

Semistructured models are surprisingly useful for user-centered design

Thomas Herrmann, Marcel Hoffmann, Kai-Uwe Loser, Klaus Moysich

Informatics & Society, Dept. Computer Science, University of Dortmund, Dortmund, Germany
{herrmann, hoffmann, loser, moysich}@iug.cs.uni-dortmund.de

Abstract. Diagrammatic representations are commonly accepted as valuable tools in requirements engineering and systems design. However, the most prominent techniques, are not sufficient for requirements negotiation with users because they focus on the design of technical systems. In user-centered design of socio-technical systems there is a strong demand for models which integrate different viewpoints. We believe that appropriate semi-formal diagramming techniques can facilitate the negotiation of the design, especially when they are combined with additional representations. Therefore we have designed a notation that supports the generation of integrated models of organizational, social, and technical structures, e.g. business processes, social relations and dependencies among protagonists, resources, work-objects, and software functionality. SeeMe, the diagramming-technique for modeling semistructured socio-technical systems moreover provides special concepts for the representation of vagueness, incompleteness, and contradictions that are inherent to user requirements. In this paper we present a first evaluation of the SeeMe-diagramming technique. The results are drawn from four different case-studies. We briefly introduce the main features of the SeeMe-Diagramming technique and subsequently present the result of our evaluation according to four aspects.

1. Diagram-based user-centered design

If user's requirements are expressed incompletely or vaguely, requirements analysts and software designers are often tempted to complement available information with their imagination. Unfortunately the designer's imagination does not always meet the customer's actual expectations. Semistructured phenomena are a well known problem in groupware design and usage. In his remarkable paper "Semistructured email are surprisingly useful for computer-supported coordination" Malone et al. suggested optional and partial structuring to use email more efficiently [13]. Similarly, in conceptual modelling neither completely structured and strong typed information nor completely unstructured information complies with the social and organizational requirements. In contrast to aiming at unambiguous and complete descriptions (e.g. [16]), we suggest emphasizing vagueness in user-centered systems design. To make vagueness explicit we propose special diagramming concepts for vague modeling. When we introduced our diagramming technique SeeMe in different case studies our partners were unfamiliar with the idea of vague modeling. However, we were surprised by how easily and naturally our partners adopted the concepts.

Diagrammatic representations are commonly accepted as valuable tools in requirements engineering and systems design. Especially in designing cooperative processes and groupware, diagrams help to overcome the limits of narrative descriptions and of demonstrations of a single user's interaction with a prototype. The most prominent techniques, as, for example, ER-diagrams, data-flow-diagrams, and the notations that are provided by the UML, focus on the design of technical systems. However, models of the technical system are not sufficient for the negotiation of requirements with users when the design aims at the reconciliation of technical, organizational, and social requirements. In user-centered design of socio-technical systems there is a strong demand for models which integrate different viewpoints. We believe that appropriate semi-formal diagramming techniques can

facilitate the negotiation of the design, especially when they are combined with additional representations. We have designed a notation that supports the creation of integrated models of organizational and social structures, including business processes, social relations and dependencies among the protagonists, resources, work-objects, and software functionality. *SeeMe* – the *diagramming-notation for modeling semistructured socio-technical systems* – moreover provides special concepts for the representation of vagueness, incompleteness, and contradictions that are inherent to user requirements.

In this paper we present a first evaluation of the SeeMe-diagramming technique. The results are drawn from four different case studies. Firstly, the design of a shared Know-How-Repository that supports the generation and customisation of training offers, training elements and the selection and sequencing and training elements into training timetables. Secondly, the planning of a groupware application in another training company. Thirdly, the negotiation of an agreement on privacy protection between the works council and the management in an international express company that introduces a Workflow-Management-System. Finally, we analysed the design process of a software system for an administration department in a governmental financial institution. In all of these projects we used SeeMe-diagrams as a tool to document requirements and as a medium for discussion and collaborative activity with users and stakeholders in workshops. In order to learn about the particular affordances and limits of different representations in some of the projects we generated redundant descriptions and combined SeeMe-diagrams with prototypes and additional representations. The evaluation relies on reviews of the participant's feedback and particularly the questions and the requirements that were generated during the workshops. We preferred to evaluate our approach in case studies, since we wanted to know how SeeMe performs on realistic problems and how different users comprehend and use our methodology.

Known evaluations that address the comprehensibility of diagrams either focus on graph layout [16], [11] or on the comparison of different representations [10]. Moody recommends usage conventions and some extensions for ER-diagrams in order to improve their comprehensibility [14]. However, there is no evaluation that focuses on vague modeling in organizational and social context.

In the following section we will briefly introduce the main features of the SeeMe-Diagramming technique. Subsequently, we present the result of our evaluation according to four aspects: explicating and integrating organizational, social and technical design, complexity and comprehensibility, handling the complexity of combined vagueness, and finally strategies against strong sequencing.

2. Introduction to SeeMe

We analysed a set of common modeling methods for their appropriateness of modeling socio-technical systems ([1], [2], [4], [7], [15], [18], [19], and [20]). We found all of them to have deficiencies in describing this kind of system especially for the mentioned modeling domains. We also analysed the underlying concepts of approaches which deal with social aspects in the context of information technology, such as activity theory [9], human-computer interaction theory [3], coordination theory [12] and the studies of [8] on informal communication as well as the Winograd-Suchman dispute (CSCW-Journal 2/1994). The result of these analyses led us to requirements and design rationale for a modeling method for socio-technical systems, consisting of semistructured and social aspects. The major deficits of the analysed methods we found are:

- ?? the possibilities to make vagueness (e.g. incompleteness) explicit are limited. By contrast, completeness is sometimes enforced
- ?? possibilities to present different kinds of attributes are restricted
- ?? altering the perspectives or points of view mostly requires different notation systems
- ?? meta-aspects are not systematically taken into consideration
- ?? social interests of roles or role playing cannot be represented

?? possibilities for free and arbitrary decisions are not sufficiently taken into account
 SeeMe – *the diagramming-notation for modeling semistructured socio-technical systems* – which we evaluated in the mentioned cases, is designed to overcome these limitations.

To understand the examples presented in this paper we have to give a brief introduction in SeeMe. For this purpose we use simplified examples from the context of the workflow management case study.

Basic Elements: SeeMe is based on the basic concepts of role, activity, entity and relation. These concepts are very common in many modeling methods. Roles describe a set of rights and responsibilities assigned to a person, a group or an organizational unit. The living parts of a socio-technical system whose choices for action are not definite, but controlled by expectations, are depicted as roles. When persons playing a role are in action, they are performing activities. Activities use entities as resources (documents, tools, computing systems etc.) or they manipulate entities. Fig. 2.1 gives a simple example with the following basic elements: the role *employee*, the activity *evaluating* and two entities *WMS database* and *analysis report*.

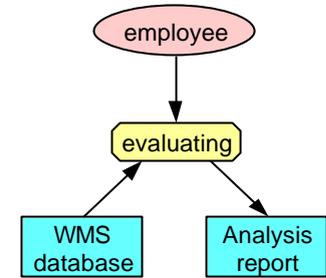


Figure 2.1

All elements in a model are at least specified by a name. More precise descriptions are possible by adding attributes or by giving detailed specifications by means of sub-elements. In fig. 2.2, for example, the activity *evaluating* is specified with the alternative sub-activities *free query* and *predefined query*.

Relations: Relations connecting basic elements are depicted as directed arcs. Relations visualize possible logical connections as a result of a relation between two elements. The establishment of a relation can be called an instantiation of a relation. The instantiation can also be understood as an event. Correspondingly, modeling relations is modeling the predicted events of the modelled system, similar to the transitions in state-transition-diagrams. To annotate conditions and probability or uncertainty of an instantiation modifiers (s. below) are used. Logical connections between relations can be expressed with connectors (s. below). In SeeMe the syntax is not limiting usage of relations: all mutual combinations are possible and have predefined meaning. Table 2.1 summarizes the predefined meanings assigned to relations that are connecting two instances of basic elements. In the example in Fig. 2.1 the role *employee* is performing the activity *evaluating*. The *employee evaluates* the *WMS database* to create an *analysis report*.

In addition to the predefined meaning it is possible to assign a new meaning by specifying names and types to new relations. Other types are also supported by shortcuts like instantiation or termination (e.g. of entities) which cannot be described in detail here.

An important special type of relation is the meta-relation depicted with a zigzag. The meaning is that one element is defining the other elements. An example is a control committee deciding what types of analysis are allowed on a given WMS database in order to assure privacy.

Finally, relations can be specified by other elements. For example, specifying a relation with an activity expresses how a relation gets instantiated. Transmitting the email connects the inbox of a recipient with a message sent by a sender, for example. All other basic elements can be used as well to specify a relation.

Connectors:
 Connectors are used to logically combine relations.

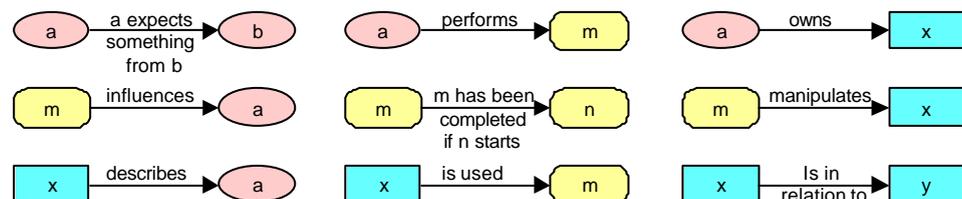


Table 2.1: Default meanings of relations between basic elements

The diamond, the graphical notation for the connector, can be filled with a “^” (or) or a “V” (and) or an X (exclusively or). These symbols represent the usual logical combination of the relations. In fig. 3 an *employee* performs the activity *evaluating*. This can be done by either a *free query* or a *predefined query*. The X filling the connector represents an “exclusively or” combination of the participating relations.

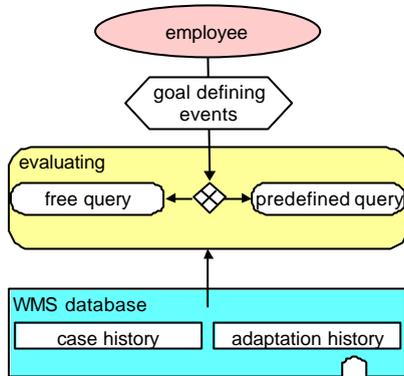


Figure 2.2

Modifiers: The instantiation of relations and basic-elements is often bound to events or conditions. Modifiers are used to describe these events and conditions. Similarly to [19], the notational symbol is the stretched hexagon which contains the condition. Modifiers are especially relevant for sequences of activities, where conditions are regularly modelled as events which necessarily have to happen before an activity can be performed. In fig. 2.2 the evaluation is only permitted if an event happens which is related to the purpose the data was stored for. In contrast to other methods [19], events are only specified if they have relevance for decisions at branches or the existence of model-elements in the proposed notation.

Conditions for relations that link basic-elements other than activities are handled similarly. Modifiers can contain several pieces of additional information which modify the instantiation of elements. The information can specify an event, a condition and/or a probability. A modifier can also be used to define the conditions of the existence of basic-elements. The modifier then relates to the instantiation of an element instead of the instantiation of a relation.

Nesting and dynamic presentation: Various types of relations between sub- and super-elements which build hierarchies are often presented using nesting of structures (e.g. [7], or [17]). An example has already been mentioned: in fig. 2.2 the activity *evaluating* is specified with the alternative sub-activities *free query* and *predefined query*. SeeMe supports multiple semantics of this depiction of elements: aggregation (“part of”), inclusion in sets (“is included in”), or specialization (“is a”) is possible to express using this technique. Using embedding in this informal notion, simply understood as unspecified hierarchical relation, is helpful in the early stages where a formal specification of the type of relation is not easy to determine and often neither useful nor necessary.

Nesting also builds a foundation for dynamic presentation of models. A modeling tool can support hiding and showing of details embedded in elements. This method helps to present large models that cannot be understood at once. A software tool can support exploration of models as well as preparation for presentations of the same. We used this in various ways in the described cases.

Explicit vagueness: Based on this notation we proposed extensions to express vagueness and uncertainty in models. A more detailed description of the concepts which support modellers to make vagueness explicit can be found in [5]. SeeMe supports basically two concepts to describe systems vaguely:

- a) Intended omission of information a modeller has, but is not willing to present in that diagram.
- b) Expressing vagueness or doubting completeness/appropriateness of contained information.

Expressing vagueness can be used in combination with all elements: models, elements, relations, modifiers, attributes. The two basic concepts also aggregate various sub-cases. In this paper we focus on basic elements and endings of relations. The subtypes for these elements are described in the following paragraphs.

Intended Omission: One of the most crucial decisions in modeling concerns with the decision of what is sensible and what is not, to the purpose of a model. To achieve comprehensibility, clarity and

readability of a model, the hiding of information is necessary. To allow the modeller to hide specific information, SeeMe uses three extensions of the basic notation:

<p>a1. Referencing knowledge of the modeller allows the modeller to express that she/he has more knowledge on a modelled aspect that she/he can present on request. The Symbol " " is used to express this.</p>
<p>a2. References to parts of the model that are not visible in the current diagram are shown with filled black areas (""). A software tool can then support zooming-in after clicking on such a symbol.</p>
<p>a3. The decision of the kind „more specific information is not of interest to this diagram“ is depicted with empty areas ("").</p>

Expressing vague information: A modeller can express knowledge about the completeness and appropriateness of modeled information. He can express that parts of a model are incomplete or that he or she is uncertain about the modelled information. This kind of knowledge is found during investigations, for example, because of inconsistencies in the information or because of questions left open. Again we differentiate three cases:

<p>b1. A modeller realizes that a specification is incomplete and that he or she is not able to complete the diagram. She/he uses three dots „...“ to express this finding.</p>
<p>b2. A modeller is uncertain about whether a specification is correct. These doubts can be expressed by annotating a question mark „?“.</p>
<p>b3. In many cases, even the decision of whether a specification is complete or not, cannot be made. This case – doubting the completeness – is expressed with three question marks „???“.</p>

The notational elements for vagueness can be combined. One example could be “?;+” expressing that the correctness of the modelled information is questioned, and, furthermore, the modeller shows that he could provide more information to the recipient.

For development projects modellers also sometimes need the possibility to express the completeness and correctness of a (part of) a diagram. For this purpose a tick expresses that the modeler takes responsibility for this part of the model.

Vagueness with relations: Relations can be connected to an element as a whole or to its parts, such as sub-elements. In SeeMe the definite specification is not necessary. Relations, at both ends, are not necessarily connected with one specific element. Fig. 2.2 gives an example where the *uses* relation connecting the activity *evaluating* and the entity *WMS database* is unspecified at the end of the *WMS database*. The modeller expresses that not always is the whole WMS database evaluated but parts of it that he/she cannot specify in this diagram.

The unspecified connection of a relation is especially helpful to model processes vaguely. Usually the semantics of process models is that one step is completed and then the next begins. In socio-technical processes where activities can also be ongoing processes, the start of a following activity can be at any time (vague information) while the predecessor is active (see fig 3.4.5). It is also helpful to reduce complexity, for example, when the complete expression of all connections between two elements with many sub-elements is too complex to show in one diagram, the connections can be reduced to simply one deflected relation or to meaningful subsets of the whole set of relations.

3. Experiences with SeeMe

3.1 Explicating and Integrating organizational, social and technical design – the case of designing an embedded Know-How-System

In a project with a training organization we used SeeMe-Diagrams to suggest usage scenarios that integrated the design of a Know-How-Repository and the development of central business processes in the organization. Furthermore, we used SeeMe to capture and to compare results from some ethnographic studies of working processes. The project resulted in a requirements definition that was passed to software developers and serves as part of the contract between the training company and the software developer in the current implementation of the software.

The application of the diagrams revealed benefits and limits of diagramming techniques in user-centered system design in general and showed some affordances of the special properties of the SeeMe-Diagramming technique.

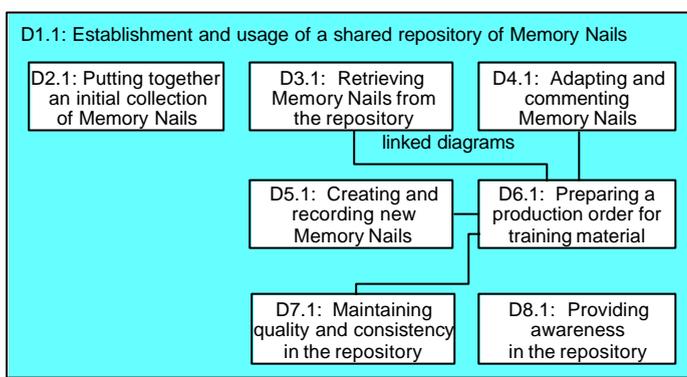


Figure 3.1.1: Metadiagram of the initial structure of SeeMe diagrams

Supporting Project-Management with diagrams:

In that particular project, the project management was challenged by different factors. Firstly, the design stage lasted relatively long. Furthermore, each prototype was implemented with a different technology, and in each workshop we welcomed new participants of our customer's organization. Keeping an information-rich project history became even more important when we integrated new members in our university team and when we started to search for software developers

who were to realize the technical system. The SeeMe diagrams provided a means to structure workshops and reflection of design. To show how the collection of diagrams evolved during the project and followed the design requirements we compared the initial diagram structure with the structure that was finally submitted to the software developers.

During requirements definition, five workshops were carried out. The initial workshop produced a general goal definition, fixed some benchmarks for later evaluation of the project, selected a subsection of training knowledge to start with, and collected usage scenarios that reflected the participants' expectations. On the basis of these usage scenarios we suggested seven activities for the establishment and usage of a shared repository (fig. 3.1.1) and in the following workshop presented some screenshots of the repository's first prototype.

Each activity was described by one SeeMe-diagram (D2-D8) that showed relations between sub-activities, resources and roles that participated in that activity. The relations suggested a division of labour among the participating roles, sequencing of sub-activities and access to resources.

The final diagram structure (fig. 3.1.2) included five sub-activities from the first version which had, of course, been revised at least once (D3, D5, D6, D7, and D8) and additionally two sub-activities (D9 and D11) that had been introduced in the meantime. To show how design requirements changed the diagram structure we will look at a few examples in more detail:

1. The design for the *retrieval of training elements* (D3) was revised according to a general shift of the project's focus from storing individual training elements to supporting the entire process of the preparation of training material. This shift was a result of a workshop presentation of a revised prototype. During the discussion it turned out that there was a strong demand to embed the storage and the retrieval of training elements in the trainer's everyday work [6]. In the next

workshop we reintroduced the idea of *preparing production orders* (D6) that had been dismissed before, and we discussed a new diagram that showed how the generation of the production order can be combined with the creation of a training folder and integrated in a *business process* (D12, cf. fig. 3.1.3). This time the design was agreed to be a useful solution.

2. Following the users requirements the activity that was used to describe the *creation of Memory Nails* (D5.1) was extended to all kinds of training elements (in version no. 3) and was divided into two sub-activities (in version no. 4). One sub-activity describes the *creation of a new Training Element* (D5.4a) and the other one described the *adaptation of an existing element* (D5.4b).
3. In order to integrate awareness support with genuine work tasks some sub-activities and their supporting functionality were moved from *Providing Awareness in the Repository* (D8.1) into the *creation and adaptation of training material* (D10).
4. Since *recording experience and comments after a training* (D11) turned out to play a major role in knowledge transfer a special sub-diagram was introduced early in the project.
5. The stored diagrams supported negotiation, too. One example was the organization of *quality management* (D7). The chief conceptionist feared that free access to training material would result in less quality, and therefore promoted a quality check by the conception department. On the other hand, the trainers asked for unlimited user rights to alter training elements and to store new material. During the project the design moved from free access to limited access to free access again. When the free access concept was questioned again during the last workshop, we reviewed past versions of the diagram and finally agreed to keep the design.

From our point of view, the diagrams proved rather useful to project management. SeeMe diagrams were the only form of design representation that was constantly used in every workshop. They provided a guiding line throughout the entire project. The overview diagram (D1) structured the discussions. Clustering and linking diagrams as supported by the SeeMe-diagramming technique proved to be especially useful.

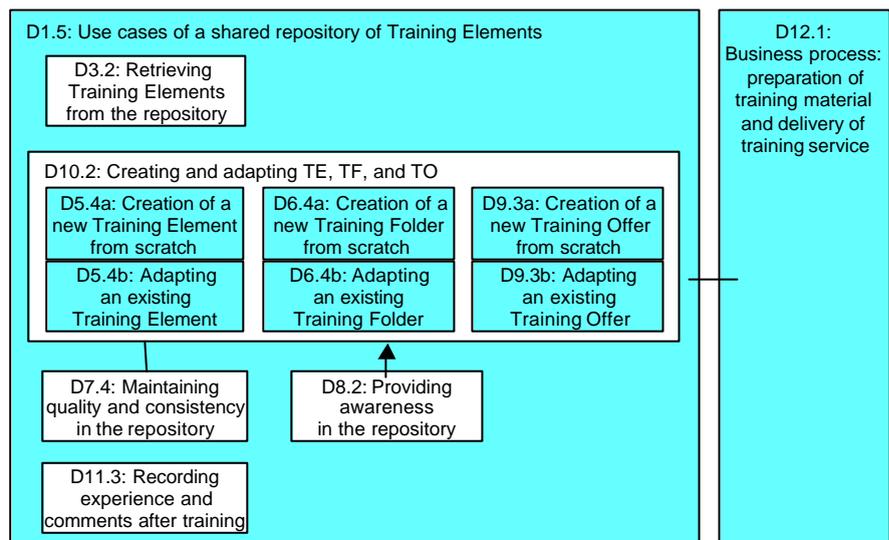


Figure 3.1.2: Metadiagram of the final structure of SeeMe diagrams

Integration of diagramming techniques and additional media facilitates understanding and provokes discussions on organizational and technical issues: Scenarios, prototypes, storyboards and the like are means of supporting communication between designers and users. An appropriate representation conveys understanding which is necessary for questioning, correcting, extending, and adapting the design. Apparently, each representation seems to provoke feedback and requirements on different aspects of the design according to the conveyed information.

We tried to emphasize the priority of organizational design throughout the entire project. For instance, we always discussed the division of labour and the sequencing of tasks before we showed how interaction might look by presenting the prototype. Furthermore, some workshops were devoted to negotiating and to improving the organizational design at first hand. Most of the adaptations and corrections of the SeeMe diagrams we gathered in these *organizational design workshops*. The re

maining workshops were intended to generate software requirements based on an established organizational design.

During the first organizational design workshop we separated the presentation of the SeeMe-diagrams from the presentation of screenshots. On that occasion we mainly gathered requirements on the sequencing of the task in the diagrams, on division of labour, and on the other hand, on screen design, and on the data that should be recorded (table 3.1.1). Of course, we could not expect much feedback on the functionality and the dialogue structure, because we did not show any interaction with the prototype, only screenshots. However, we were disappointed that there was little interaction between requirements of different classes.

Organizational requirements concerning	Technical requirements concerning
1. Exclusion of certain tasks of "preparing production orders" from the design .	1. Recording material that belongs to the Memory Nail so that it can be prepared by the assistance.
2. Integration of additional resources (e.g. sample Memory Nails to facilitate the collection).	2. Providing awareness on new Memory Nails only, but not on new comments and adaptations.
3. Integration of additional activities (e.g. quality check).	3. Providing a selection of given keywords to index a new Memory Nail
4. Adaptation of the division of labour (e.g. recording of Memory Nails must be supported by the trainer's assistance, quality check by the chief conceptionist).	4. Including Audiofiles of spoken language to show how a Memory Nail's "story has to be told"
...	...

Table 3.1.1 Examples of requirements negotiated during the first organizational design workshop (separate presentation of diagrams and screenshots)

In the next organizational design workshops we combined the prototype with the SeeMe-Diagrams by linking the demonstrations and the diagrams. The links allowed us to switch more flexibly from the presentation of the diagrams to the demonstration of the prototype. It showed that this approach generated more requirements on additional functionality. Furthermore, most of the requirements were gathered during the discussion of the demonstrator referred to activities that we presented in the process diagrams. Table 3.1.2 lists some of the requirements we gathered during the second organizational design workshop.

1. Include functionality to browse a training folder from page to page according to the chronological order of the training!	3. I need a hardcopy of the table of contents of a training folder in order to evaluate the sequencing of elements!
2. How can a training element be retrieved when I do not know anything about the element, but that it belonged to a certain training folder?	4. When I generate a new training folder I usually rely on existing folders. How can I adapt existing training folders?
...	...

Table 3.1.2 Examples of requirements negotiated during the second organizational design workshop (combined presentation of diagrams and prototype)

SeeMe diagrams provoked many requirements on the division of labour between trainers and their assistance, on embedding the repository into business process, and on supporting additional activities, for instance. Screenshots focussed on the data that was stored in the repository. Demonstrations of interaction produced requirements on the presented dialogue structures.

Diagrams provide an additional representational level for the reflection of design: In between the workshops and the final documentation of the requirements we compared different presentations of the usage processes. This comparison revealed some inconsistencies between the written scenarios and the SeeMe scenario diagrams. For instance, the diagrams were more explicit and unambiguous than the process description in written language. On the other hand, the written scenarios omitted some conditional events and activities and did not make incompleteness explicit.

When we reviewed the requirements that were explicated in the diagrams and the written specifications, it turned out that we had not fully understood and had even misunderstood some of our customer's requirements. In the diagrams we had closed the gaps through rational inference. Unfortunately, our conclusions did not always reflect the user's ideas of the requirement. The written descriptions tended to be less explicit and more ambiguous. In that respect they were closer to the customer's requirements. On the other hand, they did not completely record requirements even when

they had been negotiated in more detail. When they stated different alternatives of the course of actions, for instance, they did not conclude every topic. Furthermore, the written descriptions did not make clear whether additional cases were possible or not.

As a result of that comparison we corrected both representations. For instance, some over-specifications of SeeMe diagrams were withdrawn by introducing vagueness symbols. The written descriptions were completed by additional remarks on further events and activities. Fig. 3.1.3 shows the revised version of a SeeMe diagram where the highlighted corrections resulted from the comparison with a written requirements description.

Sometimes evaluating design representations with users and customers is not sufficient for correction. Users are reluctant to review a specification systematically. Of course they provide valuable and irreplaceable advice but that does not ensure consistency and correctness. We had good experience using redundant representations to evaluate the design documents, especially to show misunderstandings, incompleteness, and over-specification.

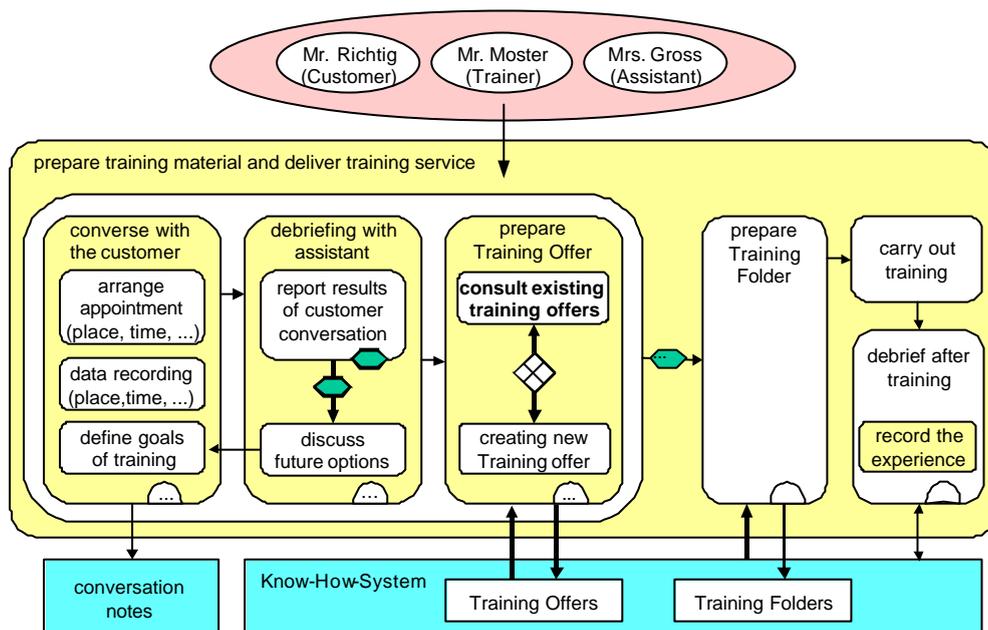


Figure 3.1.3:

3.2 Complexity and Comprehensibility

Requirements analysis and design for groupware applications: We intend to introduce the SeeMe diagramming technique as an instrument supporting the participatory design of groupware applications in organizations.

The research project *LOOK* develops and evaluates a training method and corresponding material to teach employees in understanding and using SeeMe for this purpose. One important goal of *LOOK* is the propagation of this training method in the field of further qualification of employees. Therefore, we chose a training company with extensive experience in teaching employees and consulting managers as one project partner. Using SeeMe, we developed a groupware system in cooperation with the employees of the training company. At the time of the beginning of the project the company had only limited knowledge of groupware. The PCs were connected to a small company network to share resources such as printers. Important collections of data were held paper-based. A few staff members had access to the internet and were able to use World Wide Web and email at their workplace. To overcome these deficits in knowledge of and experience with groupware we carried out a workshop to introduce groupware concepts and possibilities to the employees as a first step.

Using a scenario from the everyday tasks of the workshop participants we demonstrated the application of various groupware systems. The scenario dealt with the development of a new course, including tasks such as preparing a proposal, discussing details of the lessons, recruiting trainers, arranging training locations and fixing the concept in a paper. We presented this scenario in a SeeMe diagram showing an overall view of the sequence of the major tasks (fig. 3.2.1). To prevent high complexity, the details of the tasks were hidden in the diagram. We pointed at this fact to the workshop participants and explained the "mouse-holes" in the diagram's elements as an indicator of this incompleteness. As a further means of reducing complexity we presented the diagram not as a whole in one step. Instead, we developed it step by step showing only those additional elements that were necessary in the process of explanation. For this purpose we used Microsoft PowerPoint as the presentation tool.

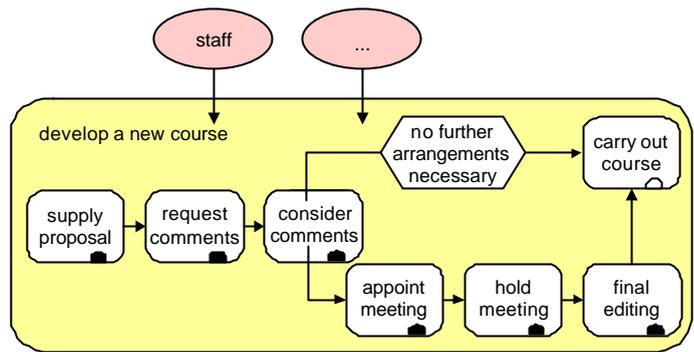


Figure 3.2.1: overall view diagram of the workshop scenario

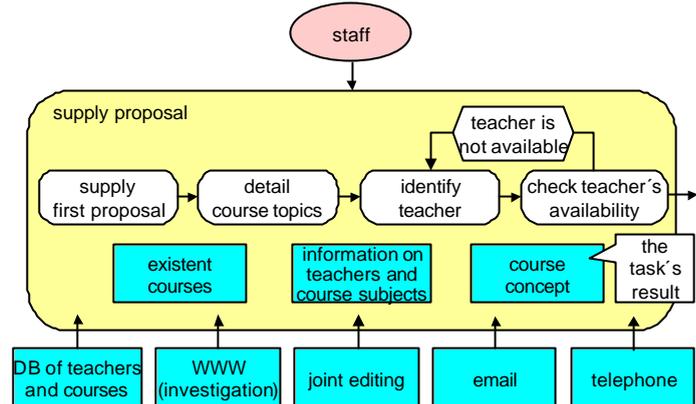


Figure 3.2.2: detailed view diagram of a task

Explaining the individual tasks, we zoomed the presented diagram into a more detailed view with particular information like sub-tasks and involved groupware systems (fig. 3.2.2). With this technique of zooming-in and out elements and information, SeeMe affords a flexible reduction of complexity. The workshop participants easily understood the relation between the overview and diagrams with a detailed view. As those present had no experience in SeeMe we explained the meaning of the used diagram elements while presenting the scenario. The discussion of the presented scenario proved that the workshop participants were able to understand the example scenario even though they had no preceding introduction to the SeeMe diagramming technique. Furthermore, they identified differences and simplifications with regard to their tasks in the diagrams of the presented scenario (see 3.2.2). Subsequently, we demonstrated the usage of groupware systems for carrying out the tasks of the example scenario in short performances. In the discussion of the demonstrations the participants were able to reference the performed tasks to the previously presented SeeMe diagrams.

After the groupware workshop the following task in the project was the participatory design of a groupware system. This system was intended to improve the flow of information and to enhance the visibility of available know-how within the training company. Additionally, an *idea-forum* was proposed to enable the joint development of new services of the company. At first we evaluated the work practice of the training company's employees and drew up corresponding SeeMe diagrams. We discussed these diagrams with the involved employees to match the current situation. Sessions and workshops were carried out to specify the requirements for the proposed groupware system and to discuss the necessary changes in the company's organizational structure. We described the findings of each session with SeeMe diagrams. Sometimes we supplied the diagrams by screenshot-prototypes of possible realizations of parts of the planned system. In the following session we presented them to the employees. Based on the diagrams and the screenshot-prototypes the employees refined the requirements and debated possible alternatives of the system's features and organizational changes. We observed that the combination of SeeMe diagrams and screenshot-prototypes

was stimulating the participants' debate. In a number of cycles of this process of evaluating, drawing diagrams, creating prototypes, presenting and discussing with the employees and managers, a groupware system was designed that was accepted by all stakeholders. Later on, this system will be implemented by the project LOOK.

Conclusions about the usage of SeeMe: The participants of the seminars and workshops showed a positive attitude towards SeeMe diagrams. They were already used to different types of diagramming techniques by the company's quality management manuals. Compared to those diagrams the employees stated that SeeMe diagrams were a better means for communication. The diagrams in the QM-manuals were criticized for being partly incomplete and too complex.

A permanently visible poster with the overall-view diagram was regarded as helpful for comprehending diagrams when zooming-in and -out details or switching to different views. We used posters with an overall view as a supplement to the agenda of a session to visualize the tasks to be done.

Analysing the participants' reactions in the process of presentations and further development of SeeMe diagrams, we can conclude that the diagrams were comprehended and that they stimulated the discussions on the matters of interest.

1. Participation and Attentiveness: About two thirds of those present participated in the discussion of the diagrams. The attentiveness decreased drastically when sessions dealt with more than about five abstract diagrams in a single sequence. This negative effect could be avoided by showing other types of views, such as screenshots.
2. Comprehensibility: The comprehensibility can be deduced from the participants' contributions of the following kind:
3. Participants remarked that certain elements were missing from their point of view. They proposed the addition of new roles, entities and relations. In particular, they made use of the possibilities to represent aspects of cooperation. For example, participants asked for the addition of a relation between a role and a connector in a diagram that depicted distribution of tasks among different roles. The new relation represented the involvement of an additional person in the activity concerned.
4. The participants asked for corrections of various diagrams. By a number of proposed corrections the comprehension of the concept of specified vs. unspecified combination of a relation with an element became obvious: For example, participants requested some executes-relations to be related to the super-activity instead of connecting it with a specific sub-activity. This should be done with an unspecified relation to express the fact that it is unclear which sub-activity is executed by the role. Other corrections referred to names of roles and entities and to the arrangement of hierarchy expressed by the nesting of elements.
5. The SeeMe concept of purposeful incompleteness was comprehended as well. Only in the first sessions did participants remark in some cases that a diagram concerning their field of work was incomplete: "There are some tasks missing in this diagram which I carry out as well!". In those situations we explained that we are aware of this kind of incompleteness and that SeeMe offers special symbols of incompleteness like semi-circles, three dots or blank modifiers. We explained them by pointing to the examples in the diagram.
6. Finally, the combination of SeeMe diagrams and screenshot-prototypes seemed to provide a helpful link to the employees' everyday tasks which the prototypes are related to. We got the impression that it is easier for an employee to realize the consequences of his/her own tasks by looking at a prototype than by the diagrams representing organizational aspects. On the other hand we observed that participants always related their contributions to organizational structures to SeeMe diagrams and never to screenshot-prototypes (cf. section 3.1)

Most of the SeeMe diagrams and screenshot-prototypes presented in workshops and seminars were rebuilt as interactive multimedia applications with additional guided tours. We offer these applications for repeated online practice to the employees of the company via World Wide Web

(<http://iundg.informatik.uni-dortmund.de/look/>). The evaluation of the usage of this material and the users' experience with it is not yet finished.

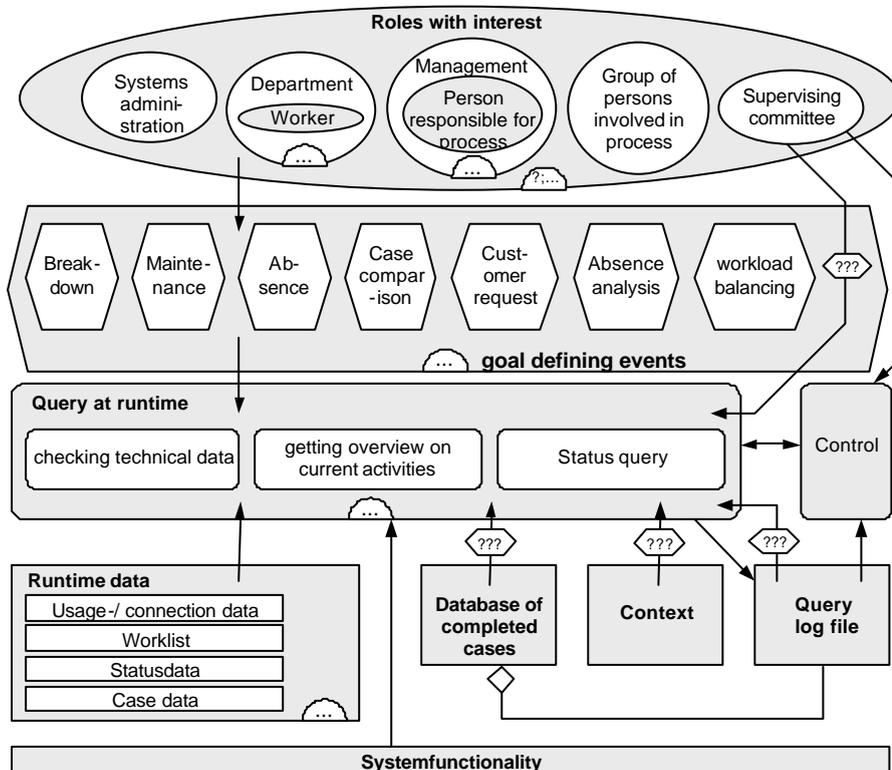
3.3 Handling the complexity of combined vagueness – the case of privacy regulations for workflow management

The project MOVE aims at continuous improvement of business processes with flexible workflow management systems. Workflow management systems offer the possibility of detailed records about how tasks are carried out. On the one hand, this recording is especially important to make the users' adaptation of the system comprehensible. These adaptations of workflows should be possible to achieve flexibility. The availability of data about who is currently working on which case is also useful to support cooperation and coordination. On the other hand, the possibilities for recording give reason for several privacy concerns. Therefore, we started to work on guidelines for legal and organizational regulations regarding privacy aspects of these systems. For this purpose various experts from the fields of science, management and shop stewards participated in a workshop to discuss necessary steps for the regulation of the use of data in this context.

During the workshop, two contrary standpoints became obvious: One part of the participants proposed the classic way of that only those activities are legal which are explicitly mentioned in a catalogue. The other group insisted that these kinds of catalogues are not possible for complex and dynamic systems like workflow management. The most urgent requirement resulting from the workshop was that we had to give a systematic overview of all potential aspects which might create the need for regulations. We felt that SeeMe was an appropriate method to deal with this requirement because vagueness had an essential role: the known facts about the functions and use of workflow management are contrasted by the case-dependent parts which have to be specified for a concrete case.

Fig. 3.3.1 differentiates between four levels for the example of queries at runtime

1. Who is (which roles are) allowed to have access to the personal data?
2. Under which conditions (events which are in accordance with the purpose for which the data is



3. Which kinds of queries are allowed?
4. Which data can be accessed by the system to answer the queries?

In this case, three concepts of SeeMe were mainly employed to deal with the complexity of uncertainty. The non-specified relations were used to express that various combinations are possible. Fig. 3.3.1 expresses that every participating role can decide under which conditions which type of query is used and which

Figure 3.3.1.

data is evaluated. To overcome this lack of specification, it would be necessary to regulate for every role under which specific conditions he/she is allowed to use which queries for which type of data.

Furthermore, the semi-circles with three points were used to express that we do not know all sub-roles, sub-conditions, sub-queries etc. and that it is a question of regulation to determine which roles, conditions etc. are valid or not. The third strategy is to use the hexagons with three question marks. If this symbol is annotated to a relation it is indicated that we are not sure whether any condition under which this relation is sensible exists. Thus it has to be principally decided whether this relations are sensible or not, e.g. whether these data of completed cases can be accessed with runtime queries (to make comparisons between current and earlier cases) or whether context can be automatically employed (if correlating data is stored on other systems).

Fig. 3.3.1 was presented to the works council of a large express company where a workflow system had to be introduced. They were able to understand the diagram and discussed several of its details. They were quickly convinced that it was too complex a task to determine all potentially relevant regulations in one step by a catalogue. They accepted a procedure of filling the specification gaps step-by-step whenever it became necessary. On the other hand, the management agreed to build a committee whose task it is to handle this step-by-step specification. This procedure was regulated in a contract and one member of the works council became a member of the committee. This process could successfully be supported by making the complex combination of aspects of incompleteness visible. With the exception of SeeMe we do not know any modeling method which includes this possibility.

3.4. Strategies against strong sequencing – the case of administration of applications

In this case SeeMe was used to support a facilitator who mediated the cooperation between an IT department and a department for the administration of governmental funds. The aim of this cooperation was to develop a software system which helps to process the administration of governmental funds. Both departments were part of a governmental finance institution which has the main task to support innovation in one of the five new states in East-Germany.

The facilitator had to support the process of requirements engineering. She started by interviewing the administration department to get a catalogue of the main functions. The result, amongst others, was a hand-sketched diagram representing the main activities and events which occur during the handling of applications for funds. In a best case procedure, the main activities were: data entering, formal checking, checking for completeness, technical checking (including aspects of economy and professionalism), calculating and preparation of the approval documents. However, there were several possible events causing a deviation from the ideal procedure: if a check reveals insufficiencies, the application is preliminarily rejected or further documents or explications are required; in some cases an expert's advice has to be asked for. If the application is rejected, a hearing is initiated giving the applicant an opportunity to improve the application. If further documents are required and the applicant does not react, it depends on the administration department how long they will wait or how often they will remind the applicant. There are several roles taking part in this process: the applicant, a secretary person, a specialist, experts and an official in charge.

Since there are a number of possibilities which cause deviations of the ideal process, the hand-made diagram was very complex and difficult to comprehend. Therefore, the facilitator decided to try a more formal method, such as SeeMe, to gain an overview without losing the possibilities for vagueness and incompleteness. The

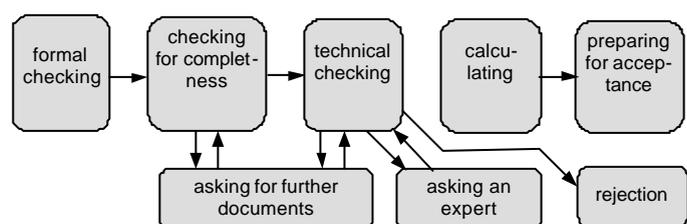


Figure 3.4.1

result was a diagram which is partly shown in fig. 3.4.1 (there is not enough space available to present the whole diagram). The facilitator expected the administration department to comment on how to assign roles and resources (entities) to the activities shown in the diagram. However, they did not focus on this question but used the diagram to discuss the problem of too strong sequencing of the activities. They claimed that the specialist is used to working more or less synchronously on the different tasks. The work can even go on while the advice of an expert or the submission of further documents are pending. Thus they might have preferred a solution as it is shown by fig. 3.4.2. which does not impose a control flow.

So far, the lesson of this case is that the procedure of an interview – which aligns the activities to a chain of events – does not provide a sequence which is in accordance with the reality of the work. The diagram was useful to reveal this problem.

Before the solution of fig. 3.4.2 could be introduced into the communication process, it was influenced by the IT-department which made an intervention with its own model. They used to model by employing state transition diagrams and summarized their point of view with a diagram as it is partly presented in fig. 3.4.3. This kind of diagram contains some dependencies which imply a kind of sequence. However, this sequence can be hardly recognized and is not in accordance with the activity orientation of the administration department. The diagram's structure does not clearly present the flow of activities. It also neglects an essential state: "further documents required". If they had introduced this state, it would sometimes have been valid together with the state *in process* – this constellation (two states at the same moment) is formally not sensible.

The problem was that the IT-department made state-orientated models while the administration department was focused on activities. The facilitator proposed a compromise by employing the nesting features of SeeMe. SeeMe allows a modeller to combine an event (as content of a modifier) with an activity by embedding it as proposed in fig. 3.4.4. A state of a process can be represented by an event. Arrows which point into a modifier indicate that a state can include sub-states which can potentially be represented by sub-modifiers. A modifier is surrounded by an activity which provides the decision on which the state transitions are based. State charts [7] offer similar concepts as they were required in this case such as nesting, concurrency and messaging. However, the decisive reason for SeeMe was the possibility of nesting different types of elements (e.g. a state represented by a modifier into an activity) and of representing incompleteness (e.g. unspecified sub-states).

Fig. 3.4.4 provides a more instructive impression of the underlying workflow. The 45 degree arrows pointing downwards lead to the "hearing"-activity, those pointing upwards indicate other kinds of repetition

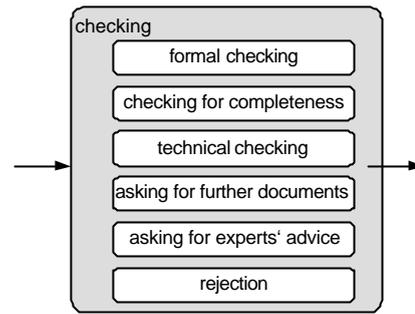


Figure 3.4.2

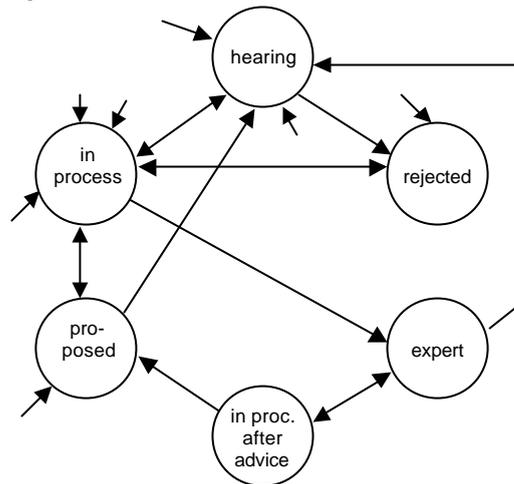


Fig. 3.4.3

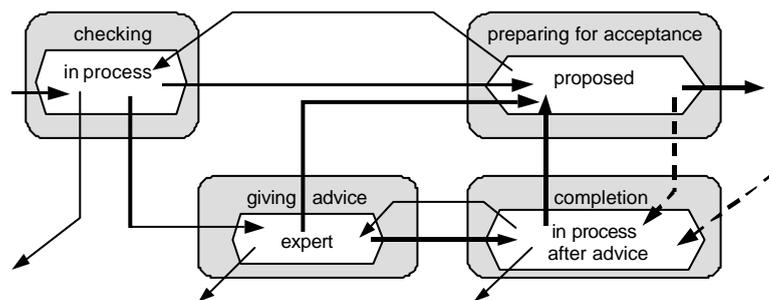


Figure 3.4.4

loops. During the process of modeling it became obvious that the state-transition diagram (see fig. 3.4.3) has left out a possible transition, and during the process of discussion with the administration department, the necessity of a further arrow became additionally obvious (see arcs with dashed lines). This discussion made clear that the diagram was understood by the departments. It is now part of the final document (fig. 3.4.4 gives a reduced, slightly altered version).

However, the discussions revealed that the problem of possible synchronous task performance is not completely solved. Fig. 3.4.5, part A represents a solution which might hypothetically be preferred by the specialists of the administration department: it proposes that it only depends on the free decision of the specialists whether and how they continue the activity of *checking* after the activity of *submitting further documents* or *giving advice* is started. However, the introduction of the state “in process after advice” indicates that the set of possible activities should be limited in this situation to the following activities: asking for further documents, rejection, preparation for acceptance. This version is given by fig. 3.4.5, part B. The diagram of this part determines that *checking* cannot be continued after *asking for documents*. This determination can be avoided by the construction of fig. 3.4.5, part C. The special connector symbol means that *asking for documents* can optionally be accompanied by another activity. This activity is not specified in the diagram. The arc which points away from *asking for documents* carries an unspecified modifier. This means that the specialists can decide by themselves whether they continue immediately or wait for the additional documents.

Conclusively, this case makes clear that a semistructured modeling method must provide possibilities to avoid strong sequencing and that SeeMe offers four sensible solutions:

- ??Sub-activities without control-flow relations (fig. 3.4.2)
- ??Control flow relations with unspecified anchor points (3.4.5A)
- ??Connector with optional ramification (3.4.5C)
- ??Ramification or continuation with non-specified conditions (3.4.5C).

4. Conclusion

At the moment, SeeMe is still an experimental modeling method. However, it has found *surprising* interest from certain consulting companies as well as scientists who have the problem of modelling socio-technical systems in fields such as knowledge management or internet services for public administrations. Most interest was provoked by the concepts of explicit incompleteness, unspecified relations and the hide and show mechanism which is connected to the black semi-circles. These three concepts represent the main advantages of SeeMe compared with other modeling methods. Although all concepts of SeeMe are integrated into one method, it is possible to select certain concepts and to export them to other modeling methods such as UML or ARIS.

The main problems becoming apparent are the lack of a guidance describing how SeeMe models can be developed systematically and efficiently. Furthermore a style-guide was required to support the aesthetic appearance of the diagrams. Correspondingly SeeMe diagrams do not sufficiently include hints how they should be read. We need also a catalogue of strategies which describe how the complexity of diagrams can be reduced. A crucial insight is that we have to find out which notation

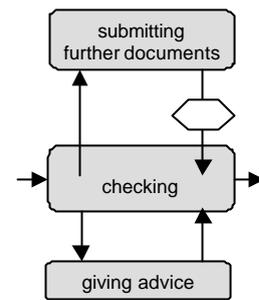


Figure 3.4.5a

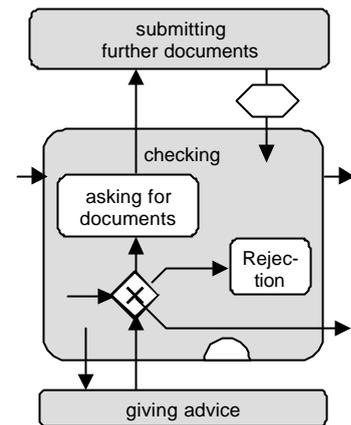


Figure 3.4.5b

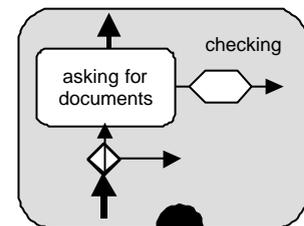


Figure 3.4.5c

elements of the method are appropriate for beginners and which elements should be exclusively used for experts. This kind of differentiation is especially relevant for the complex symbols of vagueness and incompleteness.

The hide and show mechanism is a crucial means to make SeeMe diagrams easier to comprehend and more feasible to communication processes. However, this mechanism cannot be sufficiently employed until an appropriate editor is available. It is a current task of our research on SeeMe to develop such an editor and to explore its possibilities for making complex models of cooperative tasks more comprehensible. Further research tasks are the development of a style guide for modeling with SeeMe and a training concept which enables people to understand and to alter SeeMe diagrams in the course of participative system development.

References

- [1] Beck, A.; Janssen, C.; Weisbecker, A.; Ziegler, J. (1995): Integrating object-oriented analysis and graphical user interface design. In: *Software Engineering*. Springer-Verlag, May 95, p. 15.
- [2] Dearden, A.M.; Harrison, M.D. (1997): Abstract models for HCI. In: *International Journal of Human-Computer Studies* **46**, pp. 151-177.
- [3] Gaines, B.R. (1988): A conceptual framework for person-computer interaction in complex systems. *IEEE Transactions on Systems, Man And Cybernetics* **18(4)**, pp. 532-541.
- [4] Green, T.R.G.; Benyon, D. R. (1996): The skull beneath the skin: entity-relationship models of information artifacts. In: *Int. J. Human-Computer Studies* **44**, pp. 801-829.
- [5] Herrmann, Th.; Loser, K.-U. (1999): Vagueness in models of socio-technical systems. *Behaviour & Information Technology* **18(5)**, pp. 313-323.
- [6] Hoffmann, M.; Loser, K.-U.; Walter, Th.; Herrmann, Th. (1999): A design process for embedding Knowledge Management in everyday work. *Proc. of Group'99*. (Phoenix, November 1999), to appear.
- [7] Harel, D. (1987): Statecharts: A Visual Formalism for Complex Systems. In: *Science of Computer Programming* **8**, pp. 231-274.
- [8] Kraut, R. E.; Fish, R. S.; Root, R. W.; Chalfonte, B. L. (1990): Informal Communication in Organizations: Form, Function, and Technology. In: *Baecker (1993): Readings in Groupware and computer-supported Cooperative Work*. Morgan Kaufman, pp. 145-199.
- [9] Kuuti, K. (1992): Identifying Potential CSCW Applications by Means of Activity Theory Concepts: A Case Example. *Proceedings of CSCW '92*, pp. 233-240.
- [10] Larsen, Tor J.; Naumann, Justus D. (1992): An experimental comparison of abstract and concrete representations in systems analysis. In: *Information & Management* **22**, pp. 29-40.
- [11] Lohse, Gerald L.; Min, Daihwan; Reitman Olson, Judith (1995): Cognitive evaluation of system representation diagrams. In: *Information & Management* **29**, pp. 79-94.
- [12] Malone, T. W. (1990): What is Coordination Theory and How Can it help design cooperative work Systems? *Proceedings of CSCW '90*, pp. 357-370.
- [13] Malone, T.W.; Grant, K.R.; Lai, K.-Y.; Rao, R, Roseblitt, D. (1988): Semistructured messages are surprisingly useful for computer-supported coordination. In: *Greif, I. (ed.): Computer-Supported Work: A Book of Readings*. San Mateo: Morgan Kaufmann, pp. 311-331.
- [14] Moody, Daniel (1996): Graphical Entity Relationship Models: Towards a More User Understandable Representation of Data. *Proceedings of the 15th International Conference on Conceptual Modeling, ER96*, (Cottbus, Germany, October 1996), pp. 227-244.
- [15] Oberquelle, H. (1987): *Sprachkonzepte für benutzergerechte Systeme*. Berlin: Springer, 1987.
- [16] Ortner, Erich (1995): Alternative Language Approach for Information System Development, in: *Proceedings International Conference in Information System Concepts – Towards a Consolidation of View, IFIP Working Group WG 8.1 und GI Working Group 2.5.2, Marburg 1995*, S. A15–1 bis A15-3.
- [17] Purchase, Helen (1997): Which Aesthetic Has the Greatest Effect on Human Understanding? *Graph Drawing. 5th International Symposium on Graph Drawing, GD '97*, (Rome, Italy, Sept. 1997), pp. 248-261.
- [18] Rational Software Corp. (Ed.) (1997): *Unified Modeling Language. Documentation Set Version 1.0*. 13. January 1997. Santa Clara, CA: Rational Software Cooperation, 1997.
- [19] Scheer, A-W. (1991): *Architektur integrierter Informationssysteme. Grundlagen der Unternehmensmodellierung*. Berlin: Springer, 1991.
- [20] Yourdon, E. (1989): *Modern structured analysis*. Yourdon Press, Englewood Cliffs, NJ, 1989.