Migrating Legacy Systems to Service-Oriented Architecture (SOA) Environments

Dennis B. Smith
Research, Technology and Systems Solutions (RTSS) Program
System of Systems Practice (SoSP) Initiative
Legacy System Reuse in the SOA Context

Reuse at a higher level

• Reuse of business functionality
• Encapsulation of technical details

Reuse across organizations

• Organizations can “sell” their core business expertise as services.
• Functionality can be acquired as opposed to developed from scratch—potential savings.

Option for leveraging legacy system investment

• In many cases, legacy components can be reused by exposing them as services, independent of vendor, platform, and technology.
Legacy System Reuse Challenges

Reuse at the service level is more complex than reuse at the module or component level.

- From the service provider perspective
  - Designing reusable services requires a different approach, skill set, and mindset
  - Bigger stakeholder community because services are typically reused at organization and sub-organization level
  - Services need to be as generic as possible so that they are of interest to multiple service consumers and at the same time need to add value to potential consumers
- From the service consumer perspective
  - Larger granularity may lead to larger incompatibilities

Challenges can come from the legacy system from itself or from the environment.
Legacy System Challenges

It may not always be possible to reuse functionality of legacy systems by exposing them as services.

• Technical constraints due to the nature of the legacy system
  — A batch system needs to be exposed as a service for an interactive online Web application.

• Immature technology or lack of technology for a particular legacy environment

Cost of exposing a legacy system as services may be higher than replacing it with a new service-oriented system.
Examples of Challenging Legacy System Characteristics

Poor separation of concerns

- User interface code tightly coupled with business function code

Tool availability

- Target is Web Services; XML and SOAP libraries are not available for all legacy platforms.

Architectural mismatch

- The asynchronous call to the service might be in conflict with legacy system synchronous behavior.

Operational mismatch

- The legacy system is batch-oriented, the service user expects an immediate response.

Dependencies on commercial products

- Licensing issues?
SOA Environment Challenges

The characteristics of SOA enable the exposure of legacy system functionality as services.

- Presumably without making significant changes to the legacy systems

The complexity of the migration will largely depend on the characteristics of the SOA environment—some examples:

- User community
- SOA infrastructure technology
- Other SOA entry points
- Legacy system operations
SOA Entry Points

Source: Adapted from AgilePath’s SOA Quad Model™
SMART identifies a pilot project that will help shape a migration strategy for an organization, along with an understanding of cost and risk involved.

SMART analyzes the viability of reusing legacy systems in an SOA environment:

• Does it make sense to migrate the legacy system to services?
• What services make sense to develop?
• What legacy system components can be used to implement these services?
• What changes to components are needed to accomplish the migration?
• What migration strategies are most appropriate?
• What are the preliminary estimates of cost and risk?
• What is an ideal pilot project that can help address some of these risks?
SMART Process Activities

1. Establish Migration Context
2. Describe Existing Capability
3. Describe Target SOA Environment
4. Analyze the Gap
5. Define Candidate Services
6. Describe Existing Capability
7. Describe Target SOA Environment
8. Analyze the Gap
9. Develop Migration Strategy

Migration Feasible? (No or Yes)

© 2009 Carnegie Mellon University

SMART: Process Activities
Version 1.8—MITRE—May 2009
© 2009 Carnegie Mellon University
Establish Migration Context

Understand the business and technical context for migration

- Rationale, goals and expectations
- Technical and business drivers
- Programmatic constraints (e.g. schedule, budget)
- Previous related efforts or analyses

Identify stakeholders

- Who is driving and paying for the effort
- Who knows what about the legacy system and the target SOA environment
- Demand or need for potential services

Understand legacy system and target SOA environment at a high level

Identify a set of candidate services for migration
# Establish Migration Context: SMIG Examples

<table>
<thead>
<tr>
<th>Discussion Topic</th>
<th>Related Questions</th>
<th>Potential Migration Issues</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goal and Expectations of Migration Effort</td>
<td>• What are the business and technical drivers for the migration effort?</td>
<td>• No SOA strategy</td>
</tr>
<tr>
<td></td>
<td>• What are the short-term and long-term goals?</td>
<td>• Goals for migration are not clear.</td>
</tr>
<tr>
<td>High-Level Understanding of Legacy System</td>
<td>• What is the main functionality provided by the legacy system?</td>
<td>• Legacy system knowledge is not available.</td>
</tr>
<tr>
<td></td>
<td>• What is the high-level architecture of the system?</td>
<td>• Architectural mismatch</td>
</tr>
<tr>
<td></td>
<td>• What is the current user interface to the system?</td>
<td>• User interface complexity hard to replicate in service consumers</td>
</tr>
<tr>
<td>High-Level Understanding of Target SOA Environment</td>
<td>• What are the main components in the target SOA environment?</td>
<td>• Target SOA environment has not been identified.</td>
</tr>
<tr>
<td></td>
<td>• Is this the organization’s first attempt to deploy services in this environment?</td>
<td>• No in-house knowledge of target SOA environment</td>
</tr>
<tr>
<td>Potential Service Consumers</td>
<td>• Who are the potential service consumers?</td>
<td>• Consumers for services have not been identified.</td>
</tr>
</tbody>
</table>
Case Study: Establish Migration Context

DoD organization tasked with developing services that can be used by mission planning and execution applications

MSS is a system for comparison of planned mission against current state to determine if corrective actions should be taken
  - In final stages of development

Drivers
  - Migration to services was already a longer-term goal for MSS
  - Make developed services available to all mission planning and execution systems

Requirement to demonstrate the feasibility of one component as a service being used by one mission planning and execution system within 6 months and to migrate the full system to services in two years
Case Study: Establish Migration Context

Standard Web Services environment is target SOA environment

- Not clear that this will be the future environment for the developed services

Representatives from the legacy system and a representative from a mission planning and execution application (service consumer) agreed on the following candidate services

- **AvailablePlans**: Provides list of available plans that are being reasoned about.
- **TrackedTasksPerPlan**: Provides list of tasks that are being tracked for a certain plan.
- **TaskStatus**: Provides the status for a given task in a given plan.
- **SetTaskAlert**: Alerts when a given task in a given plan satisfies a certain condition
Checkpoint for Migration Feasibility

Decision to continue with the process has to be made

Potential outcomes at this point are

- The migration is initially feasible
- The migration has potential but requires additional information to make an informed decision
- The migration is not feasible
Case Study: Migration Feasibility

Decision: Migration feasible

- Availability of stakeholders from the service provider and a service consumer
- Good understanding of the legacy system
- Request-response nature of the identified services
- Reasonable initial mapping of services to components

Migration issues identified in this activity

- Short-term goal for the migration is different from long-term goal migration
  - Work to accomplish the short-term goal might have to be redone in order to accomplish the long-term goal
- System is a single-user, single-plan system
  - When capabilities are migrated to services, it will have to support multiple users and multiple plans
Define Candidate Services

Select a small number of services, usually 3-4, from the initial list of candidate services.

For these candidate services, the end goal is to fully specify inputs and outputs.
Case Study: Define Candidate Services

The list of services identified in the previous step was considered reasonable for analysis.

Inputs and outputs were next identified in detail for each of these services.

Migration issues identified in this activity:

- *SetTaskAlert* requires (1) alert is set up to respond to certain conditions and (2) service consumer is alerted when the condition is reached.
  - Handling of events in service-oriented environments is relatively new—SOA 2.0
- Unclear how the alert mechanism is going to be implemented.
  - SOA infrastructure would need to have a way to call back the service consumer.
  - There might also be firewall issues on the consumer side.
- Complexity of alert conditions is high.
  - Service consumer interface will have to replicate this complexity or conditions would have to be made simpler or limited.
Describe Existing Capability

Obtain descriptive data about legacy components
- Name, function, size, language, operating platform, age of legacy components, etc.

Question technical personnel about
- Architecture and design paradigms
- Complexity, coupling, interfaces
- Quality of documentation
- Component/product dependencies

Gather data about
- Quality, maturity, existing problems
- Change history
- User satisfaction
## Describe Existing Capability: SMIG Examples

<table>
<thead>
<tr>
<th>Discussion Topic</th>
<th>Related Questions</th>
<th>Potential Migration Issues</th>
</tr>
</thead>
<tbody>
<tr>
<td>Legacy System Characteristics</td>
<td>• What is the history of the system?</td>
<td>• Planned development concurrent with service migration</td>
</tr>
<tr>
<td></td>
<td>• Is the system a proof of concept, prototype, under development, in testing, or a fielded system?</td>
<td>• Limited system documentation</td>
</tr>
<tr>
<td></td>
<td>• What system documentation is available?</td>
<td>• Interfaces to other systems will open doors to service consumers.</td>
</tr>
<tr>
<td></td>
<td>• Does the system have interfaces to other systems?</td>
<td>• Single-user system may have problems in a multi-user environment.</td>
</tr>
<tr>
<td></td>
<td>• What are potential locking, persistence, or transaction problems if accessed by multiple users when migrated to services?</td>
<td></td>
</tr>
<tr>
<td>Legacy System Architecture</td>
<td>• What architecture views are available?</td>
<td>• Lack of architecture documentation may lead to underestimation of complexity.</td>
</tr>
<tr>
<td></td>
<td>• What are the major modules of the system and dependencies between modules?</td>
<td>• Tight coupling between user interface code and business logic code increases effort.</td>
</tr>
<tr>
<td></td>
<td>• Is user interface code separate from the business logic code?</td>
<td>• Undocumented violations of design patterns may cause problems.</td>
</tr>
<tr>
<td></td>
<td>• Are there any design paradigms or patterns implemented in the system?</td>
<td>• Key quality attributes may not hold true in a services environment.</td>
</tr>
<tr>
<td></td>
<td>• What are the key quality attributes built into the current architecture of the system?</td>
<td></td>
</tr>
<tr>
<td>Code Characteristics</td>
<td>• What code documentation is available?</td>
<td>• Poor coding practices will increase migration effort.</td>
</tr>
<tr>
<td></td>
<td>• What coding standards are followed?</td>
<td></td>
</tr>
</tbody>
</table>

---

© 2009 Carnegie Mellon University
Case Study: Describe Existing Capability

MSS characteristics

- In demonstration state
- Written in C++, C# and Managed C++ in a Visual Studio 2005 development environment
- Runs on a Windows XP platform
- Size of the full system is approximately 13,000 lines of code
- Code documentation was rated between *Fair* and *Good* by its developers

Several architecture views were presented that were useful for understanding the system

MSS relies on an external planning system (PS) for plan data and situational awareness data

- PS is being targeted for migration to services in the future

Migration issues identified in this activity

- Documentation for most of the analyzed classes was determined *Fair*
  - Could be an issue if original developers do not perform the migration
- Currently a large amount of communication between MSS and PS
  - Unclear how performance will be affected when this communication takes place using services (they currently reside on the same machine)
- Task alert functionality is not currently implemented in MSS
  - Still unknowns about the specifics of the implementation
Describe Target SOA Environment

- Identify the impact of specific technologies, standards, and guidelines for service implementation
- Determine state of target SOA environment
- Identify how services would interact with the SOA environment
- Determine QoS expectations and execution environment for services
## Describe Target SOA Environment: SMIG Examples

<table>
<thead>
<tr>
<th>Discussion Topic</th>
<th>Related Questions</th>
<th>Potential Migration Issues</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SOA Environment</strong></td>
<td>• What is the status of the target SOA environment?</td>
<td>• Target SOA environment undefined</td>
</tr>
<tr>
<td><strong>Characteristics</strong></td>
<td>• What are the major components of the SOA infrastructure?</td>
<td>• Redundancy/conflicts between infrastructure services and legacy code</td>
</tr>
<tr>
<td></td>
<td>• Does the target SOA environment provide infrastructure services (i.e., communication, discovery, security, data storage)?</td>
<td>• Lack of tools to support legacy code migration to target infrastructure</td>
</tr>
<tr>
<td></td>
<td>• What is the communication model?</td>
<td>• Compliance with constraints requires major effort.</td>
</tr>
<tr>
<td></td>
<td>• What constraints does the target SOA environment impose on services?</td>
<td>• Architectural mismatch</td>
</tr>
<tr>
<td></td>
<td>• Does the legacy system have any behavior that would be incompatible with the target SOA environment?</td>
<td>• No thought given to service deployment and execution</td>
</tr>
<tr>
<td></td>
<td>• Once developed, where will services execute?</td>
<td></td>
</tr>
<tr>
<td><strong>Support</strong></td>
<td>• Do you have to provide automated test scripts for the services and make them publicly available?</td>
<td>• Underestimation of effort to provide service consumer support</td>
</tr>
<tr>
<td></td>
<td>• How will service consumers report problems and provide feedback?</td>
<td>• Lack of awareness of support requirements</td>
</tr>
<tr>
<td></td>
<td>• How will service consumers be informed of potential changes in service interfaces and downtime due to upgrades or problems?</td>
<td></td>
</tr>
</tbody>
</table>
Case Study: Notional Service-Oriented System Architecture

SMART Case Study: Describe Target SOA Environment
Case Study: Describe Target SOA Environment

Migration issues identified in this activity

- Not known if the identified publish-subscribe component to facilitate alerts will allow someone to subscribe on behalf of a third party
  - If not, the service consumer will have to be aware of the dependency on the publish-subscribe component
  - Ideal situation would be for the SetTaskAlert service code to subscribe on behalf of the service consumer, so that the service consumer is not affected if the alert mechanism changes
- If the service consumer has to be set up as a Web server, it would have to be configured so that it accepts incoming messages from the publish-subscribe component
  - Potential security concern
Analyze the Gap

- Define effort, risk and cost to migrate legacy components, given candidate service requirements and target SOA characteristics
- Determine need for additional analyses
Case Study: Analyze the Gap

Developers were asked to

- Describe the details of the changes that would have to be made to the code given the service requirements, the service inputs and outputs, as well as the characteristics and components of the target SOA environment
- Provide an estimate of the effort required to make these changes

No code analysis or architecture reconstruction was necessary because

- Original developers were involved in the process
- Input was credible
- Architecture documentation and knowledge of the system were acceptable
## Exercise: Analyze the Gap—Updated Component Table

<table>
<thead>
<tr>
<th>COMPONENT</th>
<th>MIGRATION FACTORS</th>
<th>MIGRATION ESTIMATES</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ID</strong></td>
<td><strong>Component Name</strong></td>
<td><strong>Migration Method</strong></td>
</tr>
</tbody>
</table>
| 1               | ComparisonEngine    | New + Extraction       | 1. Add methods to store and retrieve plan name and IDs  
                        2. Add class to process service requests from all 4 services  
                        3. Make changes to handle multiple plans  
                        4. Define structure of a condition | Medium | Low | 5 | $ |
| 2               | Analyzer            | New + Extraction       | 1. Add methods to get tasks by plan  
                        2. Modify all methods that retrieve tasks to retrieve tasks per plan | Low | Low | 1 | $ |
| 3               | Task                | New + Extraction       | 1. Add methods to get and set plan that a task is connected to  
                        2. Modify constructor to set new attribute  
                        3. Modify toXML and fromXML to serialize and deserialize new attribute | Low | Low | 1 | $ |
| 5               | AlertCondition      | New + Extraction       | Option 1:  
                        1. Add method to allow dynamically created parameters  
                        2. Modify constructor to initialize parameters  
                        3. Modify toXML to serialize parameters  
                        4. Add fromXML method to deserialize a condition | Medium | Low | 2 | $ |
| 6               | Query               | New + Extraction       | Option 2:  
                        - Add class for nodes to represent a task  
                        - Add class for nodes to represent a task status  
                        - Modify xml2Query class to serialize task and task status | Medium | Medium | 2 | $ |
| 7               | Alert               | New + Extraction       | Option 2:  
                        - Add triggers to send an alert to alert component  
                        - Make changes to constructor to deserialize task and task status | Medium | Medium | 2 | $ |
| 8               | AlertEngine         | New + Extraction       | Option 2:  
                        - Send alert to alert component | Medium | Medium | 2 | $ |
| **TOTALS**      |                      |                      |                                |                      |          |                      |          |
| Option 1 for SetTaskAlert | 20 | | | | | |
| Option 2 for SetTaskAlert | 24 | | | | | |
| Without SetTaskAlert | 11 | | | | | |
| Without SetTaskAlert and without separation from PS | 7 | | | | | |
Develop a migration strategy that makes sense for the organization and addresses the identified migration issues, e.g.

- Feasibility, risk and options for proceeding with the migration effort
- Identification of a pilot project
- Order in which to create additional services
- Guidelines for identification and creation of services
- Options for source of service implementation code
- Mechanisms for providing service functionality
- Specific migration paths to follow
- Needs for additional information, training, technology evaluation, …
1. Define Scope of Initial Migration for the Short-Term Feasibility Demonstration

If the decision is not to separate the services from PS for this short-term feasibility demonstration, it should be done as part of a subsequent iteration in preparation for the long-term goal for MSS.

Implementation of `SetTaskAlert` should meet the long-term goals for MSS and have the least impact on service consumers in terms of usability (from a service interface perspective) and performance.

<table>
<thead>
<tr>
<th>Migration Option</th>
<th>Effort (person-weeks)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implement <code>SetTaskAlert</code> service using a Query package developed for use in another system (Option 2)</td>
<td>24</td>
</tr>
<tr>
<td>Implement <code>SetTaskAlert</code> service using functionality in the legacy system (Option 1)</td>
<td>20</td>
</tr>
<tr>
<td>Do not implement the <code>SetTaskAlert</code> service</td>
<td>11</td>
</tr>
<tr>
<td>Do not implement the <code>SetTaskAlert</code> service and do not separate MSS out from PS</td>
<td>7</td>
</tr>
</tbody>
</table>
2. Define the Scope of Later Iterations

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Focus of Iteration</th>
<th>Effort (person-weeks)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Implement AvailablePlans, TrackedTasksPerPlan, and TaskStatus</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>Separate MSS from PS</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>Implement SetTaskAlert</td>
<td>Option 1: 9 Option 2: 13</td>
</tr>
<tr>
<td>4</td>
<td>Add support for multiple users and multiple plans</td>
<td>TBD</td>
</tr>
<tr>
<td>5</td>
<td>Migrate to the proprietary SOA environment</td>
<td>TBD</td>
</tr>
</tbody>
</table>

Later iterations will depend on additional services to be created from MSS as well as progress made in the migration of PS to services.
3. Finalize Service Inputs and Outputs

Service inputs and outputs in the Service Table need to be concretely defined in WSDL documents.

The structure for conditions in SetTaskAlert still needs to be defined—for this or a future iteration, depending on scope selection.
4. Gather More Information about the Alert Component

Gather additional information about the publish-subscribe component to be used as the mechanism for alert capability to answer these questions:

- Is it possible for the SetTaskAlert component to subscribe on behalf of the service consumer?
  - If it is, the internet protocol (IP) address for the service consumer has to be passed as an input.
  - If it is not, the service consumer has to be aware that it needs to subscribe to the publish-subscribe component.

- What type of alert should SetTaskAlert or the service consumer subscribe to?

- What are the requirements on the service consumer side to receive alerts?
A reference architecture to be followed by all services would provide

- A framework for service development
- The reusability of common service operations
- If done properly, the isolation of service code from changes due to the differences between short-term and long-term goals for MSS

---

**Service Interface Layer**
Performs transformations between messages from service consumers and service code

**Service Code Layer**
Contains existing service code plus new code developed to meet service requirements

**Data Access Layer**
Contains code to access external data sources

**Alert Setup Layer**
Contains code to set up alerts

---

Isolation from changes in data sources and incompatibilities in data structures as PS migrates to services.

Isolation from the evolution of the messages as the target SOA environment changes.

Isolation from changes in alert mechanism.
6. Adjust Estimates

The estimates provided in the *Component Table* are based on a preliminary understanding of the inputs and outputs, as well as a high-level look at the code.

After scope, inputs, outputs, and requirements are refined, the estimates will need to be adjusted.
7. Create MSS Services Incrementally Using the Service Reference Architecture

After defining the scope for the initial and later iterations, migration and development should start as soon as possible to take advantage of MSS developer knowledge.

In parallel with the migration and development, the service reference architecture should be implemented and refined.
8. Conduct a Workshop with Potential Service Consumers

Goal of the Workshop

• Share MSS migration plans
• Share timetable for service release schedule
• Gather and refine service consumer needs
• Discuss any support to be provided for use of MSS services
The pre-requisite of the current SMART is the identification of a target SOA environment

Reality is that

- Many organizations are at earlier stages in the SOA adoption process
- There are multiple entry points to SOA adoption

We have begun to identify variations on the SMART process to deal with these differences

The members of the SMART Family follow the same overall structure described earlier, but the emphasis is on certain activities in the process where the SMIG has been enhanced to go into more detail in specific areas
SMART Family

SOA Adoption Feasibility
Helps an organization establish the feasibility of SOA adoption and creates a high-level migration strategy if it is feasible

SMART-AF

SMART-ENV

SOA Environment
Helps an organization understand a target SOA environment in detail, including associated costs and risks of migrating to that environment

SMART-MP

Migration Pilot
Helps an organization select a pilot project that includes a migration strategy with understanding of costs and risks involved

SMART-ESP

Enterprise Service Portfolio
Helps an organization select and create services from its systems portfolio

SMART-SYS

Service-Oriented Systems Development
Helps an organization understand a complete service-oriented system—services, consumers, environment—including risk and cost data
Resources and Training

SMART Report

- [http://www.sei.cmu.edu/publications/documents/08.reports/08tn008.html](http://www.sei.cmu.edu/publications/documents/08.reports/08tn008.html)

Public Courses

- Migration of Legacy Systems to SOA Environments
  [http://www.sei.cmu.edu/products/courses/p59b.html](http://www.sei.cmu.edu/products/courses/p59b.html)
- SMART Training Workshop
  [http://www.sei.cmu.edu/products/courses/p73.html](http://www.sei.cmu.edu/products/courses/p73.html)

Certification

- SMART Team Lead
  [http://www.sei.cmu.edu/certification/soasmart.html](http://www.sei.cmu.edu/certification/soasmart.html)
NO WARRANTY

THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN “AS-IS” BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

Use of any trademarks in this presentation is not intended in any way to infringe on the rights of the trademark holder.

This Presentation may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.
# Three Elements of SMART

<table>
<thead>
<tr>
<th>Process</th>
<th>SMART Interview Guide (SMIG)</th>
<th>Artifacts</th>
</tr>
</thead>
</table>
| Gathers information about | Guides discussions in initial SMART activities | • Stakeholder List  
• Characteristics List  
• Migration Issues List  
• Business Process-Service Mapping  
• Service Table  
• Component Table  
• Notional Service-Oriented System Architecture  
• Service-Component Alternatives  
• Migration Strategy |
| • Goals and expectations of migration effort  
• Candidate services  
• Legacy systems  
• Target SOA environment | | |
| Analyzes gap between legacy and target state | | |