Assessing Business Processing Modeling Languages Using a Generic Quality Framework

Anna Gunhild Nysetvold and John Krogstie

Norwegian University of Science and Technology,
Institute of Computer and Information Sciences
and SINTEF Telecom and Informatics, Norway.
krogstie@idi.ntnu.no

Abstract. We describe in this paper an insurance company that has recently wanted to standardize on business process modeling language. To perform the evaluation, a generic framework for assessing the quality of models and modeling languages was specialized to the needs of the company. Three 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 exists a large number of business process modeling languages. Deciding which modeling language to use for a specific task is often done in an ad-hoc fashion by different organizations. In this paper we present the work within an insurance company, which have a perceived need for using process modeling to support the integration of their business systems across different functions of the organization.

We have earlier developed a general framework for assessment of quality of models, where one type of means to support different quality goals, is criteria for the language to be used for modeling, also termed language quality (Krogstie, 2001). This paper presents an example of using and specializing the quality framework for the evaluation and selection of a modeling language for enterprise process modeling for the insurance company. The need for such specialization is grounded on work on task-technology fit (Goodhue et al, 1995). A similar use of the framework for comparing process modeling languages in an oil company has been reported in (Krogstie and Arnesen, 2004). Although similar, we will see that due do different overall goals of process modeling, the criteria derived from the quality framework ended up different in the work reported in this paper. This is also due to that we in this investigation have added aspects of organizational appropriateness of the approach.

1.1 The structure of this paper

The next section describes the quality framework, with a focus on language quality. Section three describes the 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

The model quality framework (Krogstie, Lindland and Sindre 1995; Krogstie and Sølvberg 2003; Krogstie 2001) is used as a starting point for the discussion on language quality. 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 defined) 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.
- ***M***, the externalized model.
- $\boldsymbol{K_s}$, the relevant explicit knowledge of the set of stakeholders being involved in modeling. A subset of the audience is those actively involved in developing models, and their knowledge is indicated by $\boldsymbol{K_M}$.
- ***I***, the social actor interpretation, i.e., the set of all statements which the audience at a given time thinks that an externalized model consists of.
- ***T***, the technical actor interpretation, i.e., the statements in the model as 'interpreted' by different modeling tools.

The model quality types are defined as relations between these sets.

- 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 to the audience.
- Empirical quality deals with HCI-ergonomics for documentation and modeling-tools.
- 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 correspondence between the audience interpretation ***I*** of a model ***M*** and his or hers current knowledge ***K*** of ***D***.
- Pragmatic quality is the correspondence between the model ***M*** and the audience's interpretation of it (***I***).
- The goal defined for social quality is 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 are being addressed through the model (organizational goal completeness).

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

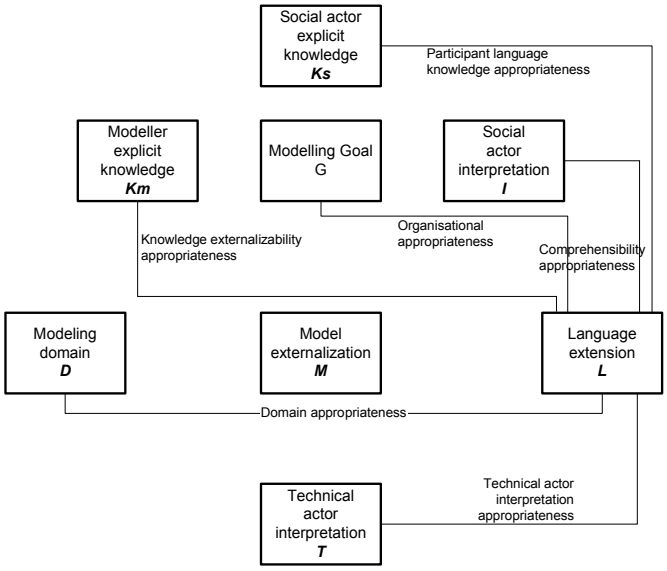

Fig. 1. Language quality related to the quality framework

As illustrated in Figure 1, six quality areas for language quality are identified:

Domain appropriateness

Ideally, the language must be powerful enough to express anything in the domain, i.e. not having construct deficit (Wand & Weber, 1993). On the other hand, you should *not* be able to express things that are not in the domain; i.e. what is termed construct excess (Wand & Weber, 1993). The only requirement to the notation is that it is able to represent all concepts in the language. Domain appropriateness is primarily a mean to achieve physical quality, and through this, to be able to achieve semantic quality.

Participant language knowledge appropriateness
This area relates the knowledge of the stakeholder 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 stakeholder or modeler. On the other hand the knowledge of the stakeholder is not static, i.e. it is possible to educate persons in the use of a specific 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 help achieve physical and pragmatic quality.

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

Comprehensibility appropriateness
This area relates the language to the social actor interpretation. For the concepts of the language we have:

- The concepts of the language should be easily distinguishable from each other. (Vs. construct redundancy (Wand & Weber, 1993)).
- The number of concepts should be reasonable. If the number has to be large, the concepts should be organized hierarchically and/or in sub-languages of reasonable size linked to specific modeling tasks. or viewpoints.
- The use of concepts should be uniform throughout the whole set of statements that can be expressed within the language.
- The language must be flexible in the level of detail.

As for the notation, 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 (What Goodman (1976) terms syntactic disjointness).
- The use of symbols should be uniform i.e. a symbol should not represent one phenomenon in one context and another one in a different context. Neither should different symbols be used for the same phenomenon in different contexts.
- One should strive for symbolic simplicity.
- One should use a uniform writing system: All symbols (at least within each sub-language) should be within the same writing system (e.g. non-phonological such as pictographic, ideographic, logographic, or phonological such as alphabetic).
- The use of emphasis in the notation should be in accordance with the relative importance of the statements in the given model

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).

Organizational appropriateness
relates the language to standards and other organizational needs within the organizational context of modeling. These are a mean to support organizational quality.

A number of sub-areas are identified for each of the six areas of language quality, and in (Østbø, 2000) approximately 70 possible criteria were identified. We will return to how this extensive list has been narrowed down and specialized for the task at hand.

3 Description of the case and the evaluation approach

The insurance company in our case has a large number of life insurance and pension insurance customers. The insurances are managed by a large number of systems of different age and using different technology. The business processes of the company go across systems, products and business areas, and the work pattern is dependant on the system being used. The company has modernized its IT-architecture. The IT-architecture is service-oriented, based on a common communication bus and an EAI-system to integrate the different system. To be able to support complete business processes in this architecture, there is a need for tools for development, evolution and enactment of business processes.

Goals for business process modeling in the insurance company

Before discussing the needs of the case organization specifically, we outline the main uses of enterprise process modeling. Five 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 with other people.
2. Computer-assisted analysis: To gain knowledge about the enterprise through simulation or deduction.
3. Business process management e.g. in connection to following up ISO-certification.

4. Model deployment and activation: To integrate the model in an information system and thereby make it actively take part in the work performed by the organization
5. To give the context for a traditional system development project, without being directly deployed and activated.

Company requirements

The main goal of the insurance company was related to the fifth area above, to give the context for traditional systems development to support the integration of their business systems across different functions of the organization, and at the same time support human sense-making and communication (area 1). A general set of requirements to a modeling language based on the discussion of language quality in section 2 is outlined in (Østbø, 2000). These were looked at relative to the requirements of the case organization, and their importance was evaluated. The analysis together with the case organization resulted in the requirements in table 1.

Table 1. Overview of evaluation criteria

No	Requirement	Type of
1.1	The language should support the following concepts (a) processes, that must be possible to decompose (b) activities (c) actors/roles (d) decision points (e) flow between activities, tasks and decision points	Domain appropriateness
1.2	The language should support (a) system resources (b) states	"
1.3	The language should support basic control patterns (van der Aalst, 2003)	"
1.4	The language should support advanced branching and synchronization patterns	"
1.5	The language should support structural patterns	"
1.6	The language should support patterns involving multiple instances	"
1.7	The language must support state based flow patterns	"
1.8	The language must support cancellation patterns	"
1.9	The language must include extension mechanisms to fit the domain	"
1.10	Elements in the process model must be possible to link to a data/information model	"
1.11	It must be possible to make hierarchical models	"
2.1	The language must be easy to learn, preferably being based on a language already being used in the organization	Participant language knowledge app.

2.2	The language should have an appropriate level of abstraction	“
2.3	Concepts should be named similarly as it is in the domain	“
2.4	The external representation of concepts should be intuitive to the stakeholders.	“
2.5	It should exist good guidelines for the use of the language	"
4.1	It must be easy to differentiate between different concepts	Comp.app.
4.2	The number of concepts should be reasonable	“
4.3	The language should be flexible in precision	"
4.4	It must be easy to differentiate between the different symbols in the language	"
4.5	The language must be consistent not having one symbol to represent several concepts, or more symbols that express the same concept.	"
4.6	One should strive for graphical simplicity	“
4.7	It should be possible to group related statements	“
5.1	The language should have a formal syntax	Technical actor app.
5.2	The language should have a formal semantics	“
5.3	It must be possible to generate BPEL –documents from the model	“
5.4	It must be possible to represent web-services in the model	“
5.5	The language should lend itself to automatic execution and testing	“
6.1	The language must be supported by tools that are either already available or can be made easily available in the organization	Organizati onal app.
6.2	The language should support traceability between the process model and any automated process support system	“
6.3	The language should support the development of models that can improve the quality of the process.	“
6.4	The language should support the development of models that help in the follow-up of separate insurance cases	“

4 Evaluation

The overall approach to the evaluation was the following: First a short-list of relevant languages was identified by us and the case organization in co-operation. The chosen languages were then evaluated according to the selected criteria on a 0-3 scale. To look upon this in more detail, all languages were used for the modeling of several real cases from the insurance company using a modeling tool that could accommodate all the selected languages (which in our case was METIS (Lillehagen, 1999)). By showing the resulting models and evaluation results to persons from the company, we got feedback and corrections both on the models and our grading. The models were

also used specifically to judge the participant language knowledge appropriateness and comprehensibility appropriateness. We also received feedback from modeling experts on the grading.

Based on discussions with persons in the case-organization and experts on business process modeling, three languages were selected for comparison. These will be briefly described: For a longer description see the report (Nysetvold, 2004) and the cited references.

EEML (Extended Enterprise Modeling Language) (EXTERNAL, 1999) was originally developed in the EU-project EXTERNAL as an extension of APM (Carlsen, 1997), and has been further developed in the EU projects UEML and ATHENA. The language has constructs to support all modeling categories described above.

The following main concepts are provided:

- Task with input and output ports (which are specific types of decision-points)
- General decision-points
- 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 has 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. In addition EEML contains constructs for goal modeling, organizational modeling and data-modeling.

UML 2.0 activity-diagrams (Fowler, 2004)

An activity-diagram can have the following symbols

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

In addition, a number of constructs is provided to support different kinds of control-flows.

BPMN (BPMN, 2005) is a notation aiming to be easily understandable and usable to both business users and technical system developers. It also tries to be formal enough to be easily translated into executable code. By being adequately formally defined, it is meant to create a connection between the design and the implementation of business processes.

BPMN defines Business Process Diagram (BPD), which can be used to create graphical models especially useful for modeling business processes and their operations. It is based on a flowchart technique - models are networks of graphical objects (activities) with flow controls between them.

The four basic categories of elements are:

- Flow Objects

- Connecting Objects
- Swimlanes
- Artifacts (not included here)

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 on 0-3 (earlier evaluations of this sort (Krogstie and Arnesen 2004) have used a 1-10 scale)

Table 2. Grading scale

Grade	Explanation
0	There is no support of the requirement
1	The requirement is partly supported
2	There is satisfactory support of the requirement
3	The requirement is very well supported

The reasoning behind the grading can be found in (Nysetvold, 2004), and is not included here due to space limitations. Table 3 gives an overview of the results. The last three rows summarize the results.

Table 3. Comparison table with all the evaluations collected

		Grading of languages		
No.	**Requirement description**	**UML AD**	**BPMN**	**EEML**
1.1	Listed concepts	3	3	3
1.2	Listed concepts	2	2	3
1.3	Basic control patterns	3	3	3
1.4	Advanced branching and synchronization patterns	0	0,5	3
1.5	Structural patterns	0	1,5	1,5
1.6	Patterns involving multiple instances	1,5	1,5	2
1.7	State based flow patterns	1	1	2
1.8	Cancellation patterns	3	3	3
1.9	Extension mechanisms to fit the domain	3	1	1
1.10	Elements in the process model must be possible to link to a data/information model	3	1	3
1.11	Hierarchical models	3	3	3
2.1	The language must be easy to learn, preferably being based on a language already being used in the organization	2	3	1

2.2	Appropriate level of abstraction	3	3	3
2.3	Concepts should be named similarly as it is in the domain	1	3	2
2.4	Intuitive representation to the stakeholders	2	2	2
2.5	Good guidelines for the use of the language	2	2	1
4.1	Easy diff. between different concepts	3	3	2
4.2	Number of concepts should be reasonable	3	3	1
4.3	The language should be flexible in precision	1	2	3
4.4	Easy to differentiate between the different symbols in the language	2	2	1
4.5	The language must be consistent.	3	3	3
4.6	One should strive for graphical simplicity	3	2	1
4.7	Grouping of related statements	1	1	2
5.1	The language should have a formal syntax	3	3	3
5.2	Formal semantics	1	3	2
5.3	Generate BPEL –documents from the model	2	3	0
5.4	Represent web-services in the model	1	3	1
5.5	Automatic execution and testing	1	3	2
6.1	The language must be supported by available tools.	3	3	1
6.2	Traceability between the process model and any automated process support system	2	3	1
6.3	Models that can improvement the quality of the process.	1	1	1
6.4	The language should support the development of models that help in the follow-up of separate cases	1	1	2
	Sum	63,5	72,5	63,5
	Sum without technical actor appropriateness	55.5	57,5	55,5
	Sum without participant language knowledge appropriateness	53,5	59,5	53,5

None of the languages satisfies all the requirements, but BPMN is markedly better overall. With 72.5 points BPMN scores 75% of maximum score, whereas the others score around 66%.

BPMN has the highest score in all categories, except for domain appropriateness, which is the category with highest weight, due to the importance of being able to express the relevant business process structures. EEML is found to have the best domain appropriateness, but loses to BPMN on technical actor appropriateness and participant knowledge appropriateness.

Comprehensibility appropriateness is the category which has the second highest weight, since the organization regards it to be very important that it is possible to use the language across the different areas of the organization, to improve communication between the IT-department and the business departments. In this category BPMN and Activity diagrams scores the same, which is not surprising given that they use the

same kind of swimlane metaphor as a basic structuring mechanism. EEML got a lower score, primarily due to the graphical complexity of some of the concepts, combined with the fact that EEML has a larger number of concepts than the others.

Participant language knowledge appropriateness and technical actor appropriateness was scored equally high, and BPMN score somewhat surprisingly high on both areas. When looking at the evaluation not taking technical actor appropriateness into account, we see that the three languages score almost equal. Thus it is in this case the focus towards the relevant implementation platforms (BPEL and web services) that in this case is putting BPMN on top. On the other hand, we see that this focus on technical aspect do not destroy for the language as a communication tool between people, at least not as it is regarded in this case.

In the category organizational appropriateness, BPMN and Activity diagrams score almost the same. The organization had for some time used activity diagrams, but it also appeared that tools supporting BPMN was available for the organization. The organization concluded that it wanted to go forward using BPMN for this kind of modeling in the future.

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 business 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, which it was also found to be by the people responsible for these aspects in the case organization. In an earlier use of the framework, with a different emphasis, UML activity diagrams got a much higher score than EEML, whereas here, they scored equally. (Krogstie and Arnesen, 2004).

It can be argued that the actual valuation is somewhat simplistic (flat grades on a 0-3 scale that is summarized). On the other hand, different kinds of requirements are weighted since the number of criteria in the different categories is different. An alternative to flat grading is to use pair-wise comparison and AHP on the alternatives (Krogstie, 1999). The weighting between expressiveness, technical appropriateness, organizational appropriateness and human 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 (where the modeling task typically has specific goals on its own). 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. This is partly achieved by the inclusion of organizational appropriateness, which is not used in earlier work applying the framework. The evaluation results are

also useful when a choice has been made, since those areas where the language does not score high can be supported through proper tools and modeling methodologies.

REFERENCES

W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, and A.P. Barros (2003). *Workflow patterns.* In Distributed and Parallel Databases, pages 5–52.

BPMN (2005) Business Process Modeling Notation. http://www.bpmn.org , last visited 12/3-2005

Carlsen, S. (1997). *Conceptual Modeling and Composition of Flexible Workflow Models*.unpublished 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.

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.

Fowler, M. (2004) UML Distilled: A Brief Guide to the Standard Object Modeling Language, third edition, Addison-Wesley

Goodhue, D. and Thompson, R. (1995) Task-Technology Fit and Individual Performance, *MIS Quarterly* June.

Goodman, N. (1976). *Languages of Art: An Approach to a Theory of Symbols*. Hackett, Indianapolis

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.

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.

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.

Krogstie, J. & Sølvberg, A. (2003) *Information Systems Engineering: Conceptual Modeling in a Quality Perspective*, Kompendiumforlaget, Trondheim, Norway.

Krogstie, J. and S. Arnesen, (2004) Assessing Enterprise Modeling Languages using a Generic Quality Framework, in Information Modeling Methods and Methodologies, J. Krogstie, K. Siau, and T. Halpin, Editors. 2004, Idea Group Publishing.

Lillehagen, F. Visual Extended Enterprise Engineering Embedding Knowledge Management, Systems Engineering and Work Execution, IEMC '99, IFIP International Enterprise Modeling Conference, Verdal, Norway, 1999.

Nysetvold, A. G. Prosessorientert IT-arkitektur, Project Thesis (In Norwegian), IDI, NTNU, November 2004.

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

Stephen A. White: *Introduction to BPMN*. IBM Corporation, 2004.

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