QoMo: A Modelling Process Quality Framework based on SEQUAL

P. (Patrick) van Bommel, S.J.B.A. (Stijn) Hoppenbrouwers,
H.A. (Erik) Proper, and Th.P. (Theo) van der Weide

Institute for Computing and Information Sciences, Radboud University Nijmegen
Toernooiveld 1, 6525 ED Nijmegen, The Netherlands, EU
{ P.vanBommel, S.Hoppenbrouwers, E.Proper, T.vanderWeide}@cs.ru.nl

Abstract This paper aims to contribute to the area of conceptual model quality assessment and improvement. We present a preliminary *modelling process*-oriented 'Quality of Modelling' framework (QoMo), mainly based on the established SEQUAL framework for quality of models. QoMo is based on knowledge state transitions, cost of the activities bringing such transitions about, and a goal structure for activities-for-modelling. Such goals are directly linked to concepts of SEQUAL. We discuss how goals for modelling can be linked to a rule-based way of describing processes for modelling. Such process descriptions hinge on strategy descriptions, which may be used descriptively (for studying/analysing real instances of processes) as well as prescriptively (for the guiding of modelling processes). Descriptive utility of the framework is critical for the quality/evaluation angle on processes-for-modelling, and reflects the main intended contribution of this paper.

1 Introduction

Interest in frameworks for quality and assessment of conceptual models has been gradually increasing for a number of years. A generic overview and discussion can be found in [10]. A key framework for analysis of the quality of conceptual models is the SEQUAL framework of Krogstie et al. [6,7,8]. This framework takes a semiotics-based view on modelling which is compatible with our own [4]. It is more than a quality framework for models as such, in that it includes not just the model but also the knowledge of the modellers, the domain modelled, the modelling languages, agreement between modellers, etc. (see section 2); it bases quality assessment on relations between such model-related items.

As argued in [5], in addition to analysis of the quality models, the *process* of which such models are a product should also be taken into account. We briefly summarize the main arguments here:

1. Though some have written about detailed stages in and aspects of "Ways of Working" in modelling, i.e. its process or procedure (for example, [3]), the detailed "how" behind the activity of creating models is still mostly art rather than science. There is, therefore, a purely scientific interest in improving our understanding of the operational details of modelling processes.
2. In addition, such a study should enable us to find ways of improving the modelling process (for example, its quality, efficiency, or effectiveness; from a more

methodological angle: reproducibility, stability, and precision; also traceability, and so on).

3. Indeed, some aspects of quality, it seems, can be better achieved through a good modelling process than by just imposing requirements on the end product and introducing a feedback cycle (iteration). This holds in particular (though not exclusively) for matters of validation and grounding in a socio-political context.
4. A score of more practical arguments follow from the ones above. For example, for improvement of case tool design, a more process-oriented approach seems promising.

As mentioned, models as such are not the only product of a modelling process. The entailing knowledge development, agreements, etc. are arguably as important. Hence, our process-oriented view suggests that we take aboard such additional *model items*, and quality concepts related to them. The latest SEQUAL version explicitly allows for this approach, though it does not make explicit use of a meta-concept such as "model item".

The importance of the modelling process in view of model quality is commonly confirmed in the literature in general, and yet we are unaware of any theoretical frameworks specifically focusing on modelling process quality.

If we want to evaluate a modelling process, we can take (at the least) the following four different points of view:

1. Measure the success of the process in fulfilling its goals. This boils down to using the current SEQUAL framework as-is, and directly link such an analysis of the model items to the success of the process. However, one might also analyse *intermediate steps* in the process against intermediate or sub-goals set. By evaluating partial or intermediary products (possibly in view of a prioritization of goals), the process may be *steered* along the way. Also, the steps described will eventually become so small that *essential* modelling goals/activities can perhaps be identified (steps in the detailed thinking process underlying modelling), opening up the black box of the modelling process.
2. The cost-benefit ratio: achievements set against the cost. Such cost again can hold for the process as a whole, but also for specific parts.
3. We can look at the process and the working environment as an information system. This then is a 2nd order information system: an IS that serves to develop (bring forth) information systems. The information system underlying the modelling process (probably but not necessarily including IT support) can be evaluated in a way similar to evaluation of information systems in general. In particular, aspects like usability and actability but also traceability and even portability are relevant here.
4. Views 1-3 concern operational evaluations of particular process instantiations. At a higher level of abstraction, we can also look at properties and control aspects of a process in terms of, for example, repeatability, optimization, etc. [2].

In this paper, we focus on 1 and 2. View 3 depends very much on implementation and support of some specific process (in particular, tooling), which is outside the grasp of our current study. View 4 is essential in the long run yet stands mostly apart from the discussion in this paper, and will not be elaborated on any further now. Admittedly, many fundamental aspects of process quality (process control) are expected to be covered by 4, rather than 1 and 2. However, 1 and 2 do provide the concepts we direly need for applying 4 in any concrete way. This paper, therefore, is arguably a SEQUAL-based 'step up' towards analysis at the level of viewpoint 4.

Our main contribution is an initial, preliminary version of a framework for Quality of Modelling (QoMo), which not only takes into account the products of modelling but also *processes*. In doing so, we will base ourselves mainly, but not exclusively, on the SEQUAL framework. Analogous to SEQUAL, QoMo is not an operational method for quality assessment, while the framework is expected to evolve as our knowledge of dealing with quality of modelling processes increases. For some aspects of process description, we use concepts similar to ones used in Situational Method Engineering or SME [9].

2 The SEQUAL framework: overview and comments

Since we cannot, nor wish to, present an elaborate overview or discussion of the SEQUAL framework, we will provide a short summary of its key concepts, as based on its latest substantial update [8]. We have in one case held on to a concept abandoned in that update, whilst we also accept the concepts that replace the abandoned one in the update. This concerns the notion of pragmatic quality: we prefer to maintain a notion of "quality of interpretation" while we also acknowledge the value of more action/change oriented pragmatics. Thus we have included the new, action-oriented notions of pragmatic quality and renamed their interpretation-oriented counterpart "Quality of interpretation" (marked below with [*]). Finally, we have rephrased some of the definitions or added some explanations/interpretations of our own, yet we believe we did not stray from the intentions of [8].

SEQUAL Model Items:

- **G**: goals of modelling (normally organizationally defined).
- **L**: language extension; set of all statements that are syntactically correct in the modelling languages used.
- **D**: the domain; the set of all statements that can be stated about the situation at hand.
- $\mathbf{D^O}$: the optimal domain; the situation the organization would or should have wanted –useful for comparison with the actual domain D in order to make quality judgments.
- **M**: the externalized model; the set of all statements in someone's model of part of the perceived reality written in a language.
- $\mathbf{K_s}$: the relevant knowledge of the set of stakeholders involved in modelling (i.e. of the audience at large).
- $\mathbf{K_m}$: a subset of K_s; the knowledge of only those stakeholders actively involved in modelling.
- $\mathbf{K^N}$: knowledge need; the knowledge needed by the organization to perform its tasks. Used for comparison with K_s in order to pass quality judgments.
- **I**: the social actor interpretation, that is, the set of all statements that the audience thinks that an externalized model consists of.
- **T**: the technical actor interpretation, that is, the statements in the model as 'interpreted' by the different modelling tools.

SEQUAL quality definitions:

- **Physical quality**: how the model is physically represented and available to stakeholders; a matter of *medium*.
- **Empirical quality**: how the model comes across in terms of *cognitive ergonomics*, e.g. layout for graphs and readability indexes for text.
- **Syntactic quality:** conformity to the syntax of the modelling language, involving L.
- **Semantic quality**: how well M reflects K_s.
- **Ideal descriptive semantic quality**: Validity: M/D=∅; Completeness: D/M=∅.
- **Ideal prescriptive semantic quality**: Validity: M/D^O=∅; Completeness: D^O/M=∅.
- **Domain quality**: how well the domain fits some desired situation: D compared with D^O.
- **Quality of socio-cognitive interpretation**: how an individual or group interprets the model, i.e. how I matches M, in view of how M was intended to be interpreted by one or more of its modellers. [*]
- **Quality of technical interpretation**: similarly, how a tool or group of tools interprets the model, i.e. how T matches M. [*]
- **Pragmatic quality -actability**: how the model, or the act of modelling, influences the actability of the organization. Note that this enables description of the effect of the modelling process even in case the model as such is discarded.
- **Pragmatic quality -learning**: how the modelling effort and/or the model as such contribute to organizational learning.
- **Knowledge quality**: how well actual knowledge K_s matches knowledge need K^N.
- **Social quality**: the level of agreement about the model among stakeholders (individuals or groups) about the statements of M.

3 Product goals and process goals

The model items and qualities of the SEQUAL framework can be used as an abstract basis for expressing product quality of a model process, and alternatively to specify *goals* for model quality. Note that during a modelling process, each model item (for example the model (M), stakeholder knowledge (K_s), or the state of agreement about M (Covered by "Social quality" in SEQUAL) may change, in principle, at any step in the process. The SEQUAL framework can therefore be used not only for expressing how well a process as a whole has done in terms of achieving its goals, but also which specific steps or series of steps in the process have contributed to specific advances in achieving specific (sub)goals. We can thus directly use SEQUAL concepts for expressing **product goals**: both **product end-goals** and **intermediary product goals.**
In addition, it should be possible to link product goals and sub-goals, and the level of achievement in view of these goals, to the notion of *benefit*, and weigh this against its *cost*. Note that cost (for example in terms of time or money) is something that can be especially well calculated in view of a work process and the people and resources involved in it. This entails that, as is common in process modelling, account should be taken of the tasks or actions involved as well as the people (typically, *roles*) performing them. Both the cost of the entire process and, again, the cost of steps/parts in the process can then be calculated. This entails that the cost-benefit ratio for an entire

process, or parts of it, can be calculated. As argued earlier, this is a very useful way of evaluating a modelling process.

In [1], we presented an initial list of *modelling goals* (slightly amended here) of which the relation to SEQUAL will be made clear. The goals are, or course, generically covered by G in SEQUAL, but they also relate to most of the other SEQUAL concepts.

Usage goals (including actability and knowledge goals): These stand apart from our "modelling goals": they represent the *why* of modelling; the modelling goals represent the *what*. In SEQUAL, the usage goals are covered primarily by the Pragmatic qualities (both learning and actability) and, related to the former, Knowledge quality. The overall cost-benefit ratio will mostly relate to Usage goals, but optimizing (aspects of) modelling methods in an operational sense requires us to look at the other goals, the "modelling goals":

Creation goals (list of model items/deliverables): This relates to what we might generalize as "required deliverables": M, in a very broad sense (i.e. also including textual documents etc.). If made explicit, K_s and/or K_m are to be included here. Creation goals are primarily related to the SEQUAL notions of *completeness* (as part of "Ideal descriptive/prescriptive semantic quality") and *validity* as defined under "Ideal descriptive/prescriptive semantic quality". Note that "Completeness" in an operational sense would in fact be defined as $K_s/M = \varnothing$ ([8] has it as $M/D = \varnothing$). Validity would then be $M/K_s = \varnothing$. There is a complication, however, because some definitions of validity also strongly involve Social Quality (see Validation goals below), linking validation with levels of agreement with (parts of) the model by specific actors. We observe that SEQUAL allows us to differentiate between these two notions of validity, and yet combine them.

Validation goals: These are related to Social Quality: the level and nature of agreement between stakeholders concerning the model. As discussed, our analysis allows us to differentiate between two common notions of "validity": one now falling under Creation Goals, one (the one related to Social Quality) under Validity Goals.

Argumentation goals: In some cases, arguments concerning particular model items may be required. Though a weak link with Social Quality can be suggested here, it seems that this type of modelling goals is not as of yet explicitly covered by SEQUAL. Argumentation goals arguably are an extension of Validation goals.

Grammar goals: Language (L) related: concerns syntactic quality.

Interpretation goals: Related to quality of socio-cognitive interpretation, and possibly also to technical interpretation. The latter may be covered by Grammar goals if the language and its use are fully formal and therefore present no interpretation challenges whatsoever. Note that Interpretation Goals may be seen as a refinement of Validation Goals.

Abstraction goals: This is as of yet a somewhat obscure category. It boils down to the question: does the model (or parts of it) strike the right level of abstraction? This seems to be a crucial matter, but also one that is terribly hard to operationalize. There seems to be a link with Semantic quality (and it is a different link than the one covered by Creation goals), but its precise nature is yet unclear to us.

4 Achieving goals by means of strategies

Given usage goals, modelling goals, and a modelling environment, strategies can be formulated to best execute the modelling process. In other words:

Usage goal + Modelling goal + Modelling environment ⇒ Modelling strategy

Moving to concepts which are more specifically related to actual modelling processes, we will now briefly present an approach to describing the detailed steps in modelling processes. This approach can be used either descriptively or prescriptively.

It is customary in run-of-the-mill method description to view procedures or processes in terms of fairly simple flows or "steps". In view of the many entwined goals and sub-goals at play in modelling, and the different sorts of equally entwined actions taken to achieve these, it seems more helpful to let go of (work)flow in the traditional sense as a metaphor for modelling procedure. Instead, we advocate a rule-based approach in which an analysis of the states of all relevant model items, in combination with a set of rules describing strategies, leads to "run-time" decisions on what to do next. Importantly, the framework is able of capturing *ad hoc* activities of modelling as well as tightly pre-structured ones. In order to achieve this, we define constraints on states and activities, i.e. not necessarily fully specified goals and strategies.

Strategies: the basic elements in a process description we call strategies[1]. They are descriptions of some *way in which some state transition is realised.* Typically, such a transition concerns a move from some state of a model (expressed in terms of one or more SEQUAL concrete model items, e.g. M or K_s) to a more advanced state of the constellation of model items. Crucially, **goals** (as discussed above) are specified as abstractions of target states. We categorize strategies according to the goals they are to fulfil. For example, they can be *creation strategies*. A creation goal might be: "create all classes required to appropriately capture the objects in domain D". Strategies are also categorised in three basic types, based on the way in which the transition is (to be) realised via further activities: **ad hoc**, **layered**, or **procedural**[2].

Ad hoc strategies: An ad hoc strategy is in fact a void representation: how to reach the goal is up to the person carrying it out. This may seem like a rather useless category, but in fact it is crucial in allowing for ad-hoc implementation of certain steps in modelling (which is realistic both descriptively and prescriptively).

Layered strategies: a transition is "zoomed in on" and the underlying activities are again described as one or more state transitions/strategies.

Procedural strategies: these are the stepwise cookbook-like activity descriptions commonly associated with basic methodical procedures. They are meant to cover the more deterministic sort of activity sequence (usually low-level, detailed instructions).

Temporal ordering of strategies: The types of transitions in the process model can be subject to some ordering over time, for example "A should take place before B" or "D should take place immediately following C".

---

[1] In [9] and other work in SME, strategies are commonly named "guidelines". We avoid the generic use of this term here because we are primarily interested in *describing* processes as they are carried out before we might set out to *guide* them, as is the chief goal in SME.

[2] In [9], a somewhat similar distinction is made between *simple, tactical,* and *strategic* guidelines. Our categorization differs in that 1. we make explicit the "ad hoc" option, 2. we distinguish between rule-based process definition (*layered* strategies; arguably covering *both* "tactical guidelines" and "strategic guidelines") and simple, textually described procedures (very similar to "simple guidelines").

The strategy to fulfil the creation goal "create all classes required to appropriately capture the objects in domain D" thus can be either ad hoc (empty strategy: "just do it"), or heavily structured (detailed, deterministic steps for identifying and formulating classes), or something in between (described as a combination of ad hoc, layered, and procedural strategies). For more on this, see section 5.
Further refinement of the notion "required" in the example creation goal (above) would be linked to refinement of the SEQUAL notion "semantic quality", as discussed (K_s/M=∅). The *start state* then represents the state before the strategy is carried out, i.e. one in which completeness of M has not been achieved.
Using the concepts above, it is possible to set up a rule-based process description that is likely to be non-deterministic: the process will unfold depending on events, decisions, and other factors as they emerge on-the-fly. Processes (even in the prescriptive sense) are thus kept *agile*. For example, goals can be changed half-way the process, or alternative strategies may be chosen, perhaps after others were tried and failed. In addition, activities related to a number of interrelated but still different goals and subgoals can be combined and interwoven into a structure of constraints on the modelling process that may or may not leave considerable freedom of action, depending on the way the constraints are set. Importantly, this allows for combinations of actions working towards various different goals, for example creation goals/actions mixed with validation goals/actions and interpretation goals/actions (to our knowledge, situational method engineering approaches do not enable this at such a basic level, and yet it is a quite realistic course of events in modelling).
We realize that the description of our process description framework as presented is very sketchy at best. Some more detail of our rule-based, goal-driven approach is provided in [1]; a full (formal) treatment of the framework is forthcoming (also see section 6).

5 An example: some strategies for achieving key goals

We will now sketch some generically occurring, simplified example strategies, in order to provide a more concrete illustration of our framework. Please note that the strategies described are merely examples, and in no way do we make claims as to their applied value. We consider two key aspects in modelling quality of modelling: completeness and validity; we already made a start with this in the previous section. A first, basic version of a set of strategies to achieve a certain degree of quality in modelling could be the following, including very basic usage, creation, grammar, and validation goals/strategies.
Usage Goal: "the model can be used as a direct input for database implementation, to be performed by a human agent" (i.e. no automated generation).
Creation Goal: "all relevant entities in the domain (drawn from the knowledge of stakeholders) are stated" (K_s/M=∅).
Grammar Goal: "entities are expressed at type level, as *classes* (taking UML as an example language"; this could in fact amount to a full description of the UML syntax).
Validation Goal: which classes are relevant is a matter of agreement between certain stakeholders. As a rather simple example, it may be stated that "*all* stakeholders actively involved in modelling must agree on every class being a proper and useful

conceptual reflection of the domain, and on the class's relevance." Note that the notion of relevance is directly related to the Usage Goal: in the example, the model is to be used for database implementation.

Usage strategy: create a database using the UML class diagram of the domain as input.
Strategy type: ad hoc. This simply means that no further strategies are provided here about usage of the model in the example method.
Start State: there is a complete and valid UML class diagram of the domain.
End state: there is a database based on the UML diagram.

Creation Strategy: create a UML class diagram.
Strategy type: layered.
Start State: there are some stakeholders that have sufficient knowledge of the domain and are duly capable of turning their knowledge into a UML class diagram.
End State: there is a complete and valid UML class diagram of the domain ($K_s/M=\varnothing$ and $M/K_s=\varnothing$)
Sub-strategies:
Creation Strategy-2

> **Strategy type**: procedural
> **Start state**: there is a UML class diagram
> **End state**: there is a UML class diagram that more complete in view of the start state (one class added)
> **Procedure**
> 1. Have someone suggest a class that is still missing
> 2. Name the class
> 3. Add it to the model
> **Validation Strategy-2**
> (see below)

Validation Strategy
Strategy type: layered.
Start State: there are classes in the UML diagram, but they may not all be agreed upon by all stakeholders
End state: all classes in the UML diagram have been agreed upon by all stakeholders.
Sub-strategies:

> **Validation Strategy-2**
> **Strategy type**: ad hoc
> **Start state**: a certain newly introduced class (created under CreationStrategy-2) has not been agreed upon by all stakeholders
> **End state:** a certain newly introduced class (created under CreationStrategy-2) has been agreed upon by all stakeholders
> **Temporal constraint**
> If a new class has been added (CreationStrategy-2), **immediately** validate that particular class (ValidationStrategy-2).

Please note that the toy example above does not aim to add anything in terms of modelling practice or methodology (though it is arguably a reasonable description of what roughly happens in a straightforward UML class modelling exercise). The example

could of course be refined and extended. Currently, we wish only to provide an example of how the SEQUAL-based goals can be linked to a network of rules describing concrete modelling activities.
A crucial property of the framework is that a rule-based dependency structure between goals and strategies is created, in which all sorts of nuances concerning dependency and ordering *can* be expressed, but may also be left out if so desired.

6 Conclusions and further research

This paper set out to present a plausible link between the SEQUAL approach to model *product* quality and our emerging QoMo approach to *process* quality in modelling. We did not aim to, nor did, present a full-fledged framework for describing and analysing modelling processes, but some fundamental concepts underlying the design of a framework for capturing and analysing 2nd order information systems were in fact put forward.

We started out describing the outline of the QoMo framework, based on knowledge state transitions, and a goal structure for activities-for-modelling. Such goals were then directly linked to the SEQUAL framework's main concepts for expressing aspects of model items and its various notions of quality, based on model items. This resulted in an abstract but reasonably comprehensive set of main modelling process goal types, rooted in a semiotic view of modelling. We then presented a simple illustration of how such goals can be linked to a rule-based way of describing processes for modelling. These process descriptions hinge on strategy descriptions. Such strategies may be used descriptively, for studying/analysing real instances of processes, as well as prescriptively, for the guiding of modelling processes. Descriptive utility of the preliminary framework is crucial for the quality/evaluation angle on processes-for-modelling, and reflects the main intended contribution of this paper. As discussed, the detailed description of a modelling process, possibly down to every small modelling action, e.g. the adding of a single class, is highly relevant for nuanced quality and cost-benefit analysis of (parts of or patterns in) modelling processes. Study and control of a process requires concrete concepts describing what happens in it, after which more abstract process analysis (efficiency, cost/benefit, levels of risk and control) may then follow. Means for such an analysis were not discussed in this paper: this most certainly amounts to future work.

Our immediate next step will be to more precisely formulate the basic conceptual framework for process/strategy description. It will be rule-based, dynamic, and goal-driven (implying quality-orientation). Since our chief, longer-term goal is to first *describe*, then, *study,* and finally to *improve* 2nd order information systems, our approach will be to use traditional (formal) techniques from the field of information systems and artificial intelligence: conceptual modelling (rooted in first order logic), process/workflow modelling (with Petri nets for underlying semantics), matching techniques, etc. We thus aim to provide a solid and well-conceived basis for systems that should support the capturing, analysis, simulation, and ultimately the support of real modelling efforts. In addition, we have already started putting our preliminary framework to use in describing various (partial) processes (both conceptual modelling

and more generic system development). The framework will be improved and extended stepwise as a result of such case studies.

7 References

1. P. van Bommel, S.J.B.A. Hoppenbrouwers, H.A. (Erik) Proper, and Th.P. van der Weide. Exploring Modelling Strategies in a Meta-modelling Context. In R. Meersman, Z. Tari, and P. Herrero, editors, *On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops*, volume 4278 of *Lecture Notes in Computer Science*, pages 1128-1137, Berlin, Germany, EU, October/November 2006. Springer.
2. M.B Chrissis, M. Konrad, and S. Shrum. *CMMI: Guidelines for Process Integration and Product Improvement*, Second Edition. Addison-Wesley, 2006.
3. T.A. Halpin. *Information Modeling and Relational Databases, From Conceptual Analysis to Logical Design*. Morgan Kaufmann, San Mateo, California, USA, 2001.
4. S.J.B.A. Hoppenbrouwers, H.A. (Erik) Proper, and Th.P. van der Weide. A Fundamental View on the Process of Conceptual Modeling. In *Conceptual Modeling - ER 2005 - 24 International Conference on Conceptual Modeling*, volume 3716 of *Lecture Notes in Computer Science*, pages 128-143, June 2005.
5. S.J.B.A. Hoppenbrouwers, H.A. (Erik) Proper, and Th.P. van der Weide. Towards explicit strategies for modeling. In T.A. Halpin, K. Siau, and J. Krogstie, editors, Proceedings of the Workshop on Evaluating Modeling Methods for Systems Analysis and Design (EMMSAD'05), held in conjunctiun with the 17th Conference on Advanced Information Systems 2005 (CAiSE 2005), pages 485-492, Porto, Portugal, EU, 2005. FEUP, Porto, Portugal, EU.
6. J. Krogstie. A Semiotic Approach to Quality in Requirements Specifications. In L. Kecheng, R.J. Clarke, P.B. Andersen, R.K. Stamper, and E.-S. Abou-Zeid, editors, *Proceedings of the IFIP TC8 / WG8.1 Working Conference on Organizational Semiotics: Evolving a Science of Information Systems*, pages 231-250, Deventer, The Netherlands, EU, 2002. Kluwer.
7. J. Krogstie and Jorgensen H.D. Quality of Interactive Models. In M. Genero, Grandi. F., W.-J. van den Heuvel, J. Krogstie, K. Lyytinen, H.C. Mayr, J. Nelson, A. Olivé, M. Piattine, G. Poels, J. Roddick, K. Siau, M. Yoshikawa, and E.S.K. Yu, editors, *21st International Conference on Conceptual Modeling (ER 2002)*, volume 2503 of *Lecture Notes in Computer Science*, pages 351-363, Berlin, Germany, EU, 2002. Springer.
8. J. Krogstie, G. Sindre, and H. Jorgensen. Process models representing knowledge for action: a revised quality framework. *European Journal of Information Systems*, 15:91-102, 2006.
9. I. Mirbel and J. Ralyte. Situational Method Engineering: combining assembly-based and roadmap-driven approaches. *Requirements Engineering*, 11:58-78, 2006.
10. D.L. Moody. Theoretical and practical issues in evaluating the quality of conceptual models: current state and future directions. *Data and Knowledge Engineering*, (55):243-276, 2006.