Requirements Validation

Requirements Management
Two Views of Quality Assessment

- **Validation**
  - Do requirements correctly capture stakeholder goals, needs, and constraints?
  - Do stakeholders have a common understanding of the requirements?
    - **Consensus** on trade-offs for conflicting needs?
  - Requirement attributes sufficiently understood?
  - Complete, unambiguous, testable, consistent, modifiable, etc.?
  - “Are we building the right product?”

- **Verification**
  - Does deployed system actually satisfy requirements?
    - Ensure correctness from activity to activity in the software development process
  - “Did we build the product right?”
Requirements Validation Challenges

- What is **truth** and what is **knowable**?
  - **Requirements** are an expression of the real world problem
  - **Validation** is observation the problem is expressed correctly
  - Cannot prove only refute correctness through observation
- Stakeholder **disagreement**
Requirements Specification Validation

- Is this **subset** of requirements ready for design and implementation?
- If valid, **baseline** this subset and manage change
Negotiation Principles

- Use a four step solution process
  - Separate the people from the problem
  - Focus on interests, not positions
  - Invent options for mutual gain
  - Insist on using objective criteria

Negotiation

- There are various theories and techniques for negotiation – good knowledge to have in your toolset
- Here is an example based on a process called Theory W Win-Win for requirements negotiation between stakeholders and software engineers

Software Requirements Negotiation and Renegotiation Aids: A Theory-W Based Spiral Approach; Barry Boehm, et al
Requirements Validation Techniques

- Reviews
- Interface prototyping
- Analysis modeling
- Architecture - incremental design (quality attributes)
- Acceptance tests
- Observation of operational system
  - Real users, systems, and world environment
  - Alpha, beta versions
Review Techniques

- Personal review
- Informal peer review
- Informal walkthrough
- **Formal inspection**

Defects found in requirements would cost …

- 10 times more to remove if not discovered until implementation
- 100 times more to remove if not discovered until deployment
Inspection Guidelines

- **Plan and prepare** for the inspection
  - Review the inspection process
  - Prepare a task list and schedule
  - Assign inspection roles
    - Author, moderator, reader, recorder
  - Assemble inspection materials
  - Prepare inspector checklists
  - Review ahead of the meeting

- **Set an agenda** for the inspection meeting and stick to it

  See the Defect Checklist in Wiegers Fig. 17-4

The inspection pattern
Inspection Guidelines (cont)

- The product should be inspected in small “chunks”
- Meetings should be at least one hour, but no more than two hours
- Inspect the work product, not the author
- Limit debate and rebuttal in the inspection meeting
- Identify problems, do not attempt to solve them
- Take written notes of the meeting; collect effort and defect data
- Limit the number of participants and insist upon advanced preparation
Acceptance Tests

- Designing test cases will reveal problems with requirements (even if you don’t execute the tests)
  - Functional requirements and quality attributes (fit criteria)
  - Vague or ambiguous requirements inhibit test case definition
- Develop requirements and tests together
- Have customers write acceptance criteria
  - Avoids tester pattern bias – results can be surprising!
- Write test cases for normal flow, alternative flows, and exceptions – i.e., achieve test coverage
Requirements Define Tests for Verification (Tests are a statement of requirements!)

System verification: Confirm that the system design and implementation satisfies the (validated) requirements
Requirements Management

- Requirements Elicitation
- Requirements Analysis
- Requirements Management
- Requirements Validation
- Requirements Specification
Requirements Management Activities

- Requirements **change control**
- **Trace** requirements element relationships through the life cycle
- **Version control**
- **Track requirements attributes** (priority, volatility, cost, benefit, etc.)

Someone on the team should “own” the requirements management activities
Establish Requirements Baseline

- Set of requirements committed to implementation in a specific increment or release
- Agreement between the stakeholders and the developers
Managing Change

- Software **requirements will change** – additions, deletions, modifications
- **Uncontrolled change** is a common source of project chaos, schedule slips, and quality problems.
- Every proposed change is carefully considered and approved
- Approved changes are communicated
- The change process is as simple as possible (but no simpler)

Change always has a price!
Requirements Change Activities

- Propose changes
- **Analyze the impact** of the proposed change
  - Software artifacts
  - Cost/benefit/risk trade-offs
  - Business impact
- **Make decisions** about the proposal
- Update plans
- **Implement the change**
  - Update ALL software artifacts
  - Test changed functionality and regression test unchanged functionality

Don’t agree to backdoor change requests!
The Change Control Board

Change Request

Business and Development Stakeholders

Impact Analysis

Change Control Board

Baseline Requirements

Update

Change Decision: Approved, Rejected, Deferred
Requirements Tracing

- Documents the dependencies between requirements and other system elements, such as:
  - Use cases
  - Business rules
  - Architecture and design components
  - Code modules
  - Test cases

- Claim: the benefits of tracing requirements, and the risks of not doing so, are greater than the cost
Why Trace Requirements?

- Help assess **change impact**
- Displays **test coverage**
- Facilitates **reuse, refactoring, maintenance**
- Helps **project management:**
  - Planning and scheduling, resource allocation
  - Estimating and tracking feature costs
  - Tracking project status
    - e.g., requirements count as a project metric
Traceability Matrix

- A requirements traceability matrix is used to maintain and trace links between software requirements and related project elements

<table>
<thead>
<tr>
<th>User Requirement</th>
<th>Functional Requirement</th>
<th>Package</th>
<th>Class</th>
<th>Method</th>
<th>Test Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>UC-28</td>
<td>ChangeGrade</td>
<td>UserFeatures</td>
<td>Student</td>
<td>ChangeGrade()</td>
<td>Student1</td>
</tr>
</tbody>
</table>
Version Control

- Requirements are allocated to development iterations
- Requirements change
- Just like code, requirements need some form of version control
- Alternatives:
  - Label requirements and SRS revisions
  - Version control tool
  - Use a requirements management tool
Track Requirements Attributes

- A requirements attribute matrix is used to maintain the state or status of requirement attributes being tracked

<table>
<thead>
<tr>
<th>Requirement ID</th>
<th>Requirement Name</th>
<th>Priority</th>
<th>Status</th>
<th>Due Date</th>
<th>Release</th>
</tr>
</thead>
<tbody>
<tr>
<td>UC-28</td>
<td>ChangeGrade</td>
<td>High</td>
<td>In Test</td>
<td>4/26/10</td>
<td>R1</td>
</tr>
</tbody>
</table>
Requirements Management: The Essential Activity

- Make *informed* decisions in response to new or changed requirements
  - Defer lower-priority requirements
  - Increase staff
  - Increase staff time (overtime)
  - Slip the schedule
  - Let the quality suffer
  - Just say No! (and explain why)

**TANSTAAFL:** “Their Ain’t No Such Thing as a Free Lunch
... anything free costs twice as much in [the] long run or turns out worthless.”

-- Manuel, in Robert A. Heinlein’s *The Moon is a Harsh Mistress*
Validation and Management Discussion

If you were the project leader at your last job, how would you improve the practices of requirements validation and management?
Requirements Management Tools

- Manage requirements content, attributes, and change
- Typical features:
  - Manage requirements versions and changes
  - Access control
  - Store and sort on requirements attributes
  - Manage and display requirements tracing
  - Track requirement status
## Some Tools

<table>
<thead>
<tr>
<th>Tool</th>
<th>Vendor</th>
<th>Database-Centric or Document-Centric</th>
</tr>
</thead>
<tbody>
<tr>
<td>Caliber Analyst</td>
<td>Borland Software</td>
<td>Database</td>
</tr>
<tr>
<td>C.A.R.E. (Computer-Aided Requirements Engineering)</td>
<td>SOPHIST Group</td>
<td>Database</td>
</tr>
<tr>
<td>DOORS</td>
<td>IBM Rational DOORS</td>
<td>Database</td>
</tr>
<tr>
<td>RequisitePro</td>
<td>IBM Rational Software</td>
<td>Document</td>
</tr>
<tr>
<td>RMTrak</td>
<td>RBC, Inc.</td>
<td>Document</td>
</tr>
<tr>
<td>Dimensions RM</td>
<td>Serena Software</td>
<td>Database</td>
</tr>
<tr>
<td>Open Source Requirements Management Tool</td>
<td>Source Forge</td>
<td>Database</td>
</tr>
</tbody>
</table>

Updated version of Wieger's Table 21-1