Process Heuristics to Achieve Requirements Specifications of Feasible Quality

Guttorm Sindre * John Krogstie

Faculty of Electrical Engineering and Computer Science
University of Trondheim, Norway

Abstract

The notion of quality in requirements specifications is poorly understood, although recent frameworks such as those of Lindland et al. and Pohl have tried to take a more systematic approach to the problem area. A major weakness of these frameworks is that they are on a high level of abstraction, and thus have little in the form of practical guidelines to offer practitioners.

Based on an extension of the Lindland framework this article proposes a set of process heuristics for guiding decisions in the requirement specification process. Although the heuristics are incomplete, the framework has thus been brought a significant step closer to practical applicability

Keywords: **Requirements engineering, conceptual modelling, quality, process heuristics**

1 Introduction

Previously propositions of quality goals for requirements specifications or conceptual models have included many useful aspects, but unfortunately in an unsystematic form. Two recent frameworks by Lindland et al. and Pohl [18, 20] have attempted to take a more structured approach to understanding this area. Still, both these need more development before they can result in concrete guidelines for requirements specification. Comparisons between the Lindland framework and the work of Pohl and of the FRISCO task group [17] on semiotic levels have increased our understanding of these areas as reported in [14, 15]. Although this extended framework contributes to our understanding of quality issues with respect to conceptual modeling, the contribution so far lies on a rather high level of abstraction. In this paper we combine the proposed framework with suggestions for process heuristics as a first step of making the framework more practically applicable.

The article is structured as follows: Section 2 gives a brief introduction to the overall framework before means to achieve these goals on the different levels are outlined in chapter 3. Use of process heuristics based on the framework is given in chapter 4. A brief comparison with related work is given in chapter 5, before summing up the contributions of the paper.

*Direct responses to this paper to Dr. Guttorm Sindre, mail: UNIT-NTH/IDT, N-7034 TRONDHEIM, NORWAY; email: Guttorm.Sindre@idt.unit.no; phone: +47 73593469, fax: + 47 73594466.

2 Framework for Quality of Conceptual Models

We will first outline the overall framework for discussing quality of conceptual models. The main concepts and their relationships are shown in figure 1. We have taken a set-theoretic approach to the discussion of the different quality aspects and quality has been defined as the correspondence between sets of statements [14, 15].

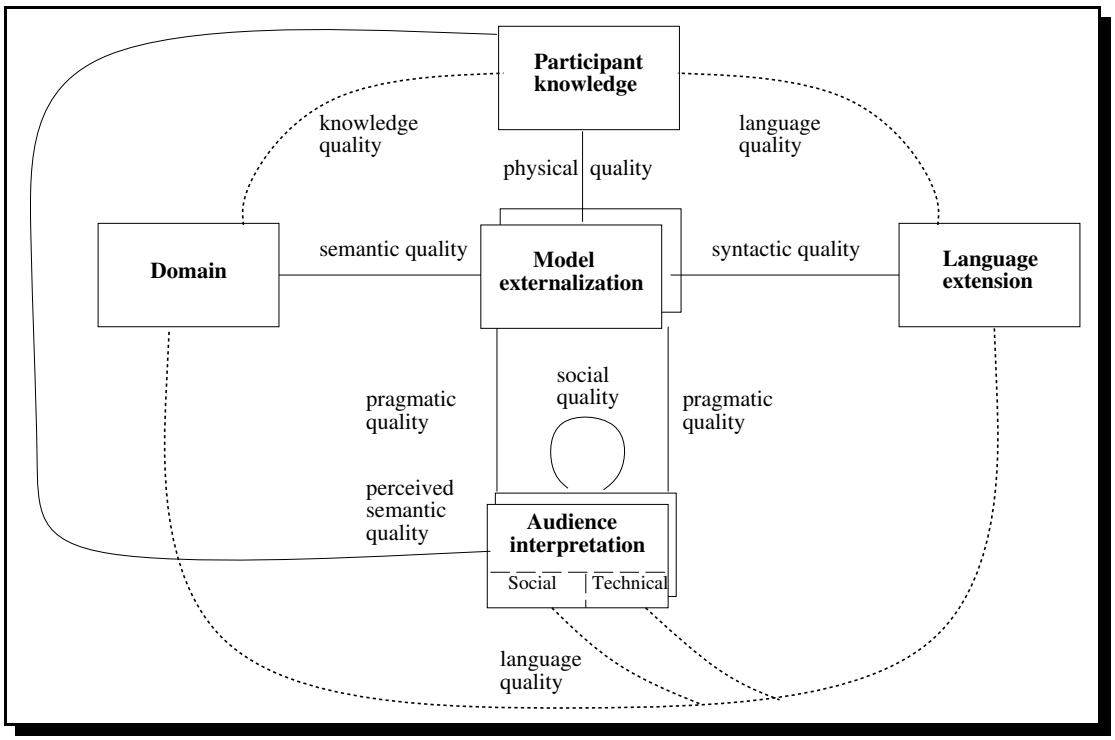

Figure 1: Framework for quality of conceptual models

- $\mathcal{A}$, the audience, the set of actors who need to relate to the model. The individual social actors are called the *participants* of the modelling process.
- $\mathcal{L}$, the language extension, i.e., the set of all statements that are possible to make according to the syntax of the modeling languages used.
- $\mathcal{M}$, the externalized model, i.e. the set of all statements in someones model of part of the perceived reality written in a language.
- $\mathcal{D}$, the domain, i.e., the set of all statements which would be regarded correct and relevant about the situation at hand. $\mathcal{D}$ denotes the 'ideal' knowledge and are used as a conceptual fixpoint to enhance terminology discussions.
- $\mathcal{K}$, the relevant explicit knowledge of the audience, i.e., the union of the set of statements, $\mathcal{K}_1$,...,$\mathcal{K}_k$, one for each participant.
- $\mathcal{I}$, the audience interpretation, i.e. the set of all statements which the audience thinks that an externalized model consists of. Just like the externalized model itself, its interpretation is projected into $\mathcal{I}_1, ..., \mathcal{I}_n$ denoting the statements in the externalized model that are understood by each social actor.

Table 1 shows an overview of the goals and means that have been identified on the different levels. Language quality goals are looked upon as means to reach other goals and are not discussed here. We will discuss each quality aspect below focussing on means before discussing process heuristics in the framework.

3 Toolbox of Means

Before looking at detailed process heuristics, we will describe the proposed quality means for semantic, pragmatic, social, and knowledge quality in some more detail. The overview below is not complete, although should contain the most important techniques for supporting the improvement of model quality.

3.1 Semantic quality

Semantic quality is the correspondence between the model and the domain, where the domain is considered the ideal knowledge about the situation to be modelled. The framework contains two semantic goals; *validity* — all statements made by the model are correct and relevant to the problem, and *completeness* — the model contains all the statements which would be correct and relevant about the domain.

Perceived semantic quality is the similar correspondence between the participants' interpretation of a model and his or hers current knowledge of the domain.

Attempts at reaching a state of total validity and completeness will lead to unlimited spending of time and money on the modeling activity. The time to terminate a modeling activity is thus not when the model is "perfect", but when it has reached a state where further modeling is less beneficial than applying the model in its current state. With respect to this, feasible completeness and validity can be defined.

Activities for establishing higher semantic quality are statement insertion and deletion. Consistency checking is another activity here. To be able to do consistency checking, the model must be made in a formal language, and to enable and assess the impact of updates, it should be modifiable. This includes properties such as structure and locality of changes. The use of driving questions to increase completeness based on the already existing model is described in TEMPORA [21].

3.2 Pragmatic quality

Pragmatic quality is the correspondence between the model and the audience's interpretation of it. Comprehension is an ideal goal, just like validity and completeness. Again it will be useful to introduce the notion of feasibility.

Some of the activities to achieve feasible pragmatic quality are:

- Inspection: Manually reading a model.
- Transformations: Generally to transform the model into another model in the same language, e.g., by *layout modification* or *filtering*.
- Translation: Translate the model from one language to another language.
- Audience training: Educate the audience in the syntax and semantics of the modelling languages.

Quality types	*Goals*	*Means*	
		Model properties	**Activities**
Physical quality	Externalisation	Domain appr.ness Knowledge appr.ness	
	Internalizability	Persistence Availability	DB-activities
Syntactic quality	Syntactic correctness	Formal syntax	Error prevention Error detection Error correction
Semantic quality	Feasible validity	Formal semantics	Consistency checking Driving questions
	Feasible completeness	Modifiability	Statement insertion Statement deletion
Perceived sem.quality	Perceived validity Perceived completeness		Participant training
Pragmatic quality	Feas. comprehension	Expressive economy Aesthetics	Inspection Visualisation Filtering Diagram layout Paraphrasing Explanation Participant training
		Executability	Execution Animation Simulation
Social quality	Feasible agreement	Inconsistency handling	Viewpoint analysis Conflict resolution Model merging
Knowledge quality	Feasible k. completeness Feasible k. validity		Stakeholder ident. Participant selection Problem selection Participant training

Table 1: Framework for model quality

Several specific applications of translations and transformations and combinations of these can be imagined:

- Model execution: Translate the model to a model in an executable language [6, 10].
- Animation: Make systems dynamics explicit by using moving pictures [19].
- Explanation generation: An explanation generator can answer questions about a model and its behaviour [7].

- Simulation: Use statistical assumptions to anticipate how a system built according to the model would behave if implemented. Simulation can be combined with execution, animation, and explanation [10, 19].

3.3 Social quality

The goal defined for social quality is *agreement* among participant interpretation.

It is not given that all participants will come to agreement. Few decisions are taken in society under complete agreement, and those that are are not necessarily good. To answer this we define the concept of *feasible agreement:*

Feasible agreement is achieved if feasible comprehension is achieved and inconsistencies between statements in the different interpretations of the model $\mathcal{I}_i$ are resolved by choosing one of the alternatives when the benefits of doing this is less than the drawbacks of working out an agreement.

Some activities for achieving feasible agreement are:

- Viewpoint analysis [16]: This includes techniques for comparing two or more externalized models and find discrepancies and similarities between the models.
- Conflict resolution: Specific techniques for this can be found in the area of computer supported cooperative work, see e.g [4, 9].

3.4 Knowledge quality

The "quality" of the participant knowledge can be expressed by the relationships between the audience knowledge and the domain.

To get a *good enough* knowledge about the domain, careful *participant selection* based on *stakeholder identification* is necessary (if you have a problem and can choose the participants), or alternatively, careful *problem selection* (if the participants are given, but not the problem to be solved). In the case that both participants and problem are more or less given, and not fitting too well, some development in terms of training of the participants may be necessary. Just as for the other aspects of quality, it will be possible to talk about *feasible knowledge quality*, meaning that the knowledge of the audience could still be improved, but the benefit of improving it through additional education or the hiring of additional experts or including additional stakeholders will be less than the drawbacks of mistakes made.

4 Process Heuristics

We now move on to discussing heuristics based on information which could be collected during requirements engineering. The information will be used to:

- guide the actions of the current project.
- guide the evolution of the organization's system development methodology.

These issues can be called process support and meta-process support, respectively [12]. Both these issues will be discussed here. Section 4.1 gives an overall view of the

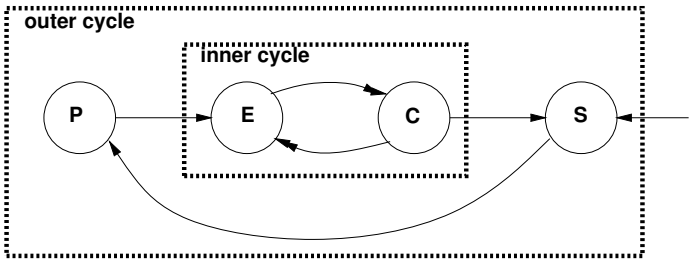

Figure 2: The SPEC cycle

RE-process, whereby section 4.2 discusses what information should be collected. Section 4.3 then presents heuristics for in-process guidance, and section 4.4 for meta-process guidance.

4.1 Overall idea of the process

In [18], the RE process is vaguely described as consisting of cycles of expansion and consolidation of $\mathcal{M}$. In the following we will elaborate more on this idea, to prepare the ground for a discussion of process heuristics. Our idea of the RE process is shown as a state machine in Figure 2. The diagram can be explained as follows:

P – preparation: In this state, the organization is performing actions in preparation of the requirements capture itself, e.g., stakeholder identification, selection of project participants, training, and planning.

E – expansion: Requirements are being stated, hence $\mathcal{M}$ is growing. During expansion, statements may be made more or less uncritically, i.e., thorough validation is not undertaken, and there will be errors introduced in $\mathcal{M}$. Still, as long as some valid statements are made, $\mathcal{M}$'s degree of *completeness* will grow.

C – consolidation: The requirements (especially those captured in the previous expansion phase) are consolidated with respect to *validity*, *comprehension*, and *agreement*, as defined in section 2.

S – suspension: The RE activity is suspended. There may be several reasons for suspension. The requirements may have been agreed upon and "frozen" (e.g., to start the design phase), or the project may have been aborted (e.g., because it was found infeasible).

As can be seen, this diagram consists of an inner cycle of expansion and consolidation, and an outer cycle including preparation and suspension. The starting state has been defined as **S**, i.e., before you do anything, you are in a state of suspension. Hence, this cycle can be called the **SPEC**-cycle. The fact that there is no accepting state reflects the view that RE is never finished (i.e., "frozen" requirements may be revisited later, models may be reused in new projects etc.).

Considering the entire requirements specification, there will be a lot of work in parallel by various participants. Hence, it is impossible to observe the simple state-changes of Figure 2 for the specification as a whole. It applies to smaller parts of the specification, the entire process then being composed of multiple such cycles.

The focus of the heuristics to be discussed here will mostly be on the inner cycle, in particular the switching between **E** and **C**. This switch can be based on the following:

- *resource limit:* you are supposed to use a certain amount of time or manpower for **E**, then go to **C**, and vice versa, and similarly for **E** and **C** together vs. **S**.
- *chunk size:* i.e., the amount of statements made at one visit in **E**. When this size has been reached, there may be a policy to switch to **C**.
- *progress:* you observe the progress made at **E** or **C** and switch when this has fallen below a certain threshold. The progress *will* decrease when the process has been in the same state for a while because of a phenomenon that may be called *exhaustion:* the most evident statements will be captured first, and the most evident errors found first, leaving ever harder problems for further work. Moreover, staying too long in **E** will yield a big chunk, for which incomprehension and disagreement is likely to hinder further growth of the model.

Of course, the three above strategies will be combined to make a practical RE process.

4.2 Information gathering

To start with, we will identify what information should be delivered to enable the heuristics. Then we will continue by discussing how this information can be obtained. Information gathering is a significant problem which has to be treated lightly here since the purpose is to arrive at heuristics.

To determine the progress of RE and the extent to which feasible quality has been reached, it is useful to know the point estimates for the quality goals, and their corresponding ratios (vs. resource consumption). In addition, model size has been included because it is important in the management of expansion, and model value because it is important in considerations about feasibility.

- perceived validity (PV), and ratio of perceived validity (PVR)
- perceived completeness (PC), and (PCR)
- perceived pragmatic quality (PP), and (PPR)
- perceived social quality (PS), and (PSR)
- perceived knowledge quality (PK), and (PKR)
- model size (MS), and (MSR)
- model value (MV), and (MVR)

For all these, there should be break-downs on separate activities, as well as possible aggregates over techniques, types of problem, types of participants etc., available.

The model size should be possible to obtain automatically. For the other information, we need to register:

- for all parts of the model, which have been perceived as complete (within their scope), which have been perceived as incomplete, and how big is this incompletion estimated to be

- for all statements, which have been acknowledged as comprehended, which have been turned out to be incomprehensible, and which have not yet been checked.
- for all statements, which have been agreed upon and by whom, which have been disagreed upon and by whom, and which have not yet been checked.
- for each activity (here: each visit at **E** or **C**), how much resources spent
- for each activity, what is the value increment to the model

Some of these are easier to obtain than others. The most complicated are perceived completeness and model value. Even if these are dropped, it will be possible to provide some useful heuristics, as will be shown later. The phrase "for all statements" may give the reader the suspicion that the above requires an enormous registration work. However, this need not be the case. E.g., discussing agreement of an ER diagram, it should be easy to implement functionality for quickly selecting larger parts of the diagram to mark them as agreed or disagreed upon.

Problems with information collection

As indicated in [18], information collected to guide the project itself becomes a part of $\mathcal{M}$, and is hence subject to the same quality considerations as the rest of $\mathcal{M}$. Information may be invalid, incomplete, miscomprehended or disagreed upon. Hence, the information mentioned above must also be subject to feasibility considerations, just as the specification itself.

For the sake of feasibility, time, cost and value estimates should probably be made at the granularity of chunks, not at the granularity of single statements, which would create both an enormous overhead and a lot of uncertain guesses . Granularity is also important when considering the application of the ratios in heuristics.

4.3 Heuristics to guide the current process

Heuristics will be presented as observed symptoms and possible suggested actions. The list is not supposed to be exhaustive, and as one symptom may have several causes, there are several possible actions for each symptom. Heuristics must be evaluated by the participants in any specific case.

Expansion heuristics

Symptom E1: resource consumption $\sim$ limit

Symptom E2: MS $\sim$ recommended chunk size

Action (E1 or E2): switch to consolidation. For E2, if the recommended size was reached very easily, it might also be that the problem is simpler than assumed, so that it can be interesting to increase the recommended chunk size.

Symptom E3: MSR $<$ min.threshold (expansion is getting unproductive)

Symptom E4: MVR < 1 (growth of model value is less than resources being spent, i.e., work currently being done yielding deficit)

Actions (E3 or E4): 1. switch to consolidation (if the problem is due to significant incomprehension or disagreement). If the chunk is well within the recommended size, it may also be sensible to lower the recommended chunk size.

2. switch to other techniques (if the problem is due to exhaustive use of some techniques and there are others which can be tried).

3. involve new participants (if the problem is due to exhaustive use of some participants and there are others which do possess relevant knowledge)

As shown here, an observed symptom from the collected information will not define a unique action; there has to be further considerations by the participants.

Consolidation heuristics

Consolidation heuristics are more complex than expansion heuristics, since there are more goals and measures involved. To save space, we will avoid to list the most obvious heuristics. Hence, it is sensible to address pragmatic quality before validity, completeness and agreement because comprehension of the model is necessary to achieve anything else. Further, it is sensible to address validity and agreement before completeness. Guidance for this sequencing can be done by heuristics investigating the values PP, PV, PS, PC. This will not be discussed below. Instead we will look at symptoms indicating problems with the consolidation being done.

Symptom C1: resource consumption $\sim$ limit.

Action: switch to expansion (if resources available) or to suspension.

Symptom C2: PVR, PPR, PSR $<$ min. threshold (i.e., consolidation is getting exhausted).

Symptom C3: MVR < 1 (i.e., value being added to the model by consolidation is less than resources being spent).

Actions (C2 and C3): 1. conclude that feasible quality has been reached (if the values for PV, PC, PP, PS are good).

2. terminate this part of the project as hopeless, or at least backtrack to some previous decision (if the values for PV, PC, PP, PS are bad and it is impossible to see any way out).

3. switch to expansion (if the value for PC worse than PV, PP, PS).

4. switch to other techniques (if one or more of the values PV, PP, PS is unacceptable; concentrate on techniques applicable for the quality aspect which is most pressing).

5. involve other participants (if one or more of the values PV, PP, PS is unacceptable).

4.4 Heuristics to guide process evolution

By aggregating collected information over several projects, one can also find heuristics to guide the evolution of the RE process. Important questions are:

- what specification languages, methods, techniques, tools etc. are good for various categories of problems in the organization?
- what is the optimal chunk size for various categories of problems?
- what is the optimal statement growth ratio for various kinds of problems?
- what team constellations are good for various kinds of problems (e.g., what combinations of skills and knowledge)
- what kind of knowledge seems to be in shortage in the organization.

The results could be used to evaluate both approaches at the cycle-level and the composition of the entire RE process, i.e., suitable breakdown in cycles. In addition, post-project evaluations can be taken into concern here.

The discussion of meta-process heuristics in more detail, in the style done for in-process heuristics, is beyond the scope of this article due to space limitations.

5 Related work and Conclusion

The inner cycle of our SPEC-model can be seen to correspond to the RE validation cycle, e.g., [8]. However, whereas the traditional validation cycle usually has four steps, our has been collapsed to two steps, expansion covering both acquisition and representation, consolidation covering both transformation and validation. The use of only two steps was found more appropriate to couple the discussion of process to the quality framework of [14].

Meta-process support has been discussed within the software process community, e.g., work on software process engines [2, 12] for the evolution of rigid, enactable processes, and CMM [11] for the more high level improvement of processes through assessment and information gathering. What we are aiming for with our heuristics corresponds to the level 5 of CMM (optimizing), where the process is continuously evaluated and improved. Our work is not an effort towards enacting the RE process, and neither is it an assessment framework. Hence it is rather different from the above. The software process community does also discuss in-process support, through monitoring and enactment. One of the most relevant works here is [3], where machine-assisted data exploration of classified defect data is used to help the participants improve their process during development. However, this work concentrates on later phases where defects can be identified as failure to fulfill the requirements. For the requirements specification itself, this comparison is not available, and hence defect identification does indeed rely on quality goals appropriate for requirements specifications, such as those summarised in section 2.

Process support of the more knowledge-based kind has been discussed in the projects ACKnowledge [1] and DAIDA [13]. ACKnowledge's workbench, KEW, does

indeed include some of the features we are advocating, e.g., the observation of exhaustion at expansion, by which the workbench can suggest the application of an alternative tool.

5.1 Concluding remarks

The major distinction between our approach and the ones mentioned above is that we have undertaken a deeper discussion of the RE process and its quality goals. Hence, heuristics have been defined based on the perceived semantic, pragmatic, and social quality of a model. Although the heuristics are only preliminary, we believe that this paper has brought the framework developed in [14, 15, 18] a significant step closer to practical applicability.

Further work will be to apply the principles on specific methodologies for conceptual modelling. This work is currently being done in connection to the use of the PPP languages [5].

References

[1] A. Anjerwierden, B. Wielinga, and N. Shadbolt. Supporting knowledge acquisition: The ACKnowledge project. In L. Steels and B Lepape, editors, *ESPRIT-92 Knowledge Engineering*. 1992.

[2] S. C. Bandinelli, A. Fuggetta, and C. Ghezzi. Software process model evolution in the SPADE environment. *IEEE Transactions on Software Engineering*, 19(12):1128–44, December 1993.

[3] I. Bhandari et al. A case study of software process improvement during development. *IEEE Transactions on Software Engineering*, 19(12):1157–70, December 1993.

[4] J. Conklin and M. J. Begeman. gIBIS: A hypertext tool for exploratory policy discussion. *ACM Transactions on Office Information Systems*, 6(4):303–331, 1988.

[5] J. A. Gulla, O. I. Lindland, and G. Willumsen. PPP - An integrated CASE environment. In R. Andersen, J. A. Bubenko jr., and A. Sølvberg, editors, *Proceedings of the Third International Conference on Advanced Information Systems Engineering (CAiSE'91)*, pages 194–221, Trondheim, Norway, May 1991. Springer-Verlag.

[6] J. A. Gulla, O. I. Lindland, G. Willumsen, and A. Sølvberg. Executing, viewing and explaning conceptual models. In *Proceedings of the First International Conference on Requirement Engineering (ICRE-94)*, pages 166–175, Colorado Springs, USA, April 18-22 1994. IEEE.

[7] J. A. Gulla and G. Willumsen. Using explanations to improve the validation of executable models. In C. Rolland, F. Bodart, and C. Cauvet, editors, *Proceedings of the 5th International Conference on Advanced Information Systems Engineering (CAiSE'93)*, pages 118–142, Paris, France, June 8-11 1993. Springer Verlag.

[8] J. Hagelstein. Declarative approach to information systems requirements. *Knowledge-Based Systems*, 1(4):211–220, 1988.

[9] U. Hahn, M. Jarke, and T. Rose. Group work in software projects: Integrated conceptual models and collaboration tools. In S. Gibbs and A. A. Verrijn-Stuart, editors, *Multi-User Interfaces and Applications: Proceedings of the IFIP WG 8.4 Conference on Multi-User Interfaces and Applications*, pages 83–102. North-Holland, 1990.

[10] D. Harel. Biting the silver bullet: Toward a brighter future for system development. *IEEE Computer*, January 1992.

[11] W. Humphrey. *Managing the Software Process*. Addison-Wesley, Reading, MA, 1989.

[12] M. L. Jaccheri and R. Conradi. Techniques for process model evolution in EPOS. *IEEE Transactions on Software Engineering*, 19(12):1145–56, December 1993.

[13] M. Jarke et al. DAIDA: an environment for evolving information systems. *ACM Transactions on Information Systems*, 10(1):1–50, 1992.

[14] J. Krogstie, O. I. Lindland, and G. Sindre. Defining quality aspects for conceptual models. Presented at ISCO3, Marburg, Germany 28-30 March 1995.

[15] J. Krogstie, O. I. Lindland, and G. Sindre. Towards a deeper understanding of quality in requirements engineering. Accepted at CAiSE'95, Jyväskylä, Finland 12-16 June, 1995.

[16] J. C. S. P. Leite and P. A. Freeman. Requirements validation through viewpoint resolution. *IEEE Transactions on Software Engineering*, 17(12):1253–1269, December 1991.

[17] P. Lindgren ed. A framework of information systems concepts. Technical Report Interrim report, FRISCO, May 1990.

[18] O. I. Lindland, G. Sindre, and A. Sølvberg. Understanding quality in conceptual modelling. *IEEE Software*, pages 42–49, April 1994.

[19] C. E. Pegden, R. E. Shannon, and R. P. Sadowski. *Introduction to simulation using Siman*. McGraw-Hill, USA, 1990.

[20] K. Pohl. The three dimensions of requirements engineering: A framework and its applications. *Information Systems*, 19(3):243–258, April 1994.

[21] B. Wangler, R. Wohed, and S-E. Öhlund. Business modelling and rule capture in a CASE environment. In *Proceedings of the Fourth Workshop on The Next Generation of CASE Tools*, Twente, The Netherlands, 1993.