Extending TPC-E to Measure Availability in Database Systems

TPC Technology Conference at VLDB
August 29, 2011

Yantao Li
Senior Program Manager
Microsoft
yantaoli@microsoft.com

Charles Levine
Principal Program Manager Lead
Microsoft
clevine@microsoft.com

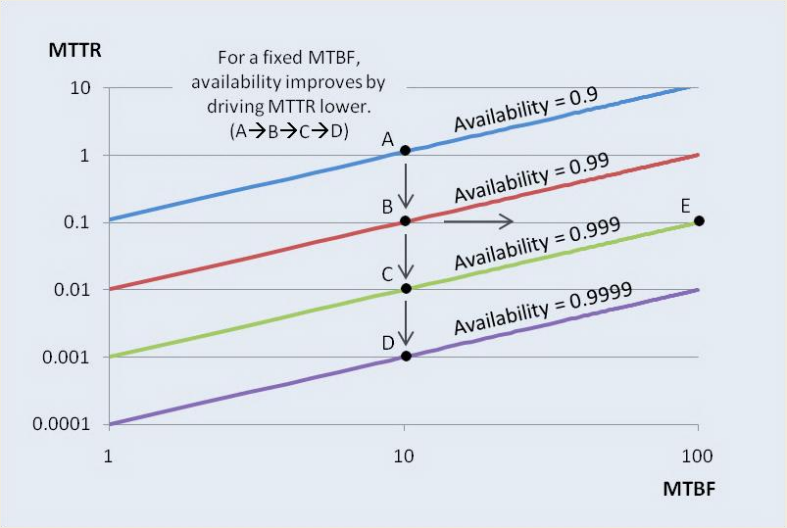
If you can't measure something, you can't improve it.

Overview

- High-availability (HA) is required for mission-critical database applications to ensure business continuity despite various faults
  - Redundancy in multiple layers depending on degree of availability an application needs
    - Redundant rower supply, storage (e.g., disk RAID levels), NICs and network switches, etc.
    - One or more standby database instances
    - Duplicated system in remote data center (for geographic disaster recovery)
  - Scenarios
    - Planned downtime: OS & SQL Server patches, service pack, hardware maintenance, etc.
    - Unplanned downtime: Hardware faults, software bugs and human errors
- Being able to measure and characterize availability is important for:
  - Driving availability improvement in RDBMS
  - Proactive understanding of the HA capability of a system
  - Guiding HA system design/development
  - Evaluating different HA technologies
- Currently, there's no industry standard availability benchmark for database systems
  - TPC-E benchmark is a representative OLTP workload for measuring performance & scalability
  - We extend TPC-E to measure database system availability

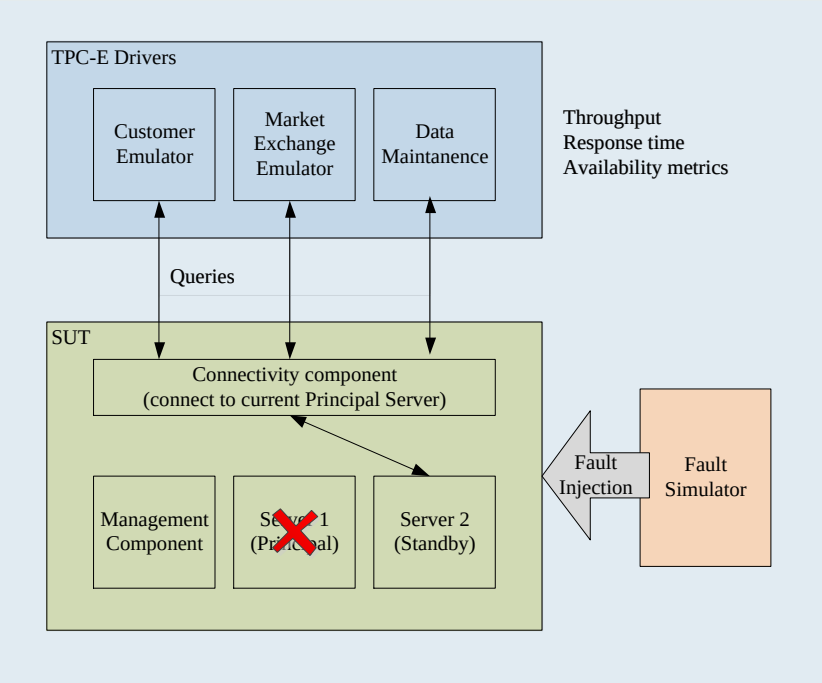
Improve Availability by Driving Time-to-Recover Lower

- Availability is usually expressed as a percentage of uptime over a period of time
  - For example, 99.999% availability means ~5 minutes downtime per year
- Availability is the product of mean-time-between-failures (MTBF) and mean-time-to-recover (MTTR)
  - Availability = MTBF/(MTBF+MTTR)
  - MTBF and MTTR are orthogonal metrics
  - Can improve availability by improving one, independent of the other
- In this paper we focus on time-to-recover
  - Measurable and actionable metric
  - To understand MTBF generally requires certain estimates or modeling exercises (e.g., the probability of power outage in one area in one year)



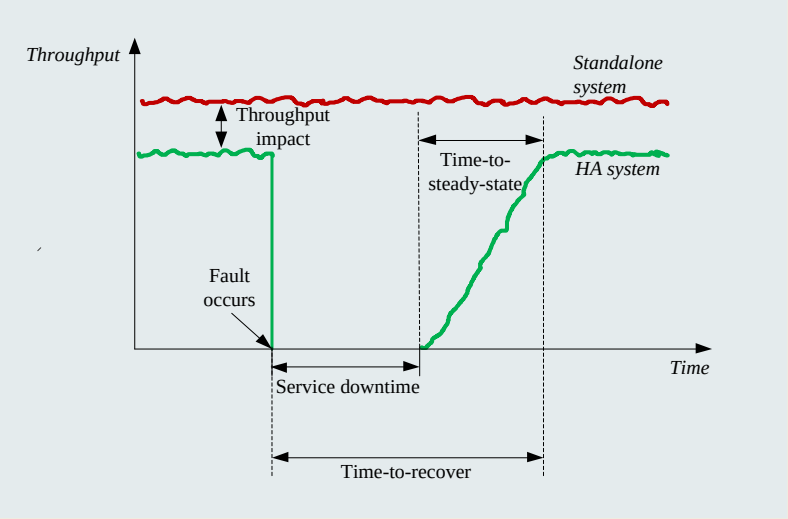
Extend TPC-E for Availability Measurement

- Extend System Under Test (SUT) to include all HA components
  - Principal server
  - Standby servers
  - Management component
  - Connectivity component
- Simulate representative fault scenarios
  - Both planned and unplanned downtime
  - Focus on how database system handles faults rather than the causes of the faults
- Automatic reconnection capability in the TPC-E driver
- Define and implement availability metric



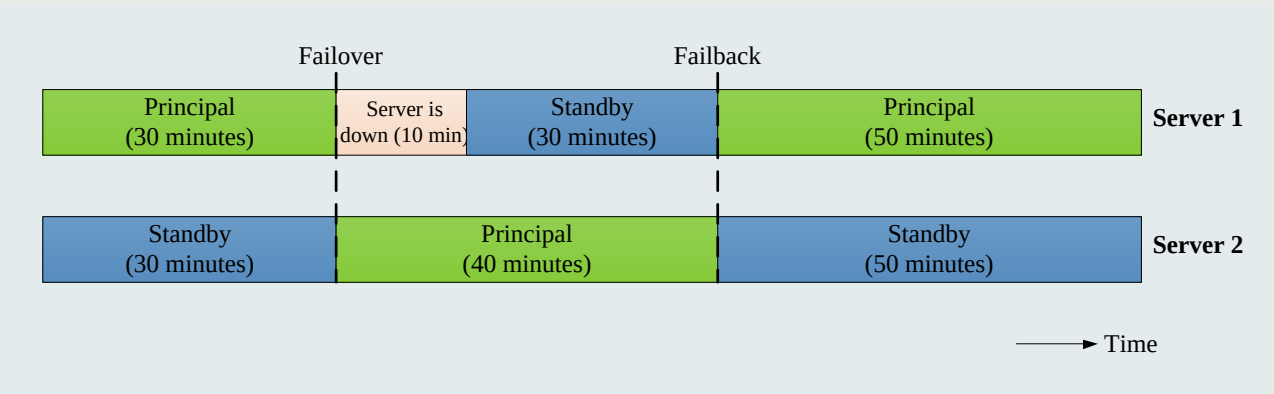
Key Metrics in HA Database Systems

- Capital cost for additional hardware and software
  - Pricing model defined in the TPC-E specification can be used as is to compute system prices for both HA and non-HA configurations
- Performance impact during normal operations
  - The impact to throughput/response time of HA capabilities compared to the non-HA system
- Recovery time: The time to restore the database service after a fault occurs
  - Service downtime
  - Time to steady state



Testing Availability on DB Mirroring

- Microsoft SQL Server Database Mirroring
  - Maintains two copies of a single database that reside on different servers
  - High-safety mode: Database status is synchronously replicated to standby
- Performance counters for monitoring data movement
  - Log Send Queue
  - Redo Queue
- System workflow in planned/unplanned downtime scenarios
  - Manual failover (planned downtime) script: *ALTER DATABASE FAILOVER*
  - Automatic failover (unplanned downtime) script: *TASKKILL MSSQLSERVER.EXE*



Test Results on DB Mirroring

DB Mirroring & Standalone Comparison

	DB Mirroring Normalized as % of Standalone	
	Principal	**Standby**
Throughput	98.6%	NA
CPU	100%	7%
DB Mirroring: Log Send Queue (KB)	0.1	NA
DB Mirroring: Redo Queue (KB)	NA	11

Automatic Failover Performance

Stage		**Metric (in seconds)**
Start the workload	**Time-to-steady-state**	21
Failover	**Service downtime**	17
	Time-to-steady-state	24
	Total time-to-recover	41

System Specification

Principal & Standby Server	Dell PE 2950 Processor: 2 x Quad Core Intel Xeon X5355, 2.66 GHz Memory: 16 GB NIC: 1Gbps
Witness	Dell PE 860 Processor: 1 x Quad Core Intel Xeon X3220, 2.4 GHz Memory: 4 GB
Storage (Both Principal & Standby)	Data: 52 x 15K SAS drives; configured to 4 LUNs (14 spindles each) Log: 4 x 15K SAS drives
Software	Microsoft Windows Server 2008 x64 Enterprise Edition Microsoft SQL Server 2008 x64 Enterprise Edition

TPC-E Configuration

TPC-E Database Size	30,000 customers
Users	120 concurrent users: Drive to maximum load (CPU is 100% busy). Zero think time.
Start Rate (Users/Minute)	300
Connect Rate (Users/Minute)	300
Transaction Mix	Standard Benchmark Mix
SQL Server Memory	14,000 MB
Database Size	240 GB raw data size. Allocated about 395 GB in data files for growth

Applications/Future Work

- The approach has been used for Microsoft SQL Server internal engineering
- Similar approach can be used by database customers to guide their HA system design and improvement
- HA metrics could be introduced into other database workloads
- The methodology could be used as a starting point to define an industry standard for availability measurement

Related Work

- Gray and Siewiorek described key concepts and techniques to build high availability computer systems
  - Gray, J., Siewiorek, D.: High-Availability Computer Systems, Computer, vol. 24, no. 9 (1991) 39–48
- International Federation for Information Processing (IFIP) Working Group 10.4 was established to identify and integrate methods and techniques for dependable computing
  - http://www.dependability.org/wg10.4/
  - Areas include understanding of various faults, methods for error detection, validation and design for testability and verifiability, etc.
  - Many workshops and conferences have been held to advance the research
- The DBench-OLTP project defined a general dependability benchmark model for OLTP systems using TPC-C
  - Vieira, M., Madeira, H.: A dependability Benchmark for OLTP Application Environments, VLDB 2003 742-753