Model-driven Planning and Design of Cooperative Information Systems

Matthias Jarke, Peter Peters, Manfred A. Jeusfeld
Informatik V, RWTH Aachen
Ahornstr. 55, 52056 Aachen, Germany
e-mail: {jarke,jeusfeld,peters}@informatik.rwth-aachen.de

Abstract:
Cooperative information systems (CIS) aim at *continued cooperativity* between user groups through componentized networks of information systems. Change management is therefore a definitional part of CIS. We advocate a conceptual modeling strategy for addressing this task, and illustrate it with experiences gained in WibQuS, a project aimed at CIS support for Total Quality Management in manufacturing organizations. These experiences emphasize the role of meta models in focusing the change process. Specific meta models and supporting environments are presented for: cooperative business process modeling in distributed organizations; simulation analysis of short-term and long-term effects of information flow designs; forward and reverse mappings between a distributed (relational) system interoperability layer and the information flow model. Models are not just analyzed at change time, but also support planned and unplanned information flows at runtime.

1 Introduction

Information systems have traditionally been characterized as hardware/software/people *systems* that maintain data about a specific *subject domain* for one or more *users*, sometimes within a formal organization. The database community has dedicated much work to subject domain modeling (e.g. relational, object-oriented and semantic data models) and efficient systems implementation (e.g. query processing, concurrency control, and recovery).

Relatively less attention has been paid to the usage side, neither at the system level of user interface research nor at the design level of modeling usage by individuals, teams, and formal organizations. Recently, this has begun to change due to massive complaints by user organizations that their central needs are not being adequately addressed by information technology.

One of the responses raising to this challenge is the vision of *cooperative information systems* (CIS [Brodie and Ceri, 1992, Jarke and Ellis, 1993, De Michelis *et al.*, 1996]). This vision presents information systems as a communications medium among user groups in and across organizations. On one hand, this brings groupware and organizational research into the information systems field. On the other, it changes conventional database wisdoms about the required information modeling, system implementation and integration technologies.

We conceptualize CIS as a layered network of user and system components as sketched in Figure 1. At the *system level*, the old idea of one-time view integration – followed by the integrated design of an integrated system – is replaced by a trend towards componentization (including the 'wrapping' of legacy software) into small and easily re-configurable objects. Coordination between these units is no longer hardcoded in applications but dynamically achieved by workflow mechanisms.

At the *usage level*, we observe very similar phenomena. Formal organizations are being decomposed into small autonomous units with market-oriented rather than hierarchical coordination [Malone *et al.*,

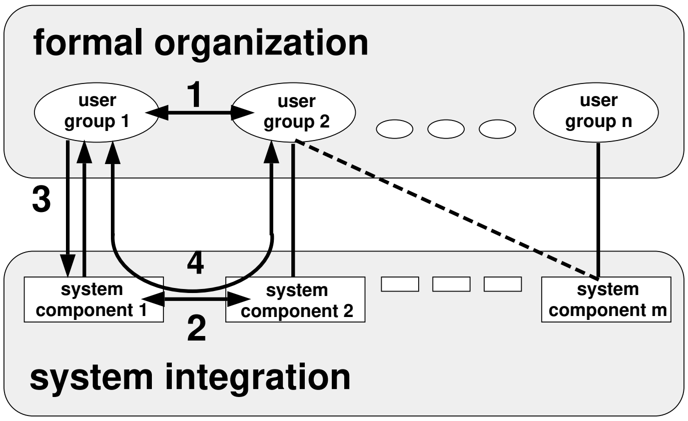

Fig. 1 Static structure of a CIS

1987]. In these federated organizations the units have their own local information systems configuration, either from new or from legacy components. But they may also be customers or suppliers of other units with respect to information services. Thus, in a CIS, we have necessary interactions between user groups (1 in Figure 1), between system components (2), between user groups and system components (3), and – most importantly for us – between user groups *through* system components (4).

Why is all this happening? The answer – at both levels – is *reactiveness to change* [Heinzl and Srikanth, 1995, Scott-Morton, 1994]. Organizations have to react quickly to ever-changing market requirements. System technology has to react quickly to organizational change as well as technical innovation. Hierarchies have turned out to be too clumsy, therefore the trend towards small largely autonomous units that can react quickly.

CIS are intended to assure this reactiveness to change (often called *agility*) by providing flexible communication mechanisms and rapid feedback between stakeholder groups such as customers, planners, designers, production managers and workers. Under these requirements lack of relevant information is as dangerous as information overload, and schema evolution due to changing user interests is a necessity.

In this chapter, *we argue that cooperativeness and agility are not just requirements in the usage of CIS but also central determinants of the change process itself.* Our main message is that conceptual modeling technology, properly supported and applied, can go a long way in addressing such CIS engineering goals. Specifically, we propose *user-definable meta meta models* as a means of coordination in requirements engineering and systems design for CIS. Elaborating this idea, we demonstrate how conceptual modeling technology can be extended to deal in an integrated manner with problems such as

- the integration of different terminologies and methodological frameworks through meta modeling and the resulting need for uncovering implicit assumptions about business processes and systems [Klein and Lyytinen, 1992] by the stakeholders in the modeling process
- the evaluation of different information flow policies through simulation models derived from the meta models by explaning the effects of local CIS changes on the overall CIS performance
- the computer-supported mapping of meta models to database structures, workflow definitions, and monitoring facilities at the operational level and therefore an enhancement the agility of the CIS
- the use of the conceptual models at runtime as a domain-specific information trader to facilitate intelligent question-answering and workflow coordination.

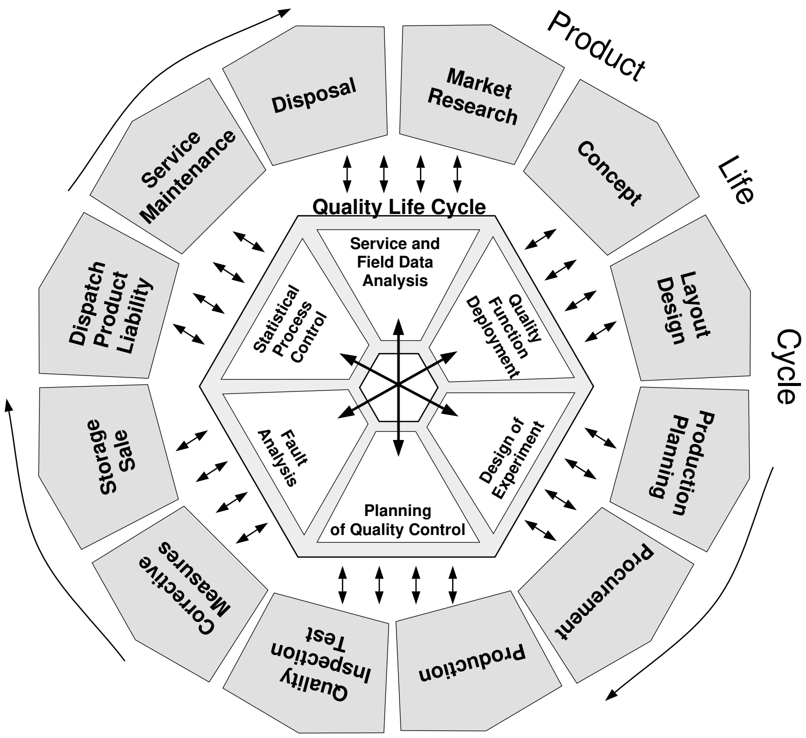

Fig. 2 The quality life cycle as a CIS

To ground the discussion, section 2 presents an example CIS – a prototype environment for life cycle wide quality management developed in a German industrial engineering project called WibQuS. In Section 3, we introduce the basics of our conceptual modeling approach which relies heavily on user-definable meta meta models to coordinate distributed CIS engineering, and discuss its technical support by the ConceptBase meta data manager. Section 4 describes how the engineering approach applies in the structural design and quantitative evaluation of organizational information flows while Section 5 discusses the mapping of such designs to the system level. Section 6 lists some limitations and summarizes ongoing work.

2 An Example CIS : The WibQuS Environment

Total Quality Management (TQM) is one of several business philosophies that can be associated with the trend of on-going change as a basis of business success. It aims at continuous business process improvement by emphasizing customer orientation throughout, relying on local ideas and improvement initiatives, influencing the organization via cleverly designed feedback cycles. Therefore, the implementation of a TQM system appears as a prototypical example of a (not necessarily fully computerized) CIS.

The question of how to design and implement such CIS was investigated since 1992 by a consortium of five German engineering centers, an organizational science group, and ourselves in a project called WibQuS[1] [Jarke *et al.*, 1993]; production data from a "real" product life cycle were provided by a manufacturer of forklifts. Figure 2 shows how the CIS components are instantiated in industrial quality management [Pfeifer, 1993].

The outer circle presents the product life cycle stages. Traditionally, the stages have been associated with different departments or companies, each with their local IT solution. Due to business process

[1] WibQuS is the German abbreviation for "Knowledge-based Systems in Quality Management".

orientation, the forward flow of information along the product life cycle is now reasonably under control. However, the feedback loop around the whole cycle takes too much time for companies to stay competitive. The idea of quality management is to introduce smaller feedback cycles that provide selected information about particular quality issues, thus enabling rapid and continuous improvement. This so-called *quality life cycle* can be supported by methods and software tools as shown in the inner part of Figure 2. The six methods in the WibQuS project which form the framework for the CIS are as follows [Pfeifer (ed.), 1996]:

- *Quality Function Deployment (QFD)*
  The QFD method [Hauser and Clausing, 1988; Akao, 1990] is a quality planning method that relates customer wishes to product and process features. In a stepwise team process, customer requirements are related to product features, which in turn are related to production process and quality control features. The process is guided by a multi-matrix stucture, the House of Quality (HoQ). In the WibQuS environment, a system called DACAPO supports the knowledge-based structuring of products and their functions in order to prepare the HoQ structure for team sessions. QFD engineers need information about products and processes from all over the company in a compressed manner in order to relate customer wishes to the engineering and technical capabilities of the company.
- *Design of Experiments (DoE)*
  Goal of the DoE [Quentin, 1989; Pfeifer, 1993] is a systematic optimization of processes and products by a set of experiments. The CADOX tool supports the DoE team in the selection of the right experimental design based on the knowledge that already exists about the processes and products (critical quality features, previous experiments, production experience) as well as the possible experiment designs.
- *Computer Aided Fault Analysis (CAFA)*
  The CAFA method [Pfeifer *et al.*, 1995] collects and provides cause-effect knowledge about products and processes in order to define corrective actions in case that faulty process behavior is detected by the SPC. The tool supports knowledge acquisition as well as the detection of appropriate actions by inference over the company's quality management knowledge base.
- *Knowledge Based Planning of Quality Control*
  This method deals with the systematic description of structural and functional knowledge of quality control techniques in order to support the engineer in the selection and preparation of the appropriate control technique, plan, and measurement instruments according to the planned product features.
- *Extended Statistical Process Control (XSPC)*
  Statistical process control [Köppe and Heid, 1989] is one of the earliest quality management methods. It checks the process by analyzing a particular feature using statistical samples. XSPC does the same for two correlated features. The XSPC tool supports the selection of the appropriate control chart and the analysis of probably critical process developments.
- *Service and Field Support (FDE)*
  This method supports the service engineer in the efficient collection and goal oriented preparation of quality data gained during customer visits. The FDE tool provides guidance for this data collection. It also supports the definition of service actions and the statistical analysis of data for further use by other methods.

Broadcasting all available quality information produced by these methods will quickly overwhelm and turn off users. The goal of our work in WibQuS was therefore to develop a set of methods and tools for CIS planning and design which would help companies identify possibly useful information flows, help them structuring the information flows according to the possible interactions in Figure 1, estimate the impact of different information flow designs, and provide different degrees of operational support for planned and unplanned information flows with respect to the chosen design. Since quality management implies continuous change, the interplay of these methods had to be organized to make change as easy as possible.

The consortium of the WibQuS project and the methods represented reflect the issues in CIS for federated organizations very well:

At the system level the participating engineering groups had their own software standards, partially existing systems and databases on which the above tools were based so that significant heterogeneity existed. In the WibQuS prototype, interoperability at the implementation level was provided by using SQL as the exchange standard for data; all subsystems had relational databases (of different vendors such as Sybase, Oracle, and Informix) to interface with the rest of the environment.

On the organizational level the groups were distributed across Germany so that major geographical distances were involved. The methods to be represented originate from different sub-disciplines and it turned out that the different engineering sub-disciplines had also developed quite different and not always consistent vocabularies and conceptual understandings. Due to the differing complexity of the methods — from complex design support methods to automatable process control tasks – the involved agents, their roles in the methods and the needed tool support differed widely. Furthermore, the degree of computing knowledge varied widely in the teams.

For the following sections, the experiences made in this exercise will serve as our main basis for examples. While the users/developers were mostly industrial engineering researchers rather than actual users in companies, and despite the technical limitation to relation-based interoperability, we believe that useful lessons can be learned from these experiences. Moreover, several of the individual analysis and design methods presented below were tested in real-world industrial environments; where appropriate, we shall mention these experiences as well.

3 Conceptual Models as a Basis for Change Management

3.1 A Basic Change Management Process for CIS

Even if an organization is federated, the user groups must still be coordinated such that they move towards common goals and, more importantly, that adaptation to *changing* goals is always possible. Given that coordination mechanisms should be loose and market-like, how can we create the necessary market infrastructure and force fields that let the organization drift in the direction management and customers want, and still provide services to the users that make working with the CIS effective?

Our solution is based on *conceptual modeling* (Figure 3). It is a generalization of the task-artefact cycle for user interface design [Carroll *et al.*, 1991] : you (1) reverse-engineer the rationale from the existing man-machine system into a conceptual model, then (2) analyze a proposed change conceptually, finally (3) implement the agreed change, (4) taking into account the existing legacy context. Then the cycle repeats, *ad infinitum*.

There are different ways to manage this evolution cycle. A commercially most successful is the ARIS/SAP approach [Scheer, 1994]: You start with an ARIS reference model for the business process in question, and customize it to your specific situation. Each component of the reference model is linked to a parameterized SAP software component. The resulting software system is largely configured by parameter tuning.

Although this approach is becoming very popular these days, it has some drawbacks. Obviously, the feasibility depends on the coverage of the reference model and, in particular, on the regularity of the

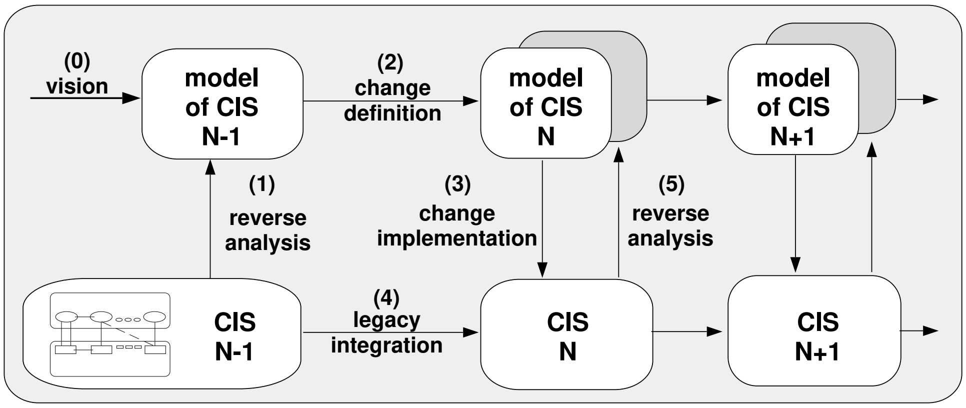

Fig. 3 A model-based evolution process for CIS

business process itself. In more complex cooperative settings which include creative aspects, it either does not cover much of the required information flows, or tends to over-constrain user cooperation.

Structuration theory in organizational behavior gives one possible explanation for these observations [Orlikowski and Robey, 1991]. According to this theory, organizational structure and organizational change are inextricably linked. If you want a structured operational workflow, a centrally driven reference model adaptation process will work well. If you want a distributed CIS, your design approach also needs to be distributed and cooperative. Moreover, if you want to support organizational change, your models of reality must necessarily reflect the purpose of the change. In Figure 3, the reverse-engineering therefore link goes back to a revised version of the model from which the system was initially constructed. In other words, the cycle actually does *not* start with step (1) but with an initial vision of step (2), the intended change. In previous work, we have expressed this observation by defining requirements engineering as 'the process of establishing a vision in the existing technical, cognitive, and social context' [Jarke and Pohl, 1993].

3.2 Meta Modeling as a Coordination Strategy

Distributed modeling is a necessity in CIS due to the diversity of knowledge and the autonomy of the network nodes. However, it also make reconciliation of viewpoints quite a difficult task. To illustrate, Figure 4 shows how designers of two WibQuS applications (WiFEA support for service technicians and DACAPO for quality function deployment) are coordinated in defining their viewpoints of the same corporate reality – step (1) in Figure 3. They abstract their observations of reality into local application models. These models may differ not only in the notations used but are also influenced by the different knowledge background, and the different interests of the modelers. In other words, viewpoint resolution must happen in all three dimensions of requirements engineering : representation (notations), specification (domain understanding), and agreement (negotiation) [Pohl, 1994].

To indicate how this can be supported by a repository, the left of the figure relates the picture to the levels of the ISO Information Resource Dictionary Standard [ISO/IEC, 1990]. In IRDS, the concepts on level *n+1* (the defining level) constitute a type system for level *n* (the defined level). A sub-repository at level *n+1* can manage the interaction of subsystems at level *n*.

In such a multi-level framework, reconciliation can happen in an ad-hoc manner at the level of the models themselves. In this case, the participating tools are seen purely as drawing tools and practically no computerized support for cross-analysis is possible.

Integrated CASE environments (e.g. the Oracle CASE family) or recently proposed viewpoint resolution mechanisms [Easterbrook and Nuseibeh, 1995] go one step further : they explicitly represent

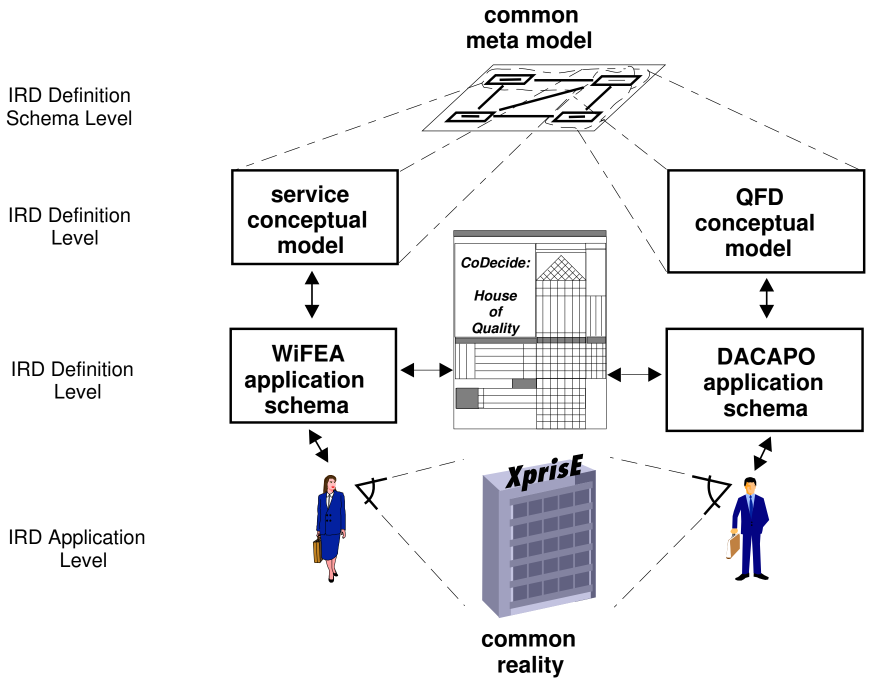

Fig. 4 Integrating different business views with meta models

possible relationships between the different notations or domain models controlling the development of each viewpoint, e.g. in the form of mapping or consistency rules. The environments then differ in whether they see these rules as strictly to be enforced or as quality control tools to be applied to a "sloppy modeling" process at the user's discretion.

However, in complex heterogeneous environments, the development of such mapping rules is itself a difficult an error-prone process. A number of environments have therefore adopted explicit *meta models* (actually: a meta meta models) defining a shared language in which the domain models, notations, etc. are represented. Similarities, transformations, and consistency checks between notations or domain abstractions can then be defined in terms of this meta model.

The first meta models have focused on the integration or generation of *design notations*. Several commercial CASE environments, including PROMOD by Cap debis, include meta models for the integration of different notations. The MetaEdit+ environment developed at the University of Jyväskylä [Tolvanen and Lyytinen, 1992] allows users to define graphically their own graphical notations, together with referential constraints across such graphics. Enforcement of constraints is strict, as all notations are seen as views on a consistent design database. Notational meta models can also support runtime tasks such as exploratory search in large networks of heterogeneous databases, where node schemata only become known opportunistically as they are visited for querying [Papazoglou *et al.*, 1995].

Integration via shared *domain meta models* is a more recent phenomenon. For example, in the medical domain, the Unified Medical Language System UMLS [National Library of Medicine, 1994] offers a semantic network structure, metathesaurus, and information sources map with the goal of providing uniform access to heterogeneous knowledge sources and aiding their integration. ARIS offers its reference models under a meta model of event-driven process chains [Scheer, 1994]. The most ambitious attempt is probably the Carnot project at MCC [Huhns *et al.*, 1993] which used the "universal" ontology of hundred thousands of terms in the CYC knowledge base for IS integration.

However, having a huge ontology in a system does not mean that users share understanding of this ontology. Indeed, given the diversity of aspects on which modeler viewpoints can differ, we believe that no single domain meta model can be at the same time specific enough to provide actual guidance, and small enough to enable shared understanding and avoid unnecessary modeling overhead. A distinguishing feature of our approach is therefore that the development team can define *their own meta model*. The meta model is negotiated by the owners of the notations and then serves as a guidelines for the modeling process, as well as a basis for communication about the coherence of different models. In other words, our *user-definable meta models represent the shared 'vision' of potential problems or solutions which drives the analysis process.*

The ConceptBase system [Jarke *et al.*, 1995] enables this by the unlimited classification hierarchy in its *Telos* language [Mylopoulos *et al.*, 1990] which directly supports modeling the multi-level repository hierarchy shown in the figure. So-called *meta formulas* attached to a meta class allow you to define deductive rules and integrity constraints that specialize automatically to each class which is an instance of this meta class. The thus specialized formulas then apply to the instances of these classes. This makes meta formulas robust with respect to notational variations and is a good means for conflict analysis across heterogeneous representations or worldviews.

The common language defined in the meta model alone is usually not enough to ensure coherence of distributed modeling since at maximum the formalizable part of a vision can be captured in it. The modeling environment is therefore augmented with additional conflict analysis and negotiation support tools which are defined at the same level as the methods and tools they try to relate. In ConceptBase, they are specialized automatically or configured semi-automatically from two generic instruments defined with the meta model: query classes and matrix-based visualizations.

Query classes [Staudt *et al.*, 1994] are parameterized and possibly materialized views defined by necessary and sufficient membership constraints similar to description logics [Borgida, 1995]. They defer the checking of consistency and completeness of the perspectives generated by parallel development to a moment definable by the teams themselves. Usually, a large number of query classes can be associated with a meta model and later applied in the negotiations between the modeling teams whose modeling work is based on this meta model.

CoDecide is a visualization toolkit for conflict analysis [Jarke *et al.*, 1996]. A CoDecide tool shows interrelationships between submodels via one or more matrix representations. The toolkit approach allows modelers to quickly develop specialized visualizations for particular kinds of conflicts. Different synchronization styles, ranging from fully shared editing to asynchronous cooperation with partial visibility of mutual views can be supported. A standard example of a conflict visualization interface based on CoDecide is the House of Quality from the QFD method shown in the middle of figure 4 which offers a comprehensive visualization of the relationships within and between two parties, plus some external context such as versions or competing solutions.

The combination of formal modeling and negotiation management applications makes ConceptBase a flexible modeling environment for CIS. The models designed cooperatively in the repository architecture implemented in ConceptBase form the starting point for other CIS engineering tools supporting system planning, design, and usage tasks, as discussed in the next sections.

4 Identifying and Planning CIS Information Flows

In this section, we discuss the CIS engineering process at the level of information flows in federated organizations. Here, meta models firstly help to link the analysis step (1) in Figure 3 with the change

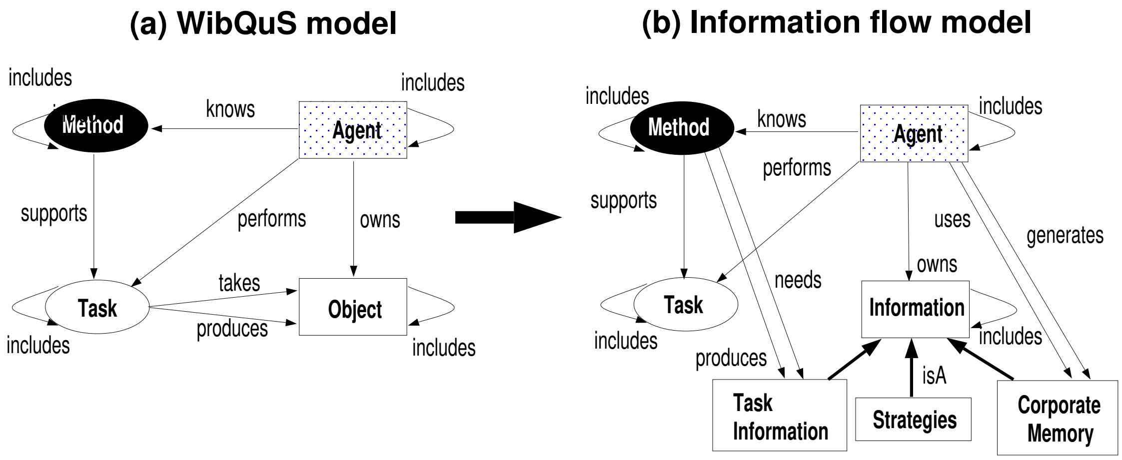

Fig. 5 The WibQuS meta model and its successor

definition step (2). secondly, they identify suitable design choices by simulation in the transition from the change definition step (2) to the change implementation step (3). Both activities will be described in the next subsections.

4.1 Cooperative Modeling of Information Flows

There is usually a large conceptual and spatial distance among the different groups involved in CIS design and use. This distance needs to be bridged to some degree before a successful modeling process can start. The modeling process does not aim at generic mutual understanding but has a *specific purpose*. In WibQuS this purpose was the capturing of information flows between the supported methods. Only after some agreement on this purpose has been reached, distributed modeling should start. The formalizable part of this agreement can be coded in a *business process meta model* that defines a language in which modelers communicate about processes, methods and their interactions.

Our strategy for defining an overall model of information flows to be implemented in the CIS consists of four phases (cf. figure 6):

1. Joint design of a shared language meta model as a basis for process definition
2. Distributed Modeling of business processes and information flows within the methods
3. Integration of the method models
4. Revision of the shared meta model based on the gained experiences, followed by optimization of the integrated method models.

The *first phase* involves significant literature studies as well as lengthy negotiations and first experiments with the six modeling teams involved. Starting from the principles of the CRIS methodology [Olle *et al.*, 1991], the business process description consists of **Tasks** and **Objects** consumed and produced by those **Tasks**. Additionally, due to the cooperative nature of the system, it refers to the **agent** as the responsible performer of a specific **Task** to whom a supporting IS should be adapted. Finally, the concept of **Method** was introduced, because there may be different ways-of-working for a specific task. Methods and tasks form an AND/OR decomposition structure in that a task can be supported by one or several methods while a method consists of a partial order of sub-tasks. Figure 5(a) shows the final version of the WibQuS meta model.

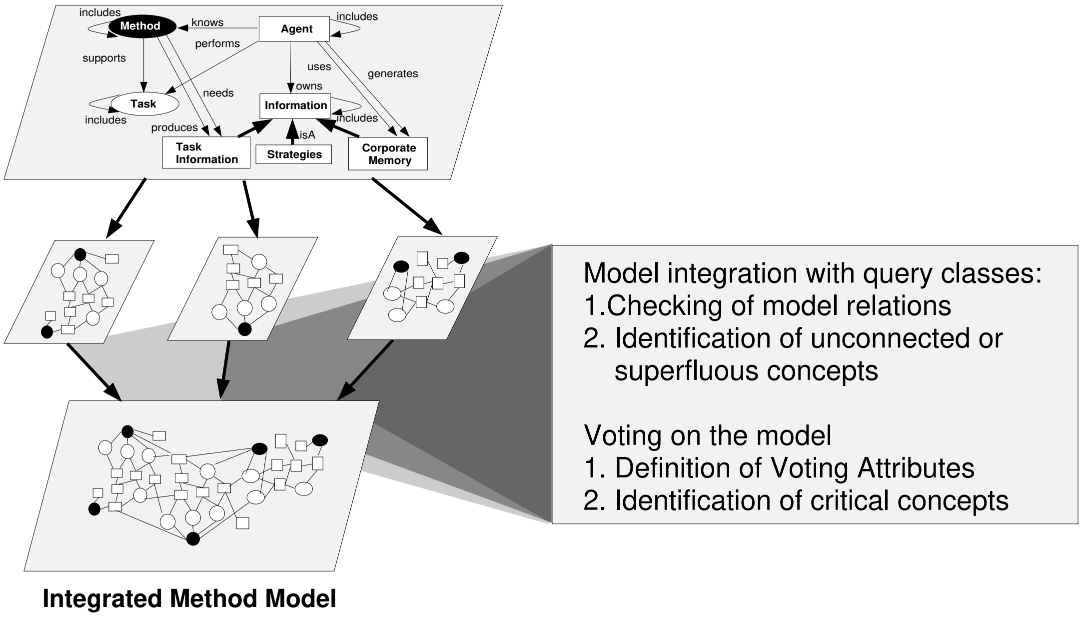

Fig. 6 Support of distributed modeling by the modeling language and ConceptBase

In the *second phase*, each team of engineers developed models of the different methods used along the product life cycle by distributed instantiation of the modeling language making use of the client-server architecture of ConceptBase.

The consistency checks of ConceptBase ensured that the models defined where in line with the meta model by strict monitoring of superficial concept correctness via the axioms of the knowledge representation language (here: Telos). The *instantiation principle* avoided modeling errors, since all concepts had to be instances of the defined meta model. The *naming principles* of ConceptBase avoided that two different concepts with the same name were defined. The detection of such inconsistencies led to direct communication between the modeling groups. If a differing meaning of concepts was detected, one of the groups had to change the name. If the same meaning was intended with the same naming, the first interfaces between methods were detected even before the model integration phase had started.

The *third phase* of our strategy, partially overlapping with the second one, concerns support for the negotiations that are intended to ensure coherence between the models. The groups discuss the defined concepts in iterating cycles until a common understanding and agreement about the models is reached (cf. Figure 6). Besides the naming principles described above ConceptBase offers two techniques for this purpose:

(a) analysis of formal conflicts such as inconsistency or incompleteness via query classes:
As an example, consider the search for concepts that fulfilled the defined semantics of the modeling language only partially. We were looking for **Objects** that were not taken or produced, **Tasks** where no **Agent** was attached to, and concepts that weren't connected to others at all[2]. A simple example for a query class looks as follows:

[2] In a commercial business process analysis environment based on ConceptBase, about 80 such query classes were identified to uncover analysis errors, differences in opinion, and problems of the business process [Nissen *et al.*, 1996].

Number of objects	592
accepted_by	195
rejected_by	2
not_understood_by	13
comment	178
accepted_by and rejected_by	90
accepted_by and not_understood_by	7

Table 1 The results of the voting process

```
QueryClass DeadEnd isA ConceptualCategory with
              comment
                 explanation :  "Information provided but not needed"
              constraint
                 qr_deadend  : $ exists t1/task (t1 produces this)
                                 and not exists t2/task (t2 takes this) $
          end
```

CoDecide was not yet available during the WibQuS modeling process; an example of its use for cooperative business process analysis is given in [Jarke *et al.*, 1996].

(b) analysis of subjective conflicts through structured annotations:
In the WibQuS project, we initiated a voting on concepts included in the knowledge base under a simple variant of the IBIS model [Conklin and Begemann, 1988]. We attached voting attributes to each concept such that modelers could annotate their opinions about the definition [Peters and Jeusfeld, 1994]. Table 1 shows the result of the voting, grouped in criticality classes represented by special queries. The results of this classification were given to the groups, who resolved the problems indicated by direct communication. The cycle of voting and discussing was performed until the number of critical concepts was getting small enough to discuss them at a meeting where all groups were represented.

The end result of the modeling process was a set of almost 600 classes which constituted the first formal ontology of the quality cycle in engineering-oriented enterprises. The pen-and-paper analysis and integration of such a complex model in a federated fashion would have been very costly or even inconductable. The meta modeling approach together with the client server architecture of ConceptBase enabled a rather fast process which lasted about half a year from the first language definition to the integrated model.

The *fourth phase* was a review of this ontology considering the goal of facilitating the planning, design and operation of CIS with high local autonomy. Therefore, the review included theoretical considerations of information flow analysis as well as experiences with the developers themselves [Peters and Jeusfeld, 1994]. Two major shortcomings identified during this analysis were:

1. The interpretation of information flows as inputs and outputs of tasks as in the CRIS model was too simplistic. Firstly, the information exchange needs do not just depend on the task but more on the method how it is performed. Secondly, there are many important information exchanges between agents, e.g. concerning product quality and improvement opportunities, which cannot be related to specific tasks but can rather be interpreted as materialized organizational memory.
2. The models differed widely in their emphasis and degree of detail since no guidelines wre available when to stop modeling. Some modelers designed the model not as pure business process models but

as a specification of the tools they wanted to build. This over-commitment hindered the integration of the models and led to high overall model complexity.

The first problem led to the refocusing of the language model towards explicit representation of information flow categories, based on the literature about information management and organizational IS in industrial environments [Peters, 1996]. The revised meta-model (cf. Figure 5(b)) distinguishes three major groups of information flows within organizations, which are to be implemented and supported differently:

1. *Task Information* drives and controls the business processes in organizations [Hammer, 1992; Olle *et al.*, 1991; Scheer, 1994]. Like e.g., the structure of a product in manufacturing industries, this information is commonly understood within an organization. Its transfer between departments or tasks is based on standardized formats. The effects of task information are *short term and local*: It results from one department, is transferred to the next, and can initiate a new task. After task execution it is stored in the IS and might become part of the corporate memory.
2. *Corporate Memory* about products and processes results from accumulated execution and analysis of business processes [Senge, 1990; Vennix *et al.*, 1994; Harmsen *et al.*, 1994]. Capturing this corporate memory can ensure higher quality and efficiency of a process execution [Chen *et al.*, 1992; Fine, 1986; Pentland, 1994]. Shared knowledge enables higher productivity and a faster learning curve of the company [Adler and Clark, 1991; Cooprider and Victor, 1993]. In contrast to task information, corporate memory cannot be formalized easily. It consists of informal documents like experience collections, stories or rationales [Goldstein, 1993; Ramesh and Sengupta, 1994] and of formal ones like design rules or process models. It needs rich communication support that reduces equivocality[3] and uncertainty in communication [Daft and Lengel, 1986].
3. *Strategies* define a common context according to which tasks are organized and information is interpreted [Hammer, 1992]. They consist of a set of visions, policies and goals under which an organization or department operates. Examples are TQM, JIT, or BPR. Strategies result from long-term experiences [Burgelman, 1988] or theories. They are implemented to ensure long-term competitiveness and should influence every part of the organization.

Besides structuring the vast amount of Objects in the WibQuS knowledge base with respect to usage criteria, the definition of these categories is of major importance for CIS analysis and process support as we will show in the following sections.

With respect to the second problem (when to stop modeling), we argue for a low modeling depth of the method model for two reasons:

- The purpose of the model is the integration of loosely coupled, autonomous methods. Hence, the internal structure of a method is regarded as a black box in terms of integration as long as the necessary information is available to the outside on time.
- According to [Reichwald, 1989] QM and design tasks are of medium complexity and planability. Information needs are dependent of the problem and cooperation is flexible but performed with defined partners. In those situations a framework for information exchange is the right level of detail for business process planning.

[3] Equivocality is like uncertainty but with a twist: it evolves in ambiguous or questionable discussion domains.

Therefore, we added a constraint to the `method` concept:

```
Individual Method in Class with
  attribute
     supports : Task;
     includes : Method;
     constraint
           nondecompose1 : $ forall a,b,c/Method
                                (a includes b) and (a includes c)
                                  ==> not (forall x/task_information
                                              (b produces x)
                                        and  (c needs x))
```

The constraint says that a `Method a` cannot be decomposed in `b` and `c` if information exchange takes place only from `b` to `c`. A similar constraint was defined for the `agent/corporate memory` relationship. These constraints ensure that the model is decomposed only as far as information flows to other `methods` are defined. Applying this modeling rule to the results of the WibQuS modeling process led to a significant reduction of model complexity. For example, in the original model the QFD was described by about 100 concepts. Using the non-decomposition constraint this was reduced to 40.

The example of the revised WibQuS meta model shows, how the selection of the modeling concepts constrains the view of the modelers and influences the model quality. The resulting model of QM methods formed the basis for defining a technical infrastructure of federated information systems. It also offered the basic structure upon which quantitative analyses of data flow and organizational learning could be piggybacked, as discussed in the next subsection.

4.2 Analyzing Information Flow Designs By Simulation

The previous subsection showed a strategy for the conceptual design of CIS in a distributed organization. We now describe how these models can be further exploited for an analysis of information flow impact on organizational perfomance in order to choose among possible change scenarios. Those are to be implemented in step (3) of Figure 3.

The selection of the analysis methods is highly dependent of the information flow category. The analysis of task information is driven by its transaction costs, its timeliness, and its completeness. The exchange of corporate memory is analyzed to find out its long term effects on quality, flexibility, or personnel qualification[4].

Task information criteria describe short-term, local effects which relate directly to the business process. The analysis of such criteria is usually performed by Petri-Net or Queuing System simulation [Deiters *et al.*, 1995; Oberweis *et al.*, 1994].

The analysis of *corporate memory* is much harder, because its effects are related to long-term feedback loops within an organization: information has to be accumulated, condensed and then transferred to the organizational units where its effects are supposed to happen. These processes are not related directly to the task workflows and cannot be measured by hard business variables like time and money. The effect of corporate memory is defined by *the way they influence* the variables that produce those time-and-money effects along multiple feedback loops.

---

4 We did not yet attempt the impact of strategy information, since the WibQuS had a predefined strategy underlying CIS design, namely TQM.

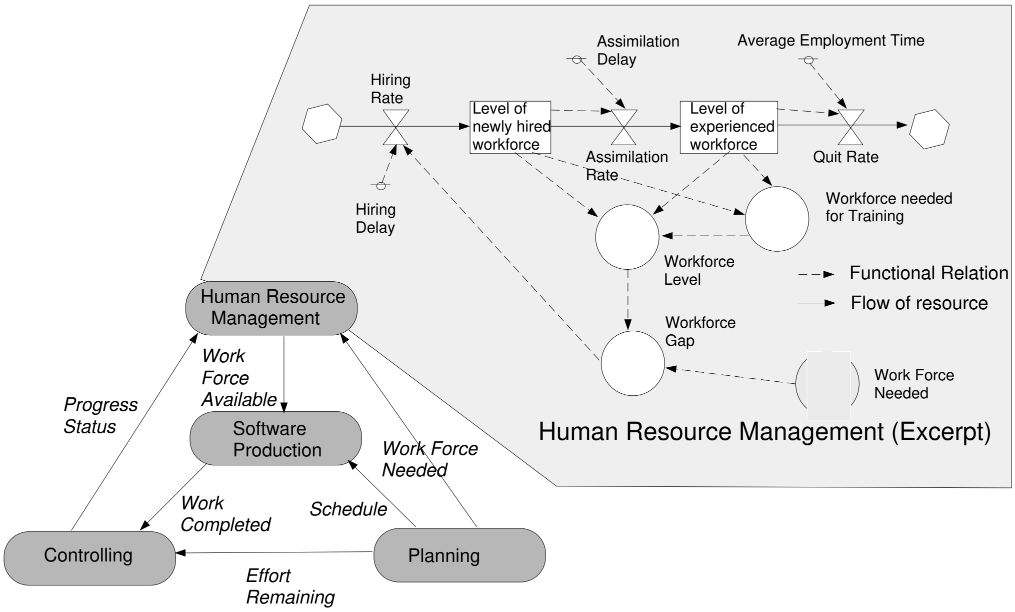

Fig. 7 Software development subsystems (adopted from [Abdel-Hamid and Madnick, 1991])

In his work on software project dynamics, Abdel-Hamid showed that System Dynamics (SD) simulation is well-suited for the analysis of multiple cause-effect feedback loops that describe the productivity behavior of a software development team performing a specific project [Abdel-Hamid and Madnick, 1991]. In the SD technique [Forrester, 1961] the interaction of variables and systems is descibed as a flow of resources between levels, influenced by the perimeter of valves (rates). This perimeter in turn is determined by the states of variables throughout the system. The philosophy of SD is that people can describe structure and local behavior of a system well, but fail to predict global behavior, especially if feedback loops of different length and complexity are part of the system.

Abdel-Hamid developed and validated a set of models for the major factors related to software development productivity and established the feedback loops within and among them (see Figure 7 for a overview of the model structure).

As an example, consider the simulation of an effect called Brooks Law: If schedule pressure is detected in the *Controlling* submodel the *Planning* module is defining the *Workforce needed* to meet the schedule. This workforce needed determines the current workforce gap and influences the hiring rate. Addition of new workforce to *Newly hired workforce* leads to additional *Workforce needed for training*. This reduces the overall workforce available, since newly hired workforce is considered being of significantly lower availability. The result is a lower productivity and even more schedule pressure.

For the analysis of CIS we generalized the model for the representation of interacting organizational units with explicit information flow management and error propagation models [Peters, 1996]. In addition, information flow planning in CIS requires modeling the *interplay* between the short-term and long-term effects. At the system level, this implies a *heterogeneous simulation* by interoperating discrete (Queueing Systems) and continuous (System Dynamics) simulation techniques. Fishwick recently showed that every quasi-continuous simulation technique can be mapped to a discrete-event technique if the time increment is sufficiently small [Fishwick, 1995]. Our MultiSim environment generalizes this idea to a graphical simulation definition and execution environment [Peters *et al.*, 1996].

The kernel of MultiSim is a ConceptBase *repository for simulation techniques*. We developed a definition language for simulation techniques by which simulation languages like SD, Queueing Systems,

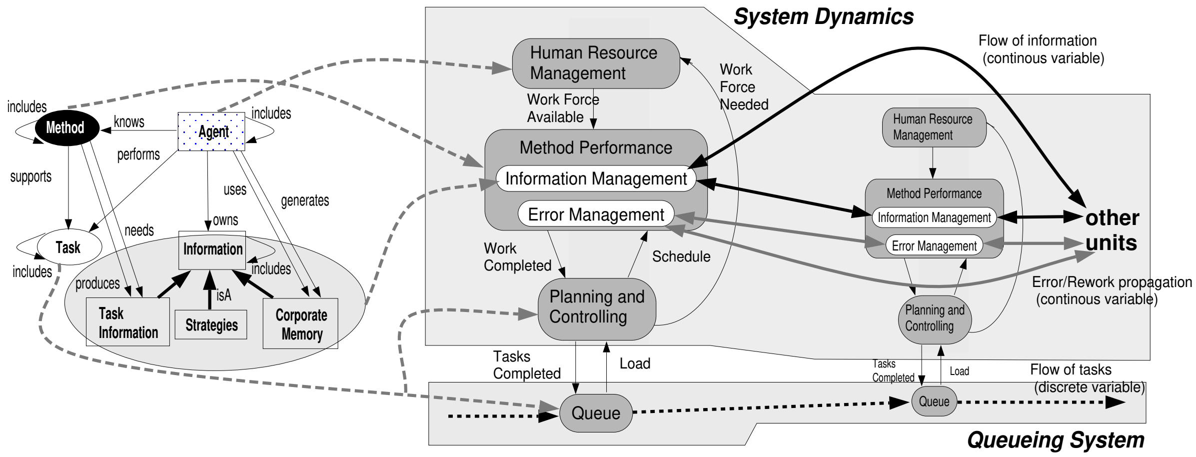

Fig. 8 Mapping of the conceptual to the simulation model

or Petri-Nets can be modelled and connected. We do not show this meta model directly but rather the mapping between its major submodels and the WibQuS meta model constructs (Figure 8).

Each *agent* is represented by a *Human Resource Management* model. *Tasks* are represented by *Planning and Controlling/Queue* models. *Methods* are mapped on *Method Performance* models which describe how manpower made available by human resource management is spent on various parts of a task. The *Error Management* module analyzes the rate by which errors are generated, detected and reworked within a method and the resulting effort in needed manpower. A variable models error propagation between Error Management models along the business process. The *Information Management* model describes the amount of work necessary to access, provide and manage the information flows defined in the conceptual model. It also provides the corporate memory as a resource that influences the productivity of other tasks in the model, e.g. training effort, task productivity, or error generation.

A first empirical validation of this combined simulation model in a business process re-engineering project for a manufacturing company showed a good match between simulation model and real business dynamics [Peters *et al.*, 1996].

4.3 Coupling Design and Simulation Repository

Figure 9 shows the interplay of meta models for the design and analysis tasks which are performed in the steps (1) and (2) of the CIS engineering process.

On the right branch, starting from the *WibQuS meta model*, process models for QM are coordinated using the method models at the second level as reference models. These process models are mapped to a multi-simulation model (left branch) which is itself based on interoperability between different simulation methods defined through a *simulation meta model*. In our special case this is the interaction between System Dynamics and Queueing Systems.

The interplay of simulation and actual process allows on-going evolution of the CIS: On the one hand, the observation of real processes leads to changes of the simulation model which in turn indicates necessary changes of the organizational process structure. On the other hand, experimenting with envisioned changes to the CIS in the simulation model leads to insights about the dynamics of those changes and therefore allows for the selection of appropriate evolution steps.

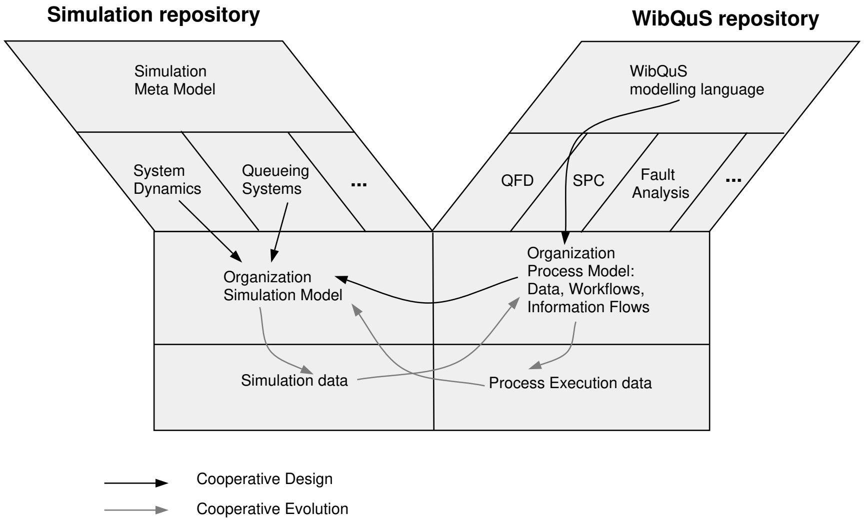

Fig. 9 CIS design and simulation process based on meta modeling

5 Operational Support for CIS Information Flows

The cooperative modeling process identifies *possible information flows* while the multi-simulation approach helps the organization evaluate which of these should be *specifically facilitated*. In this section, we show that choices can be *implemented* by linking certain views on the conceptual model to the system level, in the WibQuS project a federated network of relational databases. This supports the integration of steps (3) and (4) in Figure 3. The mappings are maintained in a *Quality Trader* which was implemented by linking ConceptBase to the Sybase OMNI-SQL gateway for distributed execution, and a relational database for maintaining temporary exchange data such as contract status [Peters *et al.*, 1995b].

Three kinds of mappings are derived from the WibQuS meta model. The ’owns’ link in Figure 5 is mapped to ownership of database nodes. In practical CIS settings, this partitioning is often given and must simply be recorded as a dependency link in the meta database. Section 5.1 shows how the conceptual structure of task and memory information is related to the underlying database schemata; this allows users to make *unplanned information searches* in a browser-based environment (Section 5.2). *Planned information flows* are mapped to stored queries and workflow structures (Section 5.3) which are more efficient to use but more difficult to change than the unplanned ones.

5.1 Mappings Between Concept and System Layer

Task information flows were defined in a subset of Telos that is basically an extended Entity-Relationship language. Many techniques exist how to map such models to efficient relational schemas; the reverse mapping, how to extract conceptual models from existing relational databases, has also been studied. Since both questions are equally relevant in the CIS context, we developed a *symmetric approach* where the mapping in both directions is constrained by the same meta model [Jeusfeld and Johnen, 1995]. In Figure 10, the organization level of figure 1 is shown to the right, the system level to the left. Steps (1)-(5) sketch how, as an example, reverse-modeling is driven by the lowest common ancestor of the meta classes of which the two concepts that have to be mapped, are instances. Since the expressiveness of both formalisms is not the same, the process is partially interactive (steps 3 and 5) : reverse mapping demands a *semantic* enrichment whereas forward mapping demands performance-oriented design choices.

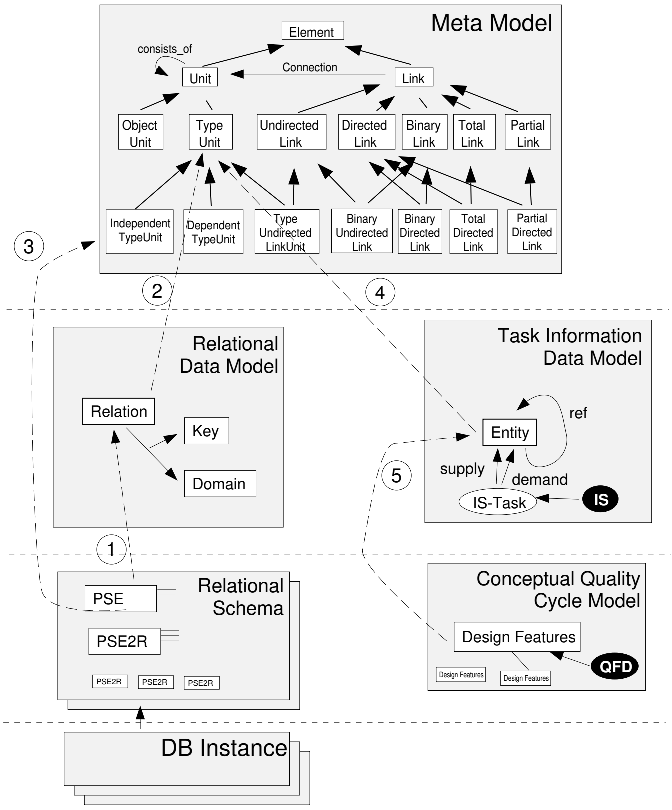

Fig. 10 Meta model constraining the mapping between concepts and relations

CIS engineering uses neither of these directions alone but demands a sophisticated interleaving. As *organizational model and system implementation co-evolve*, business and system modelers should benefit form each others' work. Typically, system designers know more details while business modelers have a broader overview. Once a link between the two models has been established, business modelers can apply reverse engineering to elaborate their conceptual task information models, and forward mapping to inform system designers about the usage context.

The main purpose of schema design in CIS is support for inter-group information flows. To interpret the thus exchanged messages, it turns out to be useful to standardize their contents using a shared meta model of the manufacturing product and process. In WibQuS, this meta model was a relational schema derived from the STEP standard, augmented with quality attributes. This *product and process model* became part of all exchange schemata of the federated databases. Now, the conceptual task information model for each subsystem must be considered a *view* on this standard model which further complicates the reverse and forward engineering task.

A complete methodology for co-engineering organization and system models is still under development. The result is in any case a set of dependencies between the two models that helps in further system evolution [Jeusfeld and Jarke, 1995], but also allows the end-user to view the distributed relational database through the conceptual model [Peters *et al.*, 1995a] in a two-level filter-browser presented in the next section.

5.2 Supporting Ad-hoc Information Flows

In distributed environment under on-going evolution only a limited amout of information requests

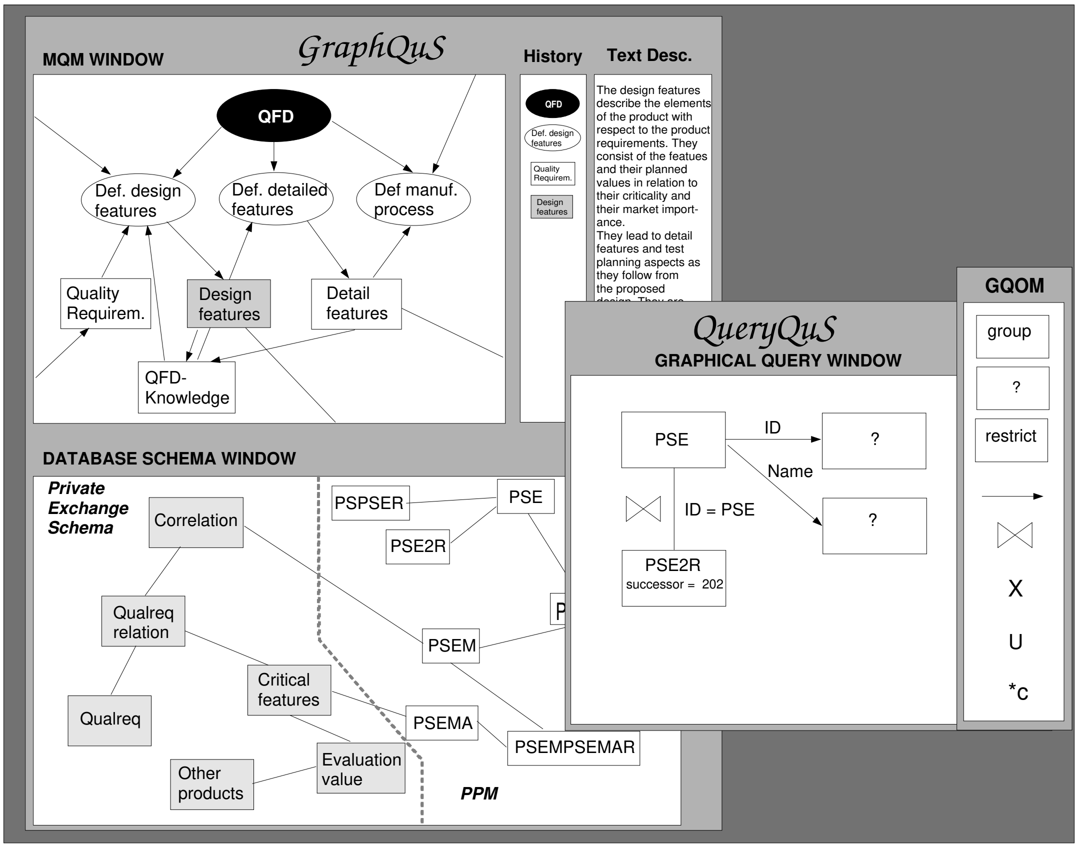

Fig. 11 The information search and access tool

can be supported by predefined mechanisms as those presented in Section 5.3. The other requests need to be supported by flexible request formulation tools which provide an acceptable compromise between tool flexibility and user friendliness.

At the system level as it is defined in the WibQuS project, one has to deal with the problems of classical query languages (hard to learn, difficult to use, not adapted to the specific domain), which are even more complicated in federated information systems. First, there are several query languages used in the system. Second, there is no central database schema known by all system members but a number of loosely coupled schemas whose structure and contents are not easily understandable for someone looking for information. The advantages of graphical database browsing and access mechanisms for central databases have been pinpointed in various publications [d'Atri, 1989; Wong and Kuo, 1982; Zhou, 1994]. They are at least as striking for federated systems if they exploit the knowledge stored in the model repository as we will show in this section.

As mentioned before, the model of quality management methods and all schema information are stored in the Quality Trader repository. This adds up to about thousand schema objects and searching the whole repository would be inefficient. Therefore, the browsing environment was structured by separating the method model from the database schema information and by dividing the schema information into PPM (general interest) and private schemas. This led to the browsing window layout shown in Figure 11.

Further, to adjust the search space to the tasks to be supported, we have developed a view concept that restricts the information provided at the windows [Peters *et al.*, 1995a]. Agent-oriented, method-oriented, product-oriented, or change-step oriented perspectives serve as filters to prevent 'getting lost in hyperspace'.

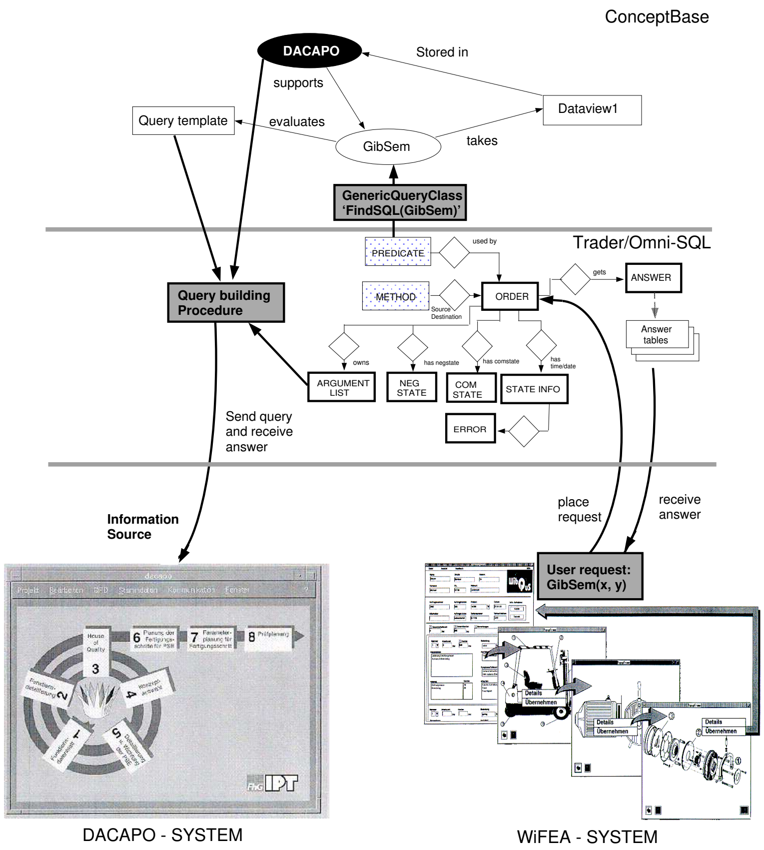

Fig. 12 The order evaluation process

In the upper window, the filtered conceptual model is shown; an exchangeable palette of graphical types facilitates user understanding. The lower window shows the corresponding part of the relational schema. The user can graphically formulate relational queries which are automatically augmented with the applied filter definitions; location transparency is provided by the quality trader.

5.3 Supporting Planned Information Flows

Units in a market-like organization interact in customer-supplier relationships. Workflows in WibQuS are therefore modeled in *request-commit-perform-evaluate cycles* [Medina-Mora *et al.*, 1992] which map task delegation between agents (Figure 5). The quality trader encodes the necessary data structures in a special relational schema, shown as an Entity-Relationship diagram in the middle of Figure 12. Active communication tasks record tuples concerning their (order, argument list) and status (negotiation, commitment, error). The task description can be just a specification for humans, it can be an executable program, or it can be a parameterized database query.

Figure 12 illustrates the latter case for an interaction between the WiFEA service support tool and the DACAPO quality planning tool in WibQuS. The service person has used her tool to zoom into the details of a forklift to be serviced, and detected some problem. She now wants to know if this problem is a known bug and therefore invokes the standard task `GibSem` which returns detailed information about a structure element in the product and process model. The task is placed in the `order` relation and abstracted to a ConceptBase query class which was defined in the conceptual model when fixing `GibSem` as a standard information flow.

In detail, the process of order evaluation is the following : Placing the **predicate** and the **argument list** in the **order** relation and setting the value of the **comstate** relation to 1 initiates the order evaluation process by the Quality Trader based on triggers defined on the **comstate** relation:

1. **comstate** = 1: The Quality Trader checks the **comstate** relation of every system frequently and starts the query evaluation procedure.
2. The Quality Trader looks up the query represented by the **predicate**. This is done by evaluating the GenericQueryClass 'findSQL(**predicate**)' on the repository. Each predicate is represented as an instance of **Task** at repository level 3.
3. The relations which are involved into the query are related to the systems in which they are stored. The Quality Trader finds those information source(s) using the **stored in** attribute added to the structure of the repository.
4. The query variables in the predefined **query template** are instantiated with the **argument list** and the locations found in step 3.
5. OmniSQL transfers the query into the database dialect(s) of the source(s). **Comstate** = 2
6. Execution of the (distributed) query.
7. The Quality Trader collects the answer(s) and transforms it into the dialect of the requesting system. **Comstate** = 3
8. The answer table is sent to the requesting system. **Comstate** = 4
9. The Quality Trader documents the querying process for statistical purposes.

ConceptBase now determines that the owner of this information is the quality planning subsystem. The corresponding SQL query is constructed in Omni-SQL and sent to DACAPO's Oracle database. After negotiation, the query is executed there and the resulting tables are made available to the WiFEA database (implemented using Informix). A monitor window of the Quality Trader allows to control this process from each involved node.

6 Summary and Conclusion

The WibQuS experiences show that many tasks involved in the planning and operation of federated organizations with cooperative information systems can be coordinated by user-definable and repository-supported meta models. Promising first commercial experiences have been gained with several parts of the overall approach though the full combination of techniques in the WibQuS prototype has only been tested with relatively small example data sets.

The meta modeling technology must be sufficiently formal to enable the largely automatic generation of the distributed coordination software. The not-too-complicated semantics of Telos in ConceptBase proved a major plus in this regard. However, our results so far have been limited to the case of heterogeneous relational databases; the follow-up project FoQuS currently investigates the generalization to object-oriented target environments. Even for the relational case, attempts are underway to increase further the degree of methodological and automated support for the co-evolution of organization and system model. Finally, our simulation approach has focused on the organizational impact of information flows alone; in future work, we intend to link this with a system-performance oriented distributed database design environment being developed in our group.

References

[Abdel-Hamid and Madnick, 1991]
T. Abdel-Hamid and S. Madnick. *Software Project Dynamics*. Prentice Hall, Englewood Cliffs, NJ, 1991.

[Adler and Clark, 1991]
P. Adler and K. Clark. Behind the Learning Curve: A Sketch of the Learning Process. *Management Science 37*, 3 (1991), pp. 267 – 281.

[Akao, 1990]
Y. Akao. *Quality Function Deployment: Integrating Customer Requirements into Product Design*. Productivity Press, Cambridge, 1990.

[Borgida, 1995]
A. Borgida. Description Logics in Data Management. *IEEE Transactions on Knowledge and Data Engineering 7*, 5 (1995), pp. 671 – 682.

[Brodie and Ceri, 1992]
M. Brodie and S. Ceri. On Intelligent and Cooperative Information Systems: A workshop summary. *Int. Jour. on Intelligent and Cooperative Information Systems 1*, 2 (1992), pp. 249–290.

[Burgelman, 1988]
R. Burgelman. Strategy Making as a Social Learning Process: The Case of Internal Corporate Venturing. *Interfaces 18*, 3 (1988), pp. 74 – 85.

[Carroll *et al.*, 1991]
J. Carroll, W. Kellogg and M. Rosson. The Task-Artifact Cycle. In J. Carroll (Ed.), *Designing Interaction: Psychology at the Human-Computer Interface*. Cambridge, 1991.

[Chen *et al.*, 1992]
M. Chen, Y. Liou and E. Weber. Developing Intelligent Organizations: A Context-Based Approach to Individual and Organizational Effectiveness. *Journal of Organizational Computing 2*, 2 (1992), pp. 181 – 202.

[Conklin and Begemann, 1988]
J. Conklin and M. Begemann. gibis: A hypertext tool for exploratory policy discussion. *ACM Trans. on Office Information Systems*, 10 (1988), pp. 140 – 151.

[Cooprider and Victor, 1993]
J. Cooprider and K. Victor. The Contribution of Shared Knowledge to IS Group Performance. In *14 th Conf. on Information Systems*, pp. 285 – 297, Orlando, Flo., 1993.

[Daft and Lengel, 1986]
R. Daft and R. Lengel. Organizational Information Requirements, Media Richness and Structural Design. *Management Science 32*, 5 (1986), pp. 554 – 571.

[d'Atri, 1989]
A. d'Atri. From Browsing to Querying. *IEEE Database Engineering 8* (1989), pp. 110 –117.

[De Michelis *et al.*, 1996]
G. De Michelis, E. Dubois, M. Jarke, F. Matthes, J. Mylopoulos, K. Pohl, J. Schmidt, C. Woo and E. Yu. Cooperative Information Systems: A Manifesto. In M. Papazoglou (Ed.), *Cooperative Information Systems*, 1996.

[Deiters *et al.*, 1995]
W. Deiters, V. Gruhn and R. Striemer. The FUNSOFT Approach to Business Process Management. *Wirtschaftsinformatik 37*, 5 (1995), pp. 459 – 466.

[Easterbrook and Nuseibeh, 1995]
S. Easterbrook and B. Nuseibeh. Managing Inconsistencies in an Evolving Specification. In *2nd IEEE Intl. Symp. on Requirements Engineering*, pp. 48 – 56, York, England, 1995.

[Fine, 1986]
C. Fine. Quality Improvement and Learning in Production Systems. *Management Science 32*, 10 (1986), pp. 1301 – 1315.

[Fishwick, 1995]
P. Fishwick. *Simulation Model Design and Execution*. Prentice Hall, Englewood Cliffs, N.J., 1995.

[Forrester, 1961]
J. Forrester. *Industrial Dynamics*. MIT Press, Cambridge, Mass., 1961.

[Goldstein, 1993]
D. Goldstein. Computer–Based Data and Organizational Learning: The Importance of Managers' Stories. *Journal of Organizational Computing 3*, 4 (1993), pp. 417 – 441.

[Hammer, 1992]
D. K. Hammer. Lean Information Management: The Integrating Power of Information. In H. J. Pels and J. C. Wortmann (Eds.), *IFIP Transactions: Integration in Production Management Systems*, pp. 147–163. Elsevier Science Publishers B.V., 1992.

[Harmsen *et al.*, 1994]
F. Harmsen, S. Brinkkemper and H. Oei. Situational Method Engineering for Information System Project Approaches. In *Methods and Associated Tools for the Information Systems Life Cycle*, pp. 169 – 194. Elsevier Science B.V., 1994.

[Hauser and Clausing, 1988]
J. R. Hauser and D. Clausing. Wenn die Stimme des Kunden bis in die Produktion vordringen soll. *Harvard Manager*, 4 (1988), pp. 57–70.

[Heinzl and Srikanth, 1995]
A. Heinzl and R. Srikanth. Entwicklung der betrieblichen Informationsverarbeitung. *Wirtschaftsinformatik 37*, 1 (1995), pp. 10 – 17.

[Huhns *et al.*, 1993]
M. Huhns, N. Jacobs, T. Ksiezyk, W.-M. Shen, M. Singh and P. Cannata. Integrating Enterprise Information Models in Carnot. In *Proc. of the Int. Conf. on Intelligent and Cooperative Information Systems*, pp. 32 – 43, Rotterdam, 1993.

[ISO/IEC, 1990]
ISO/IEC. Information Technology – Information Resource Dictionary System (IRDS) - Standard ISO/IEC 10027. Technical report, ISO/IEC International Standard, 1990.

[Jarke and Ellis, 1993]

M. Jarke and C. Ellis. Distributed Cooperation in Integrated Information Systems. *Int. Jour. of Intelligent and Cooperative Information Systems 2*, 1 (1993), pp. 85 – 103.

[Jarke and Pohl, 1993]

M. Jarke and K. Pohl. Establishing Visions in Context: Towards a Model of Requirements Processes. In *14th Conf. on Information Systems*, Orlando, Flo., 1993.

[Jarke *et al.*, 1993]

M. Jarke, M. Jeusfeld and P. Szczurko. Three Aspects of Intelligent Cooperation in the Quality Life Cycle. *Int. Jour. of Intelligent and Cooperative Information Systems 2*, 4 (1993), pp. 355–374.

[Jarke *et al.*, 1995]

M. Jarke, R. Gallersdörfer, M. Jeusfeld, M. Staudt and S. Eherer. ConceptBase – A Deductive Object Base for Meta Data Management. *Journal of Intelligent Information Systems 4*, 2 (1995).

[Jarke *et al.*, 1996]

M. Jarke, M. Gebhardt, S. Jacobs and H. Nissen. Conflict Analysis Across Heterogeneous Viewpoints: Formalization and Visualization. In *29th Annual Hawaii Conf. on System Sciences (Vol. 3)*, pp. 199 – 208, Wailea, Hawaii, 1996.

[Jeusfeld and Jarke, 1995]

M. Jeusfeld and M. Jarke. Repository Structures for Evolving Federated Database Schemas. In *IFIP Working Conference on Models and Methodologies for Enterprise Integration*, Heron Island, Australia, 1995.

[Jeusfeld and Johnen, 1995]

M. Jeusfeld and U. Johnen. An Executable Meta Model for Re-Engineering of Database Schemas. *International Journal of Cooperative Information Systems 4*, 2 (1995), pp. 237 – 258.

[Klein and Lyytinen, 1992]

H. Klein and K. Lyytinen. Towards a New Understanding of Data Modeling . In C. Floyd, H. Zullighoven, R. Budde and R. Keil-Slawik (Eds.), *Software Development and Reality Construction*, pp. 203 – 219. Springer-Verlag, 1992.

[Köppe and Heid, 1989]

D. Köppe and W. Heid. Möglichkeiten und Grenzen der SPC. *Qualität und Zuverlässigkeit 34*, 12 (1989).

[Malone *et al.*, 1987]

T. Malone, J. Yates and R. Benjamin. Electronic Markets and Electronic Hierarchies. *Communications of the ACM 30*, 6 (1987), pp. 484 – 497.

[Medina-Mora *et al.*, 1992]

R. Medina-Mora, T. Winograd, R. Flores and C. Flores. The Action Workflow Approach to Workflow Management Technology. In *4th Intl. Conf. on Computer-Supported Cooperative Work*, pp. 281 – 288, Toronto, Canada, 1992.

[Mylopoulos *et al.*, 1990]

J. Mylopoulos, A. Borgida, M. Jarke and M. Koubarakis. Telos: a Language for Representing Knowledge about Information Systems. *ACM Transactions on Informations Systems 8*, 4 (1990), pp. 325–362.

[National Library of Medicine, 1994]
National Library of Medicine. Unified Medical Language System (5th edition). Technical report, US Department of Health and Human Services, 1994.

[Nissen *et al.*, 1996]
H. Nissen, M. Jeusfeld, M. Jarke, G. Zemanek and H. Huber. Managing Multiple Requirements Perspectives with Meta Models. *IEEE Software, March* (1996).

[Oberweis *et al.*, 1994]
A. Oberweis, G. Scherrer and W. Stucky. INCOME/STAR: Methodology and Tools for the Development of Distributed Information Systems. *Information Systems 19*, 8 (1994), pp. 643 – 660.

[Olle *et al.*, 1991]
T. Olle, J. Hagelstein, I. MacDonald, C. Rolland, H. Sol, F. van Assche and A. Verrijn-Stuart. *Information Systems Methodologies - A Framework for Understanding*. North-Holland, Amsterdam, 1991.

[Orlikowski and Robey, 1991]
W. Orlikowski and D. Robey. Information Technology and the Structuring of Organizations. *Information Systems Research 2* (1991), pp. 143 – 169.

[Papazoglou *et al.*, 1995]
M. Papazoglou, N. Russell and D. Edmond. A Semantic-Oriented Translation Protocol for Heterogeneous Federated Database Systems. Technical report, Queensland University of Technology, Australia, 1995.

[Pentland, 1994]
B. Pentland. Information Systems and Organizational Learning: The Social Epistemology of Organizational Knowledge Systems. Technical report, 1994.

[Peters and Jeusfeld, 1994]
P. Peters and M. Jeusfeld. Structuring Information Flow in Quality Management. In *Int. Conf. on Data and Knowledge Systems for Manufacturing and Engineering*, pp. 258 – 263, Hong Kong, 1994.

[Peters *et al.*, 1995a]
P. Peters, U. Löb and A. Rodriguez Pardo. A Task-Oriented Graphical Interface to Federated Databases. In *Proc. of 3rd Int. Conf. of the Int. Soc. for Decision Support Systems*, pp. 223 – 231, Hong Kong, 1995.

[Peters *et al.*, 1995b]
P. Peters, P. Szczurko, M. Jarke and M. Jeusfeld. A Federated Information System for Quality Management Processes. In *IFIP WG8.1 Working Conference on Information Systems for Decentralized Organizations*, pp. 100 – 117, Trondheim, Norway, 1995.

[Peters *et al.*, 1996]
P. Peters, M. Mandelbaum and M. Jarke. Simulation-Based Method Engineering in Federated Organizations. In *Method Engineering 96*, pp. 246–262, Atlanta, Georgia, 1996.

[Peters, 1996]
P. Peters. *Planning and Analysis of Information Flow in Quality Management*. PhD thesis, RWTH Aachen, 1996.

[Pfeifer *et al.*, 1995]

T. Pfeifer, R. Grob and P. Klonaris. Teamgestützte Erfassung von Erfahrungswissen zur Fehleranalyse. In *3. Deutsche Expertensystemtagung*, Kaiserslautern, 1995.

[Pfeifer (ed.), 1996]

T. Pfeifer (ed.). *Wissensbasierte System in der Qualitätssicherung - Verteiltes Wissen nutzbar machen (in German)*. Springer Verlag, Heidelberg, Berlin, New York, 1996.

[Pfeifer, 1993]

T. Pfeifer. *Quality Management (in German)*. Carl Hanser Verlag, München, 1993.

[Pohl, 1994]

K. Pohl. The Three Dimensions of Requirements Engineering: a Framework and its applications. *Information Systems 19*, 3 (1994), pp. 243 – 258.

[Quentin, 1989]

H. Quentin. Statistische Versuchsmethodik. *Qualität und Zuverlässigkeit 34*, 5 (1989), pp. 229 – 232.

[Ramesh and Sengupta, 1994]

B. Ramesh and K. Sengupta. Managing Cognitive and Mixed-Motive Conflicts in Concurrent Engineering. *Concurrent Engineering Research and Application 2*, 3 (1994).

[Reichwald, 1989]

R. Reichwald. Die Entwicklung der Arbeitsteilung unter dem Einfluss von Technologieeinsatz im Industriebetrieb. In W. Kirsch and A. Picot (Eds.), *Die Betriebswirtschaft im Spannungsfeld zwischen Generalisierung und Spezialisierung*, pp. 300 – 321. Gabler Verlag, 1989.

[Scheer, 1994]

A.-W. Scheer. *Business Process Reengineering - Reference Models for Industrial Business Processes*. Springer Verlag, Berlin, Heidelberg,..., 1994.

[Scott-Morton, 1994]

M. Scott-Morton. The 1990s research program: Implications for Management and the Emerging Organization. *Decision Support Systems 12*, 2 (1994), pp. 251–256.

[Senge, 1990]

P. Senge. *The Fifth Discipline: The Art and Practice of the Learning Organization*. Currency, New York, 1990.

[Staudt *et al.*, 1994]

M. Staudt, M. Jeusfeld and M. Jarke. Query by Rule, Class, and Concept. *Journal of Applied Intelligence 4*, 2 (1994).

[Tolvanen and Lyytinen, 1992]

J.-P. Tolvanen and K. Lyytinen. Flexible Method Adaptation in CASE Environments – The Metamodelling Appraoch. In Bjerknes et al. (Ed.), *Proc. of the 15th IRIS*, pp. 388 – 405, 1992.

[Vennix *et al.*, 1994]

J. Vennix, D. Andersen, G. Richardson and J. Rohrbaugh. Model Building for Group decision Support: Issues and Alternatives fir Knowledge Elicitation. In J. Morecroft and J. Sterman (Eds.), *Modeling for Learning Organizations*. Productivity Press, 1994.

[Wong and Kuo, 1982]
H. Wong and I. Kuo. GUIDE: Graphical User Interface for Database Exploration. In *Proc. of the 8th Very Large Databases Conference*, 1982.

[Zhou, 1994]
J. Zhou. GUI to Databases – Survey and Practice. *GI Datenbank Rundbrief 13* (1994), pp. 49–51.