Cooperative Information Systems Engineering

Matthias Jarke, Manfred A. Jeusfeld, Peter Peters, Peter Szczurko
Informatik V, RWTH Aachen
Ahornstr. 55, 52056 Aachen, Germany
e-mail: {jarke,jeusfeld,peters,szczurko}@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 Cooperative Information Systems and the WibQuS Project

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, both at the system level of user interface research and 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 [3, 10, 5]) which see 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 fields. On the

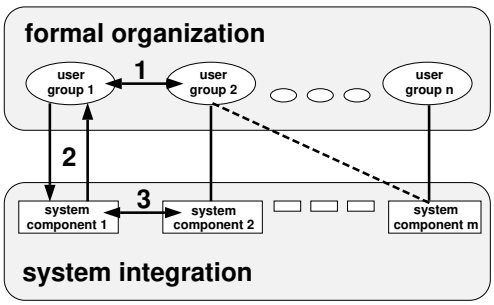

Fig. 1 Static structure of CIS

other, it changes conventional database wisdoms about the required information modeling, system implementation and integration technologies.

A CIS is a layered network of user and system components as sketched in figure 1. At the *system level*, we observe a trend towards componentization of software (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 [17]. Each of these 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, between system components, between user groups and system components, and – most importantly for us – between user groups *through* system components.

Why is all this happening? The answer – at both levels – is *reactiveness to change* [8, 30]. 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.

Total Quality Management (TQM) is one of several business philosophies that can be associated with this trend. It aims at continuous improvement of all processes in a company by emphasizing customer orientation throughout, relying on local ideas and improvement initiatives, propagated through the organization via cleverly designed feedback cycles. Therefore, TQM appears as a prototypical example of a (not necessarily 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] [13]. Figure 2 shows how the CIS levels are instantiated in TQM [28].

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

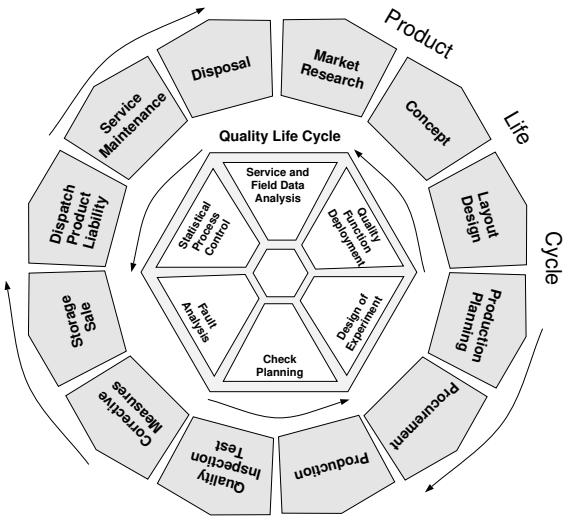

Fig. 2 The quality life cycle as a CIS

The outer circle of product life cycle stages corresponds to the organization part of figure 1. Traditionally, the stages have been associated with different departments or companies, each with their local IT solution. Due to business process orientation, the forward flow of task information 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 TQM is to introduce smaller feedback cycles that provide selected information about particular quality issues in the backward direction, thus enabling rapid and continuous improvement. This so-called *quality cycle* can be supported by additional local methods and software tools as shown in the inner part of figure 2, which corresponds to the system half of figure 1.

Broadcasting all quality information to everyone will quickly overwhelm and turn off users. The goal of our work in WibQuS was therefore to develop a set of methods and tools which would help companies *analyze possible information flows, estimate the impact of different information flow designs, and provide different degrees of operational information flow support for planned and unplanned information flows wrt. the chosen strategies*. Since quality problems are constantly changing, the interplay of all these tasks must be organized in a way that makes change as easy as possible.

Obviously, this was a tall order. We do not claim that we have solved all of the issues involved. To get anywhere, some hopefully not too unrealistic simplifications were made. Most importantly, the system integration layer was based on distributed relational databases. Thus, system interoperability problems were reduced to making the major RDB products interoperate[2]. Despite such limitations, we hope that some general lessons for CIS can be drawn from this experience.

[2] Subsystems did use SYBASE, ORACLE, and INFORMIX databases which turned out to be enough of a challenge to integrate in the easily reconfigurable way we wanted.

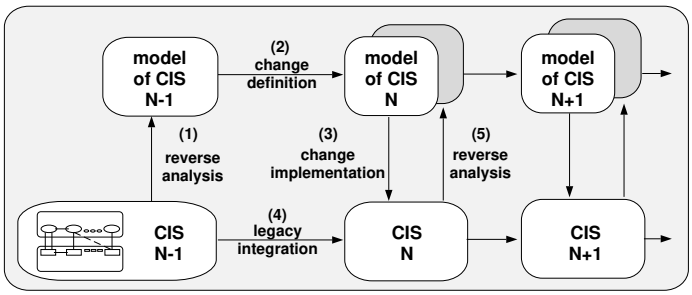

Fig. 3 A model-based change process for CIS

This paper presents an overview of some issues faced in CIS engineering, using WibQuS as an example. In section 2, we advocate a conceptual modeling approach which relies heavily on *user-definable meta models* to coordinate distributed change, and discuss its technical support by the ConceptBase meta data manager. Section 3 describes how it applies in the structural design and quantitative evaluation of organizational information flows while section 4 discusses the mapping to the system level. Section 5 lists some limitations and summarizes ongoing work.

2 Change Management through Conceptual Models

2.1 A Basic Change Process for CIS

Even if the organization is federated, the units 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?

Our solution is based on *conceptual modeling* (figure 3). It is a generalization of the task-artefact cycle for user interface design [4] : 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 how to manage the change cycle. One of the commercially most successful is the ARIS/SAP approach [29]. You start with an ARIS reference model for the business process and customize it to your specific situation. Each component of the reference model is linked to a parameterized SAP software component. Your software system is largely configured by parameter tuning.

Where feasible, this approach has been shown to reduce IS development costs by up to 80%. However, the feasibility obviously depends on the coverage of the reference model and, in particular, on the regularity of the 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.

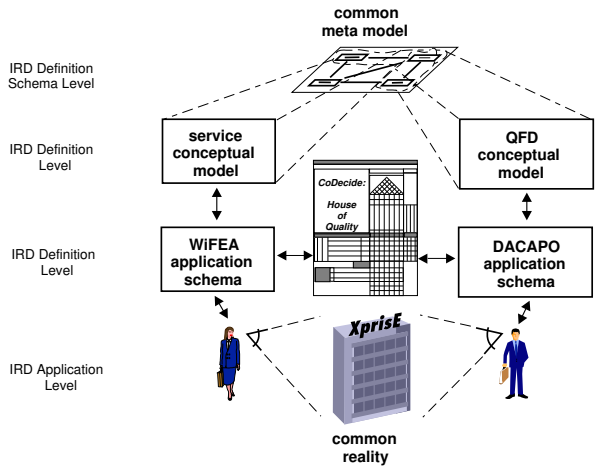

Fig. 4 Integrating different business views with meta models and CoDecide

Structuration theory [23] gives one possible explanation for these observations. 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 change 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 idea by defining requirements engineering as 'the process of establishing a vision in the existing technical, cognitive, and social context' [14].

2.2 Meta Modeling Support in ConceptBase

In WibQuS, *user-defined meta models* serve as the 'compass' which keeps the distributed change process aligned with the vision. 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 [12] – step (1) in figure 3. They abstract their observations of reality into local application models using their own conceptual notation. These notations are further abstracted into a common meta model which is defined jointly by the owners of the notations and then serves as a basis for communication about the coherence of different models.

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 [9]. 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 thus coordinate

subsystems at level *n*. The ConceptBase system developed in our group supports exactly this kind of multi-level meta modeling [11], and thus serves as an mediator between design and runtime CIS environment.

The idea of using meta models for large-scale model integration is not new. For example, in the medical domain, the Unified Medical Language System UMLS [20] 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 [29]. 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 [24].

A distinguishing feature of ConceptBase is that the development team can define *their own meta model*. ConceptBase enables this by the unlimited classification hierarchy in its *Telos* language [19]. 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. It 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 are parameterized and possibly materialized views defined by necessary and sufficient membership constraints similar to description logics [2]. 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[3].

CoDecide is a visualization toolkit for conflict analysis [12]. A CoDecide 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 so-called 'House of Quality' shown in the middle of figure 4 which offers a comprehensive visualization of the interrelationships within and between two parties, plus some external context such as versions or competing solutions.

---

[3] In a commercial 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 [21].

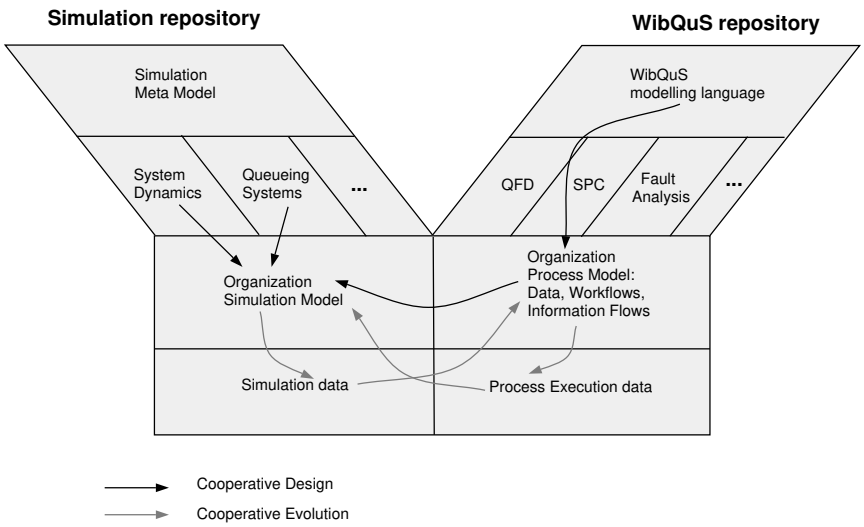

Fig. 5 CIS design and simulation process based on meta modeling

3 Designing and Simulating CIS Information Flows

In this section, we discuss the change management process at the level of organizational information flows. Here, meta models firstly help to link the analysis step (1) in figure 3 with step (2), and secondly to identify suitable design choices by simulation in the transition from step (2) to step (3).

Figure 5 shows the interplay of meta models for these tasks.. On the right branch, starting from the *WibQuS meta model*, organizational quality management process models are coordinated using reference method models at the second level. 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*. The interplay of simulation and actual process allows ongoing evolution of the distributed organizational knowledge base: the observation of real processes leads to changes of the simulation model which in turn indicates necessary changes of the organizational process structure. In the remainder of this section, we discuss the two branches in more detail.

3.1 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*. Only after some agreement on this purpose has been reached, distributed modeling should start. Much of the purpose can be coded in a *business process meta model* that defines the language in which modelers communicate about models and model interactions.

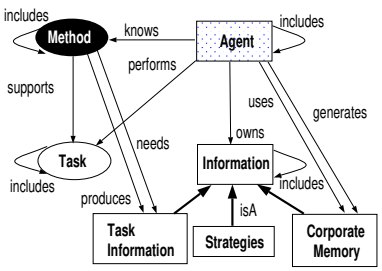

Fig. 6 The WibQuS meta model for CIS in TQM

Figure 6 shows the WibQuS meta model. It clearly reflects the paradigm of TQM, by linking agents and their interactions to tasks and methods. Each *task* in the business process is performed by a responsible *agent*. It is supported by *methods* that define the way-of-working for a specific task[4]. They are linked by three types of *information flows*:

1. *Task information* drives and monitors the operational business processes. It is provided and consumed by the methods along the process chain.
2. *Corporate memory* is important organizational knowledge about products and processes that results from accumulated execution and analysis of business processes. It is generated and used by the agent that performs the task.
3. The goal of a *strategy* is the definition of a common context according to which tasks are organized and information is interpreted. This is reflected in the meta model itself.

The development and application of this meta model in WibQuS comprised three phases. The *first phase* was the joint design of the meta model itself. These negotiations proved a useful investment because they gave the teams a shared ontology for future cooperation. In the *second phase*, the six engineering teams developed models of the different methods by parallel instantiation of the modeling language making use of the client-server architecture of ConceptBase. The *third phase*, partially overlapping with the second one, consisted of negotiations to ensure coherence between the models. In addition to the tools mentioned in section 2.2, a discussion structure based on the IBIS model represented subjective conflicts that could not be expressed in terms of inconsistency or incompleteness of the model itself. At the end, query classes were used to vote on the still unclear concepts.

The end result was a network of almost 600 Telos classes which constituted the first formal model of the quality cycle in manufacturing enterprises. Quantitative analyses of data flow and organizational learning, as well as mappings to the system level, were piggybacked on this conceptual infrastructure.

---

[4] Methods and tasks form an AND/OR decomposition structure: a task can be supported by one or several methods while a method consists of a partial order over subtasks.

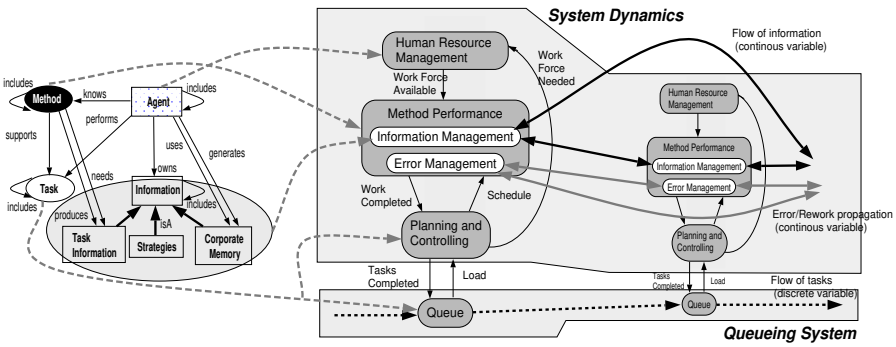

Fig. 7 Mapping of the conceptual to the simulation model

3.2 Analyzing Information Flows By Multi-Simulation

Each of the information flow categories must be analyzed according to different criteria and with different quantitative modeling techniques. 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.

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 [6; 22].

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 just related to the task workflows and cannot be measured by hard business variables like time and money, but by *the way they influence* the variables that produce those time-and-money effects. In his work on software project dynamics, Abdel-Hamid [1] showed that System Dynamics simulation is well-suited for such tasks. We generalized his model to the case of multiple interacting organizational units with explicit information flow management and error propagation models [25].

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 [7]. Our MultiSim environment generalizes this idea to a graphical simulation definition and execution environment [27].

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, 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 7).

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 integrated environment in a business process re-engineering project for a manufacturing company showed a surprisingly good match between model and reality [27].

4 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*. These choices can be *implemented* by linking certain views on the conceptual model to the system level, 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.

Three kinds of mappings are derived from the WibQuS meta model. The 'owns' link in figure 6 is mapped to ownership of database nodes. This partitioning is often given and must simply be recorded as a dependency link in the meta database. Section 4.1 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. *Planned information flows* are mapped to stored queries and workflow structures (section 4.2) which are more efficient to use but more difficult to change than the unplanned ones.

4.1 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 [16]. In figure 8, 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

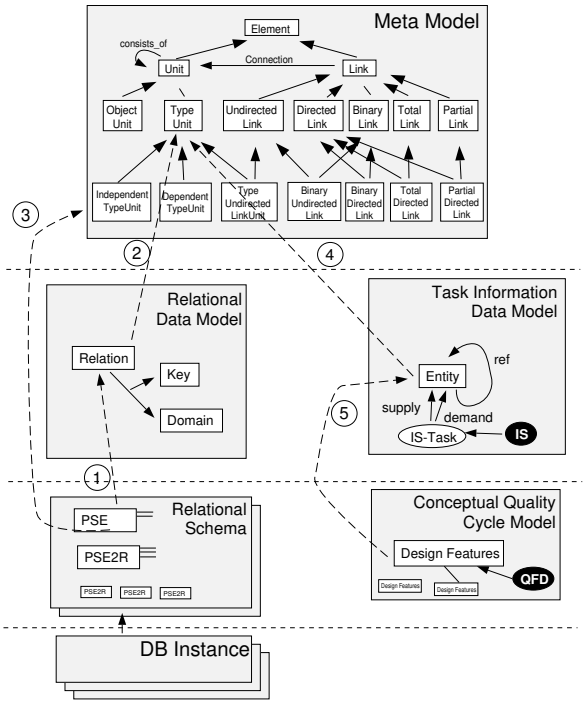

Fig. 8 Meta model constraining the mapping between concepts and relations

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.

CIS engineering uses neither of these directions throughout 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

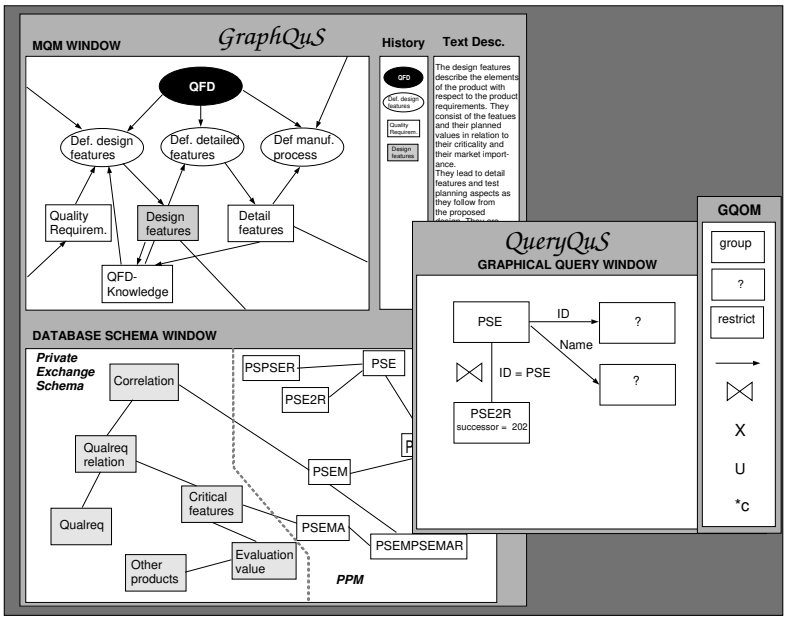

Fig. 9 The information search and access tool

the two models that helps in further system evolution [15], but also allows the end-user to view the distributed relational database through the conceptual model [26] in a two-level filter-browser (figure 9). Agent-oriented, method-oriented, product-oriented, or change-step oriented perspectives serve as filters to prevent 'getting lost in hyperspace'. 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.

4.2 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*[18] which map task delegation between agents (figure 6). The quality trader encodes the necessary data structures in a special relational schema, shown as an Entity—Relationship diagram in the middle of figure 10. 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 10 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

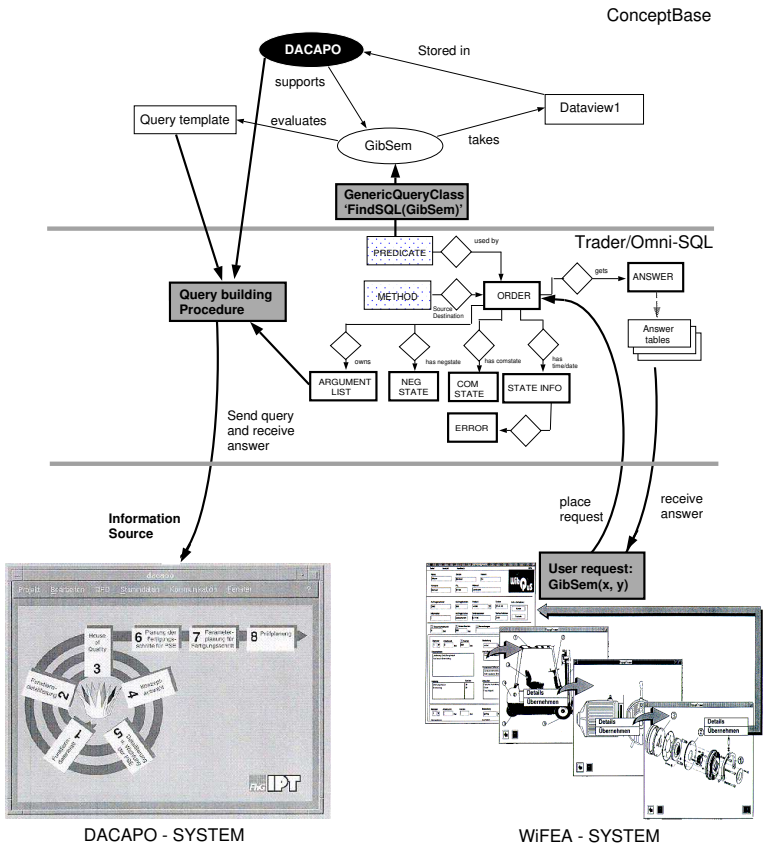

Fig. 10 The order evaluation process

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. 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.

5 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

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

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

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

[4] J.M. Carroll, W.A. Kellogg, and M.B. Rosson. The Task-Artifact Cycle. In J.M. Carroll, editor, *Designing Interaction: Psychology at the Human-Computer Interface*. Cambridge, 1991.

[5] 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 *4th Intl. Conf. on Cooperative Information Systems*, Brussels, Belgium, 1996.

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

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

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

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

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

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

[12] 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)*, pages 199 – 208, Wailea, Hawaii, 1996.

[13] 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):355–374, 1993.

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

[15] 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.

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

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

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

[19] 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):325–362, 1990.

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

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

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

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

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

[25] P. Peters. *Modeling Information Flow in Quality Management (forthcoming)*. PhD thesis, RWTH Aachen, 1996.

[26] 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*, pages 223 – 231, Hong Kong, 1995.

[27] P. Peters, M. Mandelbaum, and M. Jarke. Simulation-Based Method Engineering in Federated Organizations. In *submitted for Method Engineering 96*, Atlanta, Georgia, 1996.

[28] T. Pfeifer. *Quality Management (in German)*. Carl Hanser Verlag, Muenchen, 1993.

[29] A.-W. Scheer. *Wirtschaftsinformatik (Reference Models for Industrial Business Processes (in German)*. Springer Verlag, Berlin, Heidelberg,..., 1993.

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