:             REG_DWORD
  Data:             0x1

Value 8
  Name:             IPAddress
  Type:             REG_MULTI_SZ
  Data:             129.103.181.131

Value 9
  Name:             NameServer
  Type:             REG_SZ
  Data:

Value 10
  Name:             NTEContextList
  Type:             REG_MULTI_SZ
  Data:             0x00000005

Value 11
  Name:             RawIPAllowedProtocols
  Type:             REG_MULTI_SZ
  Data:             0

Value 12
  Name:             SubnetMask
  Type:             REG_MULTI_SZ
  Data:             255.255.255.0

Value 13
  Name:             TCPAllowedPorts
  Type:             REG_MULTI_SZ
  Data:             0

Value 14
  Name:             UDPAllowedPorts
  Type:             REG_MULTI_SZ
  Data:             0

Value 15
  Name:             UseZeroBroadcast
  Type:             REG_DWORD
  Data:             0

Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\{AD95CD3F-
D209-4F78-8AD3-167D84FE9B25}
Class Name:         <NO CLASS>
Last Write Time:    5/18/2000 - 4:20 PM
Value 0
  Name:             DefaultGateway
  Type:             REG_MULTI_SZ
  Data:

Value 1
  Name:             DefaultGatewayMetric
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
  Type:            REG_MULTI_SZ
  Data:

Value 2
  Name:            DisableDynamicUpdate
  Type:            REG_DWORD
  Data:            0x1

Value 3
  Name:            Domain
  Type:            REG_SZ
  Data:

Value 4
  Name:            EnableAdapterDomainNameRegistration
  Type:            REG_DWORD
  Data:            0

Value 5
  Name:            EnableDeadGWDetect
  Type:            REG_DWORD
  Data:            0x1

Value 6
  Name:            EnableDHCP
  Type:            REG_DWORD
  Data:            0

Value 7
  Name:            InterfaceMetric
  Type:            REG_DWORD
  Data:            0x1

Value 8
  Name:            IPAddress
  Type:            REG_MULTI_SZ
  Data:            129.103.152.2


Value 9
  Name:            NameServer
  Type:            REG_SZ
  Data:

Value 10
  Name:            NTEContextList
  Type:            REG_MULTI_SZ
  Data:            0x00000002


Value 11
  Name:            RawIPAllowedProtocols
  Type:            REG_MULTI_SZ
  Data:            0


Value 12
  Name:            SubnetMask
  Type:            REG_MULTI_SZ
  Data:            255.255.255.0


Value 13
  Name:            TCPAllowedPorts
  Type:            REG_MULTI_SZ
  Data:            0


Value 14
  Name:            UDPAllowedPorts
  Type:            REG_MULTI_SZ
  Data:            0


Value 15
  Name:            UseZeroBroadcast
  Type:            REG_DWORD
  Data:            0


Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\{AED5412A-
08F2-459B-BC77-DD710DC51898}
Class Name:        <NO CLASS>
Last Write Time:   5/18/2000 - 4:20 PM
Value 0
  Name:            DefaultGateway
  Type:            REG_MULTI_SZ
  Data:

Value 1
  Name:            DefaultGatewayMetric
  Type:            REG_MULTI_SZ
  Data:

Value 2
  Name:            DisableDynamicUpdate
  Type:            REG_DWORD
  Data:            0x1

Value 3
  Name:            Domain
  Type:            REG_SZ
  Data:

Value 4
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
  Name:            EnableAdapterDomainNameRegistration
  Type:            REG_DWORD
  Data:            0

Value 5
  Name:            EnableDeadGWDetect
  Type:            REG_DWORD
  Data:            0x1

Value 6
  Name:            EnableDHCP
  Type:            REG_DWORD
  Data:            0

Value 7
  Name:            InterfaceMetric
  Type:            REG_DWORD
  Data:            0x1

Value 8
  Name:            IPAddress
  Type:            REG_MULTI_SZ
  Data:            129.103.153.2


Value 9
  Name:            NameServer
  Type:            REG_SZ
  Data:

Value 10
  Name:            NTEContextList
  Type:            REG_MULTI_SZ
  Data:            0x00000007


Value 11
  Name:            RawIPAllowedProtocols
  Type:            REG_MULTI_SZ
  Data:            0


Value 12
  Name:            SubnetMask
  Type:            REG_MULTI_SZ
  Data:            255.255.255.0


Value 13
  Name:            TCPAllowedPorts
  Type:            REG_MULTI_SZ
  Data:            0


Value 14
  Name:            UDPAllowedPorts
  Type:            REG_MULTI_SZ
  Data:            0


Value 15
  Name:            UseZeroBroadcast
  Type:            REG_DWORD
  Data:            0


Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\{FAEE3419-
626D-4AB7-828D-EE0929036888}
Class Name:        <NO CLASS>
Last Write Time:   5/18/2000 - 4:20 PM
Value 0
  Name:            DefaultGateway
  Type:            REG_MULTI_SZ
  Data:

Value 1
  Name:            DefaultGatewayMetric
  Type:            REG_MULTI_SZ
  Data:

Value 2
  Name:            DisableDynamicUpdate
  Type:            REG_DWORD
  Data:            0x1

Value 3
  Name:            Domain
  Type:            REG_SZ
  Data:

Value 4
  Name:            EnableAdapterDomainNameRegistration
  Type:            REG_DWORD
  Data:            0

Value 5
  Name:            EnableDeadGWDetect
  Type:            REG_DWORD
  Data:            0x1

Value 6
  Name:            EnableDHCP
  Type:            REG_DWORD
  Data:            0
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Value 7
  Name:             InterfaceMetric
  Type:             REG_DWORD
  Data:             0x1

Value 8
  Name:             IPAddress
  Type:             REG_MULTI_SZ
  Data:             129.103.151.2


Value 9
  Name:             NameServer
  Type:             REG_SZ
  Data:

Value 10
  Name:             NTEContextList
  Type:             REG_MULTI_SZ
  Data:             0x00000003


Value 11
  Name:             RawIPAllowedProtocols
  Type:             REG_MULTI_SZ
  Data:             0


Value 12
  Name:             SubnetMask
  Type:             REG_MULTI_SZ
  Data:             255.255.255.0


Value 13
  Name:             TCPAllowedPorts
  Type:             REG_MULTI_SZ
  Data:             0


Value 14
  Name:             UDPAllowedPorts
  Type:             REG_MULTI_SZ
  Data:             0


Value 15
  Name:             UseZeroBroadcast
  Type:             REG_DWORD
  Data:             0
```

```
Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\PersistentRoutes
Class Name:         <NO CLASS>
Last Write Time:    2/25/2000 - 1:49 PM

Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Winsock
Class Name:         <NO CLASS>
Last Write Time:    2/25/2000 - 1:49 PM
Value 0
  Name:             HelperDllName
  Type:             REG_EXPAND_SZ
  Data:             %SystemRoot%\System32\wshtcpip.dll

Value 1
  Name:             Mapping
  Type:             REG_BINARY
  Data:
00000000   0b 00 00 00 03 00 00 00 - 02 00 00 00 01 00 00 00
................
00000010   06 00 00 00 02 00 00 00 - 01 00 00 00 00 00 00 00
................
00000020   02 00 00 00 00 00 00 00 - 06 00 00 00 00 00 00 00
................
00000030   00 00 00 00 06 00 00 00 - 00 00 00 00 01 00 00 00
................
00000040   06 00 00 00 02 00 00 00 - 02 00 00 00 11 00 00 00
................
00000050   02 00 00 00 02 00 00 00 - 00 00 00 00 02 00 00 00
................
00000060   00 00 00 00 11 00 00 00 - 00 00 00 00 00 00 00 00
................
00000070   11 00 00 00 00 00 00 00 - 02 00 00 00 11 00 00 00
................
00000080   02 00 00 00 03 00 00 00 - 00 00 00 00
............

Value 2
  Name:             MaxSockAddrLength
  Type:             REG_DWORD
  Data:             0x10

Value 3
  Name:             MinSockAddrLength
  Type:             REG_DWORD
  Data:             0x10

Value 4
  Name:             UseDelayedAcceptance
  Type:             REG_DWORD
  Data:             0
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Key Name:          SYSTEM\CurrentControlSet\Services\Tcpip\Performance
Class Name:        <NO CLASS>
Last Write Time:   8/23/2000 - 1:36 PM
Value 0
  Name:            Close
  Type:            REG_SZ
  Data:            CloseTcpIpPerformanceData

Value 1
  Name:            Collect
  Type:            REG_SZ
  Data:            CollectTcpIpPerformanceData

Value 2
  Name:            FileSize
  Type:            REG_DWORD
  Data:            0xa310

Value 3
  Name:            FileTime
  Type:            REG_BINARY
  Data:
00000000   00 80 bf 6c 1f fb be 01 -                              ..¿l.û¾.

Value 4
  Name:            Library
  Type:            REG_SZ
  Data:            Perfctrs.dll

Value 5
  Name:            Open
  Type:            REG_SZ
  Data:            OpenTcpIpPerformanceData

Value 6
  Name:            WbemAdapFileSize
  Type:            REG_DWORD
  Data:            0xa310

Value 7
  Name:            WbemAdapFileTime
  Type:            REG_BINARY
  Data:
00000000   00 60 4e 96 aa 40 bf 01 -                              .`N.ª@¿.

Value 8
  Name:            WbemAdapStatus
  Type:            REG_DWORD
  Data:            0


Key Name:          SYSTEM\CurrentControlSet\Services\Tcpip\Security
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM
Value 0
  Name:            Security
  Type:            REG_BINARY
  Data:
00000000   01 00 14 80 a0 00 00 00 - ac 00 00 00 14 00 00 00
.... ...¬.......
00000010   30 00 00 00 02 00 1c 00 - 01 00 00 00 02 80 14 00
0...............
00000020   ff 01 0f 00 01 01 00 00 - 00 00 00 01 00 00 00 00
ÿ...............
00000030   02 00 70 00 04 00 00 00 - 00 00 18 00 fd 01 02 00
..p.........ý...
00000040   01 01 00 00 00 00 00 05 - 12 00 00 00 02 00 00 00
................
00000050   00 00 1c 00 ff 01 0f 00 - 01 02 00 00 00 00 00 05
....ÿ...........
00000060   20 00 00 00 20 02 00 00 - 03 00 00 00 00 00 18 00   ...
...........
00000070   8d 01 02 00 01 01 00 00 - 00 00 00 05 0b 00 00 00
................
00000080   20 02 00 00 00 00 1c 00 - fd 01 02 00 01 02 00 00
.......ý.......
00000090   00 00 00 05 20 00 00 00 - 23 02 00 00 03 00 00 00  ....
...#........
000000a0   01 01 00 00 00 00 00 05 - 12 00 00 00 01 01 00 00
................
000000b0   00 00 00 05 12 00 00 00 -                           ........


Key Name:
SYSTEM\CurrentControlSet\Services\Tcpip\ServiceProvider
Class Name:        <NO CLASS>
Last Write Time:   2/25/2000 - 1:49 PM
Value 0
  Name:            Class
  Type:            REG_DWORD
  Data:            0x8

Value 1
  Name:            DnsPriority
  Type:            REG_DWORD
  Data:            0x7d0

Value 2
  Name:            HostsPriority
  Type:            REG_DWORD
  Data:            0x1f4

Value 3
  Name:            LocalPriority
  Type:            REG_DWORD
  Data:            0x1f3
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Value 4
  Name:             Name
  Type:             REG_SZ
  Data:             TCP/IP

Value 5
  Name:             NetbtPriority
  Type:             REG_DWORD
  Data:             0x7d1

Value 6
  Name:             ProviderPath
  Type:             REG_EXPAND_SZ
  Data:             %SystemRoot%\System32\wsock32.dll


Key Name:           SOFTWARE\Microsoft\TPCC
Class Name:         <NO CLASS>
Last Write Time:    8/11/2000 - 11:16 AM
Value 0
  Name:             COM_SinglePool
  Type:             REG_SZ
  Data:             YES

Value 1
  Name:             DB_Protocol
  Type:             REG_SZ
  Data:             DBLIB

Value 2
  Name:             DbName
  Type:             REG_SZ
  Data:             tpcc

Value 3
  Name:             DbPassword
  Type:             REG_SZ
  Data:

Value 4
  Name:             DbServer
  Type:             REG_SZ
  Data:             h400

Value 5
  Name:             DbUser
  Type:             REG_SZ
  Data:             sa

Value 6
  Name:             MaxConnections
  Type:             REG_DWORD
  Data:             0x1b58

Value 7
  Name:             MaxPendingDeliveries
  Type:             REG_DWORD
  Data:             0x3e8

Value 8
  Name:             NumberOfDeliveryThreads
  Type:             REG_DWORD
  Data:             0x4

Value 9
  Name:             Path
  Type:             REG_SZ
  Data:             c:\inetpub\wwwroot\

Value 10
  Name:             TxnMonitor
  Type:             REG_SZ
  Data:             COM


Key Name:           SYSTEM\CurrentControlSet\Services\W3SVC\Parameters
Class Name:         <NO CLASS>
Last Write Time:    6/13/2000 - 11:34 AM
Value 0
  Name:             AcceptExOutstanding
  Type:             REG_DWORD
  Data:             0x28

Value 1
  Name:             AccessDeniedMessage
  Type:             REG_SZ
  Data:             Error: Access is Denied.

Value 2
  Name:             CertMapList
  Type:             REG_SZ
  Data:             C:\WINNT\System32\inetsrv\iiscrmap.dll

Value 3
  Name:             Filter DLLs
  Type:             REG_SZ
  Data:

Value 4
  Name:             InstallPath
  Type:             REG_SZ
  Data:             C:\WINNT\System32\inetsrv

Value 5
  Name:             LogFileDirectory
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
  Type:            REG_SZ
  Data:            C:\WINNT\System32\LogFiles

Value 6
  Name:            MajorVersion
  Type:            REG_DWORD
  Data:            0x5

Value 7
  Name:            MinorVersion
  Type:            REG_DWORD
  Data:            0
```

Component Services Configuration:
COM+ Component TPCC.AllTXns Settings:

Transactions not supported
Enable object pooling
Minimum pool size 37
Maximum pool size 37
Creation timeout 60,000
Enable object construction
Enable just in time activation
Concurrency required

This section discloses the RTE parameters used on the PRIMERGY 870-40 system.

```
Profile:       HTML_970user_faster
File Path:     D:\benchcrf\N400\HTML_970user_faster.pro
Version:       1.0.1

       Number of Engines: 16

              Name: DRIVER29
              Description: Grau0
              Directory: d:\log_grau0.log
              Machine: schwarz
              Parameter Set: All_Times
              Index: 800000000
              Seed: 39578
              Configured Users: 1940
              Pipe Name: DRIVER29241052125
              Connect Rate: 400
              Start Rate: 400
              CLIENT_NURAND: 233
              CPU: 0

              Name: DRIVER30
              Description: Grau1
              Directory: d:\log_grau1.log
              Machine: schwarz
              Parameter Set: All_Times
              Index: 850000000
              Seed: 39578
              Configured Users: 1940
              Pipe Name: DRIVER30241091984
              Connect Rate: 400
              Start Rate: 400
              CLIENT_NURAND: 233
              CPU: 1

              Name: DRIVER31
              Description: Grau2
              Directory: d:\log_grau2.log
              Machine: schwarz
              Parameter Set: All_Times
              Index: 900000000
              Seed: 39578
              Configured Users: 1940
              Pipe Name: DRIVER31241125500
              Connect Rate: 350
              Start Rate: 350
              CLIENT_NURAND: 233
              CPU: 2

              Name: DRIVER32
              Description: Grau3
              Directory: d:\log_grau3.log
              Machine: schwarz
              Parameter Set: All_Times
              Index: 950000000
              Seed: 39578
              Configured Users: 970
              Pipe Name: DRIVER32241158296
              Connect Rate: 150
              Start Rate: 150
              CLIENT_NURAND: 233
              CPU: 3

              Name: DRIVER36
              Description: Weinrot0
              Directory: c:\log_weinrot0.log
              Machine: raccon
              Parameter Set: All_Times
              Index: 1000000000
              Seed: 39578
              Configured Users: 1940
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Pipe Name: DRIVER3632791734
Connect Rate: 400
Start Rate: 400
CLIENT_NURAND: 233
CPU: 0

Name: DRIVER37
Description: Weinrot1
Directory: c:\log_weinrot1.log
Machine: raccon
Parameter Set: All_Times
Index: 1050000000
Seed: 39578
Configured Users: 1940
Pipe Name: DRIVER3733118687
Connect Rate: 400
Start Rate: 400
CLIENT_NURAND: 233
CPU: 1

Name: DRIVER38
Description: Weinrot2
Directory: c:\log_weinrot2.log
Machine: raccon
Parameter Set: All_Times
Index: 1100000000
Seed: 39578
Configured Users: 1940
Pipe Name: DRIVER3833156671
Connect Rate: 350
Start Rate: 350
CLIENT_NURAND: 233
CPU: 2

Name: DRIVER39
Description: Weinrot3
Directory: c:\log_weinrot3.log
Machine: raccon
Parameter Set: All_Times
Index: 1150000000
Seed: 39578
Configured Users: 970
Pipe Name: DRIVER3933195937
Connect Rate: 150
Start Rate: 150
CLIENT_NURAND: 233
CPU: 3

Name: DRIVER43
Description: Rot0
Directory: c:\log_rot0.log
Machine: raccon
Parameter Set: All_Times
Index: 1200000000
Seed: 39578
Configured Users: 1940
Pipe Name: DRIVER4333451828
Connect Rate: 400
Start Rate: 400
CLIENT_NURAND: 233
CPU: 0

Name: DRIVER44
Description: Rot1
Directory: c:\log_rot1.log
Machine: raccon
Parameter Set: All_Times
Index: 1250000000
Seed: 39578
Configured Users: 1940
Pipe Name: DRIVER4433532421
Connect Rate: 400
Start Rate: 400
CLIENT_NURAND: 233
CPU: 1

Name: DRIVER45
Description: Rot2
Directory: c:\log_rot2.log
Machine: raccon
Parameter Set: All_Times
Index: 1300000000
Seed: 39578
Configured Users: 1940
Pipe Name: DRIVER4533559046
Connect Rate: 350
Start Rate: 350
CLIENT_NURAND: 233
CPU: 3

Name: DRIVER46
Description: Rot3
Directory: c:\log_log3.log
Machine: raccon
Parameter Set: All_Times
Index: 1350000000
Seed: 39578
Configured Users: 970
Pipe Name: DRIVER4633591437
Connect Rate: 150
Start Rate: 150
CLIENT_NURAND: 233
CPU: 2

Name: DRIVER50
Description: Hellblau0
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Directory: c:\log_hellblau0.log
Machine: P870
Parameter Set: All_Times
Index: 1400000000
Seed: 39578
Configured Users: 1940
Pipe Name: DRIVER5035901359
Connect Rate: 400
Start Rate: 400
CLIENT_NURAND: 233
CPU: 0

Name: DRIVER51
Description: Hellblau1
Directory: c:\log_hellblau1.log
Machine: P870
Parameter Set: All_Times
Index: 1450000000
Seed: 39578
Configured Users: 1940
Pipe Name: DRIVER5135970250
Connect Rate: 400
Start Rate: 400
CLIENT_NURAND: 233
CPU: 1

Name: DRIVER52
Description: Hellblau2
Directory: c:\log_hellblau2.log
Machine: P870
Parameter Set: All_Times
Index: 1500000000
Seed: 39578
Configured Users: 1940
Pipe Name: DRIVER5236011546
Connect Rate: 350
Start Rate: 350
CLIENT_NURAND: 233
CPU: 2

Name: DRIVER53
Description: Hellblau3
Directory: c:\log_hellblau3.log
Machine: P870
Parameter Set: All_Times
Index: 1550000000
Seed: 39578
Configured Users: 970
Pipe Name: DRIVER5336051578
Connect Rate: 150
Start Rate: 150
CLIENT_NURAND: 233
CPU: 3
```

```
Number of User groups: 28

        Driver Engine: DRIVER29
        IIS Server: grau0
        SQL Server: h400
        User: sa
        Protocol: Html
        w_id Range: 1 - 97
        w_id Max Warehouse: 2716
        Scale: Normal
        User Count: 970
        District id: 1
        Scale Down: No

        Driver Engine: DRIVER29
        IIS Server: grau1
        SQL Server: h400
        User: sa
        Protocol: Html
        w_id Range: 98 - 194
        w_id Max Warehouse: 2716
        Scale: Normal
        User Count: 970
        District id: 1
        Scale Down: No

        Driver Engine: DRIVER30
        IIS Server: grau2
        SQL Server: h400
        User: sa
        Protocol: Html
        w_id Range: 195 - 291
        w_id Max Warehouse: 2716
        Scale: Normal
        User Count: 970
        District id: 1
        Scale Down: No

        Driver Engine: DRIVER30
        IIS Server: grau3
        SQL Server: h400
        User: sa
        Protocol: Html
        w_id Range: 292 - 388
        w_id Max Warehouse: 2716
        Scale: Normal
        User Count: 970
        District id: 1
        Scale Down: No

        Driver Engine: DRIVER31
        IIS Server: grau4
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 389 - 485
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER31
IIS Server: grau5
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 486 - 582
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER32
IIS Server: grau6
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 583 - 679
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER36
IIS Server: weinrot0
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 680 - 776
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER36
IIS Server: weinrot1
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 777 - 873
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER37
IIS Server: weinrot2
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 874 - 970
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER37
IIS Server: weinrot3
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 971 - 1067
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER38
IIS Server: weinrot4
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 1068 - 1164
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER38
IIS Server: weinrot5
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 1165 - 1261
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Driver Engine: DRIVER39
IIS Server: weinrot6
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 1262 - 1358
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER43
IIS Server: rot0
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 1359 - 1455
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER43
IIS Server: rot1
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 1456 - 1552
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER44
IIS Server: rot2
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 1553 - 1649
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER44
IIS Server: rot3
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 1650 - 1746
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER45
IIS Server: rot4
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 1747 - 1843
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER45
IIS Server: rot5
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 1844 - 1940
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER46
IIS Server: rot6
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 1941 - 2037
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER50
IIS Server: hellblau0
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 2038 - 2134
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
Scale Down: No

Driver Engine: DRIVER50
IIS Server: hellblau1
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 2135 - 2231
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER51
IIS Server: hellblau2
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 2232 - 2328
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER51
IIS Server: hellblau3
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 2329 - 2425
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER52
IIS Server: hellblau4
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 2426 - 2522
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER52
IIS Server: hellblau5
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 2523 - 2619
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Driver Engine: DRIVER53
IIS Server: hellblau6
SQL Server: h400
User: sa
Protocol: Html
w_id Range: 2620 - 2716
w_id Max Warehouse: 2716
Scale: Normal
User Count: 970
District id: 1
Scale Down: No

Number of Parameter Sets: 2


~Default
Default Parameter Set
                Txn     Think   Key     RT      RT      Menu
                Weight  Time    Time    Delay   Fence   Delay
New Order         10.00         12.05   18.01   0.10    5.00  0.10
Payment           10.00         12.05    3.01   0.10    5.00  0.10
Delivery           1.00          5.05    2.01   0.10    5.00  0.10
Stock Level        1.00          5.05    2.01   0.10   20.00  0.10
Order Status       1.00         10.05    2.01   0.10    5.00  0.10

All_Times
HTML Param. Set
                Txn     Think   Key     RT      RT      Menu
                Weight  Time    Time    Delay   Fence   Delay
New Order         44.79         12.05   18.01   0.10    5.00  0.10
Payment           43.06         12.05    3.01   0.10    5.00  0.10
Delivery           4.05          5.05    2.01   0.10    5.00  0.10
Stock Level        4.05          5.05    2.01   0.10   20.00  0.10
Order Status       4.05         10.05    2.01   0.10    5.00  0.10
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

This section discloses the Microsoft SQL Server 2000 Enterprise Edition parameters used on the PRIMERGY N400 server system.

```
Microsoft SQL Server Startup Parameters:

sqlservr -c -x -T3502 -g100

where:
-c Start SQL Server independently of the Windows NT Service Control Manager
-x Disables the keeping of CPU time and cache-hit ratio statistics
-T3502 Prints a message to the SQL Server log at start and end of each checkpoint
-g100 memory in MB reserved for memory requests outside the buffer pool

Microsoft SQL Server Stack Size:

The default stack size for Microsoft SQL Server 2000 was changed using the EDITBIN utility:
editbin /STACK:131072

Microsoft SQL Server Configuration Parameters:


1> 2> 3> 4> 5> 6> 7> 8> 9> 10> 11>
--  File:     VERSION.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.21
--            Copyright Microsoft, 1999, 2000
--  Purpose:  Returns SQL Server version string



print " "
select convert(char(30), getdate(),9)
print " "

 ------------------------------
 Aug 29 2000 12:27:04:037PM

(1 row affected)

1> 2> 3>
select @@version



 -------------------------------------------------------------------------------
       ---------------------------------------------------------------------
       ---------------------------------------------------------------------
       ----------------------------------
 Microsoft SQL Server  2000 - 8.00.194 (Intel X86)
       Aug  6 2000 00:57:48
       Cop
       yright (c) 1988-2000 Microsoft Corporation
       Enterprise Edition on Windo
       ws NT 5.0 (Build 2195: Service Pack 1)



(1 row affected)
1> 2>
1> 2> 3> 4> 5> 6> 7> 8> 9> 10>
--  File:     CONFIG.SQL
--            Microsoft TPC-C Benchmark Kit Ver. 4.21
--            Copyright Microsoft, 1999, 2000
--  Purpose:  Collects SQL Server configuration parameters


print " "
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

```
select convert(char(30), getdate(),9)
print " "

 ------------------------------
 Aug 29 2000 12:27:04:880PM

(1 row affected)

1> 2> 3> DBCC execution completed. If DBCC printed error messages, contact your system administrator.
Configuration option 'show advanced options' changed from 1 to 1. Run the RECONFIGURE statement to
install.

sp_configure "show advanced",1
1> 2> reconfigure with override
1> 2> sp_configure
 name                                 minimum     maximum     config_value run_value
 ------------------------------------ ----------- ----------- ------------ -----------
 affinity mask                                  0  2147483647           15          15
 allow updates                                  0           1            0           0
 awe enabled                                    0           1            1           1
 c2 audit mode                                  0           1            0           0
 cost threshold for parallelism                 0       32767            5           5
 cursor threshold                              -1  2147483647           -1          -1
 default full-text language                     0  2147483647         1033        1033
 default language                               0        9999            0           0
 fill factor (%)                                0         100            0           0
 index create memory (KB)                     704  2147483647            0           0
 lightweight pooling                            0           1            1           1
 locks                                       5000  2147483647            0           0
 max degree of parallelism                      0          32            1           1
 max server memory (MB)                         4  2147483647   2147483647  2147483647
 max text repl size (B)                         0  2147483647        65536       65536
 max worker threads                            32       32767          182         182
 media retention                                0         365            0           0
 min memory per query (KB)                    512  2147483647         1024        1024
 min server memory (MB)                         0  2147483647            0           0
 nested triggers                                0           1            1           1
 network packet size (B)                      512       65536         4096        4096
 open objects                                   0  2147483647            0           0
 priority boost                                 0           1            0           0
 query governor cost limit                      0  2147483647            0           0
 query wait (s)                                -1  2147483647           -1          -1
 recovery interval (min)                        0       32767           40          40
 remote access                                  0           1            1           1
 remote login timeout (s)                       0  2147483647           20          20
 remote proc trans                              0           1            0           0
 remote query timeout (s)                       0  2147483647          600         600
 scan for startup procs                         0           1            0           0
 set working set size                           0           1            0           0
 show advanced options                          0           1            1           1
 two digit year cutoff                       1753        9999         2049        2049
 user connections                               0       32767            0           0
 user options                                   0       32767            0           0
```


© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

Appendix D – Space Calculation

						Microsoft SQL Serve
Note : Numbers are in KBytes unless otherwise specified						Updated for Version 7 (
Warehouses	2716	**tpmC**	34150	**tpmC/W**	12,57	
Table	**Rows**	**Data**	**Index**	**5% Space**	**8H Space**	**Total Space**
Warehouse	2.716	296	48	17		361
District	27.160	3.024	48	154		3.226
Item	100.000	9.528	64	480		10.072
New-order	24.444.000	386.472	896		217.280	604.648
History	81.480.000	4.526.680	80		910.685	5.437.445
Orders	81.480.000	2.497.472	1.135.704		730.916	4.364.092
Customer	81.480.000	59.258.184	3.533.584	3.139.588		65.931.356
Order-line	814.797.124	50.924.824	107.800		10.266.652	61.299.276
Stock	271.600.000	86.912.000	162.480	4.353.724		91.428.204
Totals		204.518.480	4.940.704	7.493.963	12.125.533	229.078.680
Segment	**LogDev Cnt.**	**Seg. Size**	**Needed**	**Overhead**		**Not Needed**
misc	5	89.600.000	72.436.310	724.363		16.439.326
customer/stock	5	158.720.000	158.933.156	1.589.332		-1.802.488
Totals		248.320.000	231.369.466	2.313.695		14.636.839
Dynamic space	57.948.976	Sum of Data for Order, Order-Line and History				
Static space	161.317.865	Data + Index + 5% Space + Overhead - Dynamic space				
Free space	14.416.320	Total Seg. Size - Dynamic Space - Static Space - Not Needed				
Daily growth	11.658.071	(Dynamic space/W * 62.5)* tpmC				
Daily spread	-3.070.787	Free space - 1.5 * Daily growth (zero if negative)				
180 day (KB)	2.259.770.622	Static space + 180 (daily growth + daily spread)				
180 day (GB)	2.155,09	180-day space in GB (excludes OS, Paging and RDBMS Logs)				
Log size (MB)	80.000	Total size of log file				
% Log used	20,6425	% of log file used during entire run				
Total N-O Txn	3276470	Total count of N-O transactions during entire run				
Log per N-O txn	5,1611	KB of log per New-Order transaction				
8 Hour Log (GB)	80,68	8 hours of log in GB (excluding space for redundancy)				
Disk Capacity	**MB**	**GB**	**disks needed**	**disks priced**	**GB priced**	
9 GB 10000 rpm	8392	8,20		144	1.180,13	
18 GB 10000 rpm	17160	16,76		96	1.608,75	
180 day (GB)		2.155,09		240	2.788,88	
Disk Capacity	**MB**	**GB**	**disks needed**	**disks priced**		
36 GB 10000 rpm	34712	33,90				
8 Hour Log (RAID 1)		80,68	2,38	3+3		

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

Appendix E - Price Quotations

Fujitsu Siemens Computer
z.Hd. Herrn Miguel Isenberg
FSC SHV Server DS 51

25. August 2000

ANGEBOT

D-Link DEUTSCHLAND

Gültigkeit bis 30.11.2000

Projekt: Fujitsu250800/2-LT

Sehr geehrter Herr Isenberg,

Wir können Ihnen für das avisierte Projekt die unten aufgeführten Produkte zu folgenden Konditionen anbieten:

DES-3225G	3 Stück	á	1.712,00 DM
DES-3251G	3 Stück	á	793,15 DM
DE-809TC	1.000 Stück	á	67,35 DM

Die angegebenen Preise verstehen sich netto/netto ex warehouse und beziehen sich auf das Gesamtprojekt, den nachfolgend genannten Auslieferungszeitraum und die oben aufgelistete Stückzahl. Da diese Preise auf dem aktuellen Dollarkurs basieren, müssen bei einer Kursänderung um +3% und mehr die Preise der Produkte entsprechend angepasst werden. Bei einer grösseren Bestellung bleibt der Preis erhalten.

Dieses Angebot gilt nur bei einer Bestellung an:
D-Link Deutschland GmbH, Herrn Tamme, Bachstrasse 22, 65830 Kriftel.

Zahlung: innerhalb 30 Tage, rein netto.

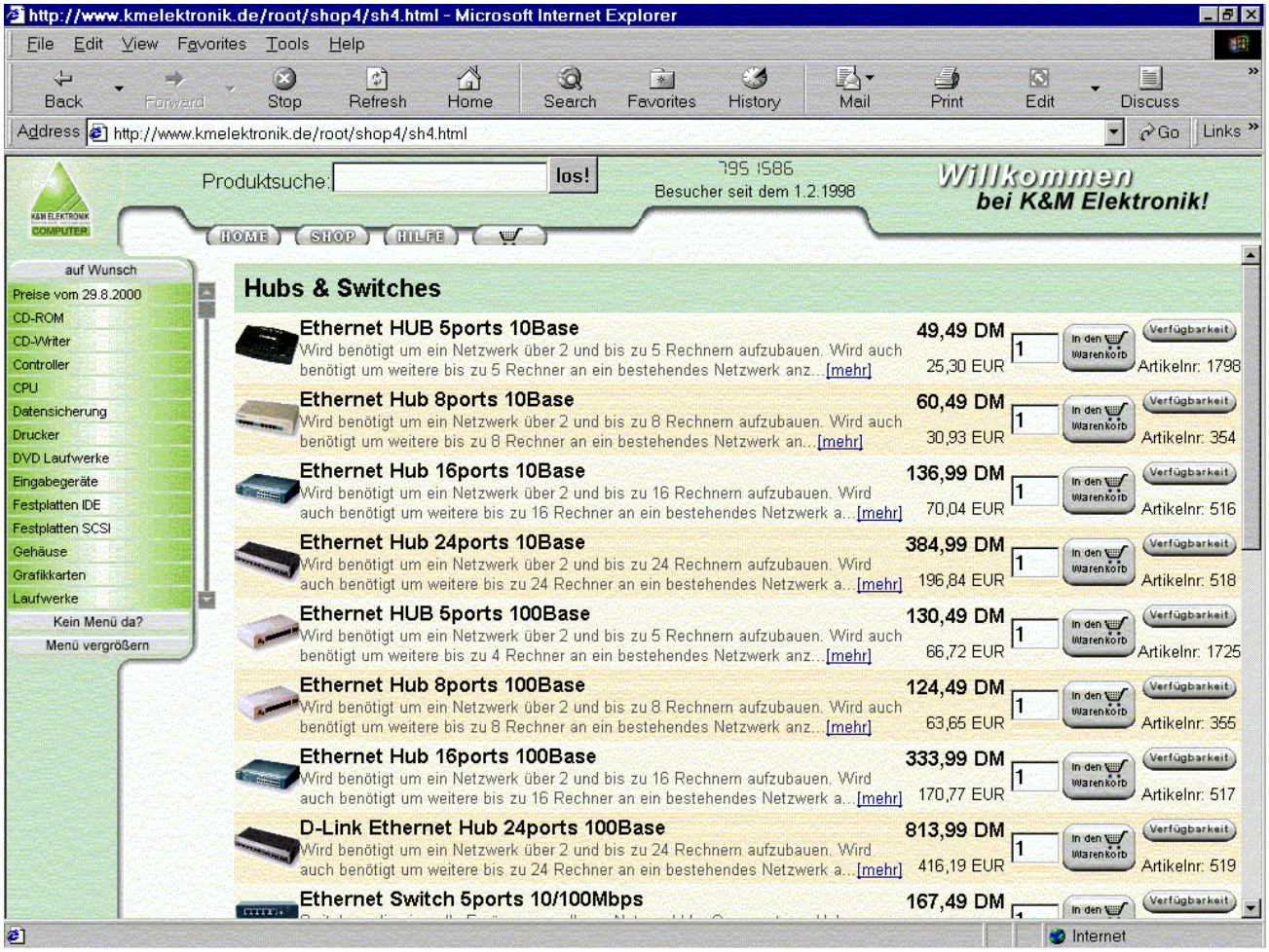
Die Lieferung erfolgt innerhalb von 4 Wochen nach Eingang der Bestellung. Bei rechtzeitige Ankündigung ist die Auslieferung anderer Lotgrößen möglich.
Bitte beziehen Sie sich bei der entsprechenden Bestellung auf die oben aufgeführte Projektkennzeichnung.

Mit freundlichen Grüßen

D-Link Deutschland GmbH

Leonhard Tamme
Key Account Manager

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.



© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

Appendix F - Attestation Letter

Benchmark Sponsor: Franz-Josef Bathe
Fujitsu Siemens Computers
Heinz-Nixdorf-Ring 1
D-33106 Paderborn, Germany

September 1, 2000

I remotely verified the TPC Benchmark™ C performance of the following Client Server configuration:

Platform:	**Siemens Primergy N400**
Operating system:	**Microsoft Windows 2000 Advanced Server**
Database Manager:	**Microsoft SQL Server 2000 Enterprise Edition**
Transaction Manager:	**Microsoft COM+ (Included in Windows 2000)**

The results were:

CPU's Speed	Memory	Disks	NewOrder 90% Response Time	**tpmC**
Server: Siemens Primergy N400				
4 x Pentium III Xeon (700 MHz)	8 GB Main (2MB L2 Cache per processor)	145 x 9 GB 96 x 18 GB 6 x 36 GB	0.40 Seconds	**34150.87**
Four (4) Clients: Primergy 170 (Specification for each)				
1 x Pentium III (750 MHz)	256 MB Main Cache: 256 KB	1 x 9 GB	n/a	n/a

In my opinion, these performance results were produced in compliance with the TPC's requirements for the benchmark. The following verification items were given special attention:

- The database records were the proper size
- The database was properly scaled and populated
- The required ACID properties were met

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.

- The transactions were correctly implemented
- Input data was generated according to the specified percentages
- The transaction cycle times included the required keying and think times
- The reported response times were correctly measured.
- All 90% response times were under the specified maximums
- At least 90% of all delivery transactions met the 80 Second completion time limit
- The reported measurement interval was 30 minutes (1800 seconds)
- The reported measurement interval was representative of steady state conditions
- One checkpoint was taken during the reported measurement interval
- The repeatability of the measured performance was verified
- The 180 day storage requirement was correctly computed
- The system pricing was verified for major components and maintenance

Respectfully Yours,

François Raab, President

Bradley J. Askins, Auditor

© 2000 Fujitsu Siemens Computers GmbH. All rights reserved.