Traceability

Even small requirements errors can produce big problems!

Content

1. Traceability according to CMMI, IEEE and INCOSE Handbook
2. Traceability
3. Procedure
4. Traceability Tools
5. Quality Function Deployment (QFD) Method

Sources:
- IEC EN 50128 Software for Railway Control and Protection Systems
- IEEE Std 1059, Guide for Verification and Validation Plans
- Capability Maturity Model Integration, Software Engineering Institute, 2006
  International Council On Systems Engineering (INCOSE).
Participants in the Requirements Process

What the customer actually wanted!
Traceability - Definitions

• **Traceability**
  - A discernable association among two or more logical entities such as requirements, system elements, verifications, or tasks. (See also “bidirectional traceability” and “requirements traceability.”)

• **Requirements traceability**
  - A discernable association between requirements and related requirements, implementations, and verifications.

• **Bidirectional traceability**
  - An association among two or more logical entities that is discernable in either direction (i.e., to and from an entity).

Traceability – IEEE –Std-1059

1. The ability to identify the relationships between originating requirements and their resulting system features.
2. It permits tracking forward or backward through the network of interrelationships that are created as requirements are decomposed and refined throughout a systems life cycle.
   - Traceability provides the thread that links one element to another. When an element is traced to another, and that element is traced on to another, a chain of cause and effect is formed.
Traceability – IEEE–Standard 1059

- Traceability allows verification of the properties set forth in the concept and that requirement specifications have been:
  1. Carried forward to the design specifications
  2. Implemented in the code
  3. Included in the test plan and cases
  4. Provided to the customer and user in the resulting system

  – Traceability Analysis

    – Each trace is analyzed for consistency, completeness, and correctness to verify that all software requirements are implemented in the software and are associated with the correct design, code, and test information.
    – An analyst examines each trace path to ensure that the connected pieces are the proper ones.
    – Analyst has to understand the intent of requirements, design and other documents.

Traceability – IEEE –Std-1059

- Traceability analysis is performed to assure that:
  1. Every requirement in the specification is identified correctly
  2. All traces are continuous from the prior phase through the current phase
  3. Forward and backward traces between adjoining phases are consistent
  4. The combined forward traces originating from each specification, fully support that specification
  5. Each current specification or feature is fully supported by traceable predecessor specifications
Traceability

• **Aim**
  – To ensure that all requirements can be shown to have been properly met and that no untraceable material has been introduced.

• **Description**
  – Traceability to requirements shall be an important consideration in the validation of a system and means shall be provided to allow this to be demonstrated throughout all phases of the lifecycle.

Source: IEC EN 50128 Software for Railway Control and Protection Systems

---

Traceability-Attributes

1. The requirement identification number
2. The source of the requirement
   • Such as the customer’s document paragraph number or the engineering report documenting the analysis that derived the requirement.
3. The full text of the requirement
4. For allocated or derived requirements, a pointer to the requirement from which it was derived, or “parent” requirement.
5. A pointer to the next lower-level area that this requirement was allocated to during the allocation process
6. Verification method (e.g. test, demonstration, analysis, inspection/examination).
7. The Test Plan name & number controlling the verification
8. The Test Procedure name & number performing the verification
9. The date and results of the final verification
10. The name of the responsible engineer.

Some key requirements traceability links.

![Diagram showing traceability links between requirements, system requirements, use cases, external interface requirements, quality attributes, change requests, software functional requirements, architecture, user interface, and system test.]

**Requirements Traceability Matrix***

<table>
<thead>
<tr>
<th>Requirement source</th>
<th>Product requirements</th>
<th>NLD section #</th>
<th>LLD section #</th>
<th>Code unit</th>
<th>UTS case #</th>
<th>STS case #</th>
<th>User manual</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business rule #1</td>
<td>R09210 credit card types</td>
<td>4.1 Parse mag strip</td>
<td>4.1.1 read card type</td>
<td>Read_card_type.c</td>
<td>UT 4.1.032</td>
<td>ST 120.000</td>
<td>Section 12</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4.1.2 Verify card type</td>
<td>Ver_Card_Type.c</td>
<td>UT 4.2.012</td>
<td>ST 120.035</td>
<td></td>
</tr>
<tr>
<td>Use Case #132 step 6</td>
<td>R09210 read gas flow</td>
<td>7.2.2 Gas flow meter interface</td>
<td>7.2.2 Read gas flow indicator</td>
<td>Read_gas_flow.c</td>
<td>UT 7.2.043</td>
<td>ST 230.002</td>
<td>Section 21.1.2</td>
</tr>
<tr>
<td></td>
<td>R09231 Calculate gas price</td>
<td>7.3 Calculate gas price</td>
<td>7.3 Calculate gas price</td>
<td>Cal_gas_price.c</td>
<td>UT 7.3.005</td>
<td>ST 231.001</td>
<td>Section 21.1.3</td>
</tr>
</tbody>
</table>

* Also called Proof of Compliance Matrix or Verification Matrix
* A column should be added to show that tests have been performed and successful

Adapted from: Linda Westfall, Bidirectional Requirements Traceability, SQP, Dec 2007
### Traceability Matrix

<table>
<thead>
<tr>
<th>Requirement ID</th>
<th>Verification Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>B5 Paragraph Number</td>
<td>Section 4 Paragraph Number</td>
</tr>
<tr>
<td>3.2.2.2.8.3</td>
<td>4.2.17.3</td>
</tr>
<tr>
<td>N/A</td>
<td>4.2.17.4</td>
</tr>
<tr>
<td>3.2.3.1. .2.3</td>
<td>4.2.18</td>
</tr>
<tr>
<td>3.2.3.1.1</td>
<td>4.2.16.1.1</td>
</tr>
<tr>
<td>3.2.3.1.2.a</td>
<td>4.2.18.1.2.a</td>
</tr>
</tbody>
</table>

#### Verification methods:
- Test, demonstration, analysis, inspection/examination.

### Four types of requirements traceability.

1. **Customer Needs**
2. **Requirements**
   - Forward to requirements
   - Backward from requirements
3. **Product**
   - Forward from requirements
   - Backward to requirements

Traceability in Standard EN 50128

• Description

2. Traceability shall be considered applicable to both functional and non-functional requirements and shall particularly address:
   1. Traceability of requirements to the design or other objects which fulfill them,
   2. Traceability of design objects to the implementation objects which instantiate them,
   3. Traceability of requirements and design objects to the operational and maintenance objects required to be applied in the safe and proper use of the system,
   4. Traceability of requirements, design, implementation, operation and maintenance objects, to the verification and test plans and specifications which will determine their acceptability,
   5. Traceability of verification and test plans and specifications to the test or other reports which record the results of their application.

Source: IEC EN 50128 Software for Railway Control and Protection Systems
Benefits of Implementing Traceability

1. Certification - Verification
   - The traceability information can be used for certification in safety-critical applications (e.g. avionics)
     - To verify and demonstrate that all requirements were implemented.

2. Change Impact Analysis
   - Traceability links help find all of the system elements that might have to be modified if you change a particular requirement.
   - Without traceability information, chances are high you’ll overlook some of the side effects of adding, deleting, or modifying a requirement.

3. Project Tracking
   - If you complete the requirements traceability matrix as development takes place, you will have accurate insight into the implementation status of planned functionality.
   - Empty space in the matrix indicates project deliverables that have not yet been created.


4. Testing
   - Links between tests, requirements, and code point toward likely parts of the code to examine for a bug when a test fails to yield the intended result

5. Reuse
   - Traceability information can facilitate the reuse of product components
     - By identifying packages of related requirements, designs, code, tests, and other artefacts.

6. Risk Management and Reduction
   - Documenting the information about system component interconnections reduces the risk associated with a key team member leaving the company with essential information residing only in that person’s brain (Ambler 1999).

Benefits of Implementing Traceability

7. Reengineering
   • If you don’t have complete requirements for the existing system.
     • You can list the functions in a legacy system you’re replacing and record where they were addressed in the requirements and software components for the new system.
     • Provide a way to capture some of what you learn through reverse engineering.

8. Identification of process improvements
   • e.g. Information about Requirements instability may be used to improve the development process/change management process

9. Allows developer, customer, supplier to follow the development of components
   • If traceability is implemented on web site

10. Help to reduce cost and delay and improve quality
    • By not developing the wrong components
    • By not forgetting components during development
    • By testing all components before delivery
    • By ensuring that all documents have been modified/updated after a change

## Requirements Traceability Matrix*

<table>
<thead>
<tr>
<th>Functional Requirement</th>
<th>Parent Use Case</th>
<th>Design Element</th>
<th>Code</th>
<th>Test Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>catalog.query.sort</td>
<td>UC-28</td>
<td>class catalog</td>
<td>catalog.sort()</td>
<td>search.7, search.8</td>
</tr>
<tr>
<td>catalog.query.import</td>
<td>UC-29</td>
<td>class catalog</td>
<td>catalog.import()</td>
<td>search.8, search.13, search.14</td>
</tr>
</tbody>
</table>

* Also called Proof of Compliance Matrix or Verification Matrix


---

## Likely Sources of Traceability Link Information

<table>
<thead>
<tr>
<th>Link Source Object Type</th>
<th>Link Target Object Type</th>
<th>Likely Information Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>system requirement</td>
<td>software requirement</td>
<td>system engineer</td>
</tr>
<tr>
<td>use case</td>
<td>functional requirement</td>
<td>requirements analyst</td>
</tr>
<tr>
<td>functional requirement</td>
<td>functional requirement</td>
<td>requirements analyst</td>
</tr>
<tr>
<td>functional requirement</td>
<td>software architecture element</td>
<td>system architect</td>
</tr>
<tr>
<td>functional requirement</td>
<td>other design elements</td>
<td>developer</td>
</tr>
<tr>
<td>design element</td>
<td>source code</td>
<td>developer</td>
</tr>
<tr>
<td>use case or functional requirement</td>
<td>test case</td>
<td>test engineer</td>
</tr>
</tbody>
</table>
Procedure: Requirements Traceability

1. Verify that all formalized requirements are traceable to a higher-level requirement
2. Verify that all higher-level requirements are traceable to a formalized requirement(s)
3. Verify that all formalized requirements are traceable to test requirement.
4. Verify that all test requirements are traceable to test case(s).
5. Identify any requirement without a parent

Traceability Tool

- Should generate the following information from a database:
  1. Requirements Statements with Project Unique Identifiers (PUID)
  2. Requirements Traceability Matrices (RTM)
  3. Verification Cross Reference Matrices (VCRM)
  4. Lists of TBD (Determined), TBR (Reviewed), and TBS (Specified)
  5. Specifications
  6. Requirements measures (e.g., requirements stability)

Quality Function Deployment (QFD)

Steps of the Method
1. Customer Requirements
2. Customer Priorities
3. Customer Assessment
   • Compare to competitors
4. Technical Requirements
5. Relationship Matrix
   • Between Whats and Hows
6. Importance Ratings
   • Priorities X Relationships
7. Engineering Assessment
   • Compare design options
8. Correlation Matrix
   • Relationships between Hows

Summary
1. Verification
2. Traceability according to IEEE Std et INCOSE Handbook
3. Traceability
4. Procedure
5. Traceability Tools
6. Quality Function Deployment (QFD) Method