An Organizational Memory for Quality-Based Software Design and Inspection: A Collaborative Multiview Approach with Hyperlinking Capabilities

Ilkka Tervonen, Pentti Kerola and Harri Oinas-Kukkonen
Department of Information Processing Science, University of Oulu
FIN-90570, Oulu, Finland
E-mail: {tervo, kerola, hok}@rieska.oulu.fi

Abstract

This paper addresses the problem of software quality information and the creation of an organizational memory founded on it. We focus on aspects such as how to organize the organizational memory, how to apply a collaborative multiview approach to it, and how to capture and use it in software design and inspection.

The presented solution is based on the integrative theories of human knowledge creation and the GRCM model, which provides an appropriate structure for the acquisition of quality information, on which to hang the theoretical and practical items found in the literature or collected from software engineers and inspectors. We place emphasis on collaborative hypermedia functionality as a key solution for fluent use.

1. Introduction

It is largely accepted that quality is like beauty, it is in the eye of the beholder. Thus, people may see quality everywhere or they do not see it at all, because they see it quite differently from different perspectives. Crosby [1] expresses this in the form: "Quality is hard to pin down, because each person thinks everyone else defines it the same way he or she does." This is true of software quality in particular, because the artifacts are very abstract things and allow freedom for people's imagination. In the light of our earlier interest in organizational memory, the wide coverage of the quality concept caused three questions: How well do software quality concepts cover the organizational memory? How well do software quality concepts support organizing the organizational memory? How well does a quality-based organizational memory support software design and inspection?

In this paper we consider these questions and focus on software quality information and knowledge creation by developing the special organizational memory in its human collaborative context. The concept of an "organizational memory" and its use is a multi-faceted one. In the present case we emphasize collaborative, complementary and communicative interaction between organizational and individual memories [2], [3]. As Konda et al. [4] express it, "one cannot have a meaningful sharing memory without shared meanings..." (for more on active and constructive views of organizational memory, see [5]). We generally agree with Conklin [6] that organizational memory extends and amplifies knowledge assets by capturing, organizing, disseminating and reusing the knowledge created by its participating human actors.

Our approach is based on our earlier experience and research in software design and inspection ([7], [8]), where a "GRCM (goal-rule-checklist-metric)" model was used to provide the structure for organizing quality information. When developing organizational memory for the domain of quality-based software design and inspection, our multiview approach has its theoretical basis in the integrative model of collaborative knowledge creation, developed from the individual-oriented experiential learning theory of Kolb [9] and from the team and organization-oriented approach of Nonaka and Takeuchi [10].

This multiview approach aroused further needs for the fluent use of quality information. We need rules and checklists tailored to the design methods and principles of a specific company, i.e. design and inspection principles gathered from practitioners and appropriate examples which illustrate the use of the GRCM model and principles in real contexts. In addition, the fluent use of quality information requires a tool which supports the construction and justification of design artifacts and the linking of diagrams with quality issues (i.e. goals, rules and checklists). We place emphasis on collaborative hypermedia functionality as a key solution for fluent use, in particular. This means that each participant may be a reader or author, and the tool provides appropriate support for each of them. We present a general concept of collaborative multiview hypermedia which is applied to the development of software quality-based organizational memory.
In summary our research problem is how to organize the organizational memory, apply a collaborative multiview approach to it, and fluently capture and use quality information throughout the software development process. Our solution follows the constructive research method in that, we provide a new approach to quality-based organizational memory and illustrate its capture and use in software design and inspection with an example. Our solution is based on the integrative theories of human knowledge creation and the GRCM model, which provides an appropriate structure for the acquisition of quality information, on which to hang the theoretical and practical items found in the literature or collected from software engineers and inspectors. As a supporting hypermedia tool we have used MetaEdit+, and especially its "subtool" for argumenting design decisions known as Debate Browser [11]. As MetaEdit+ is a tailorable graphical editor for any design method, it and Debate Browser can provide hypermedia functionality support for both design and inspection.

This paper is structured as follows. We first introduce the GRCM model in Section 2 and illustrate its principles with an example which explains the development of a quality training tool (i.e. a supporting tool for our GRCM approach). Then, in Section 3 we consider knowledge creation and conversion for an organizational memory as a theoretical background to the multiview model. After that, in Section 4, we discuss the hyperlinking capabilities of these, and especially their collaborative aspects, as an technique for implementing the multiview quality model. Finally the results are summarized and discussed.

2. The GRCM model

The GRCM model presented here provides a limited but appropriate common background for participants in software development and inspection. As depicted in Figure 1, the general quality goals are prioritized for a specific project and the ultimate objectives for software development are set. The origin of the GRCM model lies in its hierarchy of quality models (cf. [7]). Although the GRCM model is based on the SQM synthesis model (due to the numerous changes made to the Software Quality Metrics model [12], we call it a SQM synthesis), we also recognize its relation to the ISO 9126 standard [13].

The GRCM model has three links with the SQM synthesis, the goals correspond to factors (e.g. usability) and the rules to criteria (e.g. ease of use), and the metrics support quality measurement in both models. The goals are broken down into rules and further into checklists. The aim of the rules is to guide software engineers in software design. Checklists are generated from specific rules and are used as guidelines by inspectors, to help of them check that specific rules have been followed.

2.1 Introduction to the example

How do we use this GRCM model in practice? Let's

---

**Figure 1. The quality-intensive approach, based on the GRCM model**
take an illustrative example, which explains the use of quality information in the development of a "Quality Training" tool. As this tool explains the concepts and terms of our GRCM model, we recognize three topics to be learned: (1) quality goals, their definitions and potential interrelationships, (2) rules, their definitions and relations to the goals, and (3) checklists, their definitions and relations to the rules and metrics.

2.2 Defining the quality goals

We now first prioritize the most important of the 11 alternative goals (presented in the SQM synthesis) for the “Quality Training” tool. Even in this tiny example, we have two alternative branches to follow. We may choose to place our focus on teaching aspects, e.g. striving for understandable presentation of abstract quality concepts and using multimedia to explain company-specific inspection principles, or we can emphasize the importance of teaching abstract concepts in general drive our prioritizing process. We choose three goals: usability, expandability and reusability, and usability of the tool as the most important goal, as depicted in Figure 2.

![Figure 2. Reasons for the choice of quality goals](image)

<table>
<thead>
<tr>
<th>Teaching of abstract concepts</th>
<th>Usability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Different users</td>
<td>Expandability</td>
</tr>
<tr>
<td>New aspects</td>
<td></td>
</tr>
<tr>
<td>Limited resources</td>
<td>Reusability</td>
</tr>
</tbody>
</table>

We justify our prioritization as follows. The learning of abstract quality issues may not require extra efforts and it should be tailor able to different types of users. Further expandability of the tool is also important, because we must be able to illustrate quality issues with examples and company practices, e.g. presented in video form. The reusability goal forces us to evaluate our solutions from the viewpoint of the effort required, e.g. we must carefully justify any solutions which require development from scratch. As we see in Figure 2 (dashed lines), there are also some conflicts between reasons and goals, e.g. we can not count on reusability in constructing of new aspects and limited resources may detract the emphasis on usability.

The definitions of the goals are based on the ISO 9126 standard, and their form is as follows.

**Usability**
A set of attributes that bear on the effort needed for using the software and on the individual assessment of such use by a stated or implied set of users, e.g. the value of a user interface to a set of users.

**Ease of Use**
Attributes of software that bear on the user's effort needed for using the software in a specific environment, within a certain time and without too many errors.

Rule: Aim at user-friendly interfaces.

| Easy of Use | Checklist: Check that usability heuristics are followed in user interface construction, i.e. use simple and natural dialogue, speak the user's language, minimize user memory load, be consistent, provide feedback, provide clearly marked exits, provide shortcuts, provide good error messages, and prevent errors. |

2.3 Using the rules and checklists

In the next phase, the designer aims to achieve the quality goals in the best way possible, using the rules as guidelines. Following the GRCM approach, the goals are broken down into rules. The rules for Usability, for example, are Usefulness, Ease of Use, Learnability, Likeability and Operability. The descriptions of rules are derived from quality criteria, which are taken partly from the ISO standard [13], partly from SQM model [12] and partly from our own (cf. Ease of Use below).

**Ease of Use**
Checklist: Check that usability heuristics are followed in user interface construction, i.e. use simple and natural dialogue, speak the user's language, minimize user memory load, be consistent, provide feedback, provide clearly marked exits, provide shortcuts, provide good error messages, and prevent errors.

2.4 Applying the GRCM model

We now illustrate the use of the GRCM model in an example in which we consider the decision lying behind the user interface principle. As stated earlier, there are three topics to be learned: (1) quality goals, their definitions and
potential interrelationships, (2) rules, their definitions and relations to the goals, and (3) checklists, their definitions and relations to the rules and metrics. We now specify three scenarios for the first use case: Learn the quality goals and their potential interrelationships

<table>
<thead>
<tr>
<th>Scenario 1: Using a specific (tailored) window for selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Select a quality goal to learn</td>
</tr>
<tr>
<td>2. Select an interrelationship</td>
</tr>
<tr>
<td>3. Select a supporting or conflicting aspect</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scenario 2: Using standard menus for selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Select a quality goal to learn</td>
</tr>
<tr>
<td>2. Select one participant in an interrelationship</td>
</tr>
<tr>
<td>3. Select another participant in an interrelationship</td>
</tr>
<tr>
<td>4. Select a supporting or conflicting aspect</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scenario 3: Typing the input in a standard query box</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Type the name of a quality goal to learn</td>
</tr>
<tr>
<td>2. Type the name of one participant in an interrelationship</td>
</tr>
<tr>
<td>3. Type the name of another participant in an interrelationship</td>
</tr>
<tr>
<td>4. Select a supporting or conflicting aspect</td>
</tr>
</tbody>
</table>

Following our priorities, the designer has used the Usability goal as the most important one and has chosen Scenario 1. He justifies this by the Ease of use rule, i.e. "Aim at user-friendly interfaces". The participants may use the checklist of the Ease of use rule in inspection phase and check that an alternative uses simple and natural dialogue, for instance, speaks the user's language, minimizes user memory load, is consistent with other tools used, provides enough feedback and provides shortcuts for professional users.

2.5 Problems encountered in the GRCM model

Although the GRCM model gives an easy way of proceeding on quality issues, this straightforward model has some drawbacks. We evaluated the understandability of the quality issues on a scale of 1 (bad) to 5 (excellent) in a course on reviewing and testing, and the students (answers from 19 participants recorded) understood the definitions to a moderate degree (2.6) and agreed with them to a moderate degree (3.2). Based on these experiments we agree that the present implementation of the GRCM model places too much emphasis on conceptual definitions and gives too abstract an introduction to the quality world. The definitions based on the SQM model and the ISO 9126 standard give a rather technical view of quality terms, and are also tailored to the object-oriented approach, particularly its design and coding phases. The definitions and interrelationships nevertheless aroused discussion and helped the students to realize the problems associated with quality assessment. The evaluation also disclosed a major deficiency in representing the interrelationships, in that it was possible to see either synergic or conflicting interrelationships between the two quality goals, whereas in the real world there may be both.

All in all, the recognized shortcomings in our present approach, in that it does not provide easily understandable definitions for all users, nor appropriate representations of quality information, are just examples of typical shortages in organizational memory, as we will note in the next section.

3. Knowledge creation for an organizational memory

The main idea of this section is to describe an integrative metamodel for human knowledge creation and to give the main reasoning behind it from two complementary perspectives: (1) the generic core concepts of Kolb's individual-oriented model for knowledge development [9] and (2) Nonaka-Takeuchi’s team and organization-oriented model for knowledge creation [10].

3.1 Kolb's generic concepts of human knowledge development

Kolb’s Experiential Learning Theory [9] has its generic base in a two-dimensional tension field of knowledge development acts producing four complementary human knowledge categories as outputs (Figure 3). Both the vertical concrete/abstract axis and the horizontal action/reflection axis are dialectic, i.e. both contradictory and complementary, and this is reflected in the whole structure and all the interactions included in it. In principle, all the knowledge-producing subactions can be precedent to each other and to their outputs, but often the interrelationships emphasize the multicircle/spiral interaction described in particular by Lewin [14].

The major source of pattern and coherence in individual development is the underlying structure of the process described in Figure 3. Over time, individuals in the interaction with other human actors, nature and the whole environment develop their typical possibility-processing structures in such a way that the dialectic tensions are contextually and consistently resolved in a characteristic fashion. People more or less consciously develop experiential learning and cognitive styles that emphasize/disregard certain knowledge categories and abilities relative to others. Using his knowledge taxonomy (which is ideal in the Weberian sense), Kolb defines the four basic categories of experiential styles, each emphasizing two knowledge types and disregarding the
opposite two.

The *converger* style relies primarily on the dominant abilities of abstract conceptualization and active experimentation. The greatest strength of this style lies in problem solving, decision making and the practical application of theoretical or metatheoretical ideas. Knowledge is organized in such a way that it can be focused on specific practical or research problems through hypothetical-deductive reasoning.

The *accommodator* style has the opposite strengths to the assimilator, emphasizing concrete experience and active experimentation. Its greatest strength lies in doing things, in carrying out plans and tasks and becoming involved in new experiences. The adaptive emphasis of this orientation is on opportunity seeking, risk taking and action. In situations where the theory or plans do not fit the facts, those with an accommodative style will most likely discard the plan or theory. Such people tend to solve problems in an intuitive trial and error manner, relying heavily on other people for information rather than on their own analytical ability.

### 3.2 Team and organization-oriented knowledge creation model

Just as Kolb declares that "knowledge results from the combination of grasping human individual experience and transforming it", Nonaka and Takeuchi also take this as a necessary assumption ([10], pp. 58-59):

"...we consider knowing as a dynamic human process of justifying personal belief towards the "truth". ... In a strict sense, knowledge is created only by individuals...".

The main essence of their theory of organizational knowledge creation is nevertheless the integration of individual, team and organization orientations. Following the same 'form of description' as with Kolb, their complementary two-dimensional knowledge field is spanned by the vertical epistemological and horizontal ontological dimensions (Figure 4).

![Figure 3. Kolb's generic model for individual knowledge development, categories and experiential styles](image)

- **Figure 3. Kolb's generic model for individual knowledge development, categories and experiential styles**

  The *diverger* style has the opposite knowledge strengths, emphasizing concrete experience and reflective observation. The greatest strength of this orientation lies in imaginative ability and awareness of meaning and differences. The primary adaptive ability of a diverger is to view concrete situations from many perspectives and to organize many relationships into a meaningful "gestalt". The emphasis is on adaptation by observation rather than action. Those oriented towards the diverger style are interested in people but not very active, and tend to be imaginative and emotionally oriented.

  The dominant knowledge abilities for the *assimilator* are abstract conceptualization and reflective observation. The greatest strength of this orientation lies in inductive reasoning and the ability to create theoretical models and assimilate disparate observations into an integrated explanation. As in the converger, this orientation is less focused on people and more concerned with ideas and abstract concepts. Ideas are judged less by their practical value in this orientation, however.

- **Figure 4. Nonaka-Takeuchi's two dimensions of knowledge creation**

  Tacit knowledge is personal, context-specific, and therefore hard to formalize, not needing any specific language for communication. Explicit or "codified" knowledge is transmittable in formal/systematic language.
The assumption that knowledge is created through the interaction between tacit (T) and explicit (E) knowledge leads to the postulation of four different knowledge conversion, as described in Figure 5.

*Socialization* (from T to T′) is a process of sharing experiences (between people) and thereby creating tacit knowledge in the form of mental models and technical skills that can be called "sympathized knowledge". *Externalization* (from T to E) is a collective reflection process articulating tacit knowledge to form explicit concepts. It is quintessential, assuming the shape of metaphors, analogies, concepts, hypotheses or models, and can be called "conceptual knowledge". *Combination* (from E to E′) is a team interaction process of systemizing concepts into a knowledge system. Individuals exchange and combine knowledge through such media as documents, meetings, telephone conversations or computerized communication networks, leading to new, explicit "systemic knowledge". *Internalization* (from E to T) is a process of embodying explicit knowledge in tacit knowledge, "learning by doing" and re-experiencing. The output can be called "operational knowledge".

![Figure 5. Four modes of knowledge creation and conversion](image)

Although in principle all the modes of knowledge conversion are in continuous, dynamic complementary interaction with each other, Nonaka and Takeuchi emphasize the spiral evolution of knowledge creation: socialization -> externalization -> combination -> internalization -> socialization etc. The interaction between T and E becomes larger in scale, starting at the individual level and moving up through expanding interaction communities that cross sectional, departmental, divisional and organizational boundaries (horizontal progress from right to left in Figure 5).

### 3.3 Implications for the development of organizational memory

We may illustrate the integrative model of Nonaka and Kolb and its implications for software inspection as follows. In the original GRCM model we provided quality information mainly from the assimilator style perspective, so that more practical viewpoints such as the company's design principles and examples are missing. According to Kolb's theory we should provide a fair view of all these aspects of quality information.

A practical illustration of Nonaka's theory (cf. Figure 5) may be presented by means of an inspection example. In socialization the participants (client's representatives or software engineers in various roles, e.g. project leaders, quality managers, software designers) share their quality experiences and discuss them in an informal way, e.g. during coffee breaks. This discussion is one way of constructing shared meanings. In externalization, inspectors organize their own tacit knowledge into explicit concepts, e.g. using the GRCM model and a supporting tool for recording purposes. The combination may be based on meetings and tools that support teamwork. The keywords are collaboration and shared knowledge, as the participants should be able to understand other people and be understood by them. We may help the combination e.g. by means of display systems [6] which support the capturing, organizing and displaying of the information. The externalization and combination activities are the most important from the viewpoint of explicit knowledge, i.e. the account for how we create explicit knowledge from informal knowledge. In internalization the explicit quality knowledge is further converted into the tacit knowledge of participants. This may be happen during the inspection process, for example, in which novices follow the work of other inspectors. We emphasize that the learning involved in internalization and socialization processes is closely related to the attitude of the participants, i.e. whether they understand their weak points in the sense of learning styles, for example.

A more theoretical summary of Kolb's and Nonaka's model can be given in terms of the following preliminary conjectures:

1. (Nonaka): Development and use of organizational memory is mainly a matter of externalization and
2. (Nonaka): Internalization and socialization are the most human-sensitive subprocesses, and especially in the problem domain where information and knowledge concerns quality aspects of abstract objects.

3. (Nonaka and Kolb): The original GRCM model is one typical case of externalization. It provides a structure with which a software engineer can organize tacit knowledge into explicit concepts.

4. (Kolb): The ideal principle of "knowledge category coverage" can be conjectured in the manner: "About any object of human interest all four knowledge categories should be expressed as explicitly as possible."

5. (Kolb): The ideal principle of collaboration can be conjectured in the manner: "Because of style preferences/biases, all human actors should collaborate as developers (authors) and users (readers) of an organizational memory."

6. (Kolb): The ideal principle of style awareness can be conjectured in the manner: "Individual and team awareness of actors’ experiential styles would lead to a more balanced process of coauthoring and reading" [15].

All in all, these statements explain why the original GRCM model-based approach is inadequate for quality-based organizational memory and why we need a multiview approach, although a more specified reasoning of these conjectures is not possible within the limits of this paper. We will now illustrate especially the externalization and combination activities by means of a design and inspection example in Section 4 and evaluate the supporting characteristics of a display system based on hypermedia technology.

4. Collaborative hypermedia functionality

Organizational memory research often refers to hypermedia technology as a means of implementing organizational memory. End-users do not need hypermedia systems per se, but they do need information systems which support their work and are reasonably easy to use. Thus, instead of introducing new hypermedia tools or environments, there should be a means of enhancing existing information systems, most of which are not hypermedia systems, with the desired hypermedia features.

This is called the hypermedia functionality approach (HMF), which means that a set of hypermedia features are incorporated into a software system [16], [17]. Hypermedia functionality is seen as a value-added support functionality for software systems, allowing the information to be investigated in a non-linear, semi-structured way. Core hypermedia functionality consists of the creation, modification and deletion capabilities of various kinds of hyperlinks and nodes, and orientation and navigation capabilities achieved via the hyperlinks. Hypermedia functionality can be incorporated into any knowledge-intensive software system so as to provide its users with an associative way of organizing, analyzing and accessing information. We are concerned here mostly with collaborative hypermedia functionality (CHMF), meaning simply HMF in collaborative software systems, i.e., systems that emphasize the groupware or team aspects of knowledge work. Even so, team support always takes place through supporting individual team members, just as organizational learning always takes place through individuals.

Collaborative HMF support of this kind has been implemented in MetaEdit+, which is a multiuser, multithread metaCASE environment [18]. MetaCASE technology in general enables the definition of CASE tool support for one's own methods and methodologies for the purpose of planning, analysis and design, and also the adaptation of existing methods and their CASE tool support. On account of the environment’s metamodelling capabilities, the tools described can equally well be utilized for any modelling activities, e.g., information architecture, planning, structured analysis, object-oriented design, organizational modelling, or business process modelling. At the same time the concept of collaborative hypermedia functionality is general enough to be applicable to other groupware tools in a similar manner. MetaEdit+ consists basically of a set of tools for constructing design artifacts, e.g., Diagram, Table and Matrix Editors, a toolset for justifying and inspecting the design decisions made, known as Debate Browser [11], and a hypermedia functionality underlying the construction and argumentation tools, known as Linking Ability [19].

Diagram Editor in MetaEdit+ simply enables the drawing of diagrams using a specified method, which can be determined by the user. The conversation on development issues is captured in Debate Browser with the fixed QAR design rationale method. The method consists of questions (q), answers (a), and arguments (t), which either support or contest the answers. Debate Browser actually provides its users with three browsers: a graph browser, a document browser and a node browser. The graph browser represents the overall structure of the design rationale in graphical form, while the document browser represents a focused view of the text structure in textual form and the node browser enables exploration of individual nodes.

Collaborative HMF behind of these tools is received through Linking Ability subsystem. Diagram editor (as well as other editors) enables to link any design objects to
each other through hyperlinks, e.g. to trace back to requirements. All hyperlinks can have attributes associated with them to provide more information. These associative hyperlinks integrate also the argumentation for constructions (i.e. design diagrams), by means of which any argumentation node can be reached from the design objects in a diagram. It also allows a user to ensure that all outgoing/incoming links from/to a diagram have been checked. Readers are able to make link attribute queries, e.g. to obtain 'all design rationale links created after 5/31/1996', 'all links to which the keyword important is attached' or 'all links created by HOK'.

The interaction history list supports backward navigation by moving to a node that has already been visited. Two kinds of marks are also provided as guides for readers. A bookmark is a marker for keeping one's place in the collection of documents, and can be attached to a specific design object in a document (if it might be of interest later), while a landmark directs the reader to an important region in the document (e.g. starting point for an inspector). Their semantic difference is that, while a bookmark is meant to be personal, a landmark is meant to be informative for all readers.

The tools described here can be utilized not only for capturing design rationales, annotations and dependences lying behind the analysis and design diagrams, but also for debates and rationales lying behind information system development methodologies or methods [20], or any other issues. When tools such as Diagram Editor and Debate Browser are empowered with hypermedia functionality, they become especially interesting from the organizational memory viewpoint, as it is the collaborative hypermedia functionality which enables the different kinds of dependences to be represented explicitly and used fluently in the information system.

The design rationales and their linkages can be effectively built up based on the GRCM software quality model. See the example in Figure 6, in which the selected quality goals for the project at hand (usability, expandability, and reusability) are defined in answer to the question “What are the selected quality goals?”.

Activation of a hypertext link in an answer node Usability (which is not represented in Figure 6 for economy of space) leads to definition of the general quality term Usability, which is stored in an annotation (see Figure 6). It is further linked to annotations that define the ease-of-use, learnability and usefulness rules. An argument for the usability quality goal can be seen in the background. Figure 7 depicts a justification of a design decision and is based on the quality terms of the GRCM model. The annotation and debate nodes have associative connections with the design diagrams via hypertext links. Activation of a hypertext link labelled “Explanation” in Figure 7 enables a jump to be made from the design diagram to the annotation and opens up an annotation “Explanation of interrelationships”.

The annotation “Explanation of interrelationships” has hyperlinks to annotations defining the Usefulness rule and Usefulness checklist. In this manner, software designers are encouraged to justify their design decisions by reference to rules. In a similar way, inspectors are supported by checklists which help them to check that designers have followed the specific rules. Another link, labelled “Independence”, explains the decision situation in which the designer has decided to implement the Checklist and Metric entities as classes of their own and has justified this in terms of the upgradeability rule. We can see these design rationale conversations as examples of externalization and combination activities, where a designer

Figure 6. An example of selected quality goals
formalizes his/her tacit knowledge to obtain explicit knowledge by accepting a specific quality definition as a justification for a decision, or by giving his/her own argumentation. At an inspection meeting the team may discuss alternatives and their justifications, and in that way collaborate with each other. In addition to these, the most human-sensitive subprocesses, internalization and socialization, may also be supported to some extent. Personalized bookmarks and unstructured annotations and hyperlinks may support the creation of new tacit knowledge (internalization). The process of capturing annotations, building hyperlinks and defining landmarks may also support the sharing of experiences between people (socialization). Thus the conjectures 1 and 5 are partially supported. Debate Browser provides an automated means of organizing tacit knowledge into explicit concepts (see conjecture 3). The collaborative hypermedia functionality through Linking Ability also enables the addition of new quality definitions to the quality model by any user with minimal disruption of the normal working process (thanks to seamless integration into the Diagram Editor) and supports the creation of linkages between actual design decisions and the quality model by any user.

The tools described in this chapter can in principle be utilized to capture all four knowledge categories of Kolb, thus also supporting the different knowledge styles (conjecture 4). Collaborative hypermedia functionality has the potential to blur the distinction between separate modes for authors and readers [21], making all users capable of creating hyperlinks and nodes (conjecture 5).

To summarize, just as a quality model is regarded as a solid basis for creating an organizational memory, collaborative hypermedia functionality implies a natural means of supporting the fluent use of this memory. Both linking and annotation features (authoring), and orientation and navigation features (reading) support the core tasks in the creation and use of the quality-based organizational memory. Annotations and debates can be used to prioritize quality goals in quality terms by setting out definitions, principles and examples, by recording new definitions, design and inspection principles and examples 'on the run', and in particular by integrating quality-based argumentation into design artifacts.

5. Conclusions

This paper addresses the problem of software quality information and the creation of an organizational memory founded on it. We focus on aspects such as how to organize the organizational memory, how to apply a collaborative multiview approach to it, and how to capture and use it in software design and inspection.

The organizing is based on a GRCM model, which provides a straightforward but limited structure for managing quality information. We searched for the theoretical background for a more flexible, multiview model and evaluated our original GRCM based approach against the models of Kolb and Nonaka recognizing some shortcomings. Kolb's theory shows that practical viewpoints on quality information such as the company's design principles and examples are missing, while from Nonaka's viewpoint we recognized various new challenges. Although the development and use of organizational memory is mainly a matter of externalization and combination subprocesses, we should also remember internalization and socialization, especially in our problem domain, where information and knowledge concerns quality aspects of abstract objects.

We have also illustrated the different subprocesses of knowledge creation in a design and inspection example, in particular externalization and combination activities, and evaluated the supporting characteristics of a design rationale tool.

Figure 7. Justifying design decisions
When tools such as diagram editors or design rationale systems are empowered with hypermedia functionality, they become especially interesting from the organizational memory viewpoint, as it is the collaborative hypermedia functionality which enables the different kinds of dependences to be represented explicitly and used fluently in the information system.

Our future work will include the development of a "Quality Training" tool which supports the presented multiview approach and allows recording of all types of quality information (e.g., display snapshots and video).

6. References


