- Observe the system context to detect context changes
- Manage the execution of requirements engineering activities
- Manage the requirements artefacts
Requirements Traceability

• The ability to describe and follow the life of requirement in both a forward and backward direction
  – From its origins
  – Through its development and specification
  – To its subsequent deployment and use
  – Through all periods of on-going refinement and iteration in any of these stages
Use of traceability

- **Verifiability and acceptance**
  - Supports the validation that requirement was considered during the implementation of the system
  - Provides evidence that requirement was implemented in the system

- **Gold plating**
  - Information, functions, or qualities of the system can be uncovered that were not specified in the requirements and, therefore, have no justification

- **Change management**
  - Allows for analysing, in the case of change, which other requirements are affected by change
  - Supports prediction of the effort to change the requirement
Use of traceability

• **Quality assurance, maintenance and repair**
  – Facilitates identification of causes and the impact of errors in the identification of parts of the system affected by an error, and the prognosis of the effort required for correcting the error

• **Re-engineering**
  – Supports re-engineering of legacy systems by relating the functions of the legacy system to the requirements for the new system and by documenting which components of the new system realise these requirements

• **Reuse**
  – Supports reuse of development artefacts related to a requirement
  – When requirement is reused, the corresponding artefacts in the old system can be identified
Use of traceability

• **Project traceability**
  – Supports tracing the project and current project status
  – Helps determining which requirements have already been considered/implemented in the system and which not

• **Risk management**
  – Facilitates identification of development artefacts that are potentially affected by a risk or threat

• **Accountability**
  – Can be used to assign development effort to individual requirements

• **Process improvement**
  – Can be used to trace problems back to their sources
Pre- and Post-traceability

- **Pre-traceability** – of requirements artefact to its predecessor artefact
  - To its source or origin

- **Post-traceability** – from a requirement to its successor artefacts
  - Architecture, implementation, test cases, and etc.
Pre- and Post-traceability

"The customer wants to be able to monitor his house or apartment with his mobile phone."

R17: The mobile phone shall allow for the surveillance of one's living space.

"We want to facilitate the control and surveillance of living space via a mobile phone as of 2008."

Page 22, Product Strategy Document V2.3
Pre- and Post-traceability

Requirement R17: The mobile phone shall be able to determine its location to an accuracy of 10 m.

refines

Goal G2: The mobile phone shall be able to offer location-based services.

realises

realises

realises

Preliminary design

Detailed design
Traceability Model

- Traceability artefact
  - Part of: 0..1
  - Source: 0..*
  - Destination: 1..*

- Traceability relationship
  - 0..*

- Goal
- Scenario
- Solution-oriented requirement

- Constraint
- Precondition
- Condition
- Content
- Abstraction
- Evolution
- Miscellaneous
Traceability Types

• **Condition**
  - **Constraint** – a solution-oriented requirement can be a constraint of another solution-oriented requirement
  - **Precondition** – the precondition realising a functional requirement might be that the hardware meets a specific performance requirement
Traceability Types

• **Content**
  - **Similar** – two associated requirements are similar in content
  - **Compares** – $A_1$ represents the result of comparison of artefacts $A_2 \ldots A_n$
  - **Contradicts** – inconsistency in the requirements artefacts
  - **Conflicts** – realisation of requirement $A$ may hinder (but does not necessarily exclude) the realisation of requirement $B$
Traceability Types

• Abstraction
  – **Classifies** – A classifies B₁…B₂
  – **Aggregates** – A is an aggregation of a set of other artefacts B₁…B₂
  – **Generalises** – artefact is a generalisation of (one or) several other artefacts
Traceability Types

• **Evolution**
  - **Replaces** – artefact B was replaced by artefact A
  - **Satisfies** – if artefact A is realised in the system, artefact B is realised as well
  - **Based on** – artefact A has influenced the definition of artefact B
  - **Formalises** – artefact A is a formal documentation of artefact B
  - **Refines** – artefact A defines the artefact B in more detail
  - **Derived** – artefact A was derived based on a set of other artefacts
Traceability Types

- **Miscellaneous**
  - **Example of** – artefact contains exemplary aspect of a set of artefacts
  - **Verifies** – e.g., test artefact is used to verify/validate the requirement artefact
  - **Rationale** – one artefact documents the justification of another artefact
  - **Responsible for** – a stakeholder is responsible for associated artefact
  - **Background** – used to assign a background information (e.g., requirement documented using some standard)
  - **Comment** – any kind of information related to a requirements artefact
Example

1. based_on

2. conflicts

3. formalises

4. classifies

5. refines

6. satisfies
1. "Based on" documents that the associated goal is based on the text fragments of a predecessor artefacts (e.g., fragment of the minutes of the interview)
2. “Conflict” documents that a conflict between textual scenario and the goal definition exists
3. “Formalises” documents that the model-based scenario formalises the associated textual scenario.
4. “Classifies” between the model-based scenario and the associated text, solution-oriented requirements document that the solution-oriented requirements are classified by the scenario.
5. “Refines” documents that the statechart refines the associated textual requirement
6. **“Satifies”** documents that the depicted fragment of the **solution-oriented requirements model** is realised if a **specific component** in the system architecture is realised.
Documenting Traceability Relationships
Documenting Traceability Relationships

- **Textual references**

```
R2-17: For selecting the trip destination, the navigation system shall display the last ten trip destinations. [based_on→R1-17] [...]
```
Documenting Traceability Relationships

- **Textual references**

R2-17: For selecting the trip destination, the navigation system shall display the last ten trip destinations. [based_on→R1-17] [...]

- **Hyperlinks**

R2-17: For selecting the trip destination, the navigation system shall display the last ten trip destinations.

hyperlink (type: conflicts)

R3-11: The system shall not store any information about the destinations of previous trips.
Documenting Traceability Relationships

- Traceability Matrix
Documenting Traceability Relationships

- **Traceability Matrix**

<table>
<thead>
<tr>
<th>Source artefacts</th>
<th>Goal 1</th>
<th>Goal 2</th>
<th>Goal 3</th>
<th>Goal 4</th>
<th>Goal 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scenario 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scenario 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scenario 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scenario 4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scenario 5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Satisfies
- Based on
- Conflicts
- Satisfies
Documenting Traceability Relationships

- Traceability Graphs

[Diagram showing traceability relationships between goals and scenarios, with annotations for different types of nodes (goal, scenario, solution-oriented requirement) and relationships (based-on, satisfies, example_of)].
Documenting Traceability Relationships

- Traceability Graphs
Documenting Traceability Relationships

- Traceability Graphs

Traceability graph ("satisfies" for Goal G-12)
Take Home