Chapter 12

Evaluating Products, Processes, and Resources

Shari L. Pfleeger
Joann M. Atlee

4th Edition
Chapter 12 Objectives

- Feature analysis, case studies, surveys, and experiments
- Measurement and validation
- Capability maturity, ISO 9000, and other process models
- People maturity
- Evaluating development artifacts
- Return of investment
12.1 Approaches to Evaluation

- Measure key aspects of product, processes, and resources
- Determine whether we have met goals for productivity, performance, quality, and other desire attributes
12.1 Approaches to Evaluation
Categories of Evaluation

- Feature analysis: rate and rank attributes
- Survey: document relationships
- Case studies
- Formal experiment
12.1 Approaches to Evaluation
Feature Analysis Example: Buying a Design Tool

- List five key attributes that the tool should have
- Identify three possible tools and rate the criterion
- Examine the scores, creating a total score based on the importance of each criterion
- Based on the score, we select the highest score (t–OO–1)
### 12.1 Approaches to Evaluation
Buying a Design Tool (continued)

- **Design tool ratings**

<table>
<thead>
<tr>
<th>Features</th>
<th>Tool 1: T–OO–1</th>
<th>Tool 2: Object Tool</th>
<th>Tool 3: Easy Design</th>
<th>Importance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good user interface</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Object–oriented design</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Consistency checking</td>
<td>5</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Use cases</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Runs on UNIX</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td><strong>Score</strong></td>
<td><strong>85</strong></td>
<td><strong>77</strong></td>
<td><strong>73</strong></td>
<td></td>
</tr>
</tbody>
</table>
12.1 Approaches to Evaluation

Surveys

- Record data
  - to determine how project participants reacted to a particular method, tool, or technique
  - to determine trends or relationships
- Capture information related to products or projects
- Document the size of components, number of faults, effort expended
12.1 Approaches to Evaluation
Case Studies

- Identify key factors that may affect an activity’s outcome and then document them
- Involve sequence of steps: conception hypothesis setting, design, preparation, execution, analysis, dissemination, and decision making
- Compare one situation with another
12.1 Approaches to Evaluation
Case Study Types

- Sister projects: each is typical and has similar values for the independent variables
- Baseline: compare single project to organizational norm
- Random selection: partition single project into parts
12.1 Approaches to Evaluation

Formal Experiment

- Controls variables
- Uses methods to reduce bias and eliminate confounding factors
- Often replicated
- Instances are representative: sample over the variables (whereas case study samples from the variables)
12.1 Approaches to Evaluation

Evaluation Steps

- Setting the hypothesis: deciding what we wish to investigate, expressed as a hypothesis we want to test
- Maintaining control over variables: identify variables that can affect the hypothesis, and decide how much control we have over the variables
- Making investigation meaningful: the result of formal experiment is more generalizable, while a case study or survey only applies to certain organization
12.2 Selecting An Evaluation Technique

- Formal experiments: research in the small
- Case studies: research in typical
- Surveys: research in the large
12.2 Selecting An Evaluation Technique

Key Selection Factors

- Level of control over the variables
- Degree to which the task can be isolated from the rest of the development process
- Degree to which we can replicate the basic situation
12.2 Selecting An Evaluation Technique
What to Believe

• When results conflict, how do we know which study to believe?
  – Using series of questions, represented by the game board

• How do you know if the result is valid?
  – Evaluation pitfalls table
## 12.2 Selecting An Evaluation Technique

### Common Pitfalls in Investigation

<table>
<thead>
<tr>
<th>Pitfall</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Confounding</td>
<td>Another factor is causing the effect</td>
</tr>
<tr>
<td>2. Cause of effect?</td>
<td>The factor could be a result, not a cause, of the treatment</td>
</tr>
<tr>
<td>3. Chance</td>
<td>There is always a small possibility that your result happened by chance</td>
</tr>
<tr>
<td>4. Homogeneity</td>
<td>You can find no link because all subjects had the same level of the factor</td>
</tr>
<tr>
<td>5. Misclassification</td>
<td>You can find no link because you can not accurately classify each subject’s level of the factor</td>
</tr>
<tr>
<td>6. Bias</td>
<td>Selection procedures or administration of the study inadvertently bias the result</td>
</tr>
<tr>
<td>7. Too short</td>
<td>The short–term effects are different from the long–term ones</td>
</tr>
<tr>
<td>8. Wrong amount</td>
<td>The factor would have had an effect, but not in the amount used in the study</td>
</tr>
<tr>
<td>9. Wrong situation</td>
<td>The factor has the desired effect, but not in the situation studied</td>
</tr>
</tbody>
</table>
12.3 Assessment vs. Prediction

- Assessment system examines an existing entity by characterizing it numerically.
- Prediction system predicts characteristic of a future entity; involves a model with associated prediction procedures.
  - Deterministic prediction (we always get the same output for an input).
  - Stochastic prediction (output varies probabilistically).
12.3 Assessment vs. Prediction
Validating Prediction System

- Comparing the model’s performance with known data in the given environment
- Stating a hypothesis about the prediction, and then looking at data to see whether the hypothesis is supported or refuted
## 12.3 Assessment vs. Prediction

### Sidebar 12.1 Comparing Software Reliability Prediction Modeling techniques

<table>
<thead>
<tr>
<th>Modeling techniques</th>
<th>Predictive validity</th>
<th>Proportion of false negatives (%)</th>
<th>Proportion of false positive (%)</th>
<th>Proportion of false classification (%)</th>
<th>Completeness (%)</th>
<th>Overall Inspection</th>
<th>Wasted Inspection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discriminant Analysis</td>
<td>P= 0.621</td>
<td>28</td>
<td>26</td>
<td>52</td>
<td>42</td>
<td>46</td>
<td>56</td>
</tr>
<tr>
<td>Principal component analysis plus discriminant analysis</td>
<td>P=0.408</td>
<td>15</td>
<td>41</td>
<td>56</td>
<td>68</td>
<td>74</td>
<td>55</td>
</tr>
<tr>
<td>Logistic regression</td>
<td>P=0.491</td>
<td>28</td>
<td>28</td>
<td>56</td>
<td>42</td>
<td>49</td>
<td>58</td>
</tr>
<tr>
<td>Principal component analysis plus logistic regression</td>
<td>P=0.184</td>
<td>13</td>
<td>46</td>
<td>59</td>
<td>74</td>
<td>82</td>
<td>56</td>
</tr>
<tr>
<td>Logical classification model</td>
<td>P=0.643</td>
<td>26</td>
<td>21</td>
<td>46</td>
<td>47</td>
<td>44</td>
<td>47</td>
</tr>
<tr>
<td>Layered neural network</td>
<td>P=0.421</td>
<td>28</td>
<td>28</td>
<td>56</td>
<td>42</td>
<td>49</td>
<td>58</td>
</tr>
<tr>
<td>Holographic network</td>
<td>P=0.634</td>
<td>26</td>
<td>28</td>
<td>54</td>
<td>47</td>
<td>51</td>
<td>55</td>
</tr>
<tr>
<td>Heads or tails</td>
<td>R=1,000</td>
<td>25</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
</tbody>
</table>
12.3 Assessment vs. Prediction

Validating Measures

- Assuring that the measure captures the attribute properties it is supposed to capture
- Demonstrating that the representation condition holds for the measure and its corresponding attributes
12.3 Assessment vs. Prediction
Sidebar 12.2 Lines of Code and Cyclomatic Number

- The number of lines of code is a valid measure of program size, however, it is not a valid measure of complexity.
- On the other hand, there are many studies that exhibit a significant correlation between lines of code and cyclomatic number.
12.3 Assessment vs. Prediction
A Stringent Requirement for Validation

- A measure (e.g., LOC) can be
  - an attribute measure (e.g., program size)
  - an input to a prediction system (e.g., predictor of number of faults)
- Do not reject a measure if it is not part of a prediction system
  - If a measure is valid for assessment only, it is called *valid in the narrow*
  - If a measure is valid for assessment and useful for prediction, it is called *valid in the wide sense*
12.4 Evaluating Products

- Examining a product to determine if it has desirable attributes
- Asking whether a document, file, or system has certain properties, such as completeness, consistency, reliability, or maintainability
  - Product quality models
  - Establishing baselines and targets
  - Software reusability
12.4 Evaluating Products

Product Quality Models

- Boehm’s model
- ISO 9126
- Dromey’s Model
12.4 Evaluating Products
Boehm’s Quality Model
12.4 Evaluating Products
Boehm’s Quality Model (continued)

• Reflects an understanding of quality where the software
  - does what the user wants it do
  - uses computer resources correctly and efficiently
  - is easy for the user to learn and use
  - is well–designed, well–coded, and easily tested and maintained
12.4 Evaluating Products
ISO 9126 Quality Model

- A hierarchical model with six major attributes contributing to quality
  - Each right-hand characteristic is related to only to exactly one left-hand attribute
12.4 Evaluating Products
ISO 9126 Quality Model (continued)

Functionality
- Suitability
- Accuracy
- Interoperability
- Security

Reliability
- Maturity
- Fault tolerance
- Recoverability

Usability
- Understandability
- Learnability
- Operability

Efficiency
- Time behavior
- Resource behavior

Maintainability
- Analyzability
- Changeability
- Stability

Portability
- Testability
- Adaptability
- Installability
- Conformance
- Replaceability
### 12.4 Evaluating Products

**ISO 9126 Quality Characteristics**

<table>
<thead>
<tr>
<th>Quality Characteristic</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality</td>
<td>A set of attributes that bear on the existence of a set of functions and their specified properties. The functions are those that satisfy stated or implied needs.</td>
</tr>
<tr>
<td>Reliability</td>
<td>A set of attributes that bear on the capability of software to maintain its performance level under stated conditions for a stated period of time.</td>
</tr>
<tr>
<td>Usability</td>
<td>A set of attributes that bear on the effort needed for use and on the individual assessment of such use by a stated or implied set of users.</td>
</tr>
<tr>
<td>Efficiency</td>
<td>A set of attributes that bear on the relationship between the software performance and the amount of resources used under stated conditions.</td>
</tr>
<tr>
<td>Maintainability</td>
<td>A set of attributes that bear on the effort needed to make specified modifications (which may include corrections, improvements, or adaptations of software to environmental changes and changes in the requirements and functional specifications).</td>
</tr>
<tr>
<td>Portability</td>
<td>A set of attributes that bear on the ability of software to be transferred from one environment to another (including the organizational, hardware or software environment).</td>
</tr>
</tbody>
</table>
12.4 Evaluating Products
Establishing Baseline and Targets

- A baseline describes the usual or typical result in an organization or category
- Baselines are useful for managing expectations
- A target is a variation of a baseline
  - minimal acceptable behavior
## 12.4 Evaluating Products
### Quantitative Targets For Managing US Defense Projects

<table>
<thead>
<tr>
<th>Item</th>
<th>Target</th>
<th>Malpractice Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fault removal efficiency</td>
<td>&gt;95%</td>
<td>&lt;70%</td>
</tr>
<tr>
<td>Original fault density</td>
<td>&lt;4 per function point</td>
<td>&gt;7 per function point</td>
</tr>
<tr>
<td>Slip or cost overrun in Excess of risk reverse</td>
<td>0%</td>
<td>&gt;=10%</td>
</tr>
<tr>
<td>Total requirements creep (function points or equivalent)</td>
<td>&lt;1% per month average</td>
<td>&gt;= 50%</td>
</tr>
<tr>
<td>Total program documentation</td>
<td>&lt;3 pages per function point</td>
<td>&gt;6 pages per function point</td>
</tr>
<tr>
<td>Staff turnover</td>
<td>1 to 3% per year</td>
<td>&gt;5% per year</td>
</tr>
</tbody>
</table>
12.4 Evaluating Products
Software Reusability

- Software reuse: the repeated use of any part of a software system
  - documentation
  - code
  - design
  - requirements
  - test cases
  - test data
12.4 Evaluating Products

Type of Reuse

• Producer reuse: creating components for someone else to use

• Consumer reuse: using components developed for some other product
  – Black-box reuse: using component without modification
  – Clear- or white-box reuse: modifying component before reusing it
12.4 Evaluating Products
Reuse Approaches

- Compositional reuse: uses components as building blocks; development done from bottom up
- Generative reuse: components designed specifically for a domain; design is top-down
- Domain analysis: identifies areas of commonality that make a domain ripe for reuse
### 12.4 Evaluating Products

#### Aspects of Reuse

<table>
<thead>
<tr>
<th>Substance</th>
<th>Scope</th>
<th>Mode</th>
<th>Technique</th>
<th>Intention</th>
<th>Product</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ideas and concepts</td>
<td>Vertical</td>
<td>Planned and</td>
<td>Compositional</td>
<td>Black-box,</td>
<td>Source Code</td>
</tr>
<tr>
<td>Artifacts and components</td>
<td>Horizontal</td>
<td>Systematic</td>
<td>Generative</td>
<td>as is</td>
<td>Design</td>
</tr>
<tr>
<td>Procedures,</td>
<td></td>
<td>Ad hoc,</td>
<td></td>
<td>Clear-box modified</td>
<td>Requirements</td>
</tr>
<tr>
<td>skills, and experience</td>
<td></td>
<td>opportunistic</td>
<td></td>
<td></td>
<td>Objects</td>
</tr>
<tr>
<td>Patterns</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Data</td>
</tr>
<tr>
<td>Architecture</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Processes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Documentation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Tests</td>
</tr>
</tbody>
</table>
12.4 Evaluating Products
Reuse Technology

- Component classification: collection of reusable components are organized and catalogued according to a classification scheme
  - hierarchical
  - faceted classification
12.4 Evaluating Products
Example of A Hierarchical Scheme

- New topic can be added easily at the lowest level
12.4 Evaluating Products
Faceted Classification Scheme

- A facet is a kind of descriptor that helps to identify the component
- Example of the facets of reusable code
  - a application area
  - a function
  - an object
  - a programming language
  - an operating system
12.4 Evaluating Products
Component Retrieval

- A retrieval system or repository: an automated library that can search for and retrieve a component according to the user’s description.
- A repository should address a problem of conceptual closeness (values that are similar to but not exactly the same as the desired component).
- Retrieval system can
  - record information about user requests
  - retain descriptive information about the component.
12.4 Evaluating Products
Sidebar 12.3 Measuring Reusability

- The measures must
  - address a goal
  - reflect perspective of the person asking the question
- Even if we had a good list of measurements, still it is difficult to determine the characteristic of the most reused component
  - Look at past history
  - Engineering judgment
  - Automated repository
12.4 Evaluating Products
Experience with Reuse

- Raytheon
  - A new system contained an average of 60% reused code increasing productivity by 50%
- GTE Data Services
  - Established incentives and rewards for program authors whenever their components were reused
  - 14% reuse on its project, valued at a savings of $1.5 million
- Nippon Novel
  - Paid 5 cents per line of code to a developer who reused a component
12.4 Evaluating Products
Sidebar 12.4 Software Reuse at Japan’s Mainframe Makers

• NEC: reuse library was established to classify, catalog, and document
• Hitachi: integrated software environment, called Eagle, to allow software engineers to reuse standard program patterns and functional procedures
• Fujitsu: created Information Support Center (ISC), that is a regular library staffed with system analysts, software engineers, reuse experts, and switching system domain experts
12.4 Evaluating Products
Benefits of Reuse

- Reuse increases productivity and quality
- Reusing component may increase performance and reliability
- A long term benefit is improved system interoperability
12.4 Evaluating Products
Example of Cost of Reuse

- **Cost to produce and reuse at HP**

<table>
<thead>
<tr>
<th></th>
<th>Air traffic control system (%)</th>
<th>Menu– and forms Management system (%)</th>
<th>Graphics Firmware (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relative cost to create Reusable code</td>
<td>200</td>
<td>120 to 480</td>
<td>111</td>
</tr>
<tr>
<td>Relative cost to reuse</td>
<td>10 to 20</td>
<td>10 to 63</td>
<td>19</td>
</tr>
</tbody>
</table>
12.4 Evaluating Products
Sidebar 12.5 Critical Reuse Success Factors at NTT

- Success factors at NTT in implementing reuse
  - senior management commitment
  - selecting appropriate target domains
  - systematic development of reusable modules based on domain analysis
  - investing several years of continuous effort in reuse
12.4 Evaluating Products

Reuse Lessons

- Reuse goals should be measurable
- Management should resolve reuse goals early
- Different perspectives may generate different questions about reuse
- Every organization must decide at what level to answer reuse questions
- Integrate the reuse process into the development process
- Let your business goals suggest what to measure
12.4 Evaluating Products
Conflicting Interpretation of Goals

- A division manager’s reuse goal may conflict with a project manager’s goal, so no reuse ever gets done.

**PROJECT GOAL:** Improve productivity

Interpreted by project manager

**ACTION:** Minimize time to code component

**DIVISION GOAL:** Improve productivity

Interpreted by division manager

**ACTION:** Take extra time to make component reusable

**CONFLICT!**
12.4 Evaluating Products

Questions for Successful Reuse

- Do you have the right model of reuse?
- What are the criteria for success?
- How can current cost models be adjusted to look at collections of projects, not just single projects?
- How do regular notions of accounting fit with reuse?
- Who is responsible for component quality?
- Who is responsible for process quality and maintenance?
12.5 Evaluating Process
Postmortem Analysis

- A postimplementation assessment of all aspects of the project, including products, process, and resources, intended to identify areas of improvement for future projects.

- Takes places shortly after a projects is completed, or can take place at any time from just before delivery to 12 months afterward.
### 12.5 Evaluating Process
When Postimplementation Evaluation Is Done

<table>
<thead>
<tr>
<th>Time period</th>
<th>Percentage of Respondent (of 92% organizations)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Just before delivery</td>
<td>27.8</td>
</tr>
<tr>
<td>At delivery</td>
<td>4.2</td>
</tr>
<tr>
<td>One month after delivery</td>
<td>22.2</td>
</tr>
<tr>
<td>Two months after delivery</td>
<td>6.9</td>
</tr>
<tr>
<td>Three months after delivery</td>
<td>18.1</td>
</tr>
<tr>
<td>Four months after delivery</td>
<td>1.4</td>
</tr>
<tr>
<td>Five months after delivery</td>
<td>1.4</td>
</tr>
<tr>
<td>Six months after delivery</td>
<td>13.9</td>
</tr>
<tr>
<td>Twelve months after delivery</td>
<td>4.2</td>
</tr>
</tbody>
</table>
Sidebar 12.6 How Many Organizations Perform Postmortem Analysis

- Kumar (1990) surveyed 462 medium-sized organizations
  - 92 organizations that responded, more than one-fifth did not perform postmortem analysis
  - Those that did, postmortem were conducted on fewer than half of the projects in the organization
12.5 Evaluating Process
Postmortem Analysis Process

- Design and promulgate a project survey to collect relevant data
- Collect objective project information
- Conduct a debriefing meeting
- Conduct a project history day
- Publish the results by focusing on lessons learned
12.5 Evaluating Process
Postmortem Analysis Process: Survey

- A starting point to collect data that cuts across the interests of project team members
- Three guiding principles
  - Do not ask for more than you need
  - Do not ask leading questions
  - Preserve anonymity
- Sample questions shown in Sidebar 12.7
12.5 Evaluating Process
Sidebar 12.7 Sample Survey Questions From Wildfire Survey

- Were interdivisional lines of responsibility clearly defined throughout the project?
- Did project–related meetings make effective use of your time?
- Were you empowered to participate in discussion regarding issues that affected your work?
- Did schedule changes and related decision involve the right people?
- Was project definition done by the appropriate individuals?
- Was the build process effective for the component area your work on?
- What is your primary function on this project?
12.5 Evaluating Process
Postmortem Analysis Process: Objective Information

- Obtain objective information to complement the survey opinions
- Collier, Demarco and Fearey suggest three kinds of measurements: cost, schedule, and quality
  - Cost measurements might include
    - person–months of effort
    - total lines of code
    - number of lines of code changed or added
    - number of interfaces
12.5 Evaluating Process
Postmortem Analysis Process: Debriefing Meeting

- Allows team members to report what did and did not go well on the project
- Project leader can probe more deeply to identify the root cause of positive and negative effects
- Some team members may raise issues not covered in the survey questions
- Debriefing meetings should be loosely structured
12.5 Evaluating Process

Postmortem Analysis Process: Project History Day

- Objective: identify the root causes of the key problems
- Involves a limited number of participants who know something about key problems
- Review schedule predictability charts
  - Show where problems occurred
  - Spark discussion about possible causes of each problem
12.5 Evaluating Process

Postmortem Analysis Process: Schedule–Predictability Charts

- For each key project milestone, the chart shows when the predictions were made compared with the completion date.
12.5 Evaluating Process
Postmortem Analysis Process: Publish Results

- **Objective:** Share results with the project team
- **Participants in the project history day** write a letter to managers, peers, developers
- **The letter has four parts**
  - Introduction to the project
  - A summary of postmortem’s positive findings
  - A summary of three worst factors that kept the team from meeting its goals
  - Suggestions for improvement activities