Method rationale in method engineering and use

H. Oinas-Kukkonen
Department of Information Processing Science
P.O. Box 400, FIN-90571 Oulu, Finland
Tel. (358) 81 553 1900, Fax. (358) 81 553 1890
Email: hok@rieska.oulu.fi

Abstract
While the major aspect in method engineering is method assembly, a second aspect is the argumentation behind the methods. This paper introduces the concept of method rationale in method engineering and use as a communication vehicle between method and software engineers, and describes tools that support the capture and management of method rationale in a computer-aided method engineering environment.

Keywords
Information systems development, method engineering, CASE, metaCASE, CAME, design rationale, method rationale, hypertext

1 INTRODUCTION

The emergence of metaCASE and Computer-Aided Method Engineering (CAME) technology has provided the field information system development (ISD) with new promises. One of them is the utilization of the same computerized environment to develop both ISD methods and target information systems. Still, even when methods can be assembled in a CAME environment, part of the important method-related knowledge normally remains implicit, e.g. experience accumulated about the methods in use (Jarke et al., 1994). Computerized support for capturing this kind of semi-structured information is also needed.

A simple means for modelling experience information in conjunction with metaCASE is to maintain a textual note for each method (Heym and Österle 1993). The solution in this paper, however, provides a method and metamodelling independent, and a more sophisticated and effective means for recording semi-structured information. This paper considers information systems development and method engineering as a special case of design, and bases its solution on the concept of **design rationale** (Fischer et al., 1991, Ramesh and Dhar, 1992). Design rationale means basically the understanding of **why an artifact has been designed the way it has**, which may include information on e.g. requirements, assumptions, decisions, and alternative solutions. The benefits of design rationale capture include the achievement of increased rigor and clarity of thinking, augmentation of the designer's memory, better communication among team members and stakeholders, and improved meetings (Conklin and Yakemovic, 1991). The concept of design rationale in conjunction with method engineering and ISD (i.e. method use) is leveraged here in the following manner. **Method rationale means method design and usage rationales and their linkages to design artifacts across various phases of method engineering and use.**

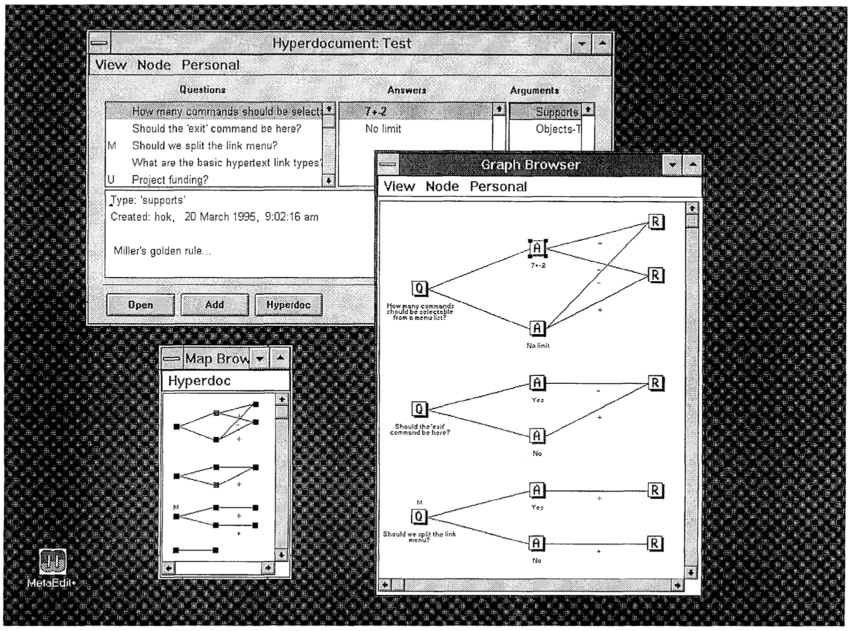
This paper introduces tools which can be utilized to support method rationale in a full-blown CAME environment, and describes an approach by which method rationale can be used as a communication vehicle between the stakeholders in this kind of environment.

2 RESEARCH ENVIRONMENT

MetaEdit+ is a fully configurable multiuser, multitool Computer-Aided Software and Method Engineering environment (Kelly et al., 1996). In addition to basic model editing and retrieval tools, MetaEdit+ also includes tools which enable the creation, modification and deletion of annotations and navigational hyperlinks between models or their parts. The model annotation and linking tools (Debate Browser and Linking Ability) are seamlessly integrated with the model editing tools (Diagram Editor, Matrix Editor and Table Editor), and they are used for commenting model instances, maintaining conversations about design issues, linking design objects for traceability and as a reminder, or finding specific locations in the design space.

Debate Browser

The Debate Browser is a hypertext-based toolset for supporting the capture and use of design rationale knowledge. It utilizes an argumentation method similar to IBIS (Conklin and Begeman, 1988), known as QAR (Question-Answer-aRgument). QAR has been abstracted from various design rationale methods for our purposes in MetaEdit+ environment, simplifying the explicit rhetorical structure of design rationale (Oinas-Kukkonen, 1996). The discussion is expressed using three kinds of nodes, questions, answers, and arguments. There is also a particular way of registering that a question has been resolved by agreement upon some answer by selecting and presenting one of the suggested answers as a decision. A node always belongs to a **hyperdocument**, a collection of discussions, consisting of **nodes and links** between the nodes. There can be various design rationale hyperdocuments for debates on different kinds of subjects, for example different organizational, research, product and other problem domains, as well as within a project for analysis, design, implementation and review concerns.



How many commands should be select
Should the 'exit' command be here?
M Should we split the link menu?
What are the basic hypertext link types
U Project funding?

7+-2
No limit

Supports
Objects-T

Type: 'supports'
Created: hok, 20 March 1995, 9:02:16 am

Miller's golden rule...

How many commands should be selectable from a menu list?
7+-2
No limit
Should the 'exit' command be here?
Yes
No
Should we split the link menu?
Yes
No

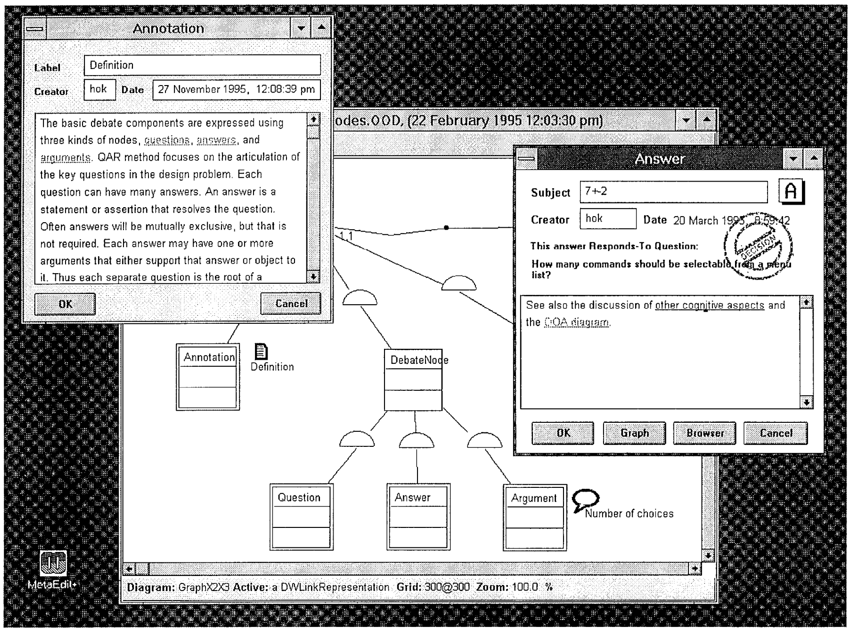
Figure 1 Graphical and textual views in Debate Browser toolset.

Debate Browser enables the investigation of design rationale hyperdocuments and their nodes and links with two browsers, a document browser and a graph browser. (See Figure 1.) The document browser gives text lists of all the question nodes of a hyperdocument, the answer nodes of an activated question, and the argument nodes of an activated answer, with the active node always visible at the bottom of the browser. The graph browser presents a graphical web view of the design rationale hyperdocument, supporting the investigation of a full hyperdocument as well as a single question and its associated answers and arguments. The zoom capability enables the investigation of the hyperdocument through map views (see lower left corner in Figure 1). The investigation of questions and their relationships is enabled from different perspectives in all views, i.e. as a plain collection of questions or from generalization-specialization, replacement-replacer, or parent-child perspectives. Nodes which have not yet been investigated by an individual reader can be highlighted, node marking is enabled, and summary reports of the design rationale hyperdocuments can be given among other features.

Linking Ability

Design rationale has to be integrated with construction environment to contextualize the rationale, and with the issues to concentrate more on design than merely philosophical discussions (Fischer et al., 1991). This can be achieved via attaching associative

hyperlinks to design diagrams and design rationale nodes (which are different from the responds, supports etc. links within design rationale hyperdocuments) through the Linking Ability tool. All hyperlinks are created by hand at will, and they can lead to any other design rationale node or diagram. More semantics can be stored into a link through link attributes, e.g. type information or keywords. The hyperlink attribute query facility helps to find specific linkages. Other sophisticated hypertext browsing features include an interaction history, filtering mechanism, and landmark and bookmark lists. This kind of hypermedia functionality also gives good modelling transparency (Brinkkemper, 1993).


Annotation

Label: Definition
Creator: hok Date: 27 November 1995, 12:08:39 pm

The basic debate components are expressed using three kinds of nodes, questions, answers, and arguments. QAR method focuses on the articulation of the key questions in the design problem. Each question can have many answers. An answer is a statement or assertion that resolves the question. Often answers will be mutually exclusive, but that is not required. Each answer may have one or more arguments that either support that answer or object to it. Thus each separate question is the root of a

odes.OOD, (22 February 1995 12:03:30 pm)

Answer

Subject: 7+-2
Creator: hok Date: 20 March 1995, 8:59:42

This answer Responds-To Question:
How many commands should be selectable from a menu list?

See also the discussion of other cognitive aspects and the COA diagram.

Diagram: GraphX2X3 Active: a DWLinkRepresentation Grid: 300@300 Zoom: 100.0 %

Figure 2 Linkages between a diagram, a design rationale node and an annotation.

Linking Ability also enables the attachment of annotations to diagrams or design rationale nodes. (See Figure 2.) There are two hyperlinks in an object-oriented diagram, represented by graphical symbols. The traversal of the hyperlinks takes the reader to corresponding nodes. The 'Definition' hyperlink leads to an annotation node 'Definition' commenting on the relationships between design rationale nodes. The annotation node includes three hyperlinks to other annotations. The 'Number of choices' hyperlink leads to a design rationale node (answer) '7+-2', which has been selected as a decision for a certain question. The design rationale node includes hyperlinks to another node and a diagram as well. Relationship representation and navigation and requirements tracing can be supported through this kind of linking capability.

3 APPLICATION OF THE MODEL ANNOTATION AND LINKING TOOLS

Let us now imagine a software project, consisting of a group of software designers and a smaller group of method engineers. Methods for business processes, information system planning, analysis and design, e.g. value chain, work flow models, and OMT (all adapted to the situation at hand), have been defined by **method engineers** using the metamodelling language and its rationale. Both the method-specific and general design rationale behind this method assembly has been captured using the Debate Browser. Figure 3 describes the role of method rationale in ISD and ME activities (it is modified from the software process support of Jarke et al. (1994)).

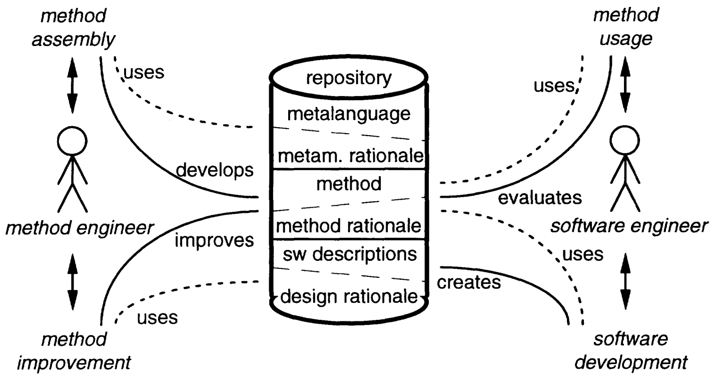


Figure 3 Method rationale in ISD and ME activities.

The methods and their design rationale guide **software designers** in their work. When software engineers use the methods to develop software artifacts they parallelly evaluate the methods in a realistic context and capture debates on them into the corresponding hyperdocuments. Software engineers are also encouraged to record software design rationale. All this takes place through the Debate Browser. When design problems or method evaluation are not mature enough for explicating debates, software engineers can attach annotations to design diagrams instead of structured discussions. Software engineers may also represent dependencies between artifacts and rationale through hyperlinks in Linking Ability, in which they may also attach specific keywords, e.g. 'method' to represent its perceived potential for method improvement.

Method engineers are interested in investigating the debates, annotations and linkages, which relate to methods and have been recorded by software engineers during their normal working process. First, they investigate the method evaluation documents, consisting of debates between software engineers regarding various aspects of method usage. Second, method engineers investigate method-related annotations through Linking Ability. Third, they are able to find and traverse the dependencies between argumentation and the design artifacts. Method engineers especially benefit from link attribute queries, such as obtaining all hyperlinks, where keyword 'method' is attached, then backtracking or traversing to either of the link end-points.

To summarize, the model annotation and linking tools can be utilized in method engineering and use in the following ways.

- Method engineers use Debate Browser to capture the design rationale behind the method assembly, which then guides software engineers in their work. Later, during development projects software engineers and reviewers capture debates behind software design, relating this kind of project performance either directly or indirectly to methods used.
- Software engineers use Linking Ability to represent the dependencies between models, rationale, and annotations through hyperlinks. A descriptive traceability process model or process trace (Jarke et al., 1994) is established among the design diagrams through linkages. When the design problems are not mature enough for explicit design rationale, software engineers can built annotations instead. This helps in avoiding premature segmentation of knowledge.

In the MetaEdit+ environment the process models and meta-models help to specify the occasions and events when method and design rationale is to be captured, e.g. phases, steps, decisions, milestones or reviews (Marttiin, 1994). Overall, the captured method rationale can be applied to method evaluation and improvement, to raise the level of consciousness and communication among the stakeholders, and to provide a help or learning system. Method rationale may play an especially important role in very large projects or in method engineering which takes place over time.

4 DISCUSSION AND CONCLUSIONS

This paper has described tools and principles for collecting and sharing experience and other information on the applicability of methods used. The proposed solution consists of capturing the method rationale in a CAME environment. Method rationale means method design and usage rationales and their linkages to design artifacts across various phases of method engineering and use. The tools described in this paper already exist in the MetaEdit+ environment, and even if they have been used so far to capture only software design rationale, we believe that they can be utilized in a similar manner to capture and share knowledge about methods.

The computerized method rationale capture takes place as an active and integral part of the ISD and ME processes, lessening the need for e.g. after-project interviews or other manual tasks. In this manner method rationale and its support tools help to achieve an advanced CAME environment. Capturing method rationale also provides a means for analysing and comparing different methods through their existing or non-existing features, e.g. requirements and assumptions. The original design rationale concept also becomes especially interesting when it is enlargened to method, process, project and business knowledge, supporting the creation and use of organizational memory.

Method rationale embeds a new conceptual structure and description language to a CAME environment, and it can be utilized on any level of abstraction or in any phase of the ISD or ME activities. In general, model annotation and linking tools in computer-aided design environments may enhance both target system quality and the quality of the process through which they are developed. One of the most important steps in future research is the development of principles for utilizing method rationale for method refinements, e.g. defining the connection between software process models and method rationale capture.

5 ACKOWLEDGEMENTS

I want to express my thanks to the other members of the MetaPHOR project, in particular Juha-Pekka Tolvanen, Steven Kelly, and Pentti Marttiin for our discussions on method engineering.

6 REFERENCES

Brinkkemper, S. (1993) Integrating Diagrams in CASE Tools Through Modelling Transparency. *Information and Software Technology*, **35**, 2, 101-105.

Conklin, J. and Begeman, M.L. (1988) gIBIS: A Hypertext Tool for Exploratory Policy Discussion. *ACM Transactions on Office Information Systems*, **6**, 4, 303-331.

Conklin, E.J. and Yakemovic, KC Burgess (1991) A Process-Oriented Approach to Design Rationale. *Human-Computer Interaction*, **6**, 3&4, 357-319.

Fischer, G., Lemke, A.G., McCall, R. and Morch, A.I. (1991) Making Argumentation Serve Design. *Human-Computer Interaction*, **6**, 3&4, 393-419.

Heym, M. and Österle, H. (1993) Computer-Aided Methodology Engineering. *Information & Software Technology*, **35**, 6&7, 345-354.

Jarke, M., Pohl, K., Rolland, C. and Schmitt, J.-R. (1994) Experience-Based Method Evaluation and Improvement: A Process Modeling Approach, in *Methods and Associated Tools for the Information Systems Life Cycle* (eds. A.A. Verrijn-Stuart and T.W. Olle), IFIP Transactions A-55, North-Holland, Amsterdam, 1-27.

Kelly, S., Lyytinen, K., and Rossi, M. (1996) MetaEdit+: A Fully Configurable Multiuser and Multitool CASE Environment, in *Proceedings of the Eigth International Conference on Advanced Information Systems Engineering (CAiSE '96)*, Crete, Greece, May 1996.

Marttiin, P. (1994) Towards Flexible Process Support with a CASE Shell, in *Advanced Information Systems Engineering* (eds. G. Wijers, S. Brinkkemper and T. Wasserman), LGNS#811, Springer-Verlag, 1994, 14-27.

Oinas-Kukkonen, H. (1996) Debate Browser - An Argumentation Tool for MetaEdit+ Environment, in *Proceedings of the Seventh European Workshop on Next Generation of CASE Tools (NGCT '96)*, Crete, Greece, May 1996.

Ramesh, B. and Dhar, V. (1992) Supporting Systems Development by Capturing Deliberations During Requirements Engineering. *IEEE Transactions on Software Engineering*, **18**, 6, June, 498-510.