Integrate End-to-End Early and Often
Driving Out Technical Risk by Blending Architecture and Process Discipline

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA  15213

James McHale, Robert Nord

In collaboration with:
Luis Carballo (Bursatec)
# A New Trading Platform

## Quality Attribute Summary

<table>
<thead>
<tr>
<th>Desired</th>
<th>Actual</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Latency:</strong></td>
<td>1 ms. avg.</td>
</tr>
<tr>
<td><strong>Throughput:</strong></td>
<td>1000/sec/thread</td>
</tr>
<tr>
<td><strong>Scalability:</strong></td>
<td>1 thread/addl. CPU</td>
</tr>
<tr>
<td><strong>Availability (hot):</strong></td>
<td>transparent</td>
</tr>
<tr>
<td><strong>Availability (warm):</strong></td>
<td>&lt; 1 min.</td>
</tr>
<tr>
<td><strong>Modifiability:</strong></td>
<td>common platform</td>
</tr>
<tr>
<td><strong>Testability:</strong></td>
<td>&lt; 1 day</td>
</tr>
</tbody>
</table>

**Actual**

- 70 µs. (.070 ms.) avg.
- >10,000/sec/thread
- √
- √
- √
- √
- √
- < 1 hr.
Experience and Results

Architecture coaching, coupled with the discipline of the Team Software Process, built a competent architecture team and an architecture with successful evaluation quickly – less than six months.

The project objectives were met.

- Schedule – finished on time.
- Quality – reliability and quality goals were met.
- No known defects carried into final cycle!
- Performance – a day’s worth of transactions is processed in seconds.
The Opportunity

Background:

• Bolsa Mexicana de Valores (BMV) operates the Mexican financial markets.

• BMV desired a new trading engine to replace the existing stock market engine and integrate the options and futures markets.

• Bursatec, the technology arm of BMV, committed to deliver a trading engine in 8-10 quarters.

• Bursatec approached the SEI for support during design and development to improve its software delivery capability.
Business Goals:

- High performance (as fast or faster than anything out there)
- Reliable and of high quality (the market cannot go down)
- Scalable (able to handle spikes and long-term growth in trading volume)
The Project -2

Architecture Decisions:
- Development in Java (lower total cost of ownership)
- Low latency communication multicast network
- In memory data storage during trading session
- Hot-hot high availability configuration
- Parallel processing in Java Virtual Machine (JVM)
- Horizontal scalability

Functional Requirements:
- Order routing with FIX protocol interconnect to current legacy systems.
- Combined cash and derivatives markets with a single control workstation.
- Separate market data and index calculation system.
Trading Engine Quality and Other Attributes

<table>
<thead>
<tr>
<th>Quality Attributes</th>
<th>Other Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Under 1ms processing latency</td>
<td>• Backward compatible with current systems</td>
</tr>
<tr>
<td>• Horizontal scalability</td>
<td>• Combined platform for both markets</td>
</tr>
<tr>
<td>• Redundant high availability system</td>
<td>• Run on commodity hardware</td>
</tr>
<tr>
<td>• Warm dual redundant system</td>
<td>• 86 order type/attribute combinations (30 in current system)</td>
</tr>
<tr>
<td>• Automatic testing framework (one day turnaround attribute)</td>
<td>• Real time updates to status of system via control workstation.</td>
</tr>
<tr>
<td>• Localize business rules changes in specific modules</td>
<td></td>
</tr>
</tbody>
</table>
Complicating Factors

Given the context, one would expect risks due to:

- Large project – scope beyond the organization’s recent experience
  - # of person-months
  - # KLOC/function points
  - # of interconnecting platforms
  - # of individual projects
- Inexperienced – available staff talented but young and key implementation technologies never used together formally
- Constant stream of new requirements and changes to business rules
## Solution Integrates High-Value Practices

<table>
<thead>
<tr>
<th>Team Software Process</th>
<th>Architecture-Centric Engineering</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Proven technology</td>
<td>• Proven technology</td>
</tr>
<tr>
<td>• Management and measurement across the project lifecycle</td>
<td>• Technical aspect of the early project lifecycle activities</td>
</tr>
<tr>
<td>• Building high-performance teams</td>
<td>• Architecting to meet business objectives</td>
</tr>
<tr>
<td>• Key managers familiar with technology through word-of-mouth and literature.</td>
<td>• Key managers familiar with technology via training courses.</td>
</tr>
</tbody>
</table>

Architecture drove the work breakdown structure and provided a robust framework for requirements management.
Architecture Drives the Lifecycle

Two iterative processes based on the architecture of the system:

- **Design cycles**
  The goal is to design a system that ensures business success.

- **Implementation cycles**
  The goal is to implement the system according to the design.
Designing a software system is defining structures that support the quality attribute requirements, such as performance, availability, extensibility, and so on.
Scenario-based Peer Reviews

The architects ensure that the current design is checked in a periodic fashion to see if the quality attribute scenarios are continuing to be fulfilled.

- Step 1: Select the scenario to analyze
- Step 2: Elicit the architecture approaches
- Step 3: Analyze architecture approaches
- Step 4: Review results

Performing scenario-based peer reviews every second week was never seen as a burden by the architects. They were actually looking forward to the next review because the reviews provided them with valuable input and they could see progress when the list of risks and the to-do list became smaller and smaller over time.¹

The implementation cycle is centered around establishing communication between architects and developers.
Conformance Reviews

The implementation must conform to the architecture to exhibit the quality attributes engineered at that level.

- A conformance review does not last longer than three hours.
- The developer uses tools to provide evidence showing how the code maps to the architecture element (module).
- For all discrepancies, the developer has to provide a written reason why it is there; the fewer the discrepancies, the less work.
- The team decides how to deal with those discrepancies by changing the code or the architecture.
Agile Practices

As part of defining their team process (a standard TSP practice), certain agile practices were adopted.

- Daily meetings (definitely NOT stand-up!)
- Continuous integration
- Time-boxed iterations early (architectural breakdown drove later iterations)
- Automated testing (testability was a major desired quality attribute)
Architecture Centric Engineering

BUSINESS AND MISSION GOALS

ACE/TSP

ARCHITECTURE

ACE/TSP

SYSTEM

Attribute Driven Design

TSP launch, release planning, checkpoints

Active Design Review

Quality Attribute Workshop

Business Thread Workshop

Architecture Trade-off Analysis Method

Scenario-based peer review

Views and Beyond

Conformance Review
Stakeholders establish and prioritize requirements
TSP-ACE Increments

Stakeholders communicate iteration requirements to the Architecture Team
TSP-ACE Increments

Stakeholders communicate next iteration requirements to the Architecture Team.

Architecture Team assigns iteration tasks to the Developer Team.
TSP-ACE Increments

At each iteration, the Architecture Team delivers status reports to the Stakeholders.

Developer Team delivers latest version of software to the Architecture Team.
TSP-ACE Increments

Developer Team releases version updates to the Stakeholders
TSP-ACE Increments

Regular status reports

Timely releases
TSP-ACE Increments
Project Timeline

Cycle 0 (Sept.-Oct. 2009)
  • Training and QAW/BTW

  • Architecture, basic test framework, and communications benchmarking

Cycle 2 (Jan.-Apr. 2010)
  • Architecture, architecture evaluation, develop functional core, full test framework

Cycle 3 (Apr.-July 2010)
  • First integration: Basic trading functions implemented, main performance loop tweaked to 10X better than quality attribute requirement

Cycle 4 (Aug. 2010)
  • Independent evaluation of JVM, communications implementation, and performance

  • Full normal trading functionality (1 mo. early), performance loops verified

Cycle 6 (Jan.-July 2011)
  • Full functionality including start- and end-of-day and maintenance modes
Select Process Data

Effort distribution by phase blocks (% of total task hours)

- ~208 eKLOC in 24 months
- Complete functionality of previous system and new functionality
- Latency target 1msec, achieved 0.1 msec

- Architectural design practices were 12% of the total cost but were key in meeting the technical requirements and are estimated to have reduced the implementation costs by 10%-15% (due to avoided functionality and clean design)

- Only 15% effort in testing – compared to normally equal distribution between coding and testing; higher than usual quality achieved
Accomplishments

Performance
• Latency and throughput metrics greatly exceeded initial expectations (0.1 msec. vs. 1.0 msec.)

Quality
• Very low defect count in validation test. Error density 0.1 error per KLOC compared to “normal” of 0.5-1.0
• Defects encountered have not modified the architecture
• Testing framework allowed a smooth continuous integration

Cost & Schedule
• Team achieved EVERY Milestone (internal and external) on time and budget (including unplanned new functionality), with the planned number of people. No “forced march”.
Key Takeaways

Investment in early and continuous architecture and team practices drive the lifecycle and plays a role is managing risk.

Constant integration allows earlier identification and resolution of defects and reduces the cost of rework.

Iterative and incremental approach fosters collaboration and facilitates handoffs reducing the cost of delay.

Architectural practices and TSP provide a disciplined framework for measuring and managing structured intellectual activity related to the product, process, and project.
Questions?

For more information: Combining Architecture-Centric Engineering with the Team Software Process, Technical Report, CMU/SEI-2010-TR-031, December 2010
Contact Information

James McHale
Software Engineering Process Management Team Software Process
jdm@sei.cmu.edu

Robert L Nord
Research, Technology, and System Solutions Architecture Practices
rn@sei.cmu.edu

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

SEI website at www.sei.cmu.edu
ADDITIONAL INFORMATION
# Architecture Training

<table>
<thead>
<tr>
<th>Certificate Programs</th>
<th>Certification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software Architecture Professional</td>
<td>ATAM Evaluator</td>
</tr>
<tr>
<td>ATAM Evaluator Training</td>
<td>✓</td>
</tr>
<tr>
<td>ATAM Leader Training</td>
<td>✓</td>
</tr>
<tr>
<td>ATAM Observation</td>
<td>✓</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Six Courses</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Software Architecture Principles and Practices*</td>
<td>✓</td>
</tr>
<tr>
<td>Documenting Software Architectures</td>
<td>✓</td>
</tr>
<tr>
<td>Software Architecture Design and Analysis</td>
<td>✓</td>
</tr>
<tr>
<td>Software Product Lines</td>
<td>✓</td>
</tr>
<tr>
<td>ATAM Evaluator Training</td>
<td>✓</td>
</tr>
</tbody>
</table>

*: required to receive certificate / certification

*: available through e-learning
PSP\textsuperscript{SM} and TSP Training

Personal Software Process (PSP\textsuperscript{SM}) training is essential to successful TSP implementation.

- \textit{TSP Executive Seminar} (1 day for top-level execs, middle managers)
- \textit{TSP Team Leader Training} (3 days for team leads, affected managers)
- \textit{PSP Fundamentals} (5 days for software developers)
- \textit{TSP Team Member Training} (3 days for other disciplines)
# Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACE</td>
<td>Architecture Centric Engineering</td>
</tr>
<tr>
<td>ATAM</td>
<td>Architecture Tradeoff Analysis Method</td>
</tr>
<tr>
<td>BTW</td>
<td>Business Thread Workshop</td>
</tr>
<tr>
<td>TSP</td>
<td>Team Software Process</td>
</tr>
<tr>
<td>QAW</td>
<td>Quality Attribute Workshop</td>
</tr>
</tbody>
</table>