A Systematic Method for Big Data Technology Selection

John Klein

Software Solutions Conference 2015

November 16–18, 2015

DM-0003040

Agenda

Background

Problem

Method

Results and Lessons Learned

Background –
Two Providers, Two EHR Systems

Military Health System

- Supports 9.7 million people – active duty, dependents, retirees
- Global – 700+ hospitals, clinics, and other facilities
  - 135,000 employees, $49B per year
- Electronic Health Record system
  - Multiple legacy systems, over 100 application interfaces
  - 1PB+ data, 99 year data retention

Veterans Health Administration

- 6 million patients
- 153 VA medical centers,1,300+ outpatient clinics and centers
- Single Electronic Health Record system (VistA)
  - Local customizations and extensions

Background – Limited EHR Interoperability

3.5 million shared patients

Multiple interoperation projects since 2001

- Federal Health Information Exchange (FHIE)
- Bidirectional Health Information Exchange (BHIE)
- Clinical Data Repository/Health Data Repository (CHDR) interface
- Laboratory Data Sharing Interface (LDSI)

Background –
Interagency Project Office (IPO)

Created by 2008 NDAA to be single point of accountability for electronic health record integration

Initially planned to develop new Integrated EHR (iEHR) for MHS and VHA

Redirected in Feb. 2013

- VHA to continue VistA evolution
- MHS to acquire COTS solution (DHMSM)

Background – Interagency Project Office (IPO)

TATRC/SEI

Engagement

Created by 2008 NDAA to be single point of accountability for electronic health record integration

Initially planned to develop new Integrated EHR (iEHR) for MHS and VHA

Redirected in Feb. 2013

- VHA to continue VistA evolution
- MHS to acquire COTS solution (DHMSM)

TATRC/SEI Partnership

U.S. Army's Telemedicine & Advanced Technology Research Center

- U.S. Army Medical Research and Materiel Command (USAMRMC) Lab
- www.tatrc.org

TATRC Advanced Concepts Team engaged SEI to help them support the IPO

The IPO's Problem

We've got "Big Data"

A best practice for big data systems is to use a NoSQL data store

Is NoSQL* a good fit for our system?

If so, can we narrow down the COTS product trade space?

* NoSQL – category of data stores that provide high scalability and performance by partitioning and replicating data across a cluster of servers, characterized by no schema on writes and simple read/query interfaces.

Big Data Technology Evaluation Challenges

"Convergence of Concerns"

- Data store technology and system architecture are intertwined
- Can't defer technology selection

Rapidly changing technology landscape

- New products emerging, multiple releases per year on existing products
- Need to balance speed with precision

Large potential solution space

- Need to quickly narrow down and focus

Scale makes full-fidelity prototyping impractical

- Data sets, compute nodes, load generation

Technology is highly configurable

- Need to focus on go/no-go criteria

“Convergence of Concerns”

SQL’s single system abstraction produces strong separation of concerns between application and database

Petascale systems are changing architecture principles by creating *convergence of concerns*

- Can’t abstract away underlying technology and topology - application, data, and deployment topology are tightly coupled
- Data layer decisions drive system architecture
- Need to select technology early in the development cycle – decision becomes embedded in the architecture and is hard to change

I. Gorton and J. Klein, “Distribution, Data, Deployment: Software Architecture Convergence in Big Data Systems,” *IEEE Software,* vol. 32, no. 3, pp. 78-85, May/June 2015. doi: 10.1109/MS.2014.51

http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=90909

“Convergence of Concerns”

SQL’s single system abstraction produces strong separation of concerns between application and database

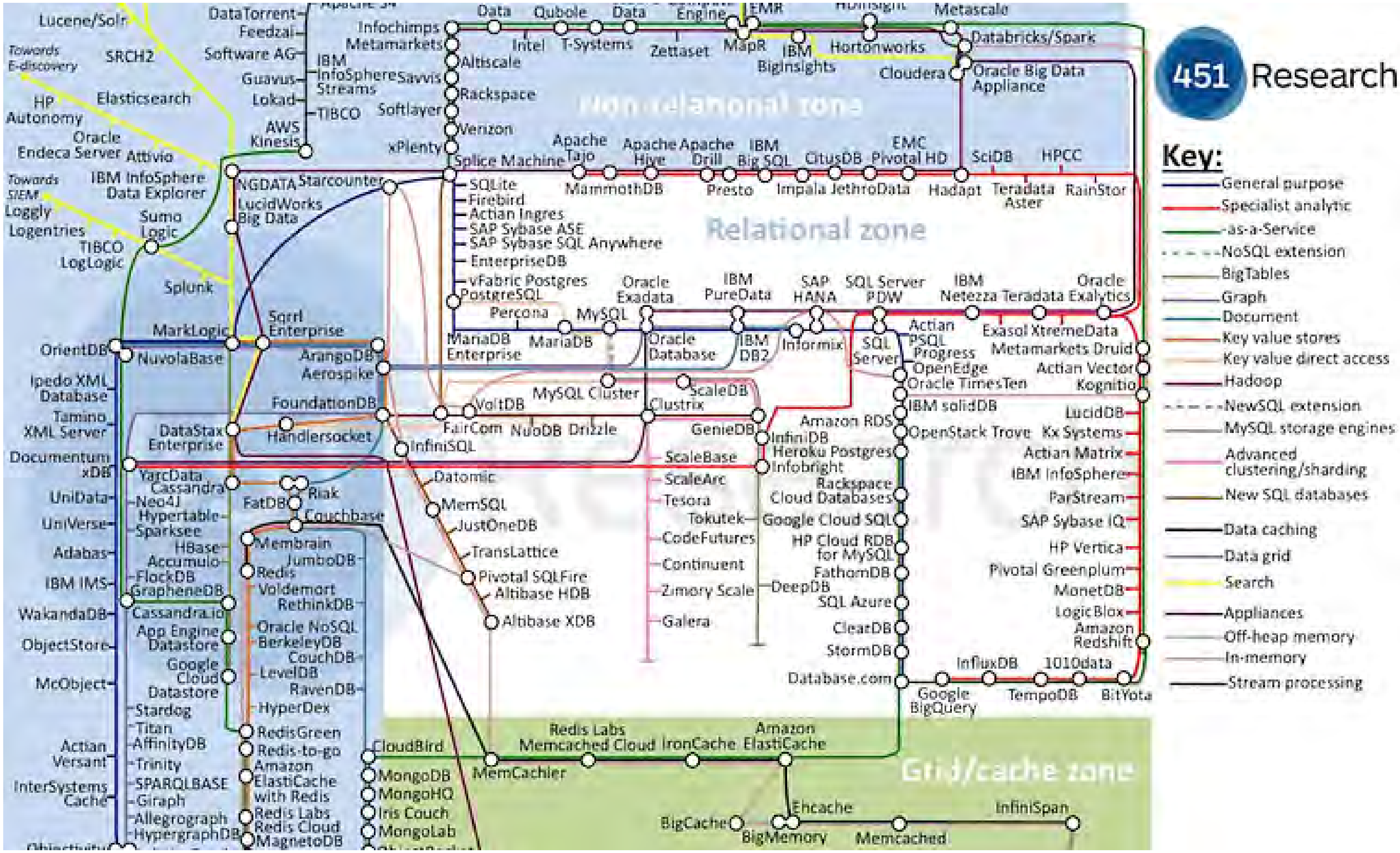
Petascale systems are changing architecture principles by creating *convergence of concerns*

- Can’t abstract away underlying technology and topology - application, data, and deployment topology are tightly coupled
- Data layer decisions drive system architecture
- **Need to select technology early in the development cycle – decision becomes embedded in the architecture and is hard to change**

I. Gorton and J. Klein, “Distribution, Data, Deployment: Software Architecture Convergence in Big Data Systems,” *IEEE Software,* vol. 32, no. 3, pp. 78-85, May/June 2015. doi: 10.1109/MS.2014.51

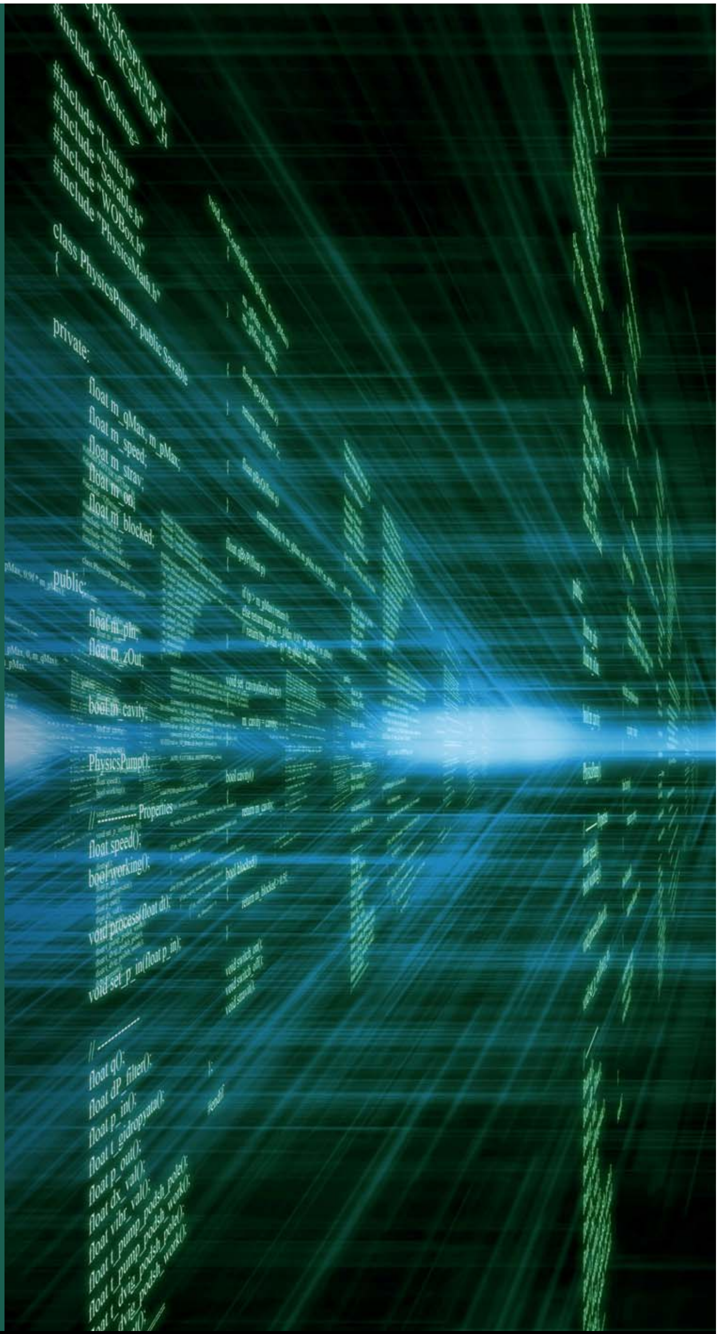
http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=90909

Solution Space – The NoSQL Landscape

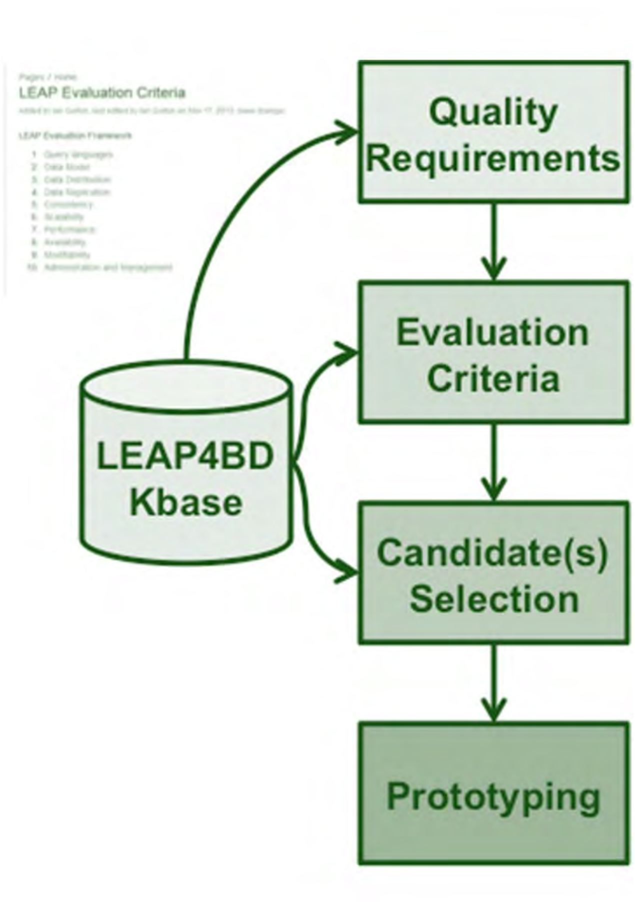


Source: http://blogs.the451group.com/information_management/2014/03/18/updated-data-platforms-landscape-map-february-2014/

Evaluation Method – Lightweight Evaluation and Architecture Prototyping for Big Data (LEAP4BD)



Evaluation Method – LEAP4BD



Lightweight Evaluation and Architecture Prototyping for Big Data (LEAP4BD)

Aims

- Risk reduction
- Rapid, streamlined selection/acquisition

Steps

- Assess the system context and landscape
- Identify the architecturally-significant requirements and decision criteria
- Evaluate candidate technologies against quality attribute decision criteria
- Validate architecture decisions and technology selections through focused prototyping

Assess the existing and future data landscape

Identify the application's fundamental data holdings, their relationships,

Identify most frequent queries and access patterns

Identify required performance and quantifies expected data and transaction growth

Output:

- The scope and context for the rest of the analysis and evaluation
- Initial insights into the suitability of a range of NoSQL candidates to support the application's requirements

Identify the architecturally-significant requirements and develop decision criteria

Focus on scalability, performance, security, availability, and data consistency

Engage with stakeholders to characterize the application's quality attribute requirements

- Quality Attribute Workshop
- Stakeholder interviews

Combine these architecture requirements with the characteristics of the data model (from previous step) to generate the necessary information for initial architecture design and technology selection

Evaluate candidate technologies against quality attribute decision criteria

Identify and evaluate candidate Big Data technologies against the applications' data and quality attribute requirements

Selects a small number of candidates (typically two to four) for validation through prototyping and testing

Riak

Images/4/48/BigData.png > Special:RunQuery/Consistency Query > Images/4/48/BigData.png > Main Page > Riak

Overview	Key-Value store based on DynamoDB
Model	Key-Value

Riak Features

Feature	View	Edit
Consistency	Riak Consistency Features	Riak Consistency
Data Model	Riak Data Model Features	Riak Data Model
Query Languages	Riak Query Language Features	Riak Query Languages
Data Distribution	Riak Data Distribution Features	Riak Data Distribution
Data Replication	Riak Data Replication Features	Riak Data Replication
Scalability	Riak Scalability Features	Riak Scalability
Performance	Example	
Security	Riak Security Features	Riak Security
Administration and Management	Example	

LEAP4BD Knowledge Base – http://quabase.sei.cmu.edu

Form-based queries

Leave "none" if you don't want to search for a criteria
Also tick some "none" if no results show, i.e. relax the search criteria

- **Object-Level isolation on updates:** ◉ None ○ supported ○ not supported ○ partially supported
- **ACID transactions in single database:** ◉ None ○ supported ○ not supported ○ partially supported
- **Distributed ACID transactions:** ◉ None ○ supported ○ not supported ○ partially supported
- **Specify Quorum Reads/Writes:** ◉ None ○ not supported ○ in the client ○ in the database configuration ○ in both the database and data center
- **Specify number of replicas to write to:** ◉ None ○ not supported ○ in the client ○ in the database configuration
- **Behaviour when write cannot complete on specified number of replicas:** ◉ None ○ no rollback: write returns replication error ○ hinted handoffs: writes are applied later when a replica recovers ○ a rollback at all replicas
- **Writes configured to never fail:** ◉ None ○ supported ○ not supported
- **Specify number of replicas to read from:** ◉ None ○ not supported ○ in the client ○ in the database configuration
- **Read from replica master only:** ◉ None ○ not supported ○ in the client ○ in the database configuration
- **Updates applied to transaction log before returning from write:** ◉ None ○ supported ○ not supported
- **Object level timestamps to detect conflicts:** ◉ None
- **Efficient protocol to rapidly propagate updates across replica**

Tabular and graphical reports

Below are your query results. Using the form below, you can enter a new query.

Result	Object-Level isolation	ACID transactions	Distributed transactions	Quorum Reads/Writes	Specify #replicas on write	Write on replica fails	Write always succeeds	Specify #replicas on read	Read master only	Update Transaction Log	Conflict detection	Efficient replication protocol
Cassandra Consistency Features	supported	not supported	not supported	in the client	in the client	hinted handoffs: writes are applied later when a replica recovers	supported	in the client	not supported	supported	supported	configurable
MongoDB Consistency Features	supported	not supported	not supported	in the database configuration	in the client	a rollback at all replicas	not supported	in the client	in the client	supported	not supported	supported by default
Riak Consistency Features	supported	not supported	not supported	in the client	in the client	no rollback: write returns replication error	supported	in the client	not supported	supported	supported	supported by default

Validate architecture decisions and technology selections

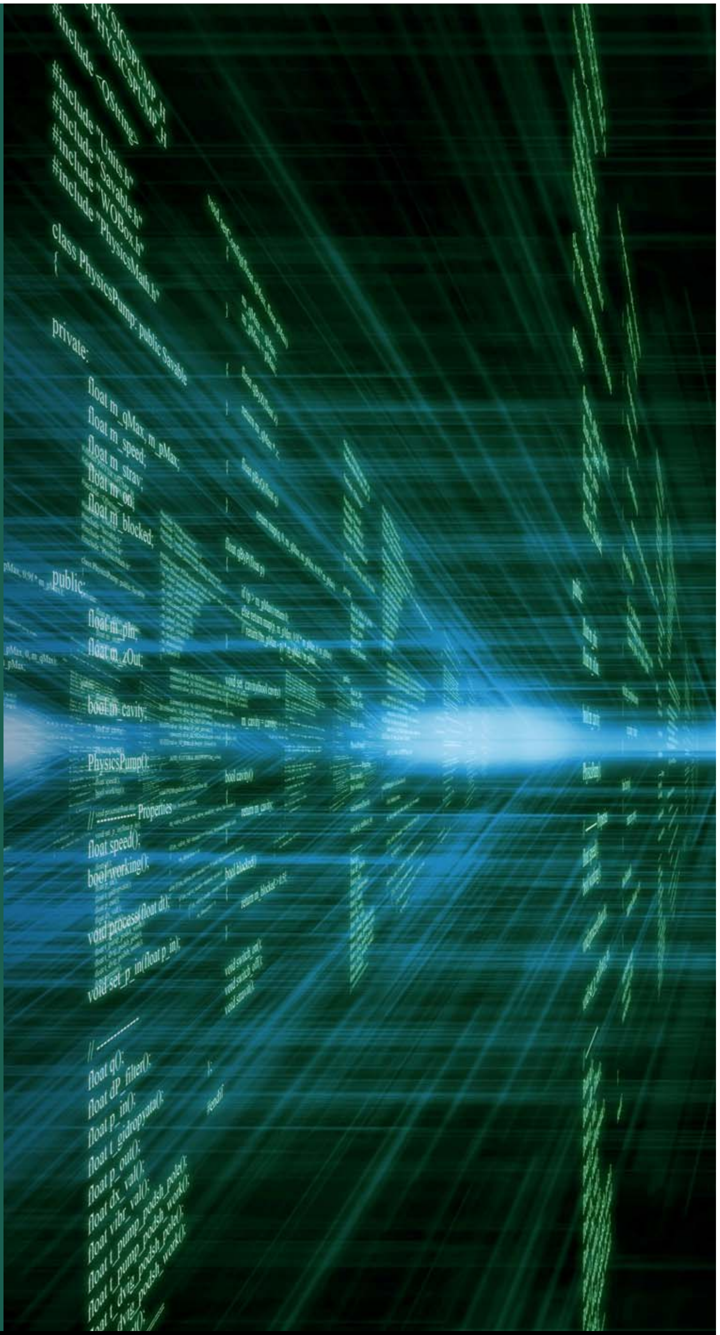
Perform focused prototyping

- Go/no-go criteria

Evaluate the prototype's behavior against a set of carefully designed, application-specific criteria (e.g., performance, scalability, etc.)

- Generate concrete evidence that can support the downstream investment decisions required to build, operate, and evolve the system
- Qualitative evidence
  - Data model fit, deployment options, etc.
- Quantitative evidence
  - Sensitivities and inflection points, look for trends rather than absolutes

Applying LEAP4BD with TATRC for the IPO



Requirements - Driving Use Cases

Conducted stakeholder workshop

Query – Retrieve the 5 latest medical test results for a patient

- Used to populate many application UI screens

Strong consistency on updates – all readers see the same data when a new test result is written

- Within a facility
- Across facilities – telemedicine, real time consultations

Core Workload – 80% read/20% write

- Also assessed a write-only workload for pre-loading a facilities cache with records for scheduled patient appointments

Select Candidate Technologies

IPO wanted to understand implications and differences among different NoSQL technologies for their application

Selected one representative from each NoSQL category (key-value, wide column, document, and graph store)

Based on feature screening and market leadership, we chose

- Riak (key-value)
- Cassandra (wide column)
- MongoDB (document)

Did not prototype and evaluate any graph store – none had horizontal partitioning needed to support the scalability requirements

Test Design and Execution

Running in SEI Virtual Private Cloud (Amazon Web Services)

- Typical configuration – 12 virtual machines, peaked 35 active virtual machines over multiple configurations

SEI/TATRC ACT Collaboration

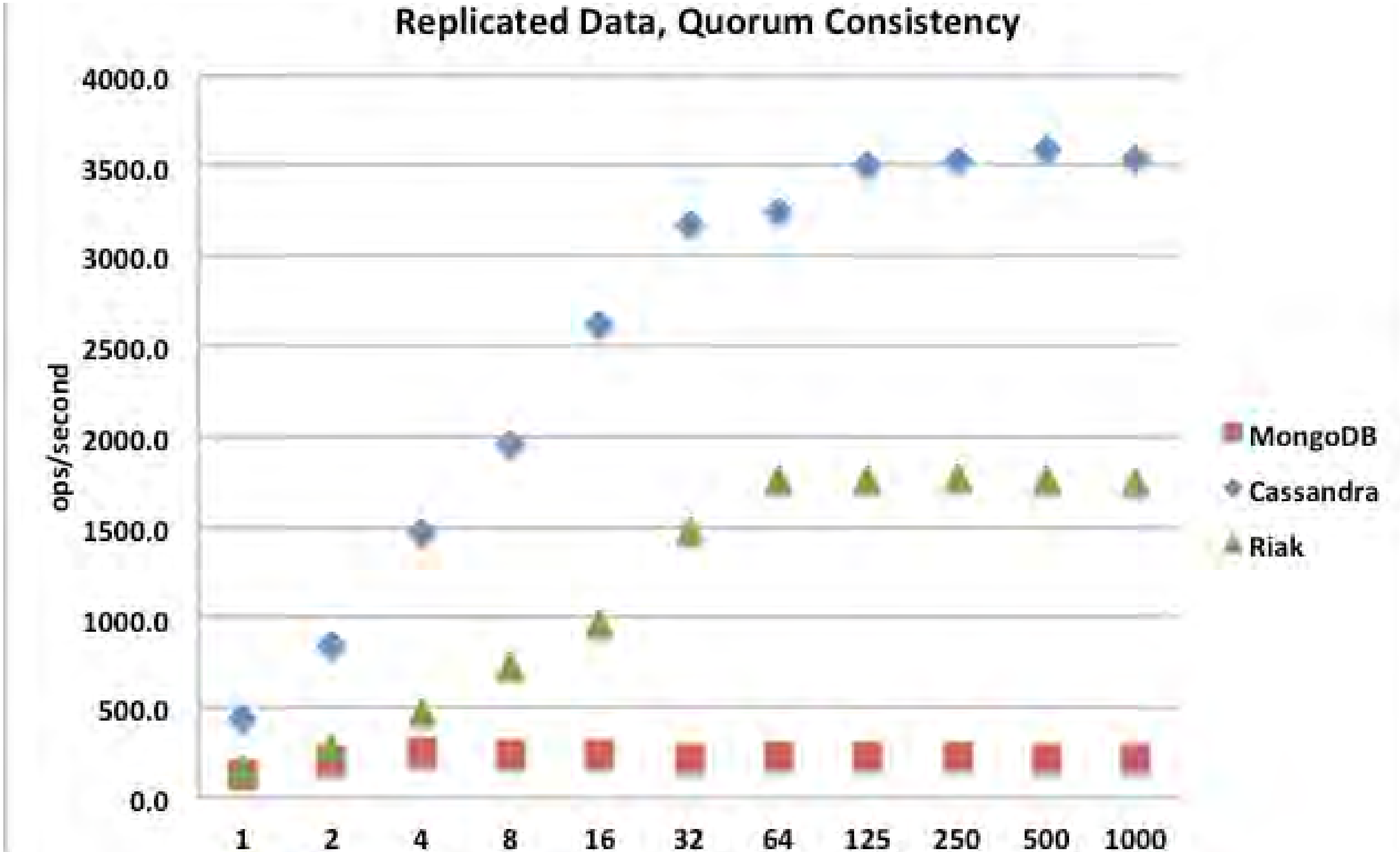
- SEI – Infrastructure management, experiment planning, analysis and synthesis of results
- TATRC ACT – database installation and configuration, workload development and test execution

Defined consistent environment - Server platform, test client platform, deployment topology

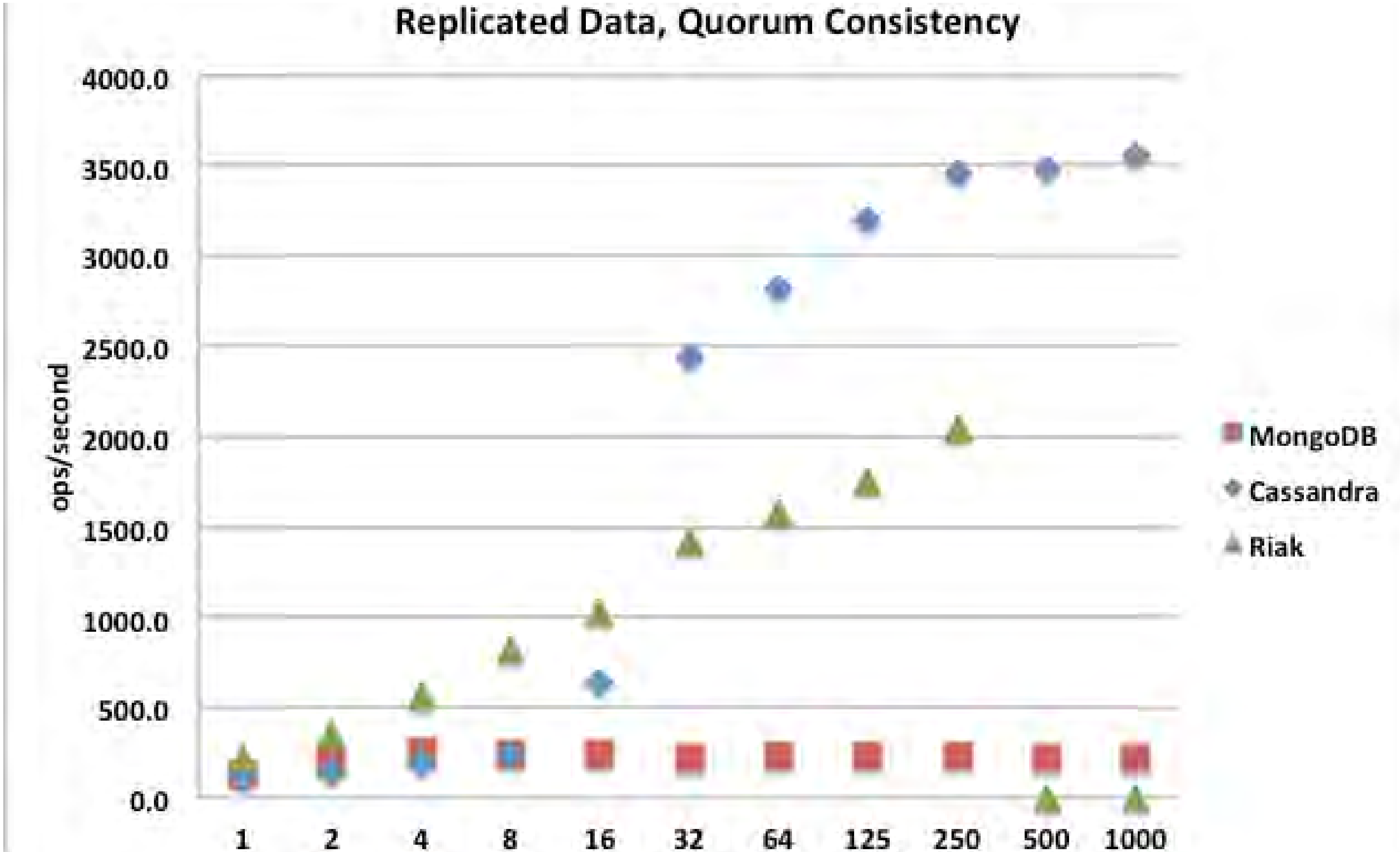
Mapped application's logical data model onto each database and load database

Defined test client to perform desired workloads, and measure and collect performance data

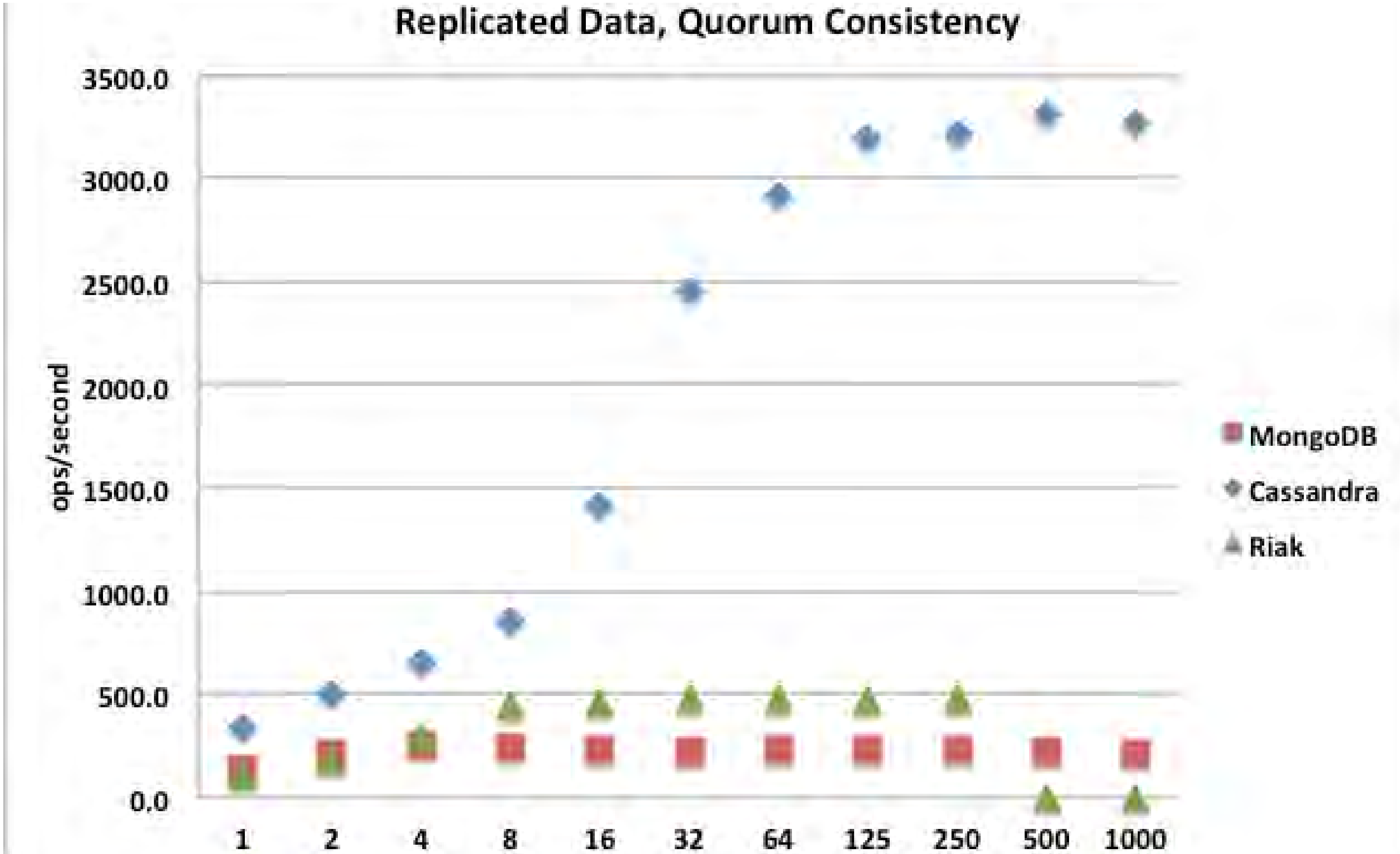
Typical Test Results – Throughput
(3x3 configuration, read-only workload)



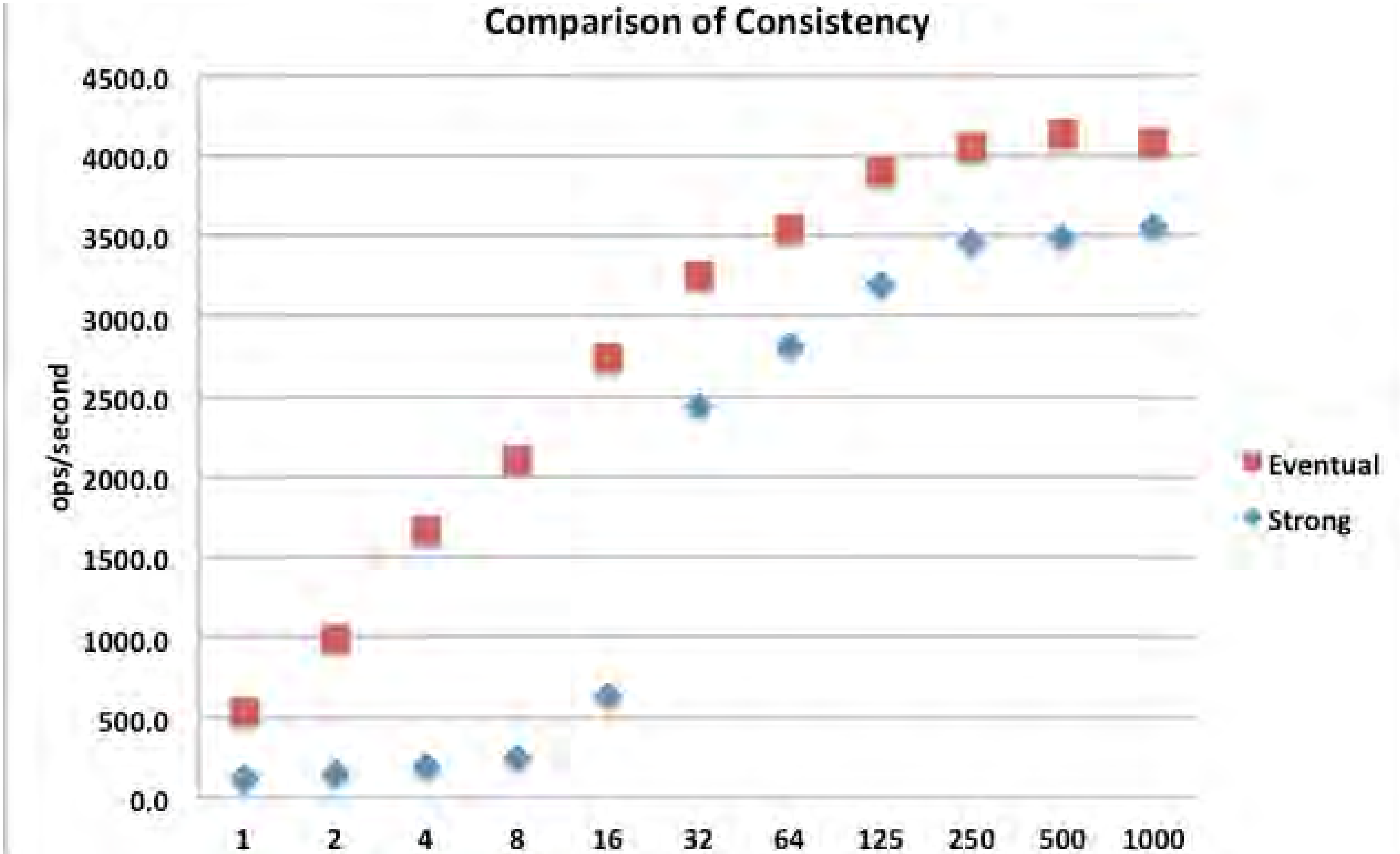
Typical Test Results – Throughput
(3x3 configuration, write-only workload)



Typical Test Results – Throughput
(3x3 configuration, read/write workload)



Test Results – "Cost of Strong Consistency"
(Cassandra data store)



LEAP4BD Outcomes

Focus on the data storage and access services addresses the major risk areas for an application

- Keeps the method lean, rapidly produces design insights that become the basis for downstream decisions

Highly transparent and systematic analysis and evaluation method significantly reduces the burden of justification for the necessary investments to build, deploy, and operate the application

- Data-driven decisions

Informed adoption of modern technologies to reduce costs while ensuring that an application can satisfy its quality attribute requirements

Increased confidence in architecture design and database technology selection

- Hands-on experience working with the technology during prototype development, reduces development risks

Identifies risks that must be mitigated in design and implementation, along with detailed strategies and measures that allow for continual assessment

Key SEI Capabilities that Contributed to Project Success

Domain

- Healthcare IT
- Technology Evaluation

Technology

- Big Data
- Distributed Systems
- Cloud

Facilities

- Virtual Private Cloud

Software Engineering Institute
Carnegie Mellon University

Contact Information

John Klein

Senior Member of the Technical Staff
Architecture Practices
+1 412-268-7700
jklein@sei.cmu.edu

U.S. mail:

Software Engineering Institute
Customer Relations
4500 Fifth Avenue
Pittsburgh, PA 15213-2612
USA

World Wide Web:

www.sei.cmu.edu
www.sei.cmu.edu/contact.html

Customer Relations

customer-relations@sei.cmu.edu
+1 412-268-5800