Comparing Languages for Enterprise Modeling using a Language Quality Framework

Sofie de Flon Arnesen and John Krogstie

Norwegian University of Science and Technology, Institute of Computer and Information Sciences
and SINTEF Telecom and Informatics, Norway.
John.Krogstie@sintef.no

Abstract: Statoil have recently wanted to standardize enterprise modeling language for sense-making and communication. To perform the evaluation, a generic framework for assessing the quality of models and modeling languages was specialized to the needs of the company. Five different modeling languages were evaluated according to the specialized criteria.

The work illustrates the practical utility of the overall framework, where language quality features are looked upon as means to enable the creation of models of high quality. It also illustrates the need for specializing this kind of general framework based on the requirements of the specific organization

1. INTRODUCTION

There exist a large number of modeling languages for enterprise modeling. Deciding which modeling language to use for a specific task is often done in an ad-hoc fashion by different organizations. Statoil have over the years developed and used several different modeling languages for modeling enterprise processes. Last year they embarked on a process for evaluating and selecting a standard modeling language for enterprise process modeling.

We have earlier developed a general framework for assessment of quality of models, where one type of means to support quality, is criteria for the language to be used for modeling, [13]. This paper presents an example of using and specializing the quality framework for the evaluation and selection of a modeling language for enterprise modeling for Statoil.

The next section describes the general quality framework, with a focus on language quality. Section three describes the Statoil-case in more detail, and is followed by the results of the evaluation. The conclusion highlights some of our experiences from using and specializing the quality framework for evaluating modeling languages for enterprise modeling.

2. FRAMEWORK FOR QUALITY OF MODELS

We here present briefly the general quality framework. The model quality framework [10,12,13] we use as a starting point has several distinguishing properties:

- Because we recognize that modeling is essentially making statements in some language, it is closely linked to linguistic and semiotic concepts.
- It is based on a constructivist world-view [2], recognizing that models are usually created as part of a dialogue between the participants involved in modeling, whose

knowledge of the modeling domain changes as modeling takes place, and where the domain is also developed (socially (re)-constructed) as part of the modeling activity.
- It distinguishes between goals and means that support the achievement of the goals.

The main concepts of the framework and their relationships are shown in Figure 1. We have taken a set-theoretic approach to the discussion of model quality at different semiotic levels, which has been defined as the correspondence between statements belonging to the following sets:

- ***G,*** the (normally organizationally founded) goals of the modeling task.
- ***L***, the language extension, i.e., the set of all statements that are possible to make according to the graphemes, vocabulary, and syntax of the modeling languages used.
- ***D***, the domain, i.e., the set of all statements which can be stated about the situation at hand. Enterprise domains are socially constructed, and are more or less inter-subjectively agreed. That the world is socially constructed does not make it any less important to model that world [5].
- ***M***, the externalized model, i.e., the set of all statements in someone's model of part of the perceived reality written in a language.
- $\boldsymbol{K_s}$, the relevant explicit knowledge of the set of stakeholders being involved in modeling (the audience ***A***). A subset of the audience is those actively involved in modeling, and their knowledge is indicated by $\boldsymbol{K_M}$.
- ***I***, the social actor interpretation, i.e., the set of all statements which the audience think that an externalized model consists of.
- ***T***, the technical actor interpretation, i.e., the statements in the model as 'interpreted' by different model activators (modeling tools).

Solid lines between the sets in the figure indicate the main quality types:
- Physical quality: The basic quality goals on the physical level are externalization, that the knowledge ***K*** of the domain ***D*** of some social actor has been externalized by the use of a modeling language, and internalizeability, that the externalized model ***M*** is persistent and available enabling the audience to make sense of it.
- Empirical quality deals with predicable error frequencies when a model is read or written by different users, coding and HCI-ergonomics.
- Syntactic quality is the correspondence between the model ***M*** and the language extension ***L*** of the language in which the model is written.
- Semantic quality is the correspondence between the model ***M*** and the domain ***D***. The framework contains two semantic goals; Validity which means that all statements made in the model are correct relative to the domain and completeness which means that the model contains all the statements which is found in the domain.
- Perceived semantic quality is the similar correspondence between the audience interpretation ***I*** of a model ***M*** and his or hers current knowledge ***K*** of the domain ***D***, and is what can actually be checked at quality control/validation.
- Pragmatic quality is the correspondence between the model ***M*** and the audience's interpretation of it (***I***).
- Social quality has as goal agreement among audience members' interpretations ***I***.
- The organizational quality of the model relates to that all statements in the model directly or indirectly contributes to fulfilling the goals of modeling (organizational goal validity), and that all the goals of modeling is taken addressed through the model (organizational goal completeness).

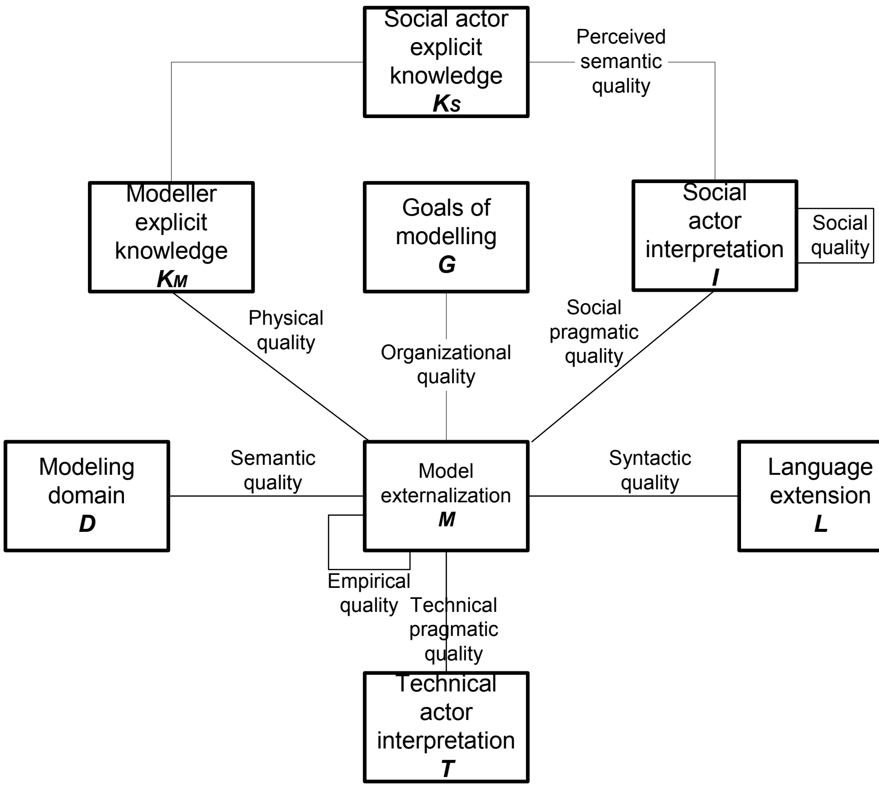
Social actor explicit knowledge K_S
Perceived semantic quality
Modeller explicit knowledge K_M
Goals of modelling G
Social actor interpretation I
Social quality
Physical quality
Organizational quality
Social pragmatic quality
Modeling domain D
Semantic quality
Model externalization M
Syntactic quality
Language extension L
Empirical quality
Technical pragmatic quality
Technical actor interpretation T

Figure 1: Main parts of the quality framework

Language quality relates the modeling languages used to the other sets. It is distinguished between two types of criteria:

- Criteria for the underlying (conceptual) basis of the language (i.e. what is represented in the abstract language model (meta-model) of the language).
- Criteria for the external (visual) representation of the language (i.e. the notation).

Domain appropriateness

Ideally, the conceptual basis must be powerful enough to express anything in the domain, i.e. not having construct deficit [17]. On the other hand, you should *not* be able to express things that are not in the domain (construct excess [17]).

There are an infinite number of statements that we might want to make, and these have to be dealt with through a limited number of phenomena classes to support comprehensibility appropriateness, as will be discussed below. This means that

- The phenomena must be general rather than specialized. This is parallel to what is termed genericity [14].
- The phenomena must be composable, which means that we can group related statements in a natural way. When only domain appropriateness is concerned, it is an advantage if all thinkable combinations are allowed.
- The language must be flexible in precision:

The only requirement to the external representation is that it does not destroy the underlying basis.

One approach to evaluating domain appropriateness is to look at how the modeling perspectives found useful for the relevant modeling tasks are covered. Seven general modeling perspectives have been identified [12]. Another approach is to base an evaluation on an ontological theory, see e.g. [15] that uses the ontology presented by [17]. Domain appropriateness is primarily a mean to achieve physical quality, and through this, semantic quality.

Participant language knowledge appropriateness

This area relates the knowledge of the modeler to the language. The conceptual basis should correspond as much as possible to the way individuals perceive reality. This will differ from person to person according to their previous experience, and thus will initially be directly dependent on the modeler. On the other hand the knowledge of the modeler is not static, i.e. it is possible to educate persons in the use of a language. In that case, one should base the language on experiences with languages for the relevant types of modeling, and languages that have been used successfully earlier in similar tasks. Participant language knowledge appropriateness is primarily a mean to achieve physical and pragmatic quality.

Knowledge externalisability appropriateness

This area relates the language to the participant knowledge. The goal is that there are no statements in the explicit knowledge of the modeler that cannot be expressd. Knowledge externalisability appropriateness is primarily a mean to achieve physical quality.

Comprehensibility appropriateness

This area relates the language to the social actor interpretation. For conceptual basis:

- The phenomena of the language should be easily distinguishable from each other. (Vs. construct redundancy [17]).
- The number of phenomena should be reasonable. If the number has to be large, the phenomena should be organized hierarchically, making it possible to approach the framework at different levels of abstraction.
- The use of phenomena should be uniform throughout the whole set of statements that can be expressed within the language. Using the same construct for different phenomena or different constructs for the same phenomena depending on the context will tend to make the language confusing (vs. construct overloading [17]).
- The language must be flexible in the level of detail. Statements must be easily extendible with other statements providing more details. Details must be easily hidden.

As for the external representation, the following aspects are important:

- Symbol discrimination should be easy.
- It should be easy to distinguish which of the symbols in a model any graphical mark in the model is part of.
- The use of symbols should be uniform.
- One should strive for symbolic simplicity.
- The use of emphasis in the notation should be in accordance with the relative importance of the statements in the model.
- Composition of symbols can be made in an aesthetically pleasing way.

Comprehensibility appropriateness is primarily a mean to achieve empirical and through that, pragmatic quality.

Technical actor interpretation appropriateness

This area relates the language to the technical actor interpretation. For the technical actors, it is especially important that the language lend itself to automatic reasoning. This requires formality (i.e. both formal syntax and semantics. The formal semantics can be operational, logical, or both), but formality is not sufficient, since the reasoning must also be efficient to be of practical use. This is covered by what we term analyzability (to exploit the mathematical semantics) and executability (to exploit the operational semantics). Different aspects of technical actor interpretation appropriateness are a mean to achieve syntactic, semantic and pragmatic quality (through formal syntax, mathematical semantics, and operational semantics respectively).

A number of sub-areas is identified for each of the five areas of language quality, and in [19] approximately 60 possible criteria were identified. We will return to how this extensive list has been narrowed down and specialized in the case. In addition one could look upon the description of a language (e.g. the notation guide and semantic description of UML) as a model of the language, and use the model quality framework to evaluate this model.

3. DESCRIPTION OF THE CASE AND THE EVALUATION APPROACH

Before discussing the needs of Statoil specifically, we outline the main uses of enterprise process modeling. Four main categories for enterprise modeling can be distinguished:

1. Human-sense making and communication: To make sense of aspects of an enterprise and communicate this to other people.
2. Computer-assisted analysis: To gain through simulation or deduction.
3. Model deployment and activation: Models can be activated in three ways:
   - Through people guided by process 'maps', where the system offers no active support.
   - Automatically, where the system plays an active role, as in most workflow engines.
   - Interactively, where the computer and the users co-operate in interpreting the model. The computer makes decisions about prescribed parts of the model, while the users resolve ambiguities.

4. To give the context for system development, without being directly implemented.

An orthogonal dimension to these four are the temporal dimension, i.e. if one are to model the past, the present (as-is) or the future (to-be). Another key differentiator is to what extent the focus is on internal processes of a company, or to support inter-organizational co-operation. Finally one can differentiate between process models on a type vs. instance level.

Statoil's requirements

The detailed requirements to the modeling language were established in discussion with Statoil. The main responsible for this process at Statoil, being responsible e.g. for methodological issues in the company, had both a long-time background in connection to several enterprise modeling tasks within different parts of Statoil, and a good overview of modeling and modeling techniques in general. Our discussions were primarily with him, whereas he communicated in addition with different parties within the company.

Statoil had the following overall requirements:

- It should be a language for sense making and communication (category 1 above).
- The language should be usable by leaders on top and medium levels, and also others that were not used to model with graphical languages.
- The language should be simple, but still have sufficient expressiveness.
- The language should be independent of any specific modeling domain.
- It should be possible to describe processes across organizational areas.
- It should be possible to model both routine and non-routine work
- It should be possible to model processes both on a type and on an instance level

The following concepts were regarded as mandatory to be able to model:

1. Processes – including decomposition of the process
2. Activities – indicating the lowest level of a process model
3. Roles
4. Decisions and decision-points
5. Flow between processes, activities, and decision points

Deliverables (process results e.g. in the form of documents) and system resources (here with a focus on information systems) were regarded as recommended (but not mandatory).

It was not a requirement that all the concepts should be expressed visually with a separate symbol, as long as it was possible to express the concept.

It should be possible to develop the model incrementally. More concretely this meant:

- It should be possible to model processes and flows.
- It should be possible to model activities and flows independently.
- It should be possible to model roles, decision points, system resources and deliverables independently.

A general set of requirements to a modeling language based on the discussion of language quality in section 2 is outlined in [19]. As already mentioned, this amounted to more than 60 sub-requirement. These were looked at relative to the requirements of Statoil, and their importance was evaluated as indicated below.

Grade	**Explanation**
0-3	Requirement has no or very limited relevance
4-6	Generally relevant requirements, but not specifically important for the case
7-9	Specifically relevant requirements for the specific needs of the case
10 (Mandatory)	An absolutely necessary requirement

Only requirements given a grade of 7 or higher were included. In addition, requirements on domain appropriateness were detailed further compared to the general framework, including the mandatory and recommended concepts mentioned above. This resulted in the following evaluation-criteria:

Domain appropriateness

Underlying basis

No	Requirement
1	The language should be independent of business domain
2	The language should be able to express the following concepts
2.1	Processes: A process can consist of several sub-processes or activities, i.e. process decomposition
2.2	Activities: The lowest level of a process model
2.3	Roles (of persons involved in the process)
2.4	Decision points/decision rules
2.5	Flow between processes, activities, and decision points.
2.6 (rec)	Deliverables (results)
2.7(rec)	System resources (information-systems used in the process).
3	It must be possible to decompose processes in as many levels as necessary.

External representation

No.	Requirement
4	Also the process-symbols should be decomposable

Participant language knowledge appropriateness

Underlying basis

No	Requirement
5	The terms used of concepts must be same as those used for these in Statoil
6	It must be easy to learn the language

External representation

No	Requirement
7	The external representation must be intuitive, meaning that the symbol represent the concept better than another symbol would.

Knowledge externalisability appropriateness

Statoil had no specific requirements in this area.

Comprehensibility appropriateness

Underlying basis

No	Requirement
8	The different concepts must be easily distinguishable
9	The number of concepts must be at a reasonable level
10	The language must be flexible in the level of detail

External representation

No	Requirement

11	Symbol discrimination should be easy
12	The use of symbols should be uniform
13	One should strive for symbolic simplicity
14	emphasis in the notation should be in accordance with the relative importance

Technical actor interpretation appropriateness

Statoil was in this connection not interested in automatic reasoning and execution/simulation, and did not want requirements on this area to decide on the choice. It was mandatory that the syntax of the language was formally defined.

4. EVALUATION

The overall approach to the evaluation was the following: First a short-list of relevant languages was identified. The chosen languages were then evaluated according to the selected criteria. To look upon this in more detail, all languages were used for the modeling of several cases (including both models on an instance and a type level) using an 'independent' modeling tool (Visio). By showing the resulting models and evaluation results to the persons from Statoil, we got feedback and corrections both on the models and our grading. The overall result identified two candidates, where aspects such as available tool support were used to come up with a final choice.

Based on discussions with Statoil and experts on Enterprise Modeling, five languages were selected on a short-list of relevant languages. These will be briefly described: For a longer description see [1] and the cited references.

Language used in the modeling conference technique [7]

Gjersvik has developed a very simple process modeling language to use in so-called participatory modeling conferences, which had been used at several occasions in Statoil earlier. The language has three symbols:

- Process, Products (intermediate and final), Flow between processes and products

EEML (Extended Enterprise Modeling Language) [6]

EEML is currently developed in the EU-project EXTERNAL as an extension of APM [4]. Version 1 of the language has constructs to support all modeling categories described in section 3, and not only for human sense-making and communication.

The following main concepts are provided:

- Task with input and output ports
- General decision-point
- Roles (Person-role, Organization-role, Tool-role, Object-role)
- Resources (Persons, Organizations and groups of persons, Tools (manual and software), Objects (material and information)

A flow links two decision points and can carry a resource. A task have several parts: An in-port and an out-port, and potentially a set of roles and a set of sub-tasks. Roles 'is filled by' resources of the corresponding types.

The current language for enterprise modeling in Statoil [1]

Through earlier enterprise modeling and reengineering-projects Statoil has developed a language that is similar to role-oriented flow-languages such as Rummler-Brache [16].

A work-process can be decomposed in several sub-processes, which can be decomposed further. On the lowest level, one has activities, which can not be decomposed.

UML activity-diagrams [3]

An activity-diagram can have the following symbols

- Start, End, Activity, Flow (between activities, either as control or as object-flows), Decision-points, and Roles using swimlanes

IDEF-0 (Integration Definition language 0) [8]

IDEF-0-diagrams have two main symbols: Functions (boxes), Flow (arrows) of different types between functions.

Overview of evaluation results

Below the main result of the evaluation is summarized. For every language, every requirement is scored according to the below scale (which was also used in [19]).

Grade	**Explanation**
0-3	There is no, or very limited support of the requirement
4-6	The requirement is partly supported
7-9	There is satisfactory support of the requirement
10	The requirement is very well supported

The reasoning behind the grading can be found in [1], and is not included here due to space limitations. The two last rows summarize the results. The last row only includes the mandatory requirements.

		Grading per language				
No	Requirement	Gjers vik	EEML	Statoil	UML - AD	IDEF-0
1	Domain independence	10	10	10	10	10
2	Expressiveness					
2.1	Processes	10	10	10	10	10
2.2	Activities	6	6	10	6	6
2.3	Roles	0	10	10	10	0
2.4	Decision points	0	7	7	7	0
2.5	Flow	5	10	10	10	10
2.6	Deliverables (results)	8	10	10	5	10
2.7	System resources	0	8	7	0	0
3	Decomposable processes	0	10	7	7	10
4	Decomposable symbols	0	10	7	7	10
5	Equal naming of concepts	9	6	8	9	7
6	Language easy to learn	10	6	7	8	6
7	Intuitive representation	7	8	9	9	10
8	Easy to separate symbols	10	6	10	10	10
9	Reasonable number of con-	4	5	7	9	4

	cepts					
10	Flexible in precision	4	10	10	5	0
11	Easy to differentiate symbols	10	5	7	9	10
12	Consistent notation	5	10	7	10	3
13	Symbolic simplicity	9	5	10	10	10
14	Use of emphasis	7	7	9	10	10
	Sum including recommended requirements 2.6 and 2.7:	114	159	172	161	136
	Sum excluding recommended requirements 2.6 and 2.7	106	141	155	156	126

Comparison table with all the evaluations collected

Based on the evaluation, two of the languages where clearly inappropriate: IDEF-0 and Gjersvik's language. The internal language developed/adapted by Statoil has the highest sum taking into account also the recommended requirements, whereas UML activity diagrams got slightly ahead when only including the mandatory requirements. EEML comes third using both summations. EEML was regarded as too complex when only looking at the support of category one (which is not surprising, since it is meant to be used across all categories), with too many concepts, symbols and constraints for inexperienced modelers. Thus looking only at the language quality, two languages were found as candidates for further investigation, both being examples of so-called role-oriented process modeling languages. Based on earlier critique of activity diagrams [13], it was somewhat surprising that this language scored as high as it did. On the other hand, when using this only for sense-making and communication, one could ignore the somewhat alienating official state-oriented semantics defined in the current version of UML (1.4), and use the activity diagram more or less as a traditional flow-chart. When looking at these languages in connection to other aspects in the model quality framework, e.g. including tool support, it appeared that for instance even if activity diagrams are intended to have decomposition, the available UML-tool in Statoil (Rational Rose) did not support decomposition of activities properly. It neither supports the more intuitive semantics of activity-diagrams (which probably will be introduced in UML2.0), but rather the official semantics.

The final choice was up to Statoil (which in this case decided to keep and extend their existing language, and provide further tool and organizational support for using this).

5. CONCLUSIONS AND FURTHER WORK

We have in this paper described the use of a general framework for discussing the quality of models and modeling languages in a concrete case of evaluating enterprise process modeling languages. The case illustrates how our generic framework can (and must) be specialized to a specific organization and type of modeling to be useful. Since the scope of modeling in the case was quite constrained, general process modeling languages such as EEML was found to be overly expressive since it is meant to be used across a larger set of modeling task (including simulation and as a basis for manual and interactive activation). We neither took into account how possible tool support can address this problem.

It can be argued that our valuation scheme is somewhat simplistic (flat grades on a 1-10 scale that is summarized). We neither used the first classification on importance of the criteria for weighting. An alternative to flat grading is to use pairwise comparison and AHP on the alternatives[11]. The weighting between expressiveness, ease of learning and understanding can also be discussed. For later evaluations of this sort, we would like to use several variants of grading schemes to investigate if and to what extent this would impact the result.

This said, we should not forget that language quality properties are never more than means for supporting the model quality. Thus instead of only evaluating modeling languages 'objectively' on the generic language quality features of expressiveness and comprehension, it is very important that these language quality goals are linked to model quality goals to more easily adapt such a generic frameworks to the task at hand.

REFERENCES

[1] Arnesen, S. (2001) *Evaluering av språk for arbeidsprosessmodellering (In Norwegian).* Project report, IDI, NTNU, Trondheim. Norway.

[2] Berger, P. and Luckmann, T. (1966) *The Social Construction of Reality: A Treatise in the Sociology of Knowledge*. Penguin

[3] Booch, G., Rumbaugh, J. & Jacobson, I (1999). *The Unified Modeling Language: User Guide* Addison-Wesley.

[4] Carlsen, S. (1997). *Conceptual Modeling and Composition of Flexible Workflow Models*. Ph.D.-thesis Information Systems Group, Department of Computer and Information Science, Faculty of Physics, Informatics and Mathematics, Trondheim, Norway, NTNU - Norwegian University of Science and Technology.

[5] Dahlbom, B. (1991) The idea that reality is socially constructed in Floyd, C., Zullighoven, H., Budde, R., and Keil-Slawik, R. *Software Development and Reality Construction Springer* 101-126.

[6] EXTERNAL (1999) *EXTERNAL - Extended Enterprise Resources, Networks And Learning, EU Project, IST-1999-10091, New Methods of Work and Electronic Commerce, Dynamic Networked Organizations.* Partners: DNV, GMD-IPSI, Zeus E.E.I.G., METIS, SINTEF Telecom and Informatics, 2000-2002.

[7] Gjersvik, R., J. Krogstie, A. Følstad (2001) Participatory Development of Enterprise Process Models. in *Evaluation of Modeling Methods in Systems Analysis and Design (EMMSAD)*, Interlaken, Switzerland, 4-5 June.

[8] IDEF-0: Federal Information Processing Standards Publication 183, (1993) December 21, Announcing the Standard for Integration Definition For Function Modeling (IDEF-0)

[9] Krogstie, J. (1995). *Conceptual Modeling for Computerized Information System Support in Organization*. Unpublished doctoral dissertation, NTH, Trondheim, Norway.

[10] Krogstie, J., Lindland, O.I., & Sindre, G. (1995). Defining Quality Aspects for Conceptual Models. In E. D. Falkenberg, W. Hesse, & A. Olive (Eds.). *Proceedings of the IFIP8.1 working conference on Information Systems Concepts (ISCO3); Towards a consolidation of views, March 28-30* (pp. 216-231), Marburg, Germany.

[11] Krogstie, J. (1999). Using Quality Function Deployment in Software Requirements Specification. In A. L. Opdahl, K. Pohl, & E. Dubois. *Proceedings of the Fifth International Workshop on Requirements Engineering: Foundations for Software Quality (REFSQ'99), June 14-15*, (pp. 171-185), Heidelberg, Germany.

[12] Krogstie, J. & Sølvberg, A. (2000) *Information Systems Engineering: Conceptual Modeling in a Quality Perspective*, Draft of Book, Information Systems Groups, NTNU, Trondheim, Norway.

[13] Krogstie, J. (2001) Using a Semiotic Framework to Evaluate UML for the Development of Models of High Quality in *Unified Modeling Language: Systems Analysis, design, and Development Issues.* K. Siau and T. Halpin, IDEA group.

[14] Oei, J. L. H. (1995). A meta model transformation approach towards harmonization in information system modeling. In E. D. Falkenberg, W. Hesse, & A. Olive (Eds.). *Proceedings of the IFIP8.1 working conference on Information Systems Concepts (ISCO3); Towards a consolidation of views, March 28-30* (pp. 216-231), Marburg, Germany.

[15] Opdahl, A., Henderson-Sellers, B., & Barbier, F. (1999). An Ontological Evaluation of the OML Meta-model. In E. Falkenberg, K. Lyytinen, & A. Verrijn-Stuart (Eds.*), Proceedings of the IFIP8.1 working conference on Information Systems Concepts (ISCO4); An Integrated Discipline Emerging September 20-22* , Leiden, , (pp. 217-232) The Netherlands.

[16] Rummler, G. A. and Brache, A. P. Improving Performance (1995) : How to Manage the White Space on the Organizational Chart. A Practical Guide for Managing Organizations, Processes, and Jobs. Jossey-Bass.

[17] Wand, Y. & Weber, R. (1993). On the Ontological Expressiveness of Information Systems Analysis and Design Grammars. *Journal of Information Systems 3*(4), 217-237.

[18] Wieringa, R. (1998). A Survey of Structured and Object-Oriented Software Specification Methods and Techniques. *ACM Computing Surveys 30* (4) December, 459-527.

[19] Østbø, M. (2000). *Anvendelse av UML til dokumentering av generiske systemer (In Norwegian).* Master Thesis Høgskolen i Stavanger, Norway, 20 June.