31.1 Fundamentals of Traceability

An essential task of requirements management is to support the traceability of requirements throughout the entire development process.

The Term "Traceability"

Traceability of development artefacts

According to [IEEE Std 610.12-1990] traceability denotes, in general, the degree to which a relationship can be established between different development artefacts, especially artefacts having a predecessor–successor– or master–subordinate-relationship to one another.

Requirements traceability

A requirement is considered to be traceable if the origin of the requirement as well as its further use in the development process can be traced (see [IEEE Std 830-1998; Edwards and Howell 1992; Hamilton and Beeby 1991; Gotel and Finkelstein 1994; Pohl 1996b]). A prerequisite for the traceability of requirements throughout the development process is that each requirement has a unique identifier. We adopt the following definition of requirements traceability from [Gotel and Finkelstein 1994]:

Definition 31-1: *Requirements traceability*

"Requirements traceability refers to the ability to describe and follow the life of a requirement, in both a forwards and backwards direction (i.e. from its origins, through its development and specification, to its subsequent deployment and use, and through all periods of on-going refinement and iteration in any of these phases)."

[Gotel and Finkelstein 1994]

Motivation for Requirements Traceability

Recorded requirements traceability information supports various system development activities. For example, an impact analysis of a requirements change depends on the quality of the recorded traceability information. Table 31-1 provides some examples of the use of requirements traceability information in different system development activities (see, e.g., [Pohl 1996a; Ramesh 1998]).

Traceability as an indicator of the maturity of system development

The quality of the traceability information influences the quality of the system developed. Establishing a sophisticated traceability approach is therefore an important aspect of maturity models such as CMMI [CMMI 2006] or SPICE [Van Loon 2004; Hörmann et al. 2006].

Tab. 31-1 Examples of using traceability information during system development

Aspect	Use of traceability information
Verifiability and acceptance	Traceability supports the validation that a requirement was considered (correctly and completely) during the implementation of the system (see e.g. [Pinheiro and Goguen 1996]). Validating the (correct and complete) implementation of requirements supports the acceptance of the system, since it provides evidence that the stakeholders' requirements have been implemented in the system.
Gold plating	By using traceability information, functions or qualities of the system can be uncovered that were not specified in the requirements and, therefore, have no justification. The development and integration of such functions and qualities is referred to as gold plating. In the same way, requirements with no justification for their existence can be uncovered.
Change management	Traceability allows for analysing, in the case of a change, which other artefacts (e.g. requirements, components, or test cases) are affected by a change (see [Edwards and Howell 1992]). Furthermore, traceability information supports the prediction of the effort required for integrating the change.
Quality assurance, maintenance, and repair	Traceability facilitates the identification of the causes and the impact of errors, the identification of parts of the system affected by an error, and the prognosis of the effort required for correcting the error (see [Brown 1987]).
Re-engineering	Traceability supports the 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	Traceability supports the reuse of development artefacts related to a requirement. When a (set of) requirement(s) is reused in the new system, the corresponding artefacts which realise this requirement(s) in the old system can be identified. These artefacts are potential reuse candidates and can be checked for whether they require adaptation or if they can be reused without adaptation.
Project traceability	Traceability information supports tracing the project and the current project status. Analysis of the recorded traceability information facilitates, for instance, determining which requirements have already been considered/implemented in the system architecture or which ones have even been implemented and tested.
Risk management	Traceability between requirements and other artefacts (e.g. components) supports risk management by facilitating the identification of development artefacts that are potentially affected by a risk or threat.
Accountability	Traceability information can be used to assign development effort to individual requirements. This enables, for example, determining the development effort of a specific stakeholder for a particular (set of) requirements artefacts (see e.g. [Cordes and Carver 1989]).
Process improvement	Traceability supports process improvement. Traceability information can be used to trace problems in the development process back to their causes. The planning and execution of improvement actions can thus be directed towards eliminating the actual causes of problems.

31.2 Pre- and Post-traceability of Requirements

As depicted in Fig. 31-1, we differentiate between requirements pre- and post-traceability (see [Gotel and Finkelstein 1994]).

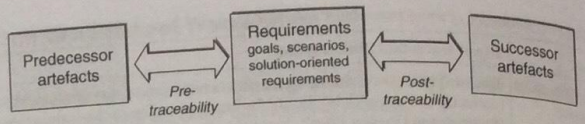

Fig. 31-1 *Pre- and post-traceability of requirements*

Predecessor and successor artefacts

Pre-traceability denotes the traceability of a requirements artefact to its predecessor artefacts. In other words, pre-traceability ensures the traceability of a requirement to its source or origin. Post-traceability denotes the traceability from a requirement to its successor artefacts, such as architectural components satisfying the requirement, the implementation of the requirement in the source code, or the test cases verifying the requirement. Post-traceability hence ensures the traceability from the requirements to, for example, design and implementation artefacts.

Extended Pre- and Post-traceability

Traceability among different types of requirements

Extended pre- and post-traceability structures the pre- and post-traceability of requirements into traceability among requirements artefacts and predecessor/successor artefacts and traceability among different types of requirements artefacts such as goals, scenarios, and solution-oriented requirements (see Fig. 31-2).

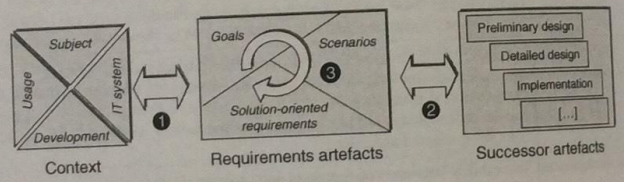

Fig. 31-2 *Extended pre- and post-traceability of requirements*

Traceability to context aspects

Pre-traceability of requirements (❶) comprises relationships from requirements artefacts (i.e. goals, scenarios, and solution-oriented requirements) to aspects in the four context facets. For example, it includes the traceability relationships from a requirements artefact to its source in one or more context facets (see Fig. 31-3 for an example).

Traceability to the realisation

Post-traceability of requirements (❷) comprises relationships between the requirements artefacts and their successor artefacts. For example, it includes traceability relationships between a requirements artefact and a class defined in the detailed design.

Traceability among requirements artefacts

In addition, extended pre- and post- traceability of requirements includes traceability among different requirements artefacts (❸), for example, between a goal and a set of scenarios that illustrate the satisfaction of the goal. Figure 31-4 depicts an example of traceability between different requirements artefacts as well as traceability between requirements and design artefacts.

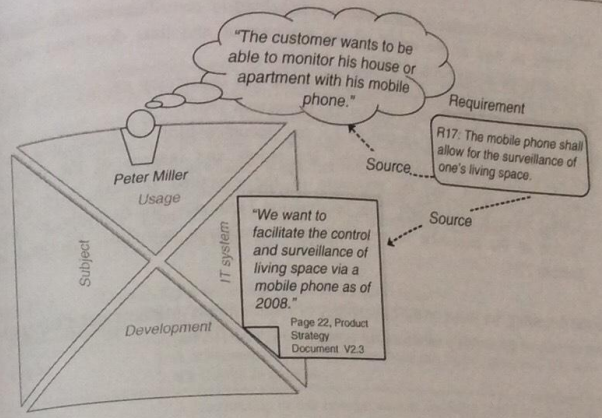

Fig. 31-3 *Example of pre-traceability of a requirement*

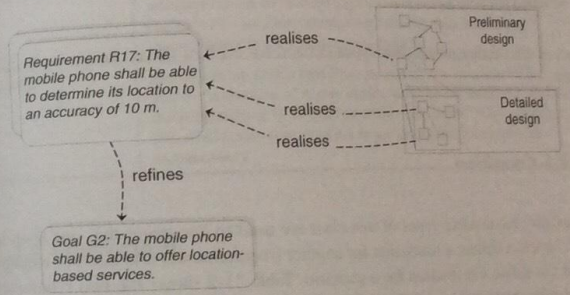

Fig. 31-4 *Example of post-traceability of requirements and traceability between requirements*

31.3 Traceability Relationship Types

Traceability information is often recorded in the form of traceability relationships. Between two development artefacts different types of traceability relationships (traceability links) can exist. For example, an artefact may refine another artefact, or an artefact may contradict another artefact.

In the literature, several suggestions for traceability relationship types have been made (see e.g. [Davis 1990; Gotel and Finkelstein 1994; Pohl 1996a; Thayer and Dorfmann 2000]). For example, one can distinguish between the following basic types of traceability (see [Dömges and Pohl 1998]):

- Traceability between requirements artefacts and successor artefacts such as derived requirements or architectural components [Pohl 1996a; Ramesh et al. 1995; Ramesh and Jarke 2001].

- Contribution structures which document stakeholders' contributions to the requirements as well as the origin of the requirement and thus document why a requirement exists [Gotel 1995].
- Traceability of design decisions, alternatives, and underlying assumptions [Pohl 1996a; Fischer et al. 1996; Conklin and Begemann 1988].
- Traceability of process execution, i.e. the documentation of actors and tools performing individual process steps, the inputs and outputs of the individual process steps, and the sequence of the executed process steps [Pohl 1996a; Ramesh et al. 1995]. This information facilitates, for example, the validation of the execution of requirements engineering activities described in Section 27.5.

Classes of traceability relationship types

In the following, we focus on the traceability of requirements artefacts and distinguish five classes of traceability relationship types (see [Pohl 1996a]:

- Condition
- Content
- Abstraction
- Evolution
- Miscellaneous

A set of traceability relationship types defined for each of these classes is described below.

31.3.1 Condition

Restriction of artefacts

Traceability relationship types of this class are used to document that one (requirements) artefact defines a restriction for another (requirements) artefact. For example, a goal can define a restriction for a scenario. Table 31-2 shows the two relationship types of the class "condition". The different relationship types are depicted in the column "relationship type" in Tab. 31-2. The column "description" contains a succinct description of each relationship type.

Tab. 31-2 *Traceability relationship types of the class "condition"*

Relationship type	Description
constraint	A relationship of this type from an artefact A to an artefact B documents that artefact A defines a constraint on artefact B. For example, a solution-oriented requirement can be constrained by another solution-oriented requirement.
precondition	A relationship of this type from an artefact A to an artefact B documents that artefact A defines a condition that must be fulfilled before artefact B can be realised. For example, the precondition for realising a functional requirement might be that the hardware meets a specific performance requirement.

31.3.2 Content

Dependencies between the contents of the artefacts

Traceability relationship types of this class are used to document dependencies between the content of the associated requirements artefacts or between requirements artefacts and other artefacts in the development process. A traceability relationship type of this class can be used, for example, to document a conflict relationship between two goals. For example, a conflict relationship can be introduced between two documented goals to document that satisfying one goal excludes the satisfaction of the other goal. Table 31-3 presents the four traceability relationship types of the class "content" and gives a brief description.

Tab. 31-3 *Traceability relationship types of the class "content"*

Relationship type	Description
similar	A relationship of this type documents that the two associated artefacts are similar in content.
compares	A relationship of this type between an artefact A_1 and a set of artefacts $A_2 \ldots A_n$ documents that A_1 represents the result of a comparison of the artefacts $A_2 \ldots A_n$.
contradicts	A relationship of this type between two artefacts documents that the two artefacts cannot be realised together. A "contradicts" relationship between two solution-oriented requirements thus indicates an inconsistency in the requirements artefacts.
conflicts	A relationship of this type from an artefact A to an artefact B documents that the realisation of A may hinder (but does not necessarily exclude) the realisation of B. In contrast to the "contradicts" relationship type, this relationship type does not hence (necessarily) document an inconsistency.

31.3.3 Abstraction

Abstraction dependencies between artefacts

This class comprises traceability relationship types that represent abstraction dependencies between requirements artefacts. Relationship types of this class can be used to document, for example, that a goal classifies a set of solution-oriented requirements or that an abstract scenario (e.g. a type scenario) is a generalisation of a set of more concrete scenarios (e.g. instance scenarios). Table 31-4 presents the three traceability relationship types of this class.

Tab. 31-4 *Traceability relationship types of the class "abstraction"*

Relationship type	Description
classifies	A relationship of this type between an artefact A and a set of artefacts $B_1 \ldots B_n$ documents that A classifies $B_1 \ldots B_n$.
aggregates	A relationship of this type documents that an artefact A is an aggregation of a set of other artefacts $B_1 \ldots B_n$.
generalises	A relationship of this type documents that an artefact is a generalisation of (one or) several other artefacts.

31.3.4 Evolution

Temporal relationships between artefacts

Traceability relationship types of this class document a kind of temporal relation between requirements artefacts or between requirements artefacts and other development artefacts. A relationship type of this class can document, for example, that a solution-oriented requirement is based on a scenario, or that an architectural component satisfies a set of requirements. Table 31-5 presents the five traceability relationship types of this class.

Tab. 31-5 *Traceability relationship types of the class "evolution"*

Relationship type	Description
replaces	A relationship of this type from an artefact A (e.g. a requirements artefact) to an artefact B documents that artefact B was replaced by artefact A.
satisfies	A relationship of this type from an artefact A to an artefact B documents that, if artefact A is realised in the system, artefact B is realised as well. The "satisfies" relationship type can be used, for instance, to relate a component to a requirement, thereby documenting that the requirement is realised by the component.
based_on	A relationship of this type from an artefact B to an artefact A documents that artefact A has influenced the definition of artefact B.
formalises	A relationship of this type from an artefact A to an artefact B documents that A is a formal documentation of the artefact B. A relationship of this type can be used, for instance, to relate a solution-oriented requirements model to a set of textual requirements.
refines	A relationship of this type from an artefact A to an artefact B documents that A refines B, i.e. artefact A defines the artefact B in more detail.
derived	A relationship of this type is used to document that a requirements artefact A was derived based on a (set of) other artefacts.

31.3.5 Miscellaneous

The class "miscellaneous" comprises additional traceability relationship types that can be defined between development artefacts. Traceability relationship types of this class document, for example, the fact that a scenario documents a concrete, exemplary interaction sequence for a dynamic aspect of a set of solution-oriented requirements. Table 31-6 presents the six traceability relationship types belonging to this class.

31.3.6 Traceability Relationship Types: Example

Figure 31-5 depicts some schematic examples of using the traceability relationship types explained in Sections 31.3.1–31.3.4 for documenting relationships between textual requirements fragments and model-based requirements artefacts.

Tab. 31-6 *Traceability relationship types of the class "miscellaneous"*

Relationship type	Description
example_of	A relationship of this type documents that an artefact contains exemplary aspects of a set of artefacts. For instance, an "example_of" relationship can be used to relate an interaction scenario to a set of solution-oriented requirements. This relationship documents that the scenario documents an exemplary sequence of interactions that a system implementing the solution-oriented requirements will support.
verifies	A relationship of this type is used to relate an artefact (e.g. a test artefact) to a requirements artefact. The former is used to verify or validate the requirements artefact.
rationale	A relationship of this type documents that one artefact documents the justification of another artefact. For instance, a "rationale" relationship can be used to relate a text fragment to a scenario, thereby documenting that the text fragment contains a justification for the existence of the scenario.
responsible_for	A relationship of this type documents that a stakeholder (or a role) is responsible for the associated artefact. Such a relationship can be used to document, for example, that a specific stakeholder is responsible for a specific requirements artefact.
background	A relationship of this type is used to assign "background information" to a requirements artefact. For example, a document containing a standard may be related to a solution-oriented requirement, thereby documenting that the standard must be considered during the specification or realisation of the requirement.
comment	A relationship of this type can be used to relate any kind of information to a requirements artefact. Use of this relationship type should be sparingly and really only if none of the other relationship types fits.

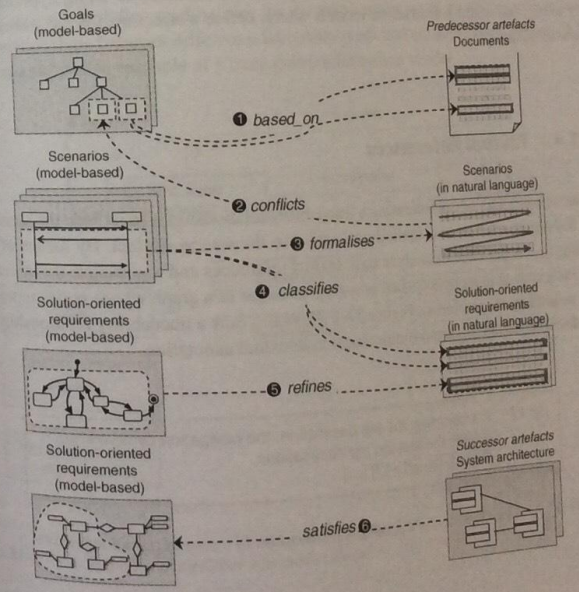

Fig. 31-5 *Examples of using the traceability relationships types*

The traceability relationships depicted in Fig. 31-5 are explained below:

❶ The traceability relationship with the label "based_on" documents that the associated goal is based on the text fragments of a predecessor artefact (e.g. fragments of the minutes of the interview).
❷ The traceability relationship with the label "conflicts" documents that a conflict between the textual scenario and the goal definition exists.
❸ The traceability relationship with the label "formalises" documents that the model-based scenario formalises the associated textual scenario.
❹ The two relationships labelled "classifies" between the model-based scenario and the associated textual, solution-oriented requirements document that the solution-oriented requirements are classified by the scenario.
❺ The traceability relationship labelled "refines" documents that the statechart (i.e. a model-based, solution-oriented requirement) refines the associated textual requirement.
❻ The traceability relationship labelled "satisfies" documents that the depicted fragment of the solution-oriented requirements model is realised if a specific component in the system architecture is realised.

31.4 Documenting Traceability Relationships

Traceability relationships can be documented in different ways. Besides the simple textual annotation, traceability relationships can be documented using hyperlinks or by using dedicated information models which define a structure for the traceability relationships.

31.4.1 Textual References

The simplest way of documenting a traceability relationship is to include the identifier of the target artefact as a textual annotation to the source artefact. By means of text analysis, it is possible to analyse such textual references and to visualise the traceability information in an appropriate way (e.g. as edges of a graph whose nodes represent the requirements artefacts). Figure 31-6 illustrates how a traceability relationship can be documented in a textual requirement as a textual annotation.

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

Fig. 31-6 *Documentation of traceability relationships by means of textual references*

31.4.2 Hyperlinks

Traceability relationships can also be documented by using hyperlinks. To document a traceability relationship, a hyperlink from the source artefact to the target artefact is created. Different types of traceability relationships can be documented by defining different hyperlink types. During traceability analysis, the hyperlinks may be visualised, for example, as edges of a graph.

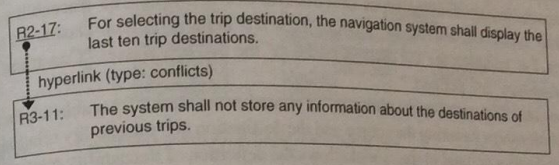

Fig. 31-7 *Documentation of traceability relationships by means of hyperlinks*

31.4.3 Traceability Models

Traceability relationship types are used to relate different types of artefacts (see Section 31.3). Use of an information model to define and structure the traceability information and relationship types is recommended. Such a traceability information model defines, for a specific project, the traceability artefact types and the traceability relationship types between the artefact types to be used in this project. Furthermore, such a model can define cardinalities for the relationship types as well as specialisation relationships between different relationship types and artefact types. Figure 31-8 depicts a simplified example of a traceability information model.

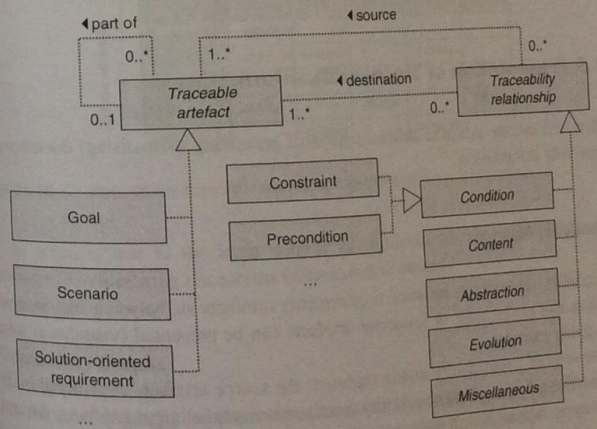

Fig. 31-8 *Simplified traceability information model*

Traceable artefacts

The information model shown in Fig. 31-8 defines an abstract class "Traceable artefact" specialised by various artefact types to be used in the project. Three specialisations are depicted in Fig. 31-8, namely "Goal", "Scenario", and "Solution-oriented requirement".

Traceability relationships

The class "Traceability relationship" is a super-class for all types of traceability relationships. It is specialised into the sub-classes "Condition", "Content", "Abstraction", "Evolution", and "Miscellaneous", denoting the different classes of traceability relationship types described in Section 31.3. The traceability relationship types introduced in Section 31.3 are defined as sub-classes of these classes. For instance, for the class "Condition" the two sub-classes "Constraint" and "Precondition" are defined.

Multiplicity of a traceability relationship

The class "Traceability relationship" is related to the class "Traceable artefact" by two relationships denoting the source and the destination of a traceability relationship. The cardinality defined in the model allows the definition of traceability relationships between sets of source artefacts and sets of destination artefacts. This can be restricted by defining more specific cardinality constraints for specific artefact types and/or traceability relationship types (see Chapter 20). For instance, a cardinality constraint could be defined which restricts all traceability relationship types defined between the class "Constraint" and the classes "Goal" or the class "Scenario" to a single instance of each class.

"Part-of" relationship between artefacts

Furthermore, the "part of" relationship of the class "Traceable artefact" enables the definition of part-of relationships between instances of the class "Traceable artefact". For instance, a class "Scenario step" and "Scenario" might be defined as sub-classes of "Traceable artefact". The class "Scenario step" might than be related to the class "Scenario" using the part-of relationship. Thereby, it would be possible to document that a scenario step is part of a scenario.

Development and use of a traceability model

In Section 31.6, we outline the creation of a project-specific traceability model based on project-specific usage strategies for traceability information and elaborate on the guidance to be provided for recording traceability information.

31.5 Presentation of Traceability Information

In this section, we describe different ways of presenting (visualising) documented traceability information.

Traceability Matrices

Traceability information between requirements artefacts or between requirements artefacts and predecessor or successor artefacts can be presented (visualised) using traceability matrices.

Traceability matrix for relationships of a single relationship type

The rows of a traceability matrix represent the source artefacts considered in this matrix. The columns of the traceability matrix represent the target artefacts. An artefact may be represented in the matrix, for instance, by its identifier. If a traceability relationship exists between the source artefact of row i and the target artefact of column j, cell (i, j) in the traceability matrix is marked. The type of traceability

relationships documented in the matrix can be stated in the top left cell of the matrix. Figure 31-9 illustrates a simple traceability matrix for the traceability relationship type "satisfies".

Target artefacts

Source artefacts

satisfies	Goal 1	Goal 2	Goal 3	Goal 4	Goal 5
Scenario 1					
Scenario 2					
Scenario 3	Traceability relationships				
Scenario 4					
Scenario 5					

Fig. 31-9 *Traceability matrix for a single relationship type*

The traceability matrix in Fig. 31-9 documents "satisfies" relationships between scenarios and goals. A scenario is represented in a row of the traceability matrix, a goal in a column. An entry "×" in the matrix documents a traceability relationship of the type "satisfies" from a scenario to a goal; i.e. the existence of a "satisfies" relationship between a scenario i and a goal j is shown is depicted as a filled cell (i, j).

Traceability matrix for multiple relationship types

In addition, also different types of traceability relationships can be documented in a single matrix. Figure 31-10 illustrates such a traceability matrix. Instead of marks, the cells of the matrix contain the relationship types. For example, the "conflicts" relationship between "Scenario 2" and "Goal 2" is represented by the entry "conflicts" in column 2, row 2.

Target artefacts

Source artefacts

	Goal 1	Goal 2	Goal 3	Goal 4	Goal 5
Scenario 1	*satisfies*				
Scenario 2	*based_on*	*conflicts*		*satisfies*	
Scenario 3		*satisfies*			*satisfies*
Scenario 4	*conflicts*		*satisfies*		
Scenario 5		*satisfies*		*based_on*	

Fig. 31-10 *Traceability matrix for several relationship types*

In practice, due to the large number of requirements, the use of traceability matrices for visualising traceability information is limited.

Traceability Graphs

Traceability relationships as edges

In a traceability graph, the nodes represent artefacts and the edges represent traceability relationships between the artefacts. To distinguish different artefact types and/or different traceability relationship types, different types of nodes and edges can be introduced. Alternatively, an attribute can be assigned to each node and/or edge of the

graph to denote the artefact or relationship types. Figure 31-11 illustrates the presentation of traceability information by means of a graph. In the traceability graph, a node type is defined for each type of requirements artefact (e.g. goal, scenario, solution-oriented requirement). In addition, three different relationship types, "based_on", "satisfies", and "example_of" (see Section 31.3), have been introduced.

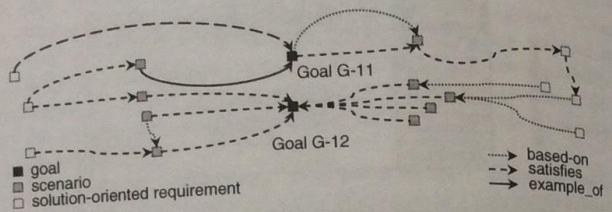

Fig. 31-11 *Representation of traceability information by means of traceability graphs*

Partial graphs

Based on the traceability graph, partial graphs may be derived. Such a graph can contain, for example, relationships of one specific type. Figure 31-12 depicts a partial graph derived from the graph in Fig. 31-11 that contains only the "satisfies" relationships between requirements artefacts.

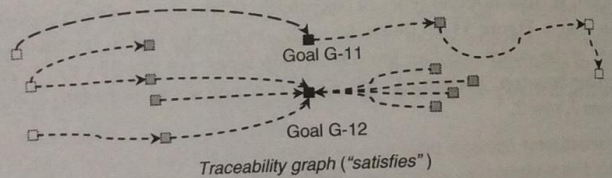

Fig. 31-12 *Partial graph derived from the graph in Fig. 31-11 showing "satisfies" relationships only*

Similarly, a specific artefact type (or set of concrete artefacts) can be used to derive a partial graph. For example, a sub-graph might show only the traceability relationships from Goal G-12 to all other artefacts.

Analysis of paths in a traceability graph

During traceability analysis, traceability graphs are often analysed along specific paths consisting of multiple nodes and edges. This kind of analysis reveals a path of artefacts (linked by traceability relationships) that logically belong together. Figure 31-13 shows a partial graph that has been derived from the graph in Fig. 31-11. This partial graph shows the traceability path for goal $G-12$ consisting of "satisfies" relationships.

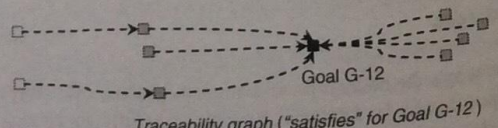

Fig. 31-13 *Traceability graph for goal G-12 consisting of "satisfies" relationships*

If relationships to predecessor artefacts (e.g. stakeholders and interview minutes) and successor artefacts (e.g. test cases and components) are also managed with the same tool, traceability graphs can be created for a requirements artefact which includes different artefact types. Even a complete traceability graph of the artefact over the entire lifecycle can be derived.

Common requirements management tools allow for choosing a search depth when creating traceability graphs. By choosing a depth of "1", only the immediate neighbours of an artefact are identified. If the depth is not restricted or a sufficiently large depth is chosen, the tool generates a complete traceability graph. Traceability paths are particularly important for change management. They are, among other things, the basis for impact analysis (see Chapter 33).

31.6 Project-Specific Traceability

Project-specific needs and constraints for recording traceability information

Recording and managing all traceability information that might potentially be of use in the further development process would require enormous resources. Recording all information is thus practically impossible. Therefore, for each project, a decision must be made regarding which traceability information should be recorded and maintained. This trade-off decision (which traceability information should be recorded and which information should not be recorded) has to take into account the resources (people, money, time) available for recording and managing traceability information. For recording and using traceability information, sophisticated strategies and guidelines should be provided in order to meet project-specific needs and constraints. In the majority of cases, the recording of traceability information without appropriate guidance leads to arbitrary accumulation of unstructured traceability information (see [Pohl et al. 1997; Dömges and Pohl 1998]).

Desired use determines the traceability information to be recorded

The project-specific traceability information to be recorded should be derived from the intended use of the traceability information and traceability constraints imposed by the organisation, the project, laws, the contract, standards or the like. For example, the customer might request that the developing organisation record the development effort needed to realise each customer requirement. This would require, among other things, an interrelation of each customer requirement with all the requirements and development artefacts derived from it as well as system maintenance artefacts such as change requests (see [Cordes and Carver 1989] for details). Furthermore, a certain law or standard could require that, for a safety-critical medical system, it must be possible to trace back each code fragment to the set of requirements realised by this code fragment.

31.6.1 *Project-Specific Traceability Environment*

Project-specific traceability environment

For defining and recording project-specific traceability information, different activities have to be performed. Figure 31-14 depicts a schematic structure of a so-called project-specific traceability environment which facilitates the definition, recording and use of project-specific traceability information.

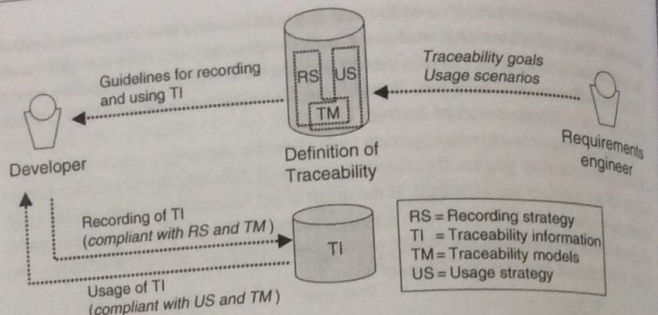

Fig. 31-14 *Scheme of a project-specific traceability environment (based on [Dömges and Pohl 1998])*

Traceability model (TM), recording strategy (RS), and usage strategy (US)

As illustrated in Fig. 31-14, the requirements engineer defines the project-specific traceability models (TM) as well as the strategies for recording (RS) and using (US) the traceability information during system development. The traceability information to be recorded (TM) is derived from the traceability usage strategies, i.e. for each type of traceability usage, the traceability information required for the particular usage is identified. The resulting traceability information required is then integrated into a coherent traceability model (TM).

Traceability information (TI)

During the development process, the members of the development team record the defined traceability information (TI) in compliance with the recording guidelines. The recorded traceability information (TI) is accessed within the development process according to the guidelines of the usage strategy. Similarly, the development team uses the defined traceability information according to the traceability usage guidelines (US).

Defining the project-specific traceability environment

To create a project-specific traceability environment, it is thus necessary to define the following elements:

- *Usage strategy*: The usage strategies define the intended use of traceability information during system development as well as during the whole lifecycle of the system. Each usage strategy requires a certain type of traceability information which is explicitly defined for each strategy. In other words, a usage strategy defines who should use which type of traceability information and when it should be used.
- *Traceability models*: The traceability models define the type of traceability information to be recorded. The type of information to be recorded is derived from the usage strategies defined for the project.
- *Recording strategy*: The recording strategy defines which type of traceability information has to be recorded for which activity and who is responsible for recording this type of information.

Goal- and scenario-based definition of project-specific traceability

Use of goals and scenarios to support the definition of project-specific traceability is recommended. Therefore, the stakeholders should first identify all relevant goals for recording traceability information. Subsequently, the stakeholders should define usage scenarios illustrating how the traceability goals are satisfied. The resulting

traceability usage scenarios are analysed, among other things, in order to identify relevant types of traceability relationships. Example 31-1 illustrates the identification of traceability relationship types based on a usage scenario. The parts of the scenario that are relevant for the identification of traceability relationship types are underlined.

Example 31-1: Scenario-based definition of traceability

Traceability goal: The traceability environment shall facilitate the calculation of the development effort spent for each requirements artefact.

Usage scenario (simplified): The project manager selects the requirements artefact for which the actual development effort shall be determined using the requirements management (RM) tool. The RM tool identifies all classes that implement aspects of the selected requirements artefact. The RM tool calculates the total implementation effort of the identified classes by summing the implementation efforts of the individual classes. Subsequently, the RM tool identifies all test cases and change requests related to the selected requirements artefact. Based on the identified test cases, the RM tool calculates the total test and correction effort that can be attributed to the implementation of the selected requirements artefact. Based on the identified change requests, the RM tool identifies all changes to the selected requirements artefact and calculates the total effort for the adaptation of the associated components and classes. Finally, the RM tool calculates the actual development effort for the selected requirements artefact by summing the total implementation effort, the total test and correction effort, and the total adaptation effort determined for this artefact.

Required traceability relationship types:

- class – satisfies – requirement
- component – satisfies – requirement
- test case – verifies – requirement
- requirement – based_on – change request

In the following, we briefly explain the definition of project-specific usage strategies for traceability information (Section 31.6.2), the definition of a project-specific traceability model based on the usage strategies (Section 31.6.3), the definition of recording strategies to collect and record the required traceability information (Section 31.6.4) as well as guidance offered to the stakeholders for recording and using traceability information (Section 31.6.5).

31.6.2 Usage Strategies for Traceability Information

After defining the goals for using the traceability information (e.g. to support change requests), the stakeholders define usage scenarios describing in which situation traceability information should be used and how it should be used. The usage scenarios are then condensed into so-called usage strategies which define the intended use of traceability information during system development. Since the usage strategies lay the

foundation for identifying the traceability information that needs to be recorded and maintained, usage strategies should be defined right at the beginning of the project. Moreover, if the usage strategies are defined later on, the traceability information recorded so far will most likely provide insufficient support for the intended usage strategies. A usage strategy should define (see [Dömges and Pohl 1996]):

- In which situation (When?)
- Which traceability information (What?) is used
- By whom, i.e. which role or agent (Who?)
- For performing which activity (For what?)

Example 31-2 shows the documentation of a single aspect of a usage strategy in natural language.

Example 31-2: Extract from a usage strategy

If a requirement is changed (When?), the requirements engineer who is responsible for the requirement (Who?) identifies which components are possibly affected by the requirements change in order to be able to estimate the effort for the integration of the requirements change. For the identification of the affected components (For what?) the recorded traceability information of the type "component – satisfies – requirement" (What?) is analysed.

Use of abstracted traceability information

Besides the recorded traceability information, abstractions (e.g. reduction, aggregation, or generalisation) derived from the recorded traceability information may also be used in a usage strategy. A simple example of an abstraction is the number of components that are affected by a requirements change. This information can be generated by aggregating the traceability relationships of type "component – satisfies – requirement".

31.6.3 Project-Specific Traceability Models

Definition of the traceability model

The traceability information required to fulfil the project-specific traceability usage strategies should be defined in a project-specific traceability information model.

For example, the usage strategy defined in Example 31-2 requires the recording of traceability relationships of the type "satisfies" between components and (solution-oriented) requirements. In addition, the strategy demands that, for each solution-oriented requirement, a responsible requirements engineer is assigned. Consequently this traceability information has to be defined in the project-specific traceability information model.

An excerpt of a project-specific traceability model which includes the information required for the usage strategy defined in Example 31-2 is depicted in Fig. 31-15.

Definition of traceability models based on a metamodel

A traceability metamodel defines the model elements to be used to define a project-specific traceability model. An example of a traceability metamodel can be found in [Ramesh et al. 1997]. A specific traceability model is created by instantiating the metamodel. The project-specific traceability models are thus instances of the traceability metamodel (see Section 19.5.2 for details on metamodels). Further

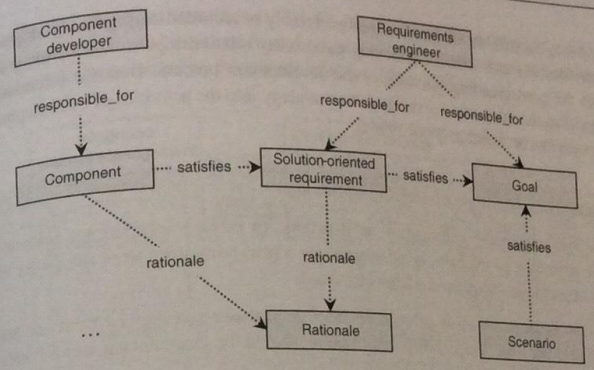

Fig. 31-15 *Simplified project-specific traceability model*

information about the project-specific definition of traceability models and the integration of the recording and usage of traceability information into process models can be found in [Pohl 1996a; Ramesh and Jarke 2001; Dömges and Pohl 1998].

31.6.4 Recording Strategies for Traceability Information

After defining the traceability information to be recorded and its structure (e.g. by means of a traceability model), project-specific strategies for recording the traceability information are defined depending on the objectives of the project and the available resources. A recording strategy determines (see [Dömges and Pohl 1998; Ramesh and Jarke 2001]):

- In which situation (When?)
- Which traceability information (What?) is recorded
- By whom, i.e. which role or agent (Who?)
- In which form (How?)

Documentation of the recording strategy

The different aspects of a recording strategy can be documented using natural language or using models (e.g. by means of an activity diagram or statechart). Example 31-3 shows the documentation of a single aspect of a recording strategy in natural language.

Example 31-3: Extract from a recording strategy

After a scenario has been documented (When?), the requirements engineer (Who?) responsible for this scenario documents the traceability relationship to the goal that is satisfied when executing the scenario (What?). The traceability relationship is recorded in the traceability module "scenarios × goals" using the traceability relationship type "scenario – satisfies – goal" (How?).

Integration of the recording strategy into existing process models

Since traceability information shall preferably be recorded during the development activities, the activities for recording traceability relationships should be integrated into the corresponding activities of the development process. Detailed information regarding the integration of the recording strategy into the activities of a development process can be found in [Pohl 1996a].

Hint 31-1: *Recording traceability information*

Record traceability information as it occurs. In other words, do not postpone the recording of traceability information to a post-processing step. Do not try to construct traceability information afterwards.

If, for example, a set of solution-oriented requirements is derived from another requirement, this traceability information should be documented immediately during the derivation.

If the information is recorded later, many aspects concerning the traceability information are no longer obvious, or documenting the traceability information is omitted because of other high-priority tasks.

31.6.5 Stakeholder Guidance

The project-specific traceability model and the project-specific recording and usage strategies lay the foundation for providing comprehensive guidance for recording and using traceability information within a development project. We distinguish two basic types of stakeholder guidance:

- Project-specific traceability handbook
- Process-integrated stakeholder guidance

Project-specific traceability handbook

The traceability information to be recorded (i.e. the traceability model) as well as the strategies for recording and using this information can be documented in a project-specific traceability handbook (see [Dömges and Pohl 1998]). In a training phase, the members of the development team first familiarise themselves with the traceability model and the recording and usage strategies defined in the handbook. During the development project, the team members apply the defined strategies. The traceability handbook then serves as a reference manual the team members can use to look up the relevant definitions while working on the project. However, using a project-specific traceability handbook does not guarantee that the traceability information is in fact recorded and used in compliance with the definitions.

Process-integrated traceability guidance

Ideally, a requirements engineering tool is available that supports process-integrated stakeholder guidance for recording and using traceability information (see [Pohl 1996a]). In this case, traceability information is recorded and used within the process steps (or development activities) executed in the process-integrated environment. Based on the traceability strategies, the process-integrated environment reminds the stakeholders about recording and using traceability information in the respective situations, i.e. the stakeholders are guided and supervised in recording and

using the traceability information (see Fig. 31-16). Moreover, a large part of the traceability information can be derived from the execution of the process steps and can thus be automatically recorded by the environment (see [Pohl 1996a]).

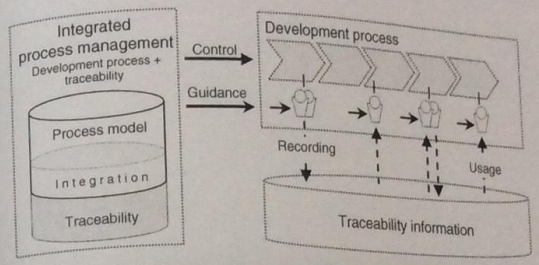

Fig. 31-16 *Process-integrated guidance of stakeholders (based on [Pohl 1996a])*

Example 31-4 illustrates the process-integrated guidance of a stakeholder based on a recording strategy for traceability information.

Example 31-4: Process-integrated guidance of stakeholders

Recording strategy: [...] For each goal bearing a high risk, decisions between alternative realisation possibilities shall be documented according to a predefined rationale structure. For each goal which bears a low risk, it is sufficient to record the corresponding decision in natural language.

Process-integrated guidance (documented as a scenario): During the development process, the stakeholder decides to choose one of several realisation possibilities for a high-risk goal. After documenting the realisation choice the requirements engineering tool executes the defined recording strategy and asks the stakeholder to document the rationale for this decision. Therefore it presents the predefined model to document the rationale. The stakeholder can decide whether to record this information right away or to create a task in his task list and record the information later on.

Detailed information regarding the process-integrated recording of traceability information can be found in [Pohl et al. 1996a].