5-2008

Supporting Electronic Collaboration in Conceptual Modeling

Peter Rittgen
University College of Borås, peter.rittgen@hb.se

73F. Supporting Electronic Collaboration in Conceptual Modeling

Peter Rittgen
University College of Borås
peter.rittgen@hb.se

Abstract

We study collaborative modeling by analyzing conversations and loud thinking during modeling sessions and the resulting models themselves. We identify the basic activities of the modeling teams on the social, pragmatic, semantic and syntactic levels and derive a schema for the pragmatic level. Our main conclusion is that team-based modeling is largely a negotiation process. Drawing on these results we derive an architecture of a system that supports the distributed development of conceptual models.

Keywords

Computer-mediated communication and collaboration, systems design and implementation, analytical modeling

1. Introduction

Descriptive approaches to understanding the modeling process are scarce. Only a few deal with collaborative modeling (see section 2). The others assume a scenario where a single expert modeler creates a formal model of some part of a business (Morris, 1967; Srinivasan & Te´eni, 1995; Willemain, 1994, 1995). These studies identify sets of general heuristics for successful modeling without going down to the level of the concrete steps that are performed in creating models. Their results are hardly applicable to business modeling in general for a number of reasons. Firstly, a business model is rarely developed by an expert alone but rather by a team involving representatives of the respective business(es) and externals. Secondly, the problem domain of general business modeling is often less well-structured and formal languages are of limited use. Thirdly and last, the goal of providing tool support for collaborative modeling requires the identification of detailed steps.

The objective of this paper is to discover the elementary activities and the structure of the modeling process, i.e. a meta-model of the modeling process. This is done by studying, in a descriptive way, the work performed by small groups of modelers that were assigned the same task: To develop business process models for a hospital based on a detailed description of the processes in natural language. This implied collective sense-making of a case text and conceptualizing the group's understanding of the text in the form of an analysis model. The group members were homogeneous concerning their modeling experience and their roles, i.e. there was no a-priori assignment of a group leader or modeling expert.

We then go on to interpret the results from the empirical study as requirements for a system that supports distributed modeling in groups. We develop an architecture of such a system that is in line with the empirical findings and supports the process that was observed in the study.

2. Research Method

Keeping this background in mind, we set out to study a situation where groups of modelers worked on a textual description of a business case with the purpose of deriving business process models. To understand the modeling process, we assumed that two factors are predominant in model creation:

- The internal mental processes of each modeler, and
- The conversations between modelers and within the group.

To get access to the former we used a think-aloud process-tracing methodology (Ericsson & Simon, 1993; Srinivasan & Te´eni, 1995) where the observants speak out what they are currently thinking. The utterances were then transcribed yielding the think-aloud protocols. The same is done with the conversations. In addition to that we also considered the product of the modeling process, the models themselves, to fill the gaps in the protocols and to help with interpreting ambiguous phrases in them. Open issues that could not be dealt with in this way were marked on the coding scheme and clarified by ex-post interviews with the respective groups.

To develop a preliminary coarse-grain categorization we turned to theories in the pertinent literature, particularly in organizational semiotics. We used the upper four 'rungs' of the semiotic ladder (Stamper, 1991): syntactic, semantic, pragmatic, and social. They refer to the structure of sign systems (e.g., a language), the meaning of the signs, their use, and the norms of a community, respectively. An initial coding phase within this framework revealed that the syntactic and semantic levels, which together make up the language level, are divided into the natural language domain and the modeling language domain depending on the kind of language used to describe the business.

The activities on the pragmatic level were classified as 'Understanding' and 'Organizing the Modeling Process'. The former term was then further refined into 'Undestanding the language' and 'Understanding the text', the latter can be divided into 'Setting the agenda' and 'Negotiation'. The social level consists of rules for acceptance and rejection in the negotiation. A detailed discussion of these categories can be found in the respective sections. The results are summarized in fig. 1.

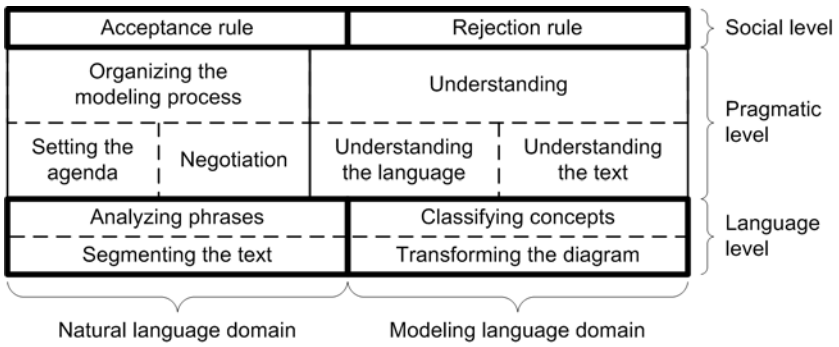

Figure 1: Levels and domains

We conducted 3 experiments that involved a total of 26 groups of 2-3 students in informatics over a period of 3 years. The students were provided with a textual description of four business processes in a hospital. They were asked to model these processes with the help of two different modeling languages that they could choose freely from a set of four languages: ARIS-EPC (Scheer, 1999), FMC-Petri nets (Keller & Wendt, 2003), UML (OMG, 2004, 2006), and DEMO (Dietz, 1999). Based on the results of these experiments we derived a layered meta-model of the modeling process that includes a model of the negotiation process.

3. Results

We carried out the main coding of the material within the framework stipulated by fig. 1. Examples of that procedure are shown in the respective section. The activities on the pragmatic level were classified as ‘Understanding’ and ‘Organizing the Modeling Process’. The former term was then further refined into ‘Understanding the language’ and ‘Understanding the text’, the latter can be divided into ‘Setting the agenda’ and ‘Negotiation’. The social level consists of rules for acceptance and rejection in the negotiation. A detailed discussion of these categories can be found in (Rittgen, 2007). Here we present only the results concerning the social and pragmatic levels.

3.1 Social level

The social norms within a modeling team are mainly made up of rules for determining whether a proposal is accepted or rejected. We observed that these rules do not have to be logical complements which allows for situations where a proposal can be neither rejected nor accepted but requires further convincing to decide one way or the other. A termination rule was applied occasionally to force a decision if a negotiation got stuck, i.e., when there were no more changes in the individuals’ convictions over an extended period of time. We witnessed two types of rules:

- *Rules of majority*, where a certain number of group members had to support or oppose a proposal in order for the whole group to accept or reject it (e.g., more than half). A tie-break rule was sometimes specified (e.g., for the case of an equal number of supporters and opponents). The tie-break could involve seniority issues.
- *Rules of seniority*, where the weight of a group member’s support or opposition was related to his or her status within the group. This status could be acquired (e.g., by experience) or associated with a position to which the member was appointed. A frequent example of this was the case of a more experienced modeler who was considered as the leader by the group and took decisions on their behalf. The other members filled the role of consultants in such a case.

These rules were sometimes set up explicitly before the group began their work, or in an early phase of this work. But in most cases they rather emerged as the result of each member’s behavior. Individuals making regular contributions of high quality were likely to acquire seniority. In homogeneous teams majority rules were used more often.

3.2 Pragmatic level

On the pragmatic level we discovered two distinct types of behavior, each of which can be classified in two sub-categories (the abbreviations of the categories are used as indices of the respective coded terms later on):

- *Understanding*, which concerns the text of the case description (index UT) or the (modelling) language (index UL), and
- Organizing the modelling process, which involves two types of activities: *setting the agenda* (index SA) and *negotiation* (index N).

Understanding was established by questions and answers. If the respondent could not provide clarification, an assumption was made. Agendas have been used by the participants in our study as an instrument for roughly structuring the modelling session. They were introduced in the beginning and then adapted during the session if necessary. On the whole most groups started by reading the case description completely and then organized their work around the flow of the text.

The majority of the activities on the pragmatic level were associated with negotiation, though. This is surprising as modelling is typically rather pictured as an intuitive act that is largely the product of a creative brain (e.g., a consultant) that possibly receives some input from other stakeholders in the modelling process (e.g., domain experts from the respective departments). According to our results modelling is a relatively well-structured process. It consists of a limited number of well-defined activities on all levels of the semiotic ladder. We are aware that further research will reveal more activities but from the experience of the three experiments that yielded a decreasing number of new ones, we are confident that the total number of activities will converge. The activities identified so far can therefore be assumed to be relatively stable. To a certain extent this is even true across different modelling languages, although the terminology of concepts may vary and not every concept is realized in each of the languages.

An analysis of the workflows on the pragmatic level revealed a structure that goes beyond the mere identification of generic activities. We found out that the negotiation process actually follows a certain pattern. This pattern is shown in Fig. 2.

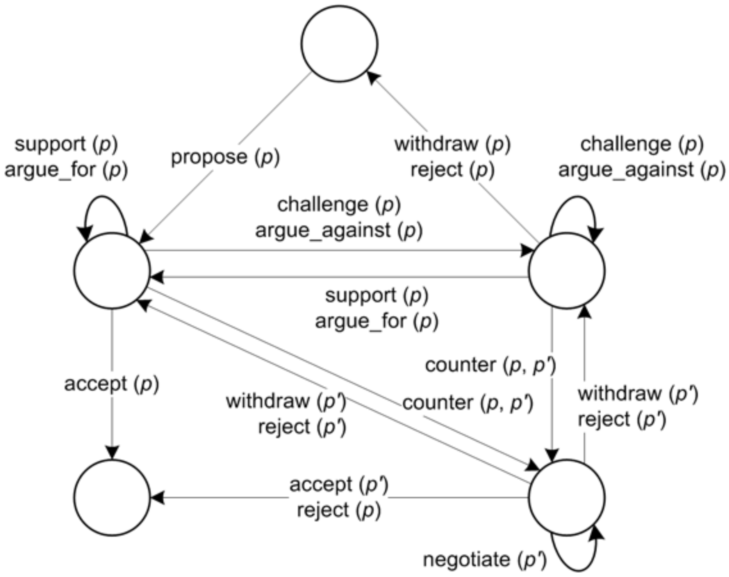

Figure 2: Negotiation pattern

It consists of an initial and reject state at the top, a state where acceptance is favored (upper left-hand corner), a state where rejection is favored (upper right-hand corner), a recursive sub-state for negotiating a counter-proposal (lower right-hand corner) and an accept state (lower left-hand corner). Each of the states allows for a set of certain pragmatic activities that take the negotiation to a different state. We have left out the parameters concerning the modeller who performs the activity and the argument (if present). In general any modeller can perform any activity but there are a few rules to be observed. A modeller making a proposal is implicitly assumed to support it. He is the only one who may withdraw it. A counter-argument is brought up by a different modeller but a counter-proposal can also be made by the proponent of the original proposal, e.g., to accommodate counter-arguments. With the help of the pattern of Fig. 2 we can control the negotiation component of a modelling support system. On the other levels we were not able to discover an equally strong pattern of activities. This will affect the kind of support a tool can provide at the language level.

4. Tool Support for the Modeling Process

Our analyses of the modeling sessions showed us that modeling is a complex process involving issues such as collective sense-making, negotiations and group decisions. It is therefore worthwhile to consider tool support for this process. This is particularly true in an interorganizational setting where participants are often geographically distributed. The tool we envision helps group members in understanding the modeling situation, creating and discussing modeling alternatives, and deciding on the best one, all in a shared internet-based environment. The following paragraphs elaborate on the components that such a tool should provide.

The architecture of a modeling support system, i.e., a system that supports a group in developing models, is still under investigation. Some authors have suggested groupware systems that help teams in collective sense-making (Boehm, Grunbacher, & Briggs., 2001; Briggs, de Vreede, & Nunamaker, 2003; Conklin, Selvin, Buckingham Shum, & Sierhuis, 2003; Hoppenbrouwers, Lindeman, & Proper, 2006) which is an important part of the modeling process. (Conklin, Selvin, Buckingham Shum, & Sierhuis, 2003) reports on an approach, Compendium, that is the result of 15 years of experience. Compendium combines three different areas: meeting facilitation, graphical hypertext and conceptual frameworks. To make them work, facilitation is viewed as essential to remove the cognitive overhead for the group members, i.e., the necessity to develop hypertext literacy, which cannot be assumed in all participants. On the technology side, the critical elements are question-based templates, metadata and maps. They allow participants to move freely between different levels of abstraction and formalization as the need dictates. The question-based templates guide the process by supplying relevant questions, the answers to which will lead the group towards a better understanding of the problem and towards the development of appropriate solutions (e.g., models). The metadata is used to provide additional information that is also considered relevant but was not anticipated in the templates or lies at the intersection of templates. The maps have a hierarchical structure and the same concept can appear in different maps so that its use in different contexts can be understood. This feature is called transclusion.

Groupware systems for collective sense-making, as the one mentioned, address an important issue in collaborative modeling. They can therefore be used as the core of a modeling support system (MSS). So far these systems are typically tailored for specific modeling languages though (in the case of Compendium, World Modeling Framework and Issue-Based Information System). For an MSS they need to be more modular so that any modeling language can be “plugged in” (e.g., other enterprise or information systems modeling languages). In addition, there is also the need for a negotiation component that facilitates structured arguments and decisions regarding modeling choices. The model shown in Fig. 2 can function as an initial workflow template controlling such a negotiation component. Once instantiated the actual workflow can then be adjusted to the concrete modeling situation.

5. Architecture of a Distributed Modeling Support System

A distributed modeling support system needs to coordinate the efforts of a number of modelers. To this end each modeler has to have a clear overview of the current status of the negotiation and model building processes. The latter involves the current stable version of the model as agreed upon so far, a version that includes the local changes made by the respective group member (the local model editor), a proposal suggesting changes to the current model and a counter proposal if applicable. Fig. 3 shows the screen layout of the model building view.

Regarding the status of the negotiation process the modeler needs access to the following information: What are the arguments for and against the proposal and the counter proposal? How many group members are in support of or against the proposal or counter proposal? What is the final decision regarding the (counter) proposal? This information can be presented in the form of a table. An example is given in Table 5.

The model building view is divided into four windows (see Fig. 3). The upper left one shows the current version that has emerged from the negotiation process so far. It is used as a reference for all other temporary versions such as the proposals and the local version. This means that suggested changes are always expressed in relation to the current version, e.g. added or deleted nodes are marked in a special way.

The lower left window contains the local version, i.e. it serves as the model editor for the group member running this particular copy of the modeling support system. It offers the model creation facilities pertaining to the syntactic level such as introducing and connecting nodes. In addition to that it provides the pragmatic functions related to making proposals and counter-proposals. Making a proposal implies that the local version is put in the respective proposal queue.

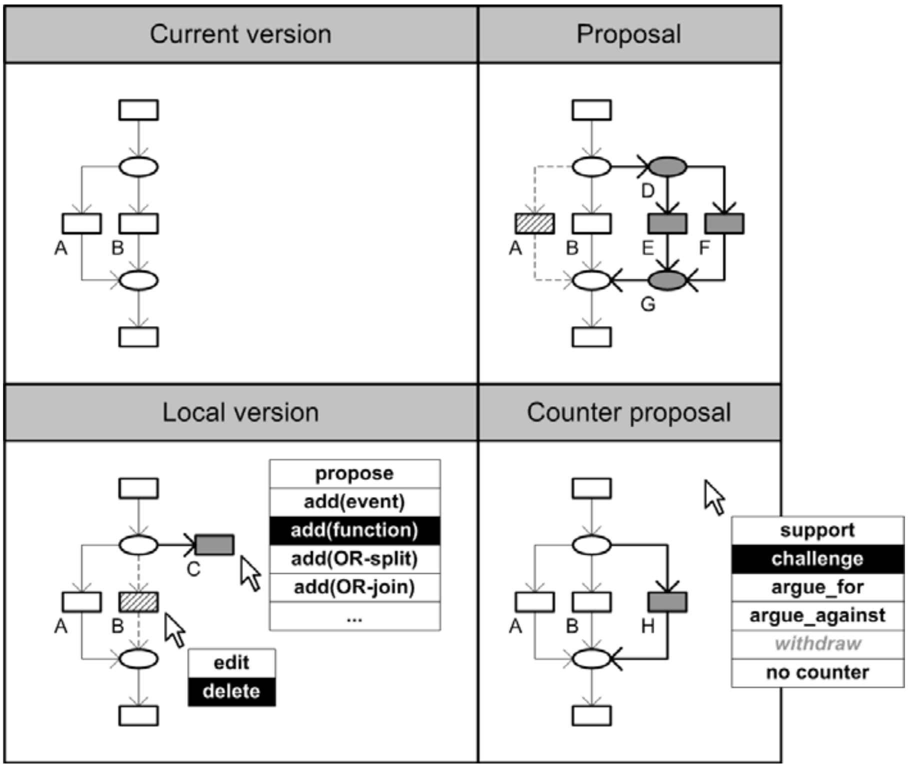

Figure 3: The model building view

In the example of Fig. 3 the modeler has already made two changes to the current version. She has decided that node B is not needed and should therefore be deleted (symbolized by the hatched area). She also suggests a new node C and connects it to a node in the current version. The respective rectangle is shaded in grey and the arrow is drawn with a larger line weight. The model in this case is an Event-driven Process Chain [8] of a business process. Another modeler has already made a proposal regarding the current version, which is displayed in the upper right window. He proposes to remove node A and to add the nodes D-G. This proposal is countered by yet another group member who thinks that the current version should only be augmented by the new node H.

In this situation the original modeler, the one sitting in front of the screen shown in Fig. 3, decides that the counter-proposal is not acceptable. He shows this to the others by voting against it, i.e. by triggering the *challenge* function in the counter-proposal window. If he wants to supply an argument for his rejection he chooses *argue_against* instead. He can also specify that he considers the counter-proposal not to be a counter to the original proposal and therefore be inadequate. This is treated in the same way as an *argue_against* vote with the argument being “Counter-proposal does not counter the original proposal.” The *challenge* serves as input to the negotiation component where the group finally accepts or rejects this counter-proposal. Observe that it is not allowed to support (or *argue_for*) both the proposal and the counter-proposal.

Proposal	
For	**Against**
Joe: We need to differentiate between two types of prescriptions. Bill: The different prescriptions must be considered in treatment planning already. Frank: A ward transfer is rather part of the anamnesis phase or treatment evaluation.	Harry: The different prescriptions should be handled later. Ralph: I am missing a third alternative for patient treatment (transfer to another ward).
4 supporters	**5 challengers**

Table 1: The negotiation view

The negotiation view (see Table 1) shows a list of pros and cons for the proposal and the counter-proposal. These are the arguments supplied when the *argue_for* or *argue_against* function is performed on a (counter-)proposal by a modeler. This view also shows the total number of supporters and challengers of this proposal. Depending on the currently active rule on the social level this determines overall acceptance or rejection of the proposal, e.g. when a required majority has been reached. A facilitator might also force a decision if the required condition cannot be reached.

If a counter-proposal is rejected the next one in the respective queue is displayed and considered by the group. If no more counter-proposals are available, the window remains empty. A rejected proposal turns the current counter-proposal into a proposal; the counter-proposals from the queue become attached to this new proposal. The first among them becomes the new active counter-proposal (if any) and is shown in the lower right window. In the absence of counter-proposals in the queue, the next proposal from the queue is considered. An accepted proposal makes all related counter-proposals obsolete; the next in the proposal queue becomes the new active proposal. If the counter-proposal is accepted, the proposal is removed and the first among the competing counter-proposals from the queue becomes the new proposal (if any) and the second will be the new counter-proposal (if any). The others remain counter-proposals but are now related to the new proposal. Observe that the deleted (counter-)proposals still exist as local copies so that the respective modelers can decide to post them again as proposals if applicable.

If a proposal or counter-proposal is accepted by the group it becomes the new current version, i.e. the upper left window is updated for all group members. This means that the local version has now become inconsistent as it is still related to the former version. It is therefore necessary to update the local version (as well as all the proposals and counter-proposals). Fig. 4 shows in which way this is done.

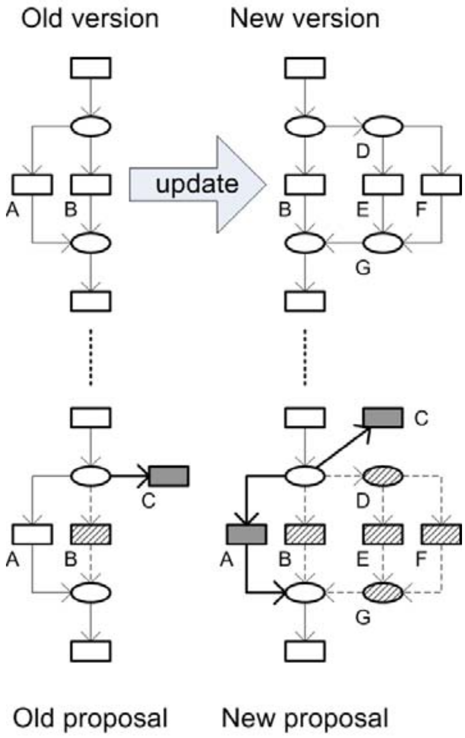

Figure 4: Update of local version following update of current version

Let us assume that the proposal of Fig. 3 has been accepted and becomes the new current version now. This scenario is depicted in Fig. 4. The old proposal, i.e. the local version in relation to the former current version (old version), involved the deletion of node B (and incident arcs) and the introduction of node C (and incident arc). In the new current version (called new proposal in Fig. 4) the situation concerning node B has not changed. It still exists in the new version and we still suggest removing it. But in our proposal the node A still exists whereas it has been deleted in the new accepted version. This means that our proposal in relation to the new version actually implies to reintroduce this node. It is therefore shaded in grey and the incident arcs get a larger line weight. Node C is part of neither the old nor the new version so nothing changes here. But the new version contains a number of new nodes that are not present in our local version (D-G). If we want to stick to our local version, then our “new” proposal is to remove these nodes and all incident arcs.

The architecture that we have described so far is a direct consequence of the results from the empirical study detailed in the previous sections. The architecture supports the activities on the social, pragmatic and syntactic levels. We are currently building a prototype that implements this architecture and that allows us to gain further insight into the modeling process. Such a prototype can be employed in a number of different ways: It can be used to test the suggested architecture and thereby indirectly confirm the study results in a broader empirical study. The additional tool support in this study makes it easier to perform such a study on a larger scale, e.g. with distributed team members, and it also provides additional information about the modeling process that was not available in the original study.

The existence of version histories, for example, makes it possible to analyze the modeling process in a more detailed manner regarding the development stages of a model. Another example is the negotiation log that gives us a deep insight into the arguing process and the competition between different model alternatives. A study supported by this tool can therefore also contribute to the development of new theories of the modeling process. On the practical side, the prototype can also help in detecting shortcomings and suggesting improvements. These suggestions can be related to the implementation (i.e., the tool itself) or the architecture behind it (as outlined above). Issues such as the design of the user interface and migration to other modeling languages are important considerations here.

6. Conclusions

We studied group modeling sessions in detail, both regarding conversations between the group members and the mental processes within each individual. By doing this we derive a sub-categorization of the upper four levels of the semiotic ladder, generic activities of business process modeling at all of these levels and a negotiation pattern at the pragmatic level. On the basis of these results we suggest a tentative architecture of a system that supports group modeling. Our aim with this research is two-fold. On the one hand we want to develop a better understanding of the modeling process that has been largely neglected by researchers so far. Such an improved understanding can lead to better modeling methods and thereby ultimately to higher quality of models.

On the other hand we are also interested in providing computer support to those modelers that work in a group environment. Modeling is a highly demanding task that is further complicated by the dynamics of group work. Effective support is therefore essential, especially if some of the group members are inexperienced as is often the case in business modeling sessions, where typically a majority of the participants does not have any modeling background. But it is precisely this latter type of participant that contributes most to the actual design of the model with his or her knowledge of the relevant business domain. Both the speed and quality of the models can therefore benefit tremendously if we can manage to involve these people directly as modelers instead of relying on the bottleneck of the modeling expert for all communication within the group. The suggested tool support can accommodate this by giving the expert seniority (i.e., the right to make the final decision) and turning the domain experts into effective consultants that make proposals (thereby reversing the traditional roles in IT consulting).

A distributed modeling support system can also be seen as a special kind of group decision support system (GDSS, (Aiken, Vanjani, & Krosp, 1995)) if we consider that the accept and reject decisions in the negotiation process are the key to model design. There is significant empirical support for the claim that GDSS are beneficial (Aiken, Vanjani, & Krosp, 1995; Bamber, Watson, & Hill, 1996; Benbasat & Lim, 1993; Bidgoli, 1996; Burke, Chidambaram, & Lock, 1995; Cass, Heintz, & Kaiser, 1991; Chudoba, 1999; Fjermestad & Hiltz, 1998/1999; Jackson, Aiken, Mahesh, & Bassam, 1995; Townsend, Whitman, & Hendrickson, 1995), particularly for larger groups and/or complex tasks. Many of these benefits carry over to modeling support systems, e.g., reduced meeting time, higher quality of the decisions, broader involvement of all participants, higher effectiveness of decisions, etc.

Our research studied text-based modeling only. This is not a realistic scenario for practical modeling situations. We are confident though that our results are relevant for real-world modeling to some extent. The social level is fairly independent of the way in which a modeling alternative was derived (text-based or other) as the decision rule rather depends on the alternatives themselves. The same is true for the language level as we can safely assume that natural language and modeling languages will play an important role in any modeling endeavor. We therefore expect differences primarily on the pragmatic level, and here especially in the areas “setting the agenda” and “understanding”. Whether modelers just interpret a text or communicate with domain experts will have considerable impact on the way the agenda is determined. Likewise the issue of understanding has to be extended to cover forms of communication other than analyzing text.

So far we have only looked at business process modeling. Other domains in the business and information systems areas remain to be explored. It should also be noted that our study has been performed in a contrived setting albeit with a realistic case. Further confirmation, and especially consolidation, is therefore required, preferably by means of a field study. The suggested architecture is only based on three of the four levels. Further research needs to address the semantic level, too. In addition to this it seems reasonable to build a prototype of a modeling support system, and to test it in a real-life modeling scenario. We assume that these measures will contribute to a better understanding of the process of modeling, both from a cognitive and a collaborative perspective, and they will eventually help us to better support modelers in their challenging task.

References

Aiken, M., Vanjani, M., & Krosp, J. (1995). Group decision support systems. *Review of Business, 16*(3), 38-42.

Bamber, E. M., Watson, R. T., & Hill, M. C. (1996). The effects of group support system technology on audit group decision-making. *Auditing: A Journal of Practice & Theory, 15*(1), 122-134.

Benbasat, I., & Lim, L. H. (1993). The effects of group, task, context, and technology variables on the usefulness of group support systems: A meta-analysis of experimental studies. *Small Group Research, 24*, 430-462.

Bidgoli, H. (1996). A new productivity tool for the 90’s: Group support systems. *Journal of Systems Management, 47*(4), 56-62.

Boehm, B., Grunbacher, P., & Briggs., R. O. (2001). Developing Groupware for Requirements Negotiation: Lessons Learned. *IEEE Software*, *18*(3), 46-55.

Briggs, R. O., de Vreede, G. J., & Nunamaker, J. (2003). Collaboration Engineering with Thinklets to Pursue Sustained Success with Group Support Systems. *Journal of MIS, 19*, 31-63.

Burke, K., Chidambaram, L., & Lock, J. (1995). Evolution of relational factors over time: A study of distributed and non-distributed meetings. In *Proceedings of the Twenty-Eighth Hawaii International Conference on System Sciences* (Vol. 4, pp. 14-23).

Cass, K., Heintz, T. J., & Kaiser, K. M. (1991). Using a voice-synchronous GDSS in dispersed locations: A preliminary analysis of participant satisfaction. In *Proceedings of the Twenty-Fourth Hawaii International Conference on System Sciences* (Vol. 3, pp. 555-563).

Chudoba, K. M. (1999). Appropriations and patterns in the use of group support systems. *Database for Advances in Information Systems, 30*(3/4), 131-148.

Conklin, J., Selvin, A., Buckingham Shum, S., & Sierhuis, M. (2003). *Facilitated Hypertext for Collective Sensemaking: 15 Years on from gIBIS.* Paper presented at the Proceedings of the 8th International Working Conference on the Language-Action Perspective on Communication Modeling (LAP'03), Tilburg, The Netherlands.

Dietz, J. L. G. (1999). Understanding and modeling business processes with DEMO. In J. Akoka, M. Bouzeghoub, I. Comyn-Wattiau & E. Métais (Eds.), *Proceedings of the 18th International Conference on Conceptual Modeling ER '99* (pp. 188-202). Berlin: Springer.

Ericsson, K., & Simon, H. (1993). *Protocol Analysis: Verbal Reports as Data*. Boston: MIT Press.

Fjermestad, J., & Hiltz, S. R. (1998/1999). An assessment of group support systems experiment research: Methodology and results. *Journal of Management Information Systems, 15*(3), 7-149.

Hoppenbrouwers, S. J. B. A., Lindeman, L., & Proper, H. A. (2006). Capturing Modeling Processes - Towards the MoDial Modeling Laboratory. In R. Meersman, Z. Tari & P. Herrero (Eds.), *On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops - OTM Confederated International Workshops and Posters, AWESOMe, CAMS, COMINF, IS, KSinBIT, MIOS-CIAO, MONET, OnToContent, ORM, PerSys, OTM Academy Doctoral Consortium, RDDS, SWWS, and SebGIS, Proceedings, Part II, Montpellier, France* (Vol. 4278, pp. 1242-1252). Berlin, Germany: Springer.

Jackson, N. F., Aiken, M. W. V., Mahesh, B. H., & Bassam, S. (1995). Support group decisions via computer systems. *Quality Progress, 28*(5), 75-78.

Keller, F., & Wendt, S. (2003). *FMC: An Approach Towards Architecture-Centric System Development.* Paper presented at the 10th IEEE Symposium and Workshop on Engineering of Computer Based Systems, Pasadena, CA.

Morris, W. T. (1967). On the Art of Modeling. *Management Science, 13*(12), B-707 - B717.

OMG. (2004). UML 2.0 Superstructure Specification. Retrieved December 20, 2005, from http://www.uml.org/

OMG. (2006). Unified Modeling Language: Infrastructure. Retrieved April 18, 2006, from http://www.uml.org/

Rittgen, P. (2007). Negotiating Models. In J. Krogstie, A. Opdahl & G. Sindre (Eds.), *Advanced Information Systems Engineering, 19th International Conference, CAiSE 2007, Trondheim, Norway, June 2007, Proceedings* (pp. 561-573). Berlin: Springer.

Scheer, A.-W. (1999). *ARIS - Business Process Modeling* (2 ed.). Berlin: Springer.

Srinivasan, A., & Te´eni, D. (1995). Modeling as Constrained Problem Solving: An Empirical Study of the Data Modeling Process. *Management Science, 41*(3), 419-434.

Stamper, R. (1991). The Semiotic Framework for Information Systems Research. In H. Nissen, H. Klein & R. Hirschheim (Eds.), *Information Systems Research: Contemporary Approaches and Emergent Traditions* (pp. 515-517). Amsterdam: North-Holland.

Townsend, A. M., Whitman, M. E., & Hendrickson, A. R. (1995). Computer support system adds power to group processes. *HRMagazine, 40*(9), 87-91.

Willemain, T. R. (1994). Insights on Modeling from a Dozen Experts. *Operations Research, 42*(2), 213-222.

Willemain, T. R. (1995). Model Formulation: What Experts Think about and When. *Operations Research, 43*(6), 916-932.