BIG DATA SYSTEM DEVELOPMENT: AN EMBEDDED CASE STUDY WITH A GLOBAL OUTSOURCING FIRM

Prof. Hong-Mei Chen
IT Management, Shidler College of Business
University of Hawaii at Manoa, USA

Prof. Rick Kazman
IT Management, Shidler College of Business
University of Hawaii at Manoa, USA
Software Engineering Institute, Carnegie Mellon University, USA

Serge Haziyev, Olha Hrytsay
SoftServe Inc.
Austin, TX, USA
OUTLINE

• Research Motivation
• Research Foundations
• Research Method
• Results
• Future Research Directions
• Conclusions
Big Data: Big Promise

• Big hype...
• Big data is the new oil
• Big data is the new gold
HOW??
Challenges

- 5V requirements
- Proliferation of Big Data Technology
- Rapid Big Data Technology Changes
- Complexity
- Paradigm Shifts
- Short history of big data system development in Enterprises
55% of big data projects were not completed

• Hype is wearing thin
• Only **13%** of respondents said their IT organizations put big data projects into production this year, but that's 5% higher than last year.
• 24% of those polled voted against the use of big data technologies in their business.
“2013 was the year of experimentation and early deployment; so is 2014”

- 73 percent of respondents have invested or plan to invest in big data in the next 24 months, up from 64 percent in 2013.
- Like 2013, much of the work today revolves around strategy development and the creation of pilots and experimental projects.

Note: The Gartner survey of 302 Gartner Research Circle members worldwide, which was conducted in June 2014.
Research Objectives

✓ To help enterprises navigate through uncharted waters and be better equipped for their big data endeavors.
✓ To uncover methodological voids and provide practical guidelines.
Research Questions

1. How does big data system development (processes and methods) differ from “small” (traditional, structured) data system development?

2. How can existing software architecture approaches be extended or modified to address new requirements for big data system design?

3. How can data modeling/design methods in traditional structured database/datawarehouse development be extended and integrated with architecture methods for effective big data system design?
“Small” Data System Development

• ANSI Standard 3-layer DBMS Architecture
  ▪ Clear Data-Program Independence (logical and physical data independence)

• Well-established RAD design process
  ▪ Iterative design of 7 phases
  ▪ Clear separation of each design phase
  ▪ Mature conceptual design tools: ER, UML, etc.

• Relational model dominance (95% market)
  ▪ Relational model easy to understand
  ▪ SQL easy to use, standardized

• Architecture Choice is relatively simple
  ▪ N-tier client-server design
Data/program Independence: ANSI 3-Layer DBMS Architecture (1980s)
Architecture Design is critical and complex in Big data System Development

I. **Volume**: Distributed and scalable architecture

II. **Variety**: Polyglot persistence architecture

III. **Velocity**: Complex Event processing + \(\rightarrow\) Lambda Architecture

IV. **Veracity**: Architecture design for understanding the data sources and the cleanliness, validation of each

V. **Value**: New architecture for hybrid, agile Analytics, big data analytics cloud, integrating the new and the Old (EDW, ETL)

VI. **Integration**: Integrating separate architectures addressing each of the 5V challenges
Research Questions

1. How does big data system development (processes and methods) differ from “small” (traditional, structured) data system development?

2. How can existing software architecture approaches be extended or modified to address new requirements for big data system design?

3. How can data modeling/design methods in traditional structured database/datawarehouse development be extended and integrated with architecture methods for effective big data system design?
Research Method

✓ Case study research is deemed suitable:
  ▪ system development, be it big or small data, cannot be separated from its organizational and business contexts.
  ▪ “How” and “Why” research questions.
  ▪ the research is largely exploratory

✓ Multiple cases: increase methodological rigor

✓ Collaborative Practice Research
  ▪ SSV, in the outsourcing industry
  ▪ who has successfully deployed 10 big data projects that can be triangulated

➔ Embedded Case Study
Reasons for selecting an outsourcer

• Outsourcing is an important and common means to realize a big data strategy

• Big data professional service is the largest segment of big data market and continues to grow.

• Outsourcing mitigates shortages of skills and expertise in the areas where they want to grow.
Big Data Market is Expected to Grow Rapidly

Professional service is the largest growing segment
Collaborative Practice Research (CPR) Steps in an Iteration

1) Appreciate problem situation
2) Study literature
3) Develop framework
4) Evolve Method
5) Action
6) Evaluate experiences
7) Exit
8) Assess usefulness
9) Elicit research results
Collaborative Practice Research (CPR)

ADD 2.0 (Cases 1-4)

ADD 2.5 -> 3.0 (Cases 5-6)

BDD (Cases 3-4, 7-10)
ADD

• ADD (Attribute-Driven Design) is an architecture design method "driven" by quality attribute concerns
  – Version 1.0 released 2000 by SEI.
  – Version 2.0 released November 2006 (on Current SEI site)
  – Version 2.5 published in 2013 by the researcher team
  – Version 3.0 to be published in 2016 by the researcher team.

• The method provides a detailed set of steps for architecture design
  – enables design to be performed in a systematic, repeatable way
  – leading to predictable outcomes.
<table>
<thead>
<tr>
<th>Case #</th>
<th>Business goals</th>
<th>Start</th>
<th>Big data</th>
<th>Technologies</th>
<th>Challenges</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Network Security, Intrusion Prevention&lt;br&gt;US MNC IT corp. (Employees &gt; 320,000)</td>
<td>Late 2010, 8.5 month</td>
<td>Machine generated data - 7.5BLN event records per day collected from IPS devices&lt;br&gt;Near real-time reporting&lt;br&gt;Reports which “touch” billions of rows should generates &lt; 1 min</td>
<td>ETL - Talend&lt;br&gt;Storage/DW – InfoBright EE, HP Vertica&lt;br&gt;OLAP – Pentaho Mondrian&lt;br&gt;BI – JasperServer Pro</td>
<td>• High throughput, different device data schemas (versions)&lt;br&gt;• keep system performance at required level when supporting IP/geography analysis: avoid join.&lt;br&gt;• Keep required performance for complex querying over billions rows</td>
</tr>
<tr>
<td>2</td>
<td>Anti-Spam Network Security System&lt;br&gt;US MNC Networking equipment corp. employees &gt; 74,000</td>
<td>2012-2013</td>
<td>20K Anti-spam rules&lt;br&gt;5M email training set&lt;br&gt;100+ Nodes in Hadoop Clusters</td>
<td>Vanilla Apache Hadoop (HDFS,MapReduce,Oozie,Zookeeper )&lt;br&gt;Perl/Python&lt;br&gt;SpamAssassin&lt;br&gt;Perceptron</td>
<td>MapReduce was written on Python and Hadoop Streaming was used. The challenge was to optimize jobs performance.&lt;br&gt;• Optimal Hadoop cluster configuration for maximizing performance and minimize map-reduce processing time</td>
</tr>
<tr>
<td>3</td>
<td>Online Coupon Web Analytics Platform&lt;br&gt;US MNC: World’s largest coupon site, 2014 Revenue &gt; US$200M</td>
<td>2012, Ongoing</td>
<td>500 million visits a year&lt;br&gt;25TB+ HP Vertica Data Warehouse&lt;br&gt;50TB+ Hadoop Cluster&lt;br&gt;Near-Real time analytics (15 minutes is supported for clickstream data)</td>
<td>Data Lake - (Amazon EMR)/Hive/Hue/MapReduce/Flume/Spark&lt;br&gt;DW: HP Vertica, MySQL&lt;br&gt;ETL/Data Integration – custom using python&lt;br&gt;BI: R, Mahout, Tableau</td>
<td>• Minimize transformation time for semi-structured data&lt;br&gt;• Data quality and consistency&lt;br&gt;• complex data integration&lt;br&gt;• fast growing data volumes,&lt;br&gt;• performance issues with Hadoop Map/Reduce (moving to Spark)</td>
</tr>
</tbody>
</table>
## Embedded Cases 4-6

<table>
<thead>
<tr>
<th>Case #</th>
<th>Business goals</th>
<th>Start</th>
<th>Big data</th>
<th>Technologies</th>
<th>Challenges</th>
</tr>
</thead>
</table>
| 4     | Social Marketing Analytical Platform | 2012, ongoing | • Volume - 45 TB  
• Sources - JSON  
• Throughput - > 20K/sec  
• Latency (1 hour – for static/pre-defined reports /real-time for streaming data) | • Lambda architecture  
• Amazon AWS, S3  
• Apache Kafka, Storm  
• Hadoop - CDH 5, HDFS(raw data), MapReduce), Cloudera Manager, Oozie, Zookeeper  
• HBase (2 clusters: batch views, streaming data) | • Hadoop upgrade – CDH 4 to CDH 5  
• Data integrity and data quality  
• Very high data throughput caused a challenge with data loss prevention (introduced Apache Kafka as a solution)  
• System performance for data discovery (introduced Redshift considering Spark)  
• Constraints - public cloud, multi-tenant |
| 5     | Cloud-based Mobile App Development Platform | 2013, 8 month | • Data Volume > 10 TB  
• Sources: JSON  
• Data Throughput > 10K/sec  
• Analytics - self-service, pre-defined reports, ad-hoc  
• Data Latency – 2 min | • Middleware: RabbitMQ, Amazon SQS, Celery  
• DB: Amazon Redshift, RDS, S3  
• Jaspersoft  
• Elastic Beanstalk  
• Integration: Python  
• Aria Subscription Billing Platform | • schema extensibility  
• minimize TCO  
• achieve high data compression without significant performance degradation was quite challenging.  
• technology selection: performance benchmarks and price comparison of Redshift vs HPVertica vs Amazon RDS) |
| 6     | Telecom E-tailing platform | End of 2013, (did only discovery) | • Analytics on 90+ TB (30+ TB structured, 60+ TB unstructured and semi-structured data)  
• Elasticity: through SDE principles | • Hadoop (HDFS, Hive, HBase)  
• Cassandra  
• HP Vertica/Teradata  
• Microstrategy/Tableau | • Data Volume for real-time analytics  
• Data Variety: data science over data in different formats from multiple data sources  
• Elasticity: private cloud, Hadoop as a service with auto-scale capabilities |
<table>
<thead>
<tr>
<th>Case #</th>
<th>Business goals</th>
<th>Start</th>
<th>Big data</th>
<th>Technologies</th>
<th>Challenges</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Social Relationship Marketing Platform</td>
<td>• Build social relationship platform that allows enterprise brands and organizations to manage, monitor, and measure their social media programs</td>
<td>2013 ongoing (redesign 2009 system)</td>
<td>&gt; one billion social connections across 84 countries</td>
<td>Cassandra • MySQL • Elasticsearch • SaaS BI Platform - GoodData • Clover ETL, custom in Java, PHP, Amazon S3, Amazon SQS • RabbitMQ</td>
</tr>
<tr>
<td></td>
<td>US private Internet Co. Funding &gt; US$100M</td>
<td>• Build an Analytics module to analyze and measure results.</td>
<td>Cassandra (~ 6Tb), ETL (&gt; 8Tb per day)</td>
<td>MySQL (~ 11 Tb)</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Web Analytics &amp; Marketing Optimization</td>
<td>• Optimization of all web, mobile, and social channels</td>
<td>2014, Ongoing (Redesign 2006-2010 system)</td>
<td>Data Volume &gt; 1 PB</td>
<td>Vanilla Apache Hadoop (HDFS, MapReduce, Oozie, Zookeeper) • Hadoop/HBase • Aster Data • Oracle • Java/Flex/JavaScript</td>
</tr>
<tr>
<td></td>
<td>US MNC IT consulting co. (Employees &gt; 430,000)</td>
<td>• Optimization of recommendations for each visitor</td>
<td>5-10 GB per customer/day</td>
<td>Data sources – clickstream data, webserver logs</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Network Monitoring &amp; Management Platform</td>
<td>• Build tool to monitor network availability, performance, events and configuration.</td>
<td>2014, Ongoing (Redesign 2006 system)</td>
<td>collect data in large datacenters (each: gigabytes to terabytes)</td>
<td>MySQL • RRDtool • HBase • Elasticsearch</td>
</tr>
<tr>
<td></td>
<td>US OSS vendor Revenue &gt; US$ 22M</td>
<td>• Integrate data storage and collection processes with one web-based user interface.</td>
<td>real-time data analysis and monitoring (&lt; 1 minute)</td>
<td>types of devices: hundreds</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IT as a service</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Healthcare Insurance Operation Intelligence</td>
<td>• Operation cost optimization for 3.4 million members</td>
<td>2014, Phase 1: 8 months, ongoing</td>
<td>Velocity: 10K+ events per second</td>
<td>AWS VPC • Apache Mesos, Apache Marathon, Chronus • Cassandra • Apache Storm • ELK (Elasticsearch, Logstash, Kibana) • Netflix Exhibitor • Chef</td>
</tr>
<tr>
<td></td>
<td>US health plan provider Employees&gt; 4,500 Revenue&gt; US$10B</td>
<td>• Track anomaly cases (e.g. control schedule 1 and 2 drugs, refill status control)</td>
<td>Complex Event Processing - pattern detection, enrichment, projection, aggregation, join</td>
<td>High scalability, High-availability, fault-tolerance</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Collaboration tool between 65,000 providers.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
RESULTS

• Big Data System Development Framework
• Big Data system Design (BDD) method
BDD Framework

Value Discovery

Innovation Process
- Use Case Development (may include prototyping)
- Strategic Development Planning (CB analysis; Sourcing decisions; Talent Management)

System Architecture
1. Business goals
2. Constraints/Concerns/Drivers
3. Quality attribute scenarios
4. Big data architecture scenarios

Big Data Modeling
1. Choose Reference Architecture
2. Form architecture landscape
3. DFD
4. Establish iteration goal
5. Choose element(s) to decompose
6. Choose design concepts + data models
7. Instantiate architectural elements
8. Sketch views & record decisions + Metadata
9. Evaluation of each iteration (may include prototyping, scale-up testing)
10. Architecture Analysis and Evaluation: BITAM

Technology Selection
- Talent/Vendors
- Reference Architecture + Frameworks
- Design patterns + tactics + data models
- Big data Technology Catalogue

Implementation

Evaluation: Technical and Business Dimensions
BDD Framework

1. **New Development Process**
   - Data-program independence undone

2. **“Futuring” big data scenario generation for innovation**
   - utilizing Eco-Arch method (Chen & Kazman, 2012).

3. **Architecture design integrated with new big data modeling techniques:**
   - Extended DFD (BD-DFD) , big data architecture template, transformation rules.

4. **Extended architecture design method**
   - ADD 2.0 (by CMU SEI) to ADD 3.0, then to BDD.

5. **Use of design concepts databases** (reference architecture, frameworks, platforms, architectural and deployment patterns, tactics, data models) and a **technology catalogue with quality attributes ratings**.

6. Adding architecture evaluation, BITAM (Business and IT Alignment Model), for risk analysis and ensuring alignment with business goals and innovation desires.
   - BITAM (Chen et.al. 2005, 2010) extended ATAM.
ECO-ARCH Method (Chen & Kazman, 2012)
ECO-ARCH Method (Chen & Kazman, 2012)

Macroscopic Level

Step 1: Goals, Scenarios Brainstorming
Vision, goals
Stakeholders
Architectural choices
TBL scorecard

Step 2: Form Architecture Landscape
Enumerate Architectural Decisions

Step 3: Develop Risk Scenarios
Multiple Quality Attribute Perspectives
Threats to Triple Bottom Line

Step 4: Map Potential Risks
Map Risk Scenarios to the Architecture Landscape

Step 5: Develop Risk Themes
For Strategic Architectural Decision Support

Step 6: 3P Impact Analysis
System-Specific 3P Impact Analysis
Cost-benefit analysis

Microscopic Level

Design based on Expandable Rationality

Design based on Engineering Principles
Big Data Architecture Design: Data Element Template

- A Scenario description includes the 6 elements: source, stimuli, environment, artifacts, response, response metrics.

1) **Data sources**: what are the data used in the scenario, where is it (are they) generated? Answer questions below for each source.

2) **Data source quality**: is this data trustworthy? How accurate does it represent the real world element it represents? Such as temperature taken?

3) **Data content format**: structured, semi-structured, unstructured? Specify subtypes.

4) **Data velocity**: what is the speed and frequency the data is generated/ingested?

5) **Data volume and Frequency**: What is the volume and frequency of data?

6) **Data Time To Live (TTL)**: How long will the data live during processing?

7) **Data storage**: What is the volume and frequency of the data generated that need to be stored.

8) **Data Life**: how long should the data need to be kept in storage? (Historical storage/time series or legal requirements).

9) **Data Access type**: OLTP (transactional), OLAP (aggregates-based), OLCP (advanced analytics)

10) **Data queries/reports by who**: what questions are asked about the data by who? What reports (real time, minutes, days, monthly?)

11) **Access pattern**: read-heavy, write-heavy, or balanced?

12) **Data read/write frequency**: how often is the data read, written?

13) **Data response requirements**: how fast of the data queries needs to respond?

14) **Data consistency and availability requirements**: ACID or BASE (strong, medium, weak)?
Ratings on Quality Attributes

**Cassandra**
*Technology/Data Storage/NoSQL Database/Column-Family*

**Description:** The Apache Cassandra database is the right choice when you need scalability and high availability without compromising performance. Linear scalability and proven fault-tolerance on commodity hardware or cloud infrastructure make it the perfect platform for mission-critical data. Cassandra's support for replicating across multiple datacenters is best-in-class, providing lower latency for your users and the peace of mind of knowing that you can survive regional outages.

**Consequences:**
- **Performance:** Cassandra supports 100% I/O (for both reads and writes) due to efficient memory usage, SSD support, online backups, locally-managed storage, and replication, etc. It is a winner of most performance benchmarks in Cassandra.
- **Reliability:** One of the most reliable and mature NoSQL databases today.

**Impala**
*Technology/Analytic/Search & Query/Distributed Query Processor*

**Description:** Impala is open-source massively parallel processing (MPP) SQL query engine developed for a distributed cluster running Apache Hadoop.

**Consequences:**
- **Performance:** Considered one of the fastest technologies at the moment. Significantly faster than Hive.
- **Reliability:** Impala supports storing data in HDFS, HBase, Amazon S3, and Cassandra. Test, animate, and query files. Uses Hive meta data, can work through ODBC/JDBC.

**Spark SQL**
*Technology/Analytic/Search & Query/Distributed Query Processor*

**Description:** Spark SQL is an in-memory distributed computing engine (an alternative to Hadoop MapReduce). Spark SQL allows running SQL and HiveQL queries over large datasets. Spark SQL is an example of using Apache Spark.

**Consequences:**
- **Performance:** Supports capabilities based on SQL-like query language. Supports most of the HiveQL features including DP/SS and DDL.
- **Reliability:** Performs well on preferred databases, can work through ODBC/JDBC.

**CouchDB**
*Technology/Data Storage/NoSQL Database/Document-Oriented*

**Description:** CouchDB is a database that can be accessed through HTTP, indexing, and transforming your documents with JavaScript. CouchDB works well with modern web and mobile apps, supports incremental replication and master/master setups with automatic conflict detection.

**Consequences:**
- **Performance:** Fast for direct CRUD-like and map/reduce jobs, but slow for CPU and other map/reduce jobs.
- **Reliability:** Various problems with reliability and availability are reported by users. Declarative functionality, like replication and automatic conflict resolution, is not available for highly available or near-tolerant solutions.

**MongoDB**
*Technology/Data Storage/NoSQL Database/Document-Oriented*

**Description:** MongoDB (from "humongous") is an open-source document database and the leading NoSQL database. Written in C++, MongoDB features JSON-style documents with dynamic schemas. It offers simplicity and power, indexes on any attribute, mirrors across URLs and deployments, scale horizontally without compromising functionality, fault aggregation and data processing, etc.

**Consequences:**
- **Performance:** Not as fast as some key-value storage, but features like auto-sharding, full index support, and reduce makes it fast enough. Written in C++ rather than Java. However, it is not as fast as Cassandra.
- **Reliability:** As a document-oriented database, MongoDB handles data loss and disk failures gracefully.

**Apache Hive**
*Technology/Analytics/Search & Query/Distributed Query Processor*

**Description:** The Apache Hive is a data warehousing software layer that provides a familiar SQL-like interface for querying and managing large datasets residing in distributed storage.

**Consequences:**
- **Performance:** Even though the Hadoop framework is still slow compared to other databases such as Impala or Spark SQL.
- **Reliability:** Supports storing data on HDFS, Amazon S3, and Apache's HBase. Can work through ODBC/JDBC.

**Spark**
*Technology/Analytics/Search & Query/Distributed Query Processor*

**Description:** Spark is a fast parallel engine for large-scale data processing. It supports various data sources and formats, including relational databases, and can be used for data munging, data analytics, and machine learning tasks.

**Consequences:**
- **Performance:** Supports capabilities based on SQL-like query language. Supports most of the HiveQL features including DDL/SS and DDL.
- **Reliability:** Performs well on preferred databases, can work through ODBC/JDBC.
BITAM
(Business-IT Alignment Model)

1) *Business Model*: drivers, strategies, revenue streams, investments, constraints, regulations

2) *Business Architecture*: applications, business processes, workflow, data flow, organization, skills

3) *IT Architecture*: hardware, software, networks, components, interfaces, platforms, standards

(Chen, Kazman, & Garg, 2005)
Work-in-Progress/Future Research

1. Prototyping vs. Architecture Analysis
2. Eco-Arch extension: More case studies
3. Decision support system (DSS) for knowledge-based big data technology selection
4. Automation of big data technology cataloguing
5. New big data design patterns for hybrid environment
6. Conceptual design for NOSQL data modeling
7. Metadata management for big data
8. Neo-Metropolis Model: BDaaS, etc.
Conclusions (1)

1. CPR approach balance rigor and relevance.

2. BDD framework describes a new process of big data system development, which is dramatically different from “small” data system development, reflecting the paradigm shifts required for big data system development.

3. Paradigm shifts and complexity in big data management underscore the importance of an architecture-centric design approach.
Conclusions (2)

4. BDD method is the first attempt to extend both architecture design methods and data modeling techniques for big data system design and integrate them in one method for design efficiency and effectiveness.

5. BDD method focuses on “futuring” for innovation.

6. BDD advances ADD 2.0 to ADD 3.0.

7. BDD method embodies best practice of complexity mitigation by utilizing quality attribute driven design strategies, reference architectures, technology catalogue (with ratings) and other design concepts databases for knowledge-based design and agile orchestration of technology.
Implications

- Disruptive Innovation Management
- Software Engineering Education
MAHALO & ALOHA!!!