Guidelines for Acquisition Planning

Ceci Albert
John Bergey
Wolf Goethert
Ed Morris

Sponsored by the U.S. Department of Defense
© 2004 by Carnegie Mellon University
System Acquisition Approach -1

“We Got it Covered” Approach

What software? I am buying a system – my contractor will take care of all of the implementation issues!
System Acquisition Approach -2

“Let’s Cross that Bridge When We Come to It” Approach

Software is inherently flexible – so define the rest of the system first and then we can define and build the software
System Acquisition Approach -3

“Attack the High Risk Issues at the Outset” Approach

Software poses major system risk – give software issues full consideration and adequately address them from the start
Purpose of the Guidelines

Help project managers select and defend acquisition strategies that explicitly consider and mitigate the software risks in their software-intensive system acquisition

• Provide a framework for effectively reasoning about the software risks in the project

• Provide the insights necessary to mitigate those risks in design of the project’s acquisition strategy

• Create a shared understanding of why specific strategies have been selected from among the myriad of possibilities
To Mitigate Software Risks

Profile the software risk in the project early - and continuously - so that stakeholders can make reasonable mitigation decisions.

Create - and update - the program’s acquisition strategy based on an understanding of the program’s exposure to software risk.

Reason about and defend the efficacy of a given acquisition strategy based on its ability to mitigate the software risk.
Determining Exposure to Software Risk

A primary concern in acquisition planning is understanding the degree to which software components in the system pose risk.

The level of software risk depends on

• The amount of software in the system
• The importance of software performance to system operation
• The precedence or difficulty of a given software component to build and/or integrate with other system component
System Software Risk Elements

Scale of Software

- Software is an insignificant portion of system
- Software is the system

Dependence on Software

- Little dependence
  - Mission not limited by software
  - Minimal consequences.
- Very dependent
  - Mission failure likely from software failure.
  - Severe consequences

Complexity of Software

- Low complexity
- High complexity

Low risk  High Risk
System Software Risk Elements

Scale of Software

- Software is an insignificant portion of system
  - Program A
- Software is the system

Dependence on Software

- Little dependence
  - Mission not limited by software
  - Minimal consequences.
  - Program A
- Very dependent
  - Mission failure likely from software failure.
  - Severe consequences

Complexity of Software

- Low complexity
  - Program A
- High complexity

The arrows represent the judgment of the program manager.
The level of software risk for a given project can be profiled in fifteen elements.
## Risk Elements

<table>
<thead>
<tr>
<th>Element</th>
<th>Low Risk</th>
<th>High Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements</td>
<td>Sub-Strategies</td>
<td>Sub-Strategies</td>
</tr>
<tr>
<td>Project Office</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contract</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Life Cycle</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

© 2004 by Carnegie Mellon University
Risk and Acquisition Strategies

Element

Low Risk  High Risk

Acquisition Planning Sub Strategies

Requirements  Program A  High Risk

Sub-Strategies  Sub-Strategies  Sub-Strategies

Project Office

Contract

Life Cycle

© 2004 by Carnegie Mellon University
For Example: Specification Risks

Stable, fully defined, unambiguous, consistent, complete, testable software requirements are rare.

- Some requirements are firm from the start
- Some requirements cannot be defined until other things about the system are known
- Some requirements may be in a constant state of flux as technology, off-the-shelf product, mission needs (or the understanding of what is needed) evolve.

Trying to fully define software requirements too early or trying to limit requirements changes in a changing environment may be riskier than having flexible requirements.

The acquisition strategy needs to accommodate the degree to which requirements can or should change.
### Specification

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Well-defined, complete, and stable</th>
<th>Incomplete or volatile</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Reqs</strong></td>
<td>Implement strong process oversight to control changes</td>
<td>Flexible, prioritized and negotiated requirements</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Nimble process to manage and communicate changes</td>
</tr>
<tr>
<td><strong>Project Office</strong></td>
<td>Limited oversight required</td>
<td>Increased need for engineering staff to monitor system design/progress</td>
</tr>
<tr>
<td><strong>Contract</strong></td>
<td>Consider fixed price contract</td>
<td>Avoid completion contracts (use Cost-plus services contract?)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Offer incentives for delivered system performance</td>
</tr>
<tr>
<td><strong>Life Cycle</strong></td>
<td>“Waterfall” approach</td>
<td>Spiral approach</td>
</tr>
<tr>
<td></td>
<td>Favorable terms for O&amp;M may be defined with development contract</td>
<td>May not be able to award an O&amp;M contract better understood</td>
</tr>
</tbody>
</table>
## Specification

<table>
<thead>
<tr>
<th>Well defined and stable</th>
<th>Incomplete or volatile</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Reqs</strong></td>
<td></td>
</tr>
<tr>
<td>Implement strong process oversight to control changes</td>
<td>Flexible, prioritized and negotiated requirements</td>
</tr>
<tr>
<td>Limited oversight required</td>
<td>Nimble process to manage and communicate changes</td>
</tr>
</tbody>
</table>

| **Project Office** |                        |
| Limited oversight required | Increased need for engineering staff to monitor system design/progress |

| **Contract** |                        |
| Consider fixed price contract | Avoid completion contracts (use Cost-plus services contract?) |
| Offer incentives for delivered system performance |

| **Life Cycle** |                        |
| “Waterfall” approach | Spiral approach |
| Favorable terms for O&M may be defined with development contract | May not be able to award an O&M contract better understood |
## Specification

<table>
<thead>
<tr>
<th>Well defined and stable</th>
<th>Strategy A2</th>
<th>Incomplete or volatile</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Reqmts</strong></td>
<td></td>
<td><strong>Strategy A2</strong></td>
</tr>
<tr>
<td>$ Implement strong process oversight to control changes</td>
<td>• Isolate the affected requirements so the changes are obvious</td>
<td>$ Flexible, prioritized and negotiated requirements</td>
</tr>
<tr>
<td>$ Limited oversight required</td>
<td>• Track technology evolution to identify commitment point</td>
<td>$ Nimble process to manage and communicate changes</td>
</tr>
<tr>
<td><strong>Project Office</strong></td>
<td></td>
<td><strong>Strategy A2</strong></td>
</tr>
<tr>
<td>$ Consider fixed price contract</td>
<td>• Separately price unknown requirements – incentivize low cost</td>
<td>$ Avoid completion contracts (use Cost-plus services contract?)</td>
</tr>
<tr>
<td>$ “Waterfall” approach $ Favorable terms for O&amp;M may be defined with development contract</td>
<td>• Plan and budget for changes across the life of the system</td>
<td>$ Spiral approach</td>
</tr>
<tr>
<td>$ May not be able to award an O&amp;M contract better understood</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Well-defined, complete, and stable**
- **Selected requirements depend on volatile technology**
- **Incomplete or volatile**
Project Profile (Composite of Elements)

Acquisition

Strategy

System

Scope of Software
Dependence on Software
Complexity of Software

Software

Implementation
External Integration
Testing
Maintenance

Program

Policy / Mandates
Cost
Schedule
Program Office
Chain of Command

Scope of Software
Dependence on Software
Complexity of Software

Acquisition

Strategy

Program

Policy / Mandates
Cost
Schedule
Program Office
Chain of Command

Scope of Software
Dependence on Software
Complexity of Software

Acquisition

Strategy

Program

Policy / Mandates
Cost
Schedule
Program Office
Chain of Command

Scope of Software
Dependence on Software
Complexity of Software

Acquisition

Strategy

Program

Policy / Mandates
Cost
Schedule
Program Office
Chain of Command

Scope of Software
Dependence on Software
Complexity of Software

Acquisition

Strategy

Program

Policy / Mandates
Cost
Schedule
Program Office
Chain of Command

Scope of Software
Dependence on Software
Complexity of Software
Next Steps in Use of Sliders

Validate the approach and the set of sliders by profiling the software risk in selected Army programs using the sliders

Show how each program’s current acquisition strategy relates to their identified software risk

Pilot use of Guidelines in a new start

Document the Guidelines
Contact Information

Ceci Albert
cca@sei.cmu.edu