Software Architecture Evaluation

Transforming a craft into a business process

2 March 2005

George Haley
Program Manager
Northrop Grumman Corporation
The Architecture Evaluation Toolkit
Why is evaluating architecture so hard?

- **Intangibility**: Architecture is only a framework, so most standard TPMs don’t apply directly, such as
  - SLOC/code size
  - Processing resources
  - Memory resources
  - Storage resources
  - I/O resources
- Evaluation criteria can be **esoteric** and **obscure**, lacking in good operational definitions
- **Interrelationships** among criteria are highly complex
  - Non-trivial hierarchy
  - Conflicting criteria*
- Certain criteria are difficult to apply & use for evaluation

* Refer to B. Boehm/H. In presentation in GSAW 1998
Presentation overview

1. Identifying bottom-line architecture evaluation criteria
2. Defining the impact tree/matrix
3. Selecting architecture evaluation criteria
4. Evaluating architecture candidates
1. Identify bottom-line criteria

- Focus is on stakeholder needs, not details
- Only five criteria really matter*
  - Utility (primary missions, new missions, product line)
  - Development Cost
  - Development Schedule
  - Development Risk
  - O&M Cost
- Development Schedule is almost always directly correlated to Development Cost

* given that requirements are met
2. Define Impact Matrix/Tree

- Determine candidate observable criteria
2. Define Impact Matrix/Tree

- Determine candidate observable criteria
- Derive any candidate intermediate criteria needed to relate to bottom-line criteria
2. Define Impact Matrix/ Tree

- Determine candidate observable criteria
- Derive any candidate intermediate criteria needed to relate to bottom-line criteria
- Establish and (preferably) quantify relationships between criteria
## 2. Define Impact Matrix/ Tree (cont.)

### Example Impact Matrix

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Quality of Service (performance, availability, etc.)</td>
<td></td>
<td>++</td>
<td>++</td>
<td>+</td>
<td>--/++</td>
</tr>
<tr>
<td>Implementability (e.g., reuse, tools avail., skills avail.)</td>
<td>N/A</td>
<td>--</td>
<td>-/+</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Interoperability</td>
<td></td>
<td>++</td>
<td>+/++</td>
<td>N/A</td>
<td>--</td>
</tr>
<tr>
<td>Extensibility</td>
<td></td>
<td>++</td>
<td>++</td>
<td>+</td>
<td>--</td>
</tr>
<tr>
<td>Portability</td>
<td></td>
<td>+/++</td>
<td>+/++</td>
<td>+</td>
<td>--</td>
</tr>
<tr>
<td>Scalability</td>
<td></td>
<td>++</td>
<td>+/++</td>
<td>+</td>
<td>--</td>
</tr>
<tr>
<td>Supportability</td>
<td>N/A</td>
<td>++</td>
<td>+</td>
<td>--</td>
<td></td>
</tr>
</tbody>
</table>

*Directly related to Development Schedule*
2. Define Impact Matrix/Tree (cont.)

Example Impact Tree (bottom part)

- Quality of Service
- Extensibility
- Supportability
- Implementability
- Interoperability
- Portability
- Scalability
- Availability
- System Performance
- Compatibility With WBS or Product Line
- Security
- Resource & Logistics Requirements
- Vendor Support
- Understandability
- Safety
- Coupling
- Factorization/Cohesion
2. Define Impact Matrix/Tree (cont.)

Example Impact – Quality of Service

- Quality of Service
  - Availability
  - System Performance
  - Security
    - Safety
2. Define Impact Matrix/Tree (cont.)

Example Impact – Extensibility

- Extensibility
  - Coupling
  - Factorization/Cohesion
  - Understandability
2. Define Impact Matrix/Tree (cont.)

Example Impact – Supportability
3. Select Architecture Evaluation Criteria

- Evaluation criteria for architecture evaluation criteria:
  - **Importance** - How strong is the relationship between the criterion and the bottom-line criteria?
  - **Understandability** – Is the operational definition of the criterion clear, unambiguous and agreed-to?
  - **Feasibility** - How feasible is it to evaluate architecture using this criterion? Can automated collection/analysis techniques be used?
  - **Canonical completeness** – Is this a complete basis set of criteria? Is there avoidable redundancy among the criteria?

- Consider impact of each criterion on each stakeholder:
  - Software developers
  - Software development managers
  - Software maintainers
  - Operators
  - System administrators
4. Evaluate Architecture Candidates

- Use only bottom-line criteria for final decision
- Transform subjective or “religious” issues into bottom-line criteria
  - Standard vs. proprietary vs. blend
  - Redundancy/fault handling
  - Reusability
- Automate whenever feasible
- Involve stakeholders appropriately in evaluation (or at least review of evaluation results)
  - Usually understand impacts to themselves better than architects do
  - Helps secure buy-in for the chosen architecture
Summary - Key Points

- Use only bottom-line criteria for final architecture evaluation
- Critically evaluate the detailed evaluation criteria
- Consider stakeholder needs and involve them whenever possible