Migrating a Large Scale Legacy IT System to a System with Current Technology

Bill Wood, Mike Gagliardi, Phil Bianco
Background

Agency wanted to Migrate two paired and tightly coupled systems

- Both are 24/7/365 and have disaster recovery at multiple sites
- Both have many classes of users

1. Service Based System - 15% functionality (going to 40%)
   - Java, Relational DB, COTS tools, Service-based, J2EE, SAP, Informatica

2. Legacy System - 85% of functionality (going to 60%)
   - COBOL, Hierarchical DB, mainframe

Migrate to a well defined target reference architecture (TRA) as a basis for a common platform infrastructure (CPI) – developmental, operational, test

• Briefing is focused on the Legacy Migration
Its Complicated

Understanding the Legacy System Architecture
- Infrastructure Tools
- Application Component Relationships (Data and Code)
- Business Process Threads

Understanding the Target System
- Target Reference Architecture (TRA)
  - SOA; Layered Infrastructure
  - Designing Services on top of TRA
  - Business Process threads

Mapping between Legacy and Target in Phases
- Architecture mismatches (development, operational, certification, sustainment, COTS)
- Operating with dual authoritative data systems, cutover, synchronization
- Relationships between Application Components (Legacy versus TRA, COTS)
It’s Worrisome

- Is there too much code/data coupling and spaghetti code to partition for migration?
- Are the current business processes and screens appropriate?
- Is the CPI (and TRA) stable?
- Are COBOL to Java transformation tools up to the job?
- Can we operate with 2 systems overlapping authoritative data?
- Do we have sufficient technology expertise in: legacy system, target reference architecture, discovery and analysis tools, transformation tools?
- Is the business logic only understandable in the legacy code?
- How can we overcome the lack of architectural documentation?
- Will the entire testing and certification process change?
- Will I be creating a maintenance nightmare?
- It’s a 24/7/365 operation- no downtime!
Keep the Right Vision

- The TRA is a good start, but an application architecture with data modeling is needed
- Synchronize with phased TRA and CPI infrastructure deployment
- Operating with multiple sources of authoritative data
- Using new technology (e.g. customized security to infrastructure security)
- Capability of transformative toolsets
Don’t Live in a Dreamworld

• Big-Bang changes usually fail
• **Conducting transactions across networks and keeping response times satisfied!**
• Transforming spaghetti code automatically
  • Separating presentation from business processing from data access!
  • Moving from green screens to windows
• **Making multiple types of changes simultaneously!**
• **Edict no changes to the legacy system!**

Paul Delvaux
Target Reference Architecture

This is a good start
But it is not enough

Need a system and application software architecture
• End-state
• Each Release
Migration Approaches

An RFI was conducted and multiple approaches suggested:

- **Lift and Shift (L&S)**
  - Reduce Operations and Maintenance (O&M) Costs ASAP
  - Fast retirement of legacy infrastructure
  - Move to the CPI
    - Don’t come close to satisfying the TRA
    - Make minimal changes- keep COBOL

- **Modernize**
  - Phased transformation to TRA
    - Keep legacy architecture
    - Change technology- Java, SOA
    - Use Conversion tools?

- **Re-engineer**
  - Re-Develop the system in Java on CPI
  - Developed system and application architecture
  - Satisfy TRA

---

Options Considered:
1. Do nothing (baseline)
2. L&S (baseline)
3. L&S and modernize
4. L&S and re-engineer
5. Re-engineer
6. Hybrid- L&S, modernize, re-engineer
Migration Process

List the Options
Develop evaluation criteria
Score the options

- Goals for sequencing
- Constraints on Phasing
- Approach to Migration
  - Data, code, user, business processes ordering
  - Quality attribute considerations
  - Migration tooling
- Define phases
  - Groups
  - Legacy: functionality, code, data BP, users
  - Tiers/layers
  - Transient code in Legacy and TA
  - Throwaway
  - Align with Infrastructure Roadmap

Determine and Score Options
Explore implementation alternatives for options
Build an End State Architecture
Build a Roadmap

Migration Planning
Determine and Score Options

Determine and Score Options
Explore implementation alternatives for options
Build an End State Architecture
Build a Roadmap
Recommend an Option

Goal- Examples
Reduce operation and maintenance (O&M) costs
Licensing, obsolescence, electric, space, people
Vendor lock
Create technical uniformity across organizational boundaries

Options
6 mentioned previously
Initial crude evaluation against TRA
Detailed evaluation against
Cost, risk, performance factors (23)

Choose an Option
Best Overall Measure - cost, risk, performance
Determine ordering: sequential, concurrently
Important Developmental and Sustainment QAs
Business Process Threads
Users, code, data, interfaces
We started with the team’s architectural knowledge of 40 or so factors. We discovered the OMB factors—19 factors—and merged them into the initial factors—50 or so. We started to weight the factors. Too many factors had an equivalent distribution of weights, so we combined factors with like distributions—23 factors.

### Evaluation Factors

<table>
<thead>
<tr>
<th>Cost</th>
<th>Performance</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>How much will the migration labor cost?</td>
<td>How is data organized wrt TRA: relational, normalized, distributed, partitioned</td>
<td>What will be the impact for data synchronization, parallel run and cutover?</td>
</tr>
<tr>
<td>How much recovery of investment cost after cancellation</td>
<td>How many internal interfaces changes will be needed</td>
<td>How modernized are the user screens wrt the TRA</td>
</tr>
<tr>
<td>How early will cost savings happen? license and support during migration?</td>
<td>How well are the TRA security conditions satisfied: easily changed, no unauthorized access, no data corruption</td>
<td>How well does the system meet the operational attributes: satisfy end-user response under maximum workload, scalability, manageability, load balancing, reliability, availability</td>
</tr>
<tr>
<td>How much will the on-going legacy license and/or support costs post migration?</td>
<td>How much will be the migration labor cost?</td>
<td>How much will be the on-going legacy license and/or support costs post migration?</td>
</tr>
</tbody>
</table>

There was a separate ROM cost team. We developed a ROM for each option and provided them with data as a basis for estimation. Their ROM sizing was about the same as our cost factor sizing.
Selected Hybrid Option- Example

How to partition: L&S and modernize; and Re-engineer

- Significant Business Process Changes – re-engineer
- Layered stable legacy code- L&S and modernize
- Experiment with modernization tools
Explore Implementation Alternatives - Lift and Shift

---

**Factors**

1. Cutover
2. Rollback
3. Synchronize
4. Data Copy
5. Performance

**Issues for Resolution**

- Functionality/Data may be too entangled to group effectively such that authorized data split can take place
- What toolset is best for converting data structure, copying data, modifying APIs & JCL, writing adaptors?
- Replacing transactional support
- Should we do some cleanup before L&S
- Performance

**Design**

- Explore implementation alternatives
- Discovery and Analysis
- Build a Roadmap

**Assumptions**

- The hybrid Analysis showed the business processes, screens, software and data elements to be Lifted and Shifted.
- It is preferred that the L&S be done in stages

---

4/18/2014 Migration Planning
Build an End State Architecture

- Determine and Score Options
- Explore alternatives for options
- Build an End State Architecture
- Build a Roadmap
- Evaluate Target Architecture

- Build a Basis
- Conduct an Architecture Evaluation
  - Identify risks and risk themes
  - Update the architecture to mitigate the risks
  - Keep a risk register

- Identify the Target Application Elements
  - Services, data models
  - References to Legacy elements
  - Interfaces to external actors
  - TRA elements used

- Build an Application Target Architecture
  - Connections between Application elements
  - Business process flows through the elements
  - User screen contents (appropriately mirror the legacy)
  - Map to TRA elements used

- TRA and CPI constraints on End State Architecture

4/18/2014 Migration Planning
Build an Architectural Roadmap

Select a Pilot Migration for First Phase
- Lightweight, low risk, few challenges
- Gain *target* and *transformation* tool experience
- Include data extraction, analysis, and reporting
- Perhaps no cutover
- Happy path functionality
  - But include some quality attribute drivers
- Aligned with TRA infrastructure development schedule
- Identify target architectural elements needed
  - Business Process Threads, Screens
  - Services, data models, COTS tools
- Identify legacy architectural items being migrated
  - Screens, Reports
  - Subsystems (partial), tables (partial), data sources

Identify technical challenges and risks
Tools to assist migration
Summary

Technical
• Criteria for selecting between options
  • OMB guidelines were good
• Plan, but don’t overdo it
• Architect
  • Understand Legacy
  • Include quality attributes
  • High Level End State
• Roadmap
  • Detailed architecture at each phase

Management
• Changing political environment
• Drift and Re-direction can become a way of life