Lecture 3 – Software Quality Factors

- The need for comprehensive software quality requirements
- Classification of requirements into software quality factors
- Product operation factors
- Product revision factors
- Product transition factors
- Alternative models of software quality factors

The need for comprehensive SQ requirements

- Often software fulfils its basic requirements (correctness etc).
- Software projects often suffer poor performance (maintenance, reliability, reuse, training).
- Caused by lack of predefined requirements to cover these aspects.

McCall’s software quality factors model

Product Operation Factors

Deal with requirements that directly affect the daily operation of the s/w.
- Correctness
- Reliability
- Efficiency
- Integrity
- Usability
Correctness

- Defined by outputs (multidimensional)
  1. The output mission (invoice printout, alarms)
  2. Required accuracy (inaccurate data/calcs)
  3. Completeness (incomplete data)
  4. Up-to-dateness (event to consideration time)
  5. Availability (reaction time, to get info or for firmware response)
  6. Standards (for coding & documenting the software system)

Correctness Example

(club membership info system)

1. Output mission: list of x reports, y standard letters, z interactive queries.
2. Required accuracy: prob of a non-accurate output < 1%.
3. Completeness: prob of missing data about a member, attendance, payments etc < 1%.
4. Up-to-dateness: no more than 2 days for info about event participation to be valid.
5. Availability: reaction time to queries < 2 seconds, to reports < 4 hours.

Reliability

- Deals with failure to provide service.
- Determine max allowed SS failure rate.
- Can refer to entire system or 1+ separate functions.
- Eg
  - Failure of heart monitor required to be < 1 per 20 years. Heart attack detection < 1 per million cases.
  - Bank branch s/w fails < 10 mins per month in office hours; prob of >30 mins recovery time < 0.5%

Efficiency

- Deal with h/w resources.
- Processing power (MIPS, MHz, Mflops)
- Storage capacity (GBytes, TBytes)
- Comms (MBPS, GBPS)
- Time between recharge for portables
- Eg 1: rival bids for system may specify different values
- Eg 2: remote traffic camera may go x days without sunshine on its solar panel
Integrity

• Security requirements
• Access to unauthorised personnel
• Access levels (read v write, user v sys)
• Deliberate attacks; Accidental losses/damage; Backups.
• Eg Local authority GIS system. Public to view maps etc, but not edit.

Usability

• Scope of staff resources needed
  – to train new employees
  – to operate software system
• Eg Help Desk specs
  – One staff to handle 60 calls per day
  – Two days to train new staff who will then be able to handle 45 calls per day

Product revision factors

Deal with requirements that affect the complete range of s/w maintenance activities (corrective, adaptive, perfective)

• Maintainability
• Flexibility
• Testability

Maintainability

• Determine efforts needed by users and maintenance personnel to
  – identify reasons for s/w failures
  – to verify success of corrections
• Refer to
  – modular structure of s/w
  – Internal program documentation
  – Programmers manual
• Typical requirements:
  – Typical s/w module <30 statements
  – Code to company standards and guidelines
Flexibility

- Capabilities & efforts to support adaptive maintenance activities
- Example: resources (man-days) to
  - adapt to different customers
  - different activities
  - different product ranges etc
- Requirements to support perfective s/w
  - improve service
  - adapt to changes in technical/commercial environment
- Eg Teacher Support s/w:
  - should adapt to schools of all levels
  - non-professionals to be able to create new reports

Testability

- Requirements deal with testing system itself & its operation
- Special program features for intermediate results and log files
- Automated diagnosis prior to start/at start/on demand for fault reporting
- Eg Testability requirement to develop standard test data against which to validate production line performance at each stage (run each morning?)

Product Transition Factors

Pertains to the adaptation of s/w to other environments and its interaction with other s/w systems
- Portability
- Reusability
- Interoperability

Portability

- Portability requirements tend to the adaptation of a s/w system to other environments
  - hardware
  - operating systems etc
- Use same s/w in diverse situations
- Eg windows NT prog required to work in Vista and Linux
Reusability

- One product’s software modules used in new project/domain.
- Benefits:
  - Save development resources
  - Save test resources
  - Shorten development period
  - Provide higher quality (already quality assured, tested and used)
- Eg hotel pool user management system considered for spa pool user management?

Interoperability

- Focus on creating interfaces with other software systems
- Requirements can specify e.g., the names of software or firmware for which interface is required.
- Can specify output structure accepted as standard.
- Eg medical lab equipment to process results in standard format for other standard lab info systems.

Alternative Models to McCall

Basically extend McCall’s factors/requirements by 5:

12. Verifiability: design & prog features for efficient verification (modularity, simplicity, adherence to doc & prog standards)
13. Expandability: future efforts for improved service, bigger system, increased user numbers. cf “flexibility”
Alternative Models to McCall 2

14. Safety: eliminate hazardous conditions as a result of errors in process control software (e.g., failure to provide alarm signals, inappropriate reactions to dangerous conditions)
15. Manageability: refer to admin tools for s/w modification during development and maintenance e.g., configuration management, change procedures
16. Survivability: continuity of service (e.g., max rate of failure mtb failures, max recovery time etc) cf “reliability”

McCall’s factor model and alternative models

<table>
<thead>
<tr>
<th>No.</th>
<th>Software quality factor</th>
<th>McCall’s classic model</th>
<th>Evans and Marciniak model</th>
<th>Deutsch and Willis model</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Correctness</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>2</td>
<td>Reliability</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>3</td>
<td>Efficiency</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>4</td>
<td>Integrity</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>5</td>
<td>Usability</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>6</td>
<td>Maintainability</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>7</td>
<td>Flexibility</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>8</td>
<td>Testability</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>9</td>
<td>Portability</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>10</td>
<td>Reliability</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>11</td>
<td>Interoperability</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>12</td>
<td>Verifiability</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>13</td>
<td>Expandability</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>14</td>
<td>Safety</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>15</td>
<td>Manageability</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>16</td>
<td>Survivability</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>