Active Models for Cooperative Information Systems

Håvard D. Jørgensen[1,2] and John Krogstie[1,2]

[1] Norwegian University of Science and Technology, Department of Information and Computer Sciences, N-7491 Trondheim, Norway

[2] SINTEF Telecom and Informatics, Department of Distributed Information Systems, PO Box 124 Blindern, N-0314 Oslo, Norway

{hdj,jok}@informatics.sintef.no

Abstract. Active models constitute a general approach for increasing the flexibility of computerised information systems. Such models are available for manipulation by the users at runtime, and they influence the behaviour of the system. The usage context for active models thus differs from that of passive models used during systems specification and design. This paper discusses requirements and potentials for the use of active models, drawn from our experience with developing support for knowledge intensive project work. As a framework for this analysis we use a model quality framework. Our results show how active models face different requirements than passive models used for information system development. This difference should be taken into account when developing modelling languages, tools and techniques for utilising active models.

1 Introduction and Motivation

Active models constitute a general technique for increasing the flexibility of cooperative information systems (CIS). The CIS makes the models available to the users at runtime, and the behaviour of the system is partly controlled by the models. By altering the models the users can thus modify the behaviour of the system to fit their local needs and adapt to changes in the environment. Therefore, active models are more tightly interwoven in the fabric of everyday organisational life than models used in analysis, requirement specification, and design. This paper explores the relationship between active models and the social environment where they are created, talked about, used and manipulated, with particular emphasis on evolving, knowledge-intensive cooperation.

Workflow [5], document classification and retrieval [4], product data management [10], cooperation support, and knowledge based and reflective systems [8] are areas where active models have been applied. In knowledge-based systems, the operational logic is stored as data rather than programmed in software, while reflective systems expose representations of their own logic to their users, and allow modification of this logic. Active models combine this behavioural reflection [7] with user interaction through conceptual modelling, model interpretation and activation [6]. Several trends in cooperative information systems engineering illustrate the increasing importance of active models, e.g. model-based assembly and customisation of standard components into solutions for a particular organisation [12]. The use of enterprise models to structure corporate intranets [25], is another example.

Change management is the main challenge of cooperative information systems [7]. Although CISs include active model technology like workflow management and dynamic ontologies, the role of models in change management is often seen in relation to systems development and integration activities [15], not to innovation in use [28].

The focus is on the role of systems developers and modelling experts, not on the users, the real agents of change. A fresh look at the role of models to manage and change system operation can complement this with a more user-oriented perspective. Although models are used to directly control the behaviour of many cooperative information systems, the distinct modelling requirements of the operations phase as opposed to the development phase, have received little attention. This paper aims to remedy that situation, but also to highlight the benefits of linking conceptual modelling to behaviour reflection, and to report on our experiences in this field.

The Structure of this Paper
Section 2 describes core characteristics of active models, using process models in workflow management systems as an example. Differences between active models and models used during IS development are then investigated with the help of a model quality framework (section 3). Challenges regarding change management, effective communication, learning and knowledge sharing are outlined. Section 4 outlines our experiences with active models in developing support for knowledge intensive inter-organisational cooperation, focusing on the innovative aspects of our active view on modelling. Related work in this area is briefly discussed in section 5. Finally we offer some conclusions and directions for further research.

2 Active Models

Models are generally defined as explicit representations of some portions of reality as perceived by some actor [29]. In information systems *development*, conceptual models have long been used to analyse the problem domain and to capture and structure user requirements. Most approaches focus on *referential* aspects, on the relationship between model elements and the real world objects that they represent, on models mimicking the real world. Individual, social and situational aspects of model *usage* have had difficulties in influencing mainstream information systems engineering [13].

At the same time, models are also being used actively during operation of some information systems. The aim of using models at runtime is often to increase the flexibility and adaptability of the systems, enabling them to better meet local user needs and changing environments. Supporting learning and knowledge management is another main motivation: "*The key criterion of a system's usability is the extent to which it supports the potential for people who work with it to understand it, to learn and to make changes*" [1].

2.1 Active and Passive Models

Compared to traditional models used during the early phases of systems development, active models are faced with a different set of requirements. A wider range of people are actively involved in modelling, amplifying the social, psychological and organisational aspects. This fact has received little attention from the various communities where active modelling methods are researched and developed. Often solutions developed for systems development are simply transferred to active modelling. The large interest in using the software engineering languages of UML (Unified Modelling Language) for enterprise modelling and workflow [14, 22, 24] illustrates this trend. The widespread

use of Petri Nets for workflow modelling is another example [33]. A look at the distinct features of active models is thus long overdue.

2.2 Characteristics of Active Models

What does it mean that a model is active? First of all, the representation must be *available* to the users of the information system at runtime. Second, the model must *influence the behaviour* of the computerised system. Third, the model must be *dynamic*, users must be supported in changing the model to fit their local reality, enabling tailoring of the system's behaviour.

Let us illustrate this with the example of workflow management systems [31]. Such systems support the coordination of work in business processes. The models used are process models. Process models are especially important because they represent ways of working, organisational routines and theories of action [2]. They reflect the tasks that are part of the process, their interdependencies and the resources that are applied to perform them. Resources include personnel, information and tools.

All workflow systems include an *enactment service* that interprets the model as work progresses, filling the users to-do-lists with new tasks when the model says they are ready to be performed. Most workflow systems also include a process definition component that enables users to build models. In *static* (production) workflow, process models are built by experts and not allowed to change while the process is being executed. This solution only works for well-understood, routine processes. Consequently, *adaptive* workflow is an important research area [5, 16, 33]. Here models are allowed to change, and change will affect ongoing processes. In *emergent* workflow [16], modelling is viewed as an integral part of the work, performed by the process participants (although a baseline model usually serves as a starting point). The focus is unique cases, especially knowledge intensive projects. Workflow management systems thus illustrate the primary characteristics of active models:

- Models are available at runtime, in the process definition component,
- Models influence system behaviour, through the enactment component,
- Models are dynamic, to a varying degree in adaptive and emergent workflow.
- Active models increase flexibility by letting users define control sequences that were previously programmed in software.

3 Requirements for Active Models

The model quality framework of Krogstie, Sindre and Lindland [19, 21] enables us to illuminate differences between requirements for active models and requirements met by conventional systems development models. This framework is closely linked to linguistic and semiotic theory, and based on a social constructivist view, recognising that models are usually created in a dialogue between the people involved. The main concepts of the framework are shown in Figure 1.

Quality is determined by the correspondence between statements belonging to the following sets:

- L, the language extension, the set of all statements that can be made in the modelling language.
- D, the domain, the set of all possible statements about the situation at hand.

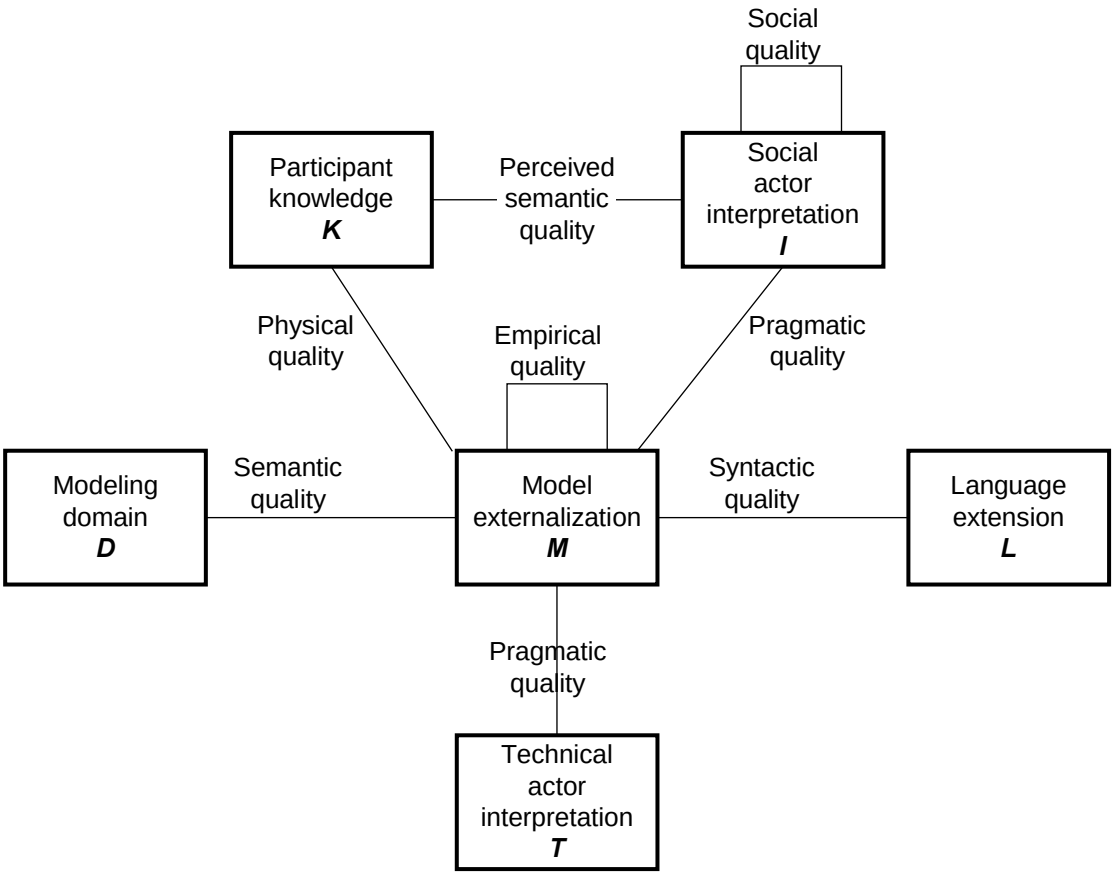

Fig. 1. Framework for analysing the quality of models.

- M, the externalised model, the set of all statements in someone's model of part of the perceived reality.
- K, the relevant explicit knowledge of the modeller.
- I, the social actor interpretation, the set of statements that the audience perceives the model to contain.
- T, the technical actor interpretation, the model as interpreted by information systems.

The main quality types are defined as relationships between these sets:

- Physical quality, involving
  - Externalisation, that the relevant explicit knowledge of the participant is reflected in the model.
  - Internalisability, that the model is available for the persons involved to make sense of it.
- Empirical quality deals with error frequencies when a model is read or written by different users, coding, and ergonomics of computer-human interaction.
- Syntactic quality is the match between the model and the language in which the model is written.
- Semantic quality is the correspondence between the model and the domain. The framework contains two semantic goals:
- Validity, that all statements in the model are correct and relevant to the problem

- Completeness, that the model contains all statements about the domain which are correct and relevant.
- Perceived semantic quality is the match between the participants' interpretation of a model and his or her current explicit knowledge. As the domain D cannot be completely known, semantic quality can only be tested indirectly through the participants' knowledge.
- Pragmatic quality is the correspondence between the model and the audience's interpretation of it.
- Social quality: The goal defined for social quality is agreement among different participants' interpretations. Social quality affects communication among participants about the contents of the model.

3.1 Physical Quality and User Participation

Active models and traditional system development models differ with respect to a number of components in the quality framework. The modellers include not only software professionals, but also all ordinary end users interacting with the system and its active models. In systems development and analysis, users are often seen more as sources of information than as model builders, and even participatory design only involves a limited number of user representatives, not the whole user community immersed in their day-to-day activities. Hence, stronger requirements for physical quality are likely, both because end users lack experience with conceptual modelling, and one will want to update the models more frequently due to learning. The possibility to rapidly update the model (and thus the system) is one of the main advantages with this approach. In systems engineering, new approaches like incremental development attempt to shorten these learning cycles, but they are still hampered with a long time-span from learning to model-change compared to the what can be achieved with active models. Also, users are likely to have more in-depth knowledge K of their domain D than software developers who have seldom taken part in the practice of the domain. Consequently, the potential for high semantic quality is greater. To achieve this potential the modelling languages need to be easy to use, both in the sense that users know the languages well, and that they are able to externalise their knowledge using the languages. Hence, simplicity, adaptability and user-orientedness [16] of the modelling language are even more crucial for active models than for their passive counterparts.

3.2 Pragmatic Quality and Model Activation

The core of active models is how models are *activated*. Activation implies *interpretation* of the model and corresponding *action* by either the social or the technical actors [6]. Hence pragmatic quality is paramount. Technical pragmatic quality demands complete formal models (i.e. models following the formal syntax and formal operational semantics of the language), while the social pragmatic quality of the models and the cognitive economy of externalisation (K→M) often demands more flexible, informal approaches. The *interaction framework* has been proposed to address this challenge [29, 30]. Accordingly, activation of models through interaction between the system and its users has been pointed to as a third way, combining the strengths of technical and user-performed activation [6]. In emergent workflow [16], *interactive enactment* has enabled simple and flexible models where users need not resolve incompleteness until the time when the flow of work reaches the incompletely specified

parts. Hence, the users and the computerised information system cooperate in bringing the process forward. The system makes decisions about what to do next when the model is conclusive; else it is up to the users. Interactive model interpretation enables models with user-controlled levels of formality, detail and preciseness, bridging the gap between theory and practice

3.3 Semantic Quality through Model-Guided Action

The second part of activation is *action* based on the interpretation of the model. Action often involves changing the domain D, and should thus be reflected in the model M. If an action is supported by a CIS with active models, it can be automatically captured, increasing the semantic quality of the active model without extra work for the users. The gap between real and modelled processes has been highlighted as a major inhibiting factor of process support systems [3] and organisational learning [2] alike. Thus active models has a great potential for flexibly supporting knowledge management and process improvement.

This immediate nature of active models, stemming from the interaction of the real world domain and active model, can also enhance the social pragmatic quality of the models. When both the real world and the model that reflects it are available and adaptable for the users, the connections between them are easier to understand. Simulation and training methods can be developed that utilises this connection. Zuboff's study of industrial control rooms [32] show great benefits for users that are able to work both with the conceptual tools of the computer and the physical environment of the factory. But the study also highlights the pitfalls of systems that isolate users inside the controls rooms' modelled world without understanding what really goes on in the plant.

3.4 Social Quality, Communication and Local Modifications

In systems development, agreement among participants about the requirements is crucial since they form the basis for a lot of detailed technical work that cannot easily be redone. Active models have a direct connection to the system and the environment it represents, so users have access also to the domain when negotiating a shared understanding. Social quality is thus perhaps not as important when assumptions readily can be tested immediately in the real world.

If an active model is to be reused in another setting, agreement on semantics is more important. Social quality of active models influences the processes of negotiating meaning and domesticating reusable model fragments into the local situation and work practice [28]. In these processes, the ability to represent conflicting interpretations and make local modifications, is just as important as the ability to represent agreement (the end result) in an unambiguous way. Also, since people learn through their work and use of the models, agreement is likely to be partial and temporary.

In conventional systems, change is regarded as exceptions to the predefined rules. Exceptions can be classified as either expected or unexpected. Expected exceptions can be caused by temporal events or by the actions of external agents. Such an exception is directly related to the system's domain and can be represented as part of the model, even if it represents a deviation from the normal or desired course of events. On the other hand, excessive amounts of detailed exception handling built into the model will make it complicated, hard to comprehend and hard to change. An unexpected exception is caused by a change in the system's domain that could not have been anticipated at

modelling time. For conventional systems without active models, modelling time coincides with systems development. Active models enable exceptions to be built in right up to the time when the models are applied. Hence a greater number of exceptions can be moved from the unexpected to the expected category, due to the learning of the participants. Active models also provide means for handling unexpected exceptions, at two levels:

1. *Local* modifications made to the running model instance that encounters the exception. This means that the generic definition is left unchanged, and other instances will execute according to that definition.
2. *General* and long-lasting modifications can be made to the type definition, so that also future instances will execute according to the new definition.

Of course, not all systems allow both local and general modifications. Workflow management systems seldom allow local modifications, they often hardwire the term 'process definition' to the class level [31]. But in an active modelling perspective, exception handling gives rise to research issues in determining when changes made to a running instance should be migrated to the generic level.

3.5 Technical Pragmatic Quality and Enterprise Integration

In the EXTERNAL project [9], our aim is to facilitate inter-organisational cooperation in knowledge intensive industries. When such cooperation moves beyond the buying and selling of well-defined goods and services, there is a need for a flexible (web) infrastructure that supports not only information exchange, but also knowledge creation, sharing, and utilisation. Cooperation is often knowledge-based, in the sense that the partners contribute with unique and complimentary competence vital for the success of the joint enterprise. We must therefore be able to form effective teams across organisational boundaries and local cultures. Also, the ability of each organisation to learn from the experiences of the joint enterprise is crucial for long term success. Such inter-organisational cooperation is difficult to support with traditional system development because of the transient and situated nature of each project. For one organisation, the number of potential partners is huge, and with each partner we may be involved in only a few projects. Consequently, integrating the infrastructures through traditional systems development practices is seldom economically viable. Standardisation is often proposed in these cases, but standards require that the domain is well understood and established. This is seldom the case for knowledge intensive work processes. Consequently, we need a more flexible approach, one that allows shared understanding and semantic interoperability to *emerge* from the cooperation, rather than being a prerequisite for cooperation. Dynamic ontologies [18] and user-definable metamodels [15] are examples of active models suitable for this kind of enterprise integration.

4 Experiences with Active Models

Our interest in the distinct characteristics of active models has arisen from our experiences in developing cooperative information systems for knowledge intensive project work, organisational learning and knowledge management [6]. The paper format does not allow for a thorough account of these experiences, but we summarise some of

the main areas where our active model solutions differ from conventional systems development models. It is also the case that these techniques are seldom used in current approaches to active models, e.g. in workflow management systems. The WORKWARE task manager and emergent workflow system [16], with integrated information management [25] and awareness services [26], is a system where the process models articulated by end users are activated in a number of ways. In the course of this work we have developed and implemented a number of ways to cope with the distinct requirements of active models:

- Local modifications are supported by *instance modelling* [16]. This limits the scope of a change to the local situation, removing much of the complexity that has prevented modelling by end users at the class level. It also is a prerequisite for establishing an immediate connection between the domain and the active model (semantic quality), as discussed above.

- *Model interpretation should be interactive*, combining the capabilities of the system to automate predefined parts and the users to handle incompletely specified parts of the model [16]. This enables the total system to handle models with varying, user-controllable degrees of specificity, where structure can emerge as the users' understanding of the domain increases (cf. pragmatic quality).

- The system architecture can benefit from *integrating multiple model-activator components*. In addition to workflow enactment, WORKWARE uses the relationships between tasks and documents for information management and the process flow structure for awareness mediation. The enactment engine and the awareness server both activate the process model to support coordination. The engine activates the predefined flows of work, and involves the users when incompleteness in the models prevents automatic reasoning. The awareness server informs users about actions in tasks that are related to their tasks, enabling coordination through mutual adjustment. This is especially important in models with little predefined structure.

- *Model interpretation should be contextual*, because semantics of data are often hidden in their context [18], and active models are developed in their usage context. Contextual interpretation implies that the meaning of a model element should depend on the current situation. In WORKWARE, the meaning of process flows depends on the states of the tasks it relates. If the source task is completed before the target is started, then the enactment engine activates the target. But if both are active in parallel, the awareness server takes control and uses the relationship as a channel for awareness mediation. Interactive and contextual interpretation of model elements enables simpler modelling languages, increasing the externalisability aspects of physical model quality, as well as the social pragmatic quality. As the previous analysis pointed out, these qualities are especially important for active models.

- *Model interpretation should be holistic*. The meaning of each element should not be solely defined by the element itself; it should also depend on the surrounding model. While Petri Net enactment [33] is based on the movement of tokens, WORKWAREs enactment is governed by the interplay of users' interaction with different model elements (tasks, flows etc.). Users are allowed to manipulate the states of each element, hence the model acts as a system of autonomous components, allowing a richer (non-deterministic) set of behaviours to emerge and be reflected in the model [30].

- For externalisation and user participation, it is important to keep the core languages *simple*. Increased expressiveness and detailed semantics may be harmful, especially if the model is expected to change and be refined as the work progresses. We need to look beyond the referential aspects of models, and see how they are used [13]. The trade-off between simplicity and expressiveness is evident in the way units of work are modelled. Traditionally, most systems have several terms for this, like process, activity, task, work item etc. These terms are used for discriminating among atomic (task) and composite (process) or generic (activity) and specific (workitem) model elements, e.g. in the Workflow Management Coalition's standards [31]. In an active model these properties will vary during the course of the model's lifecycle, e.g. an atomic task might be decomposed later by the people responsible for performing it, as part of their detailed planning. Consequently, WORKWARE does not separate between processes and tasks, but use one concept for a unit of work at any level of granularity, making the language simpler and more flexible.

More work is needed to verify the extent of application of these techniques. In the EXTERNAL project [9], three different case studies are currently underway; one in business consulting, the second a network of small and medium-sized enterprises, and the third an international research project (EXTERNAL itself).

5 Related Work

As pointed out in the introduction, the literature on active models is sparse. Some work on workflow management [5], cooperative information systems [7] and knowledge management [6] touch on related topics, like the role of the information system as a mediator of knowledge, but seldom with emphasis or fresh perspectives on the role of the models. The work of Greenwood et al. [11] is a notable exception. They present a notion of active models similar to the one given here, but with more emphasis on the active relationship between the domain and the model, and less on externalisation. Their work on a methodology for active software process support is complemented here by an in-depth look at the nature of active models and the requirements they face. We also add proposals for model activation and interpretation mechanisms, derived from our experience with the WORKWARE prototype.

Wegner and Goldin [30] have developed an *interaction framework for modelling* where models are interpreted in an interactive process involving multiple autonomous users and computerised components. This theoretical work provides a foundation for our research into interactive workflow enactment mechanisms, and connects active modelling to a wider paradigm for IS engineering.

Holm and Karlgren [13] discuss the relationship between modelling perspectives and theories of meaning, arguing the referential aspects (expressiveness) has over-shadowed individual, social, situational and organisational aspects of *models in use*. Their framework is similar in scope to the one used here, but not as detailed. Though they do not explicitly discuss the use of models during system operation, they offer insights that are even more relevant for active models than during systems development.

Jarke et al. [15] discuss the application of conceptual modelling to change management in cooperative information systems, but from a systems development perspective. They combine user-definable metamodels with negotiation support to integrate the steps of an incremental systems development process. Models are used

actively to support the systems development process, but not to the same extent during normal system operation.

Enterprise ontologies have been proposed as a solution to the communication problems arising from different interpretative frameworks [27]. This approach is based on conventional notions of model interpretation, where the technical actor interpretation is fully automated. Active models are also directed towards ongoing modelling, model interpretation, and activation by the end users. With active models, the objective is not to maximise the expressiveness and inference power of the computerised information system, but to increase the effectiveness of the whole system including the users. Another popular solution for enterprise integration is middleware frameworks like OMG's CORBA. The recent shift in the focus of OMG to modelling (UML), model driven architectures, meta-objects, business object frameworks and workflow management facilities [23] indicate an interest in model-driven enterprise integration also from the technical side. Whether these software engineering approaches are directly transferable to active modelling remains to be investigated thoroughly, and the work presented here gives some guidelines as to potential problems. UML has previously been analysed according to the model quality framework by one of the authors [20].

6 Conclusions and Further Work

This paper has pointed to active models, created and used during the operation of information systems, as a general methodology for increasing the flexibility of cooperative systems. Differences between these active models and the passive models conventionally used for systems development have been described, showing both the difficulties and the potentials of the active models approach. Our experiences in using active process models for flexible cooperation in knowledge intensive projects have been outlined, highlighting the need for innovative approaches and perspectives on the role of models.

Further experimentation with this approach as a basis for modelling languages, infrastructure, and a modelling methodology is the major challenge for the remainder of the EXTERNAL project [9]. Our ongoing research also targets the problem of generalising local modifications and innovations, packaging them in a manner suitable for reuse, thus enabling knowledge management and learning from practice [17].

References

1. Adler, P. S. and Winograd, T. A. *Usability - Turning Technologies into Tools*. New York, USA: Oxford University Press, (1992).
2. Argyris, C. and Schön, D. *Organizational Learning: A Theory of Action Perspective*. Reading, MA, USA: Addison Wesley, (1978).
3. Bandinelli, S., Fuggetta, A., Lavazza, L., Loi, M. and Picco, G. P. *Modeling and Improving an Industrial Software Process*, IEEE Transactions on Software Engineering, vol. 21, no. 5, (1995).
4. Brasethvik, T. and Gulla, J. A. *Semantically accessing documents using conceptual model descriptions*, Workshop on Web and Conceptual Modelling, ER 99, Paris, France, (1999).
5. Carlsen, S. *Action Port Model: A Mixed Paradigm Conceptual Workflow Modeling Language*, 3rd IFCIS Conference on Cooperative Information Systems, New York, (1998).

6. Carlsen, S., Johnsen, S. G., Jørgensen, H. D., Coll, G. J., Mæhle, Å., Carlsen, A. and Hatling, M. *Knowledge re-activation mediated through knowledge carriers*, MICT'99, Copenhagen, Denmark, (1999).
7. De Michelis, G., Dubois, E., Jarke, M., Matthes, F., Mylopoulos, J., Pohl, K., Schmidt, J., Woo, C. and Yu, E. *Cooperative Informations Systems: A Manifesto*, in *Cooperative Information Systems*, M. Papazoglou and G. Schlageter, Eds.: Academic Press, (1998).
8. Dourish, P. *Developing a Reflective Model of Collaborative Systems*, ACM Transactions on Computer-Human Interaction, vol. 2, no. 1, (1995).
9. EXTERNAL *EXTERNAL - Extended Enterprise Resources, Networks And Learning*, EU Project, IST-1999-10091, *New Methods of Work and Electronic Commerce, Dynamic Networked Organisations*. DNV, GMD-IPSI, Zeus, METIS, SINTEF, (2000-2002).
10. Farshchian, B. A. *Gossip: An Awareness Engine for Increasing Product Awareness in Distributed Development Projects*, in *Advanced Information Systems Engineering - Proceedings of CAiSE 2000*, Stockholm, Sweden, Springer LNCS 1789, (2000).
11. Greenwood, R. M., Robertson, I., Snowdon, R. A. and Warboys, B. C. *Active Models in Business*, 5th Conference on Business Information Technology CBIT '95, (1995).
12. Gulla, J. A. and Brasethvik, T. *On the Challenges of Business Modelling in Large-Scale Reengineering Projects*, Proceedings of the 4th International Conference on Requirements Engineering, Schaumburg, Illinois, USA, (2000).
13. Holm, P. and Karlgren, K. *Theories of Meaning and Different Perspectives on Information Systems*, Proceedings of Third International Working Conference on Information System Concepts - ISCO3, Marburg, Germany, (1995).
14. Hruby, P. *Structuring Specification of Business Systems with UML (with an Emphasis on Workflow Systems)*, OOPSLA'98 Business Object Workshop IV, (1998).
15. Jarke, M., Peters, P. and Jeusfeld, M. A. *Model-Driven Planning and Design of Cooperative Information Systems*, in *Cooperative Information Systems*, M. Papazoglou and G. Schlageter, Eds.: Academic Press, (1998).
16. Jørgensen, H. D. and Carlsen, S. *Emergent Workflow: Integrated Planning and Performance of Process Instances*, Workflow Management '99, Münster, Germany, (1999).
17. Jørgensen, H. D. and Carlsen, S. *Writings in Process Knowledge Management*, SINTEF Telecom and Informatics, Oslo, Norway STF40 A00011, (2000).
18. Kahng, J. and McLeod, D. *Dynamic Classificational Ontologies: Mediation of Information Sharing in Cooperative Federated Database Systems*, in *Cooperative Information Systems*, M. Papazoglou and G. Schlageter, Eds.: Academic Press, (1998).
19. Krogstie, J. *Conceptual Modeling for Computerized Information Systems Support in Organizations*, PhD-thesis, University of Trondheim, The Norwegian Institute of Technology, Trondheim, Norway, (1995).
20. Krogstie, J. *Using a Semiotic Framework for the Development of Models of High Quality*, in *Unified Modelling Language: Systems Analysis, design, and Development Issues*, K. Siau and T. Halpin, Eds.: IDEA group, (2000).
21. Krogstie, J., Lindland, O. I. and Sindre, G. *Defining Quality Aspects for Conceptual Models*, Proceedings of the IFIP8.1 working conference on Information Systems Concepts (ISCO3): Towards a Consolidation of Views, Marburg, Germany, (1995).

22. Loos, P. and Allweyer, T. *Process Orientation and Object-Orientation - An Approach for Integrating UML with Event-Driven Process Chains (EPC)*, University of Saarland, Saarbrücken, Germany, (1998).
23. Manola, F., Georgakopoulos, D., Heiler, S., Hurwitz, B., Mitchell, G. and Nayeri, F. *Supporting Cooperation in Enterprise-Scale Distributed Object Systems*, in *Cooperative Information Systems*, M. Papazoglou and G. Schlageter, Eds.: Academic Press, (1998).
24. Marshall, C. *Enterprise Modeling with UML*: Addison-Wesley, (1999).
25. Natvig, M. K. and Ohren, O. *Modelling shared information spaces (SIS)*, GROUP '99, Phoenix, Arizona USA, (1999).
26. Rolfsen, R. K., Jørgensen, H. D. and Carlsen, S. *Contextual Awareness: Survey and Proposed Research Agenda*, submitted to ECSCW'99, Copenhagen, (1999).
27. Uschold, M. and Gruninger, M. *ONTOLOGIES: Principles, Methods and Applications*, Knowledge Engineering Review, vol. 11, no. 2, (1996).
28. Voss, A., Procter, R. and Williams, R. *Innovation in Use: Interleaving day-to-day operation and systems development*, Participatory Design Conference, New York, NY, USA, (2000).
29. Wegner, P. *Why interaction is more powerful than algorithms*, Communications of the ACM, vol. 40, no. 5, (1997).
30. Wegner, P. and Goldin, D. *Interaction as a Framework for Modeling*, in *Conceptual Modeling. Current Issues and Future Directions*, P. P. Chen, J. Akoka, H. Kangassalo, and B. Thalheim, Eds. Berlin, Germany: LNCS 1565, Springer, (1999).
31. WfMC *Workflow Handbook 2001*. Lighthouse Point, Florida, USA: Workflow Management Coalition, Future Strategies Inc., (2000).
32. Zuboff, S. *In the Age of the Smart Machine*. USA: Basic Books Inc, (1988).
33. Aalst, W. v. d., Desel, J. and Oberweis, A. *Business Process Management*. Berlin, Germany: LNCS 1806, Springer, (2000).