COMMUNITY-BASED ICT DEVELOPMENT AS A MULTI-PLAYER GAME

S.J.B.A. (Stijn) Hoppenbrouwers

Dept. of Computing and Information Sciences
Radboud University Nijmegen
The Netherlands

stijnh@cs.ru.nl

Abstract

In this paper, we advocate both the metaphorical and practical use of a gaming approach to the analysis, design, and support of the creative, collaborative, yet also strongly constrained processes that lead to formal models for use in ICT system development and business analysis. Making formal (i.e. rationalistic, mathematics-oriented) modeling accessible to people with little or no modeling expertise (i.e. people from the Organizational Participant communities) is becoming an urgent, critical issue in Business-IT alignment.

Formal Modeling involves languages aspects (informal-formal language), which is commonly known, but also strong analytic and procedural aspects, which is often ignored in modeling studies and practice. Recent research has led to insights concerning the goal-driven, conversational nature of collaborative modeling processes. Human factors and interaction deserve to be accepted as primary in the study, design, and support (tooling) concerning operational (real-life) formal modeling processes.

Taking the gaming approach to operational modeling has a number of advantages, both practical and methodological, and seems a promising context for study and development of methods and (importantly) digital tools for improving and supporting collaborative formal modeling.

Community-Based ICT Development as a Multi-Player Game

1. Introduction

For operations in organizations to be effectively supported by ICT, organizational knowledge needs to be made explicit. What is more, it somehow has to be explicated in a way that literally fits the mathematics-based conceptual structures ("formal" structures) required by computer technology and rational/mathematical analysis. For decades, we have seen a gradual shift from technology-oriented to organization-oriented *modeling* as part of an ever intensifying, yet still largely unsuccessful, business-IT alignment effort. Standard, contemporary examples of the sort of modeling we talk about here concerns business process models, conceptual models, ontologies, business rules, use cases, value models, goal models, and so on. Closely related artifacts include strategy statements, policies, principles, scenarios, requirements, terminologies, etc.

Along with a growing realization (as voiced especially in organization-oriented systems an cybernetics literature) that organizations today are typically complex, adaptive socio-technical systems, the awareness has come that organizational change is permanently taking place, and is for a considerable part unpredictable; additionally, that organization-oriented modeling should therefore be a *continuous, adaptive process* that is somehow interwoven with operational processes in the organization; furthermore, that such a process should be *embedded firmly in the organizational community* to genuinely take off and get a hold (Veldhuijzen van Zanten et al., 2004; de Moor, 1999).

Meanwhile, the rise of the "Semantic Web" (Berners-Lee et al., 2001), hyped up as it might be, has caused a serious boost of *technological* developments aiming for the modeling of formal semantics. Mostly, the semantic web community has taken an approach centered round standardization: of semantic exchange languages, exchange protocols, vocabularies (ontologies), etc. Though this approach may be successful in certain areas of organization and automation, it cannot be fully successful, on fundamental grounds (Hoppenbrouwers, 2003). What is lacking is an approach that takes into account the communal factor as well as the wider evolutionary view, and the "requisite diversity" it enables.

A group of researchers have recently set up an initiative to augment the Semantic Web with the "Pragmatic Web" (Schoop et al., 2006). The initiative is rooted in speech act theory (Searle, Austin) and Habermas's theory of communicative action (Weigand, 2003). It takes aboard interaction and language issues that are heavily context-oriented, adding analysis and engineering of a contextual layer to the generic, "dead" core of syntax and semantics installed piecemeal by the Semantic Web. The Pragmatic Web community is closely related to the Organizational Semiotics community (Kecheng et al., 2002).

Creating an effective running process for community-based, evolutionary modeling that drives dynamic ICT development is far from trivial. It requires massive cultural, organizational, and technological changes to take place. However, some slow progress is being made towards bringing to life an *interactive modeling cycle*. Such an interaction cycle will have to combine human-human interaction with "human-computer interaction trough modeling". As mentioned, this sort of interaction differs from "regular" human-human communication in that its deliverables are subject to many technology-based, rationally motivated constraints. They inevitably boil down to the creation of *formal structures* of some sort. They require elaborate and precise *definition and specification* rooted in engineering rather than human-human communication. Combining this with intuitive, natural interaction has proved extremely hard, but wishing away the fundamental conceptual gap between formal semantics and natural semantics, for example by assuming a yet unforeseeable breakthrough in AI and NLP, is totally unsatisfactory in the long run. Furthermore, as will be discussed in section 2, the informal-formal language divide is only part of the problem.

To add to the challenge, as ICT gets more and more embedded in diverse communities, an increasing number of increasingly localized/personalized models will have to be produced and altered by non-experts with no formal modeling skills. This too justifies intensive study of the process of modeling from a community perspective, from an *interaction* point of view. The "formal modeling threshold" *will* have to be lowered.

I intend to take the study and support of *modeling as interaction* further by taking an alternative approach to interaction for model-based systems development: a gaming approach. This involves the metaphor, but also the practice, of viewing, studying, and supporting model-related interactions as a set of combined games, of different kinds; for example puzzles, role playing, negotiation games, language games; collaborative but possibly also competitive games, and so on. The relation with existing "management games" (Elgood, 1993) is obvious, but the goals and, consequently, the nature of the game(s) will be different.

Note that I do not refer to game theory here (Rasmusen, 2006), which focuses on finding winning strategies for finite-state games/problems. For now, I refer mostly to the *setting of rules of play*, including the setting of clear and achievable goals and designing a score system. In realistic gaming situations, interaction may be tightly structured or left open to creativity and improvisation, as best fits the specific gaming/modeling context (depending on, among other things, the roles, goals, and expertise of the players).

Consequently, the rules of the game will be different per situation (though finding generic patterns will of course be a main aim in research). The big challenge is to find the right *game designs* for particular situations, as an alternative way of thinking about *methods*. There is a clear link here with the field of Situational Method Engineering (Ralyté et al., 2007). Various questions are immediately raised here: when does a modeling team perform well (score system)? And what are a “good model” or “good modeling performance” in the first place? Answers relate the topic of *quality and goals of modeling* (section 3).

Crucially, my focus is on real, operational, feasible model-oriented interaction. One of the advantages of a gaming approach is that it forces researchers to define playable (i.e. *workable*) rules of play. In addition, we can introduce other typical game elements, for example score systems that guide players to high quality results (as defined at the hand of situational goals that are based on community and technical concerns and priorities). “*Playability*”, in any case, amounts to “methodological realism”; something that is much looked for but hard to find. Also, playability is very much *testable*, which is another advantage for method engineering practice.

The Big Idea, then, is that if community members play a well-designed “modeling game”, this guarantees the rendering of a “good model”, which combines issues like required model utility, systematic underlying argumentation, good old correctness and validity, but also knowledge sharing and development, consensus, even commitment.

Finally, there is a strong link with the world of Video Gaming. Community-based, evolutionary modeling calls for intelligent and automated support and will increasingly have to take place in (distributed) digital environments. It involves such software as CASE tools, negotiation support systems, and specialized Groupware. I propose to create the required tools as an integrated, game-like environment. Primarily, this should open designers’ eyes to the much neglected interaction aspects of (collaborative) modeling environments. Viewing and shaping modeling interaction as gameplay may help uncover and reduce the many and diverse factors standing in the way of operationalizing realistic, large scale, collaborative, community-based, evolutionary modeling.

However, we will allow ourselves a considerable run-up before we arrive at a discussion of the gaming approach. First, we will explore some crucial cultural differences between communities involved in business-IT alignment, and observe that the formal modeling that *combined* communities should partake in involves challenging *language* and *analysis* aspects, the latter of which has been mostly ignored. We continue by discussing a comprehensive conceptual framework for "quality of modeling", and a set of modeling goals that can be derived thereof. We then discuss how these goals are pursued in interactive modeling systems, which are conversational environments rendering formal models. This finally allows us to introduce the Gaming approach as a specific and promising view on the design, testing, and implementation of model-oriented interaction systems. We round off with a summary of main arguments and a sketch of future directions.

2. Formal Business Modeling: Language and Analysis

As explained, we are concerned with ongoing and so far rather unsuccessful attempts to bridge the gap between two worlds that in many respects are still truly alien to one another: the world of the engineering of computational systems and the world of the understanding and improvement of (the operation of) organizations.

The bridge across the gap has been envisioned and discussed widely, and even has been given at least one commonly accepted name: "Business-IT Alignment". And indeed, given that information technology (IT, or ICT, as some prefer to call it –Information and Communication Technology) is applied widely to support organizations, alignment of some sort can simply be observed to exist. It is, of course, the *extent* and *quality* of the alignment that is questioned here, and the *process* of achieving sufficient alignment.

I will refrain from discussion of the often-mentioned failure of office automation and the never ending list of ICT projects gone awry inside and outside business contexts. I merely observe that evidently, the fundamental elements and processes in the world of organizations do not seem to be matched easily with technologies and approaches as brought forth by computer science and systems engineering. And yet, in the name of "progress" (whatever that may mean), business and IT are condemned to each other, like partners in a bad but necessarily faithful marriage.

Two communities alien to one another

Importantly, the “worlds apart” are populated by *communities*: groups of people with some characteristics that are of interest to the question of what makes these “worlds apart” so different. Let us call the two generically distinguishable groups “Organization Participants“ and “ICT Developers”, respectively. Let us also discard the obvious fact that most if not all ICT Developers are inherently part of some organization, and therefore double as Organizational Participants.

What then makes the groups so different? The Organization Participants have a primarily social outlook on their working context. Even if they have technical jobs (like computer programmers), they are part of a social system in which they play their roles based on socio-cognitive functions and capabilities engrained in their biology, their mind/brain, and their life experience. Also, individual and social drives, goals, and needs need not be rational in this world; in fact, many irrational (or not so rational) goals and ideas are embraced as “truly human” and are regularly valued *superior* to their rational counterparts. Rationality is an important part of the way Organization Participants think about and plan their actions (on the work floor and in the board room alike), but it is rarely practiced in the form of explicit and methodic calculations or reasoning. *Working, collaborating, and even organizing is not engineering* (despite the existence of fields like “Enterprise Engineering”). In line with this, the language and communication Organizational Participants consider proper and preferable is “natural language”, which is capable of expressing rational concepts to some degree but also enables rich and free expression of emotional, social, subjective, irrational, associative thoughts and concepts. Natural language is not fundamentally rational in the mathematical sense (or so I believe), nor an ideal vehicle for the expression of hard-core rational meaning —but a superb vehicle for expression of human thought in general, as it is in fact part of what we are as a species (including all its cultural variations).

ICT Developers, by contrast, specialize in the design and construction of machines and (increasingly) other engineered structures that are very much of a “formal” nature, in the mathematical sense: they are built according to engineering patterns that are, at least in principle, mostly rational. Mathematics, providing language systems supporting strictly rational concepts and allowing only for rational conceptual constructs, forms the basis for their professional view on what language should be and how it should be used.

Even engineers that have not touched "pure" mathematics since they attended university have been imprinted with the rational/mathematical way of thinking, and are proud if this. It is a fundamental part of their professional identity, and they have acquired it through considerable personal effort. Also, the practical advantages of rational approaches (especially in technology and natural science) are evident.

Informal and formal language

Though language systems are often described as static entities (grammar, lexicon; syntax, semantics), in "natural" (often called "informal") and "formal" contexts alike they are also systems that are *used*. In this vein, we can talk about formal or informal *use* of language (the domain of pragmatics). After all, syntax and semantics are functionally subordinate to *communication*, and clearly a carefully crafted text in, for example, academic English can be used so rationally that it is as formal as a set of mathematical axioms (just less compactly expressed). In other words: it is the rationality underlying expressions that counts most; the language system is secondary, but of course in many cases for a hard core rational communication/analysis job it is preferable to use a language system that is engineered to be fundamentally rational —mathematical language.

The rational outlook on the world is not the only reason why ICT developers favor formal language in professional communication. On a more practical level (yet obviously as a result of the rational foundations of computer engineering), we are now simply forced to build and instruct our machines through languages with syntax and semantics that neatly fit the rational internal mechanics of those machines. Programming languages etc. are inherently formal, even if they do not look like classic mathematics. For computational machinery to be useful to organizations (primarily, to the people working in them), its internal structures to some extent have to mirror the conceptual structures that exist in, are shared in, and are used in, thinking and communication among Organization Participants. In rational, systemic terms, *models* of the organization (as conceived, perceived, and experienced by Organization Participants) have to be used as a basis for ICT design (Dietz, 2006). However, such models also need to be formal (to some extent) in order to fit both the world view of the ICT Developers and the technical instruments they wield.

It has been acknowledged widely and early on in the relatively brief history of ICT that ideally, models describing parts of Organization Participants' worlds should be put together by Organization Participants themselves, since 1. they know best what their world looks like, and, presumably, 2. what it should look like if supported by ideal computational devices. While the latter half of this conjunctive statement is perhaps debatable, at least the first half seems to be a plain fact. Consider the following nuances concerning this statement.

First, indeed it transpires that deep and detailed inside knowledge is required as a basis for adequate description of the intricacies of procedures, conventions, concepts, norms etc. that are to be respected for good Business-IT alignment to be achieved: knowledge that is rarely available to people external to the Organization Participant community, until extensive knowledge engineering procedures are deployed —which is exactly what formal modeling in this context boils down to. Many bad system designs boil down to insufficient knowledge of domain detail and well-meant but misguided creativity of ICT Developers in inventing domain structures (often based on technological solutions rather than domain understanding).

Second, "detailed inside knowledge" of the kind referred to above rarely if ever exists in its *explicit form* in the Organization Participant's mind. Making it explicit is in itself a huge challenge (Nonaka and Takeuchi, 1995), both intellectually and in terms of resources like time and money.

Crucially, the "explication" problem has two sides: a language side and an analytical side. Contrary to what is often assumed, these are not the same in modeling practice.

To be useful for ICT Developers, knowledge eventually needs to be expressed formally, which requires knowledge of mathematical language and concepts (linguistic problem). Yet even if informal language can be successfully used in the description at some intermediary stage (as is common practice), and if in a next step, "translation"[1] from informal to formal language is successfully achieved, the concepts expressed *will not inherently be well-organized* (analytical problem). Formal or not, it takes much knowledge and reasoning, about many interlinked aspects and patterns within the domain, to produce a good model. *Language alone does not do the trick* — though it is part of the solution.

Rational analysis is complementary to formal language

We are faced with situations in which typically, the expert knowledge of the Organization Participants needs to be complemented by the expertise of

[1] Given the rather fundamental difference between formal semantics and "socio-cognitive" or natural semantics, the term "translation" is a somewhat pretentious one here, and one that often installs false and inconvenient expectations in many of the

the ICT Developers: *formalization* (translating informal to formal language) and *analysis* (the rational combination of various aspects and patterns underlying the model). Interestingly, the diverse aspects and patterns that need to be combined to create a good model are usually only partially represented in the explicit, formal models that are directly useful to ICT Developers.

For example, as illustrated in the middle column of figure 1, a Business Process Model in the standard language BPMN (Business Process Modeling Notation; OMG, 2006) typically shows an ordering of activities, e.g. activities D and E must be completed before activity F can be started. However, the reason why this is the case is that D and E respectively produce entities n and o that are needed in F (resulting in what is technically called an "AND-join"). This is illustrated by the text in the leftmost and rightmost columns of figure 1 in which these entities and dependencies are made explicit. However, such dependencies and entities are *not made explicit in a regular BPMN diagram*, even if they are crucial for creating a useful, "good" one. As a consequence, the entities and dependencies involved are usually left implicit and exist only in the head of the modeler –if you are lucky. Even if the objects in the process *are* made explicit, perhaps in another model, they are not explicitly used as a basis for deriving AND-joins.

Creating a model is commonly viewed, in both ICT and "Business Engineering", as an assignment to "create a description of aspect X of the domain in language Y", for example a Business Process Diagram in BPMN. The (formal) language focus prevails, and its elementary concepts need to be somehow combined to compose a useful, good quality model. Crucially, such concepts are normally based on the conceptual requirements posed by *intended primary use* of the model (i.e. technical or analytical use[2]), not by the concepts needed to express and use the knowledge that *leads to* a good model. If all expressions that constitute a model have (at least in principle) a *rationale* that underlies them, i.e. an argumentation, then the key concepts needed for expressing and validating that argumentation should also be available as part of the modeling setup. However, the mathematical practice of careful but rigorous *abstraction* tends to dismiss "argumentation concepts" as not relevant for models and this results in neat and purpose-oriented modeling languages that leave argumentation implicit.

---

people involved.

[2] It is quite possible for models to be aimed at, for example, selling an idea, explaining a situation to laymen, and so on. In such cases the primary use does not

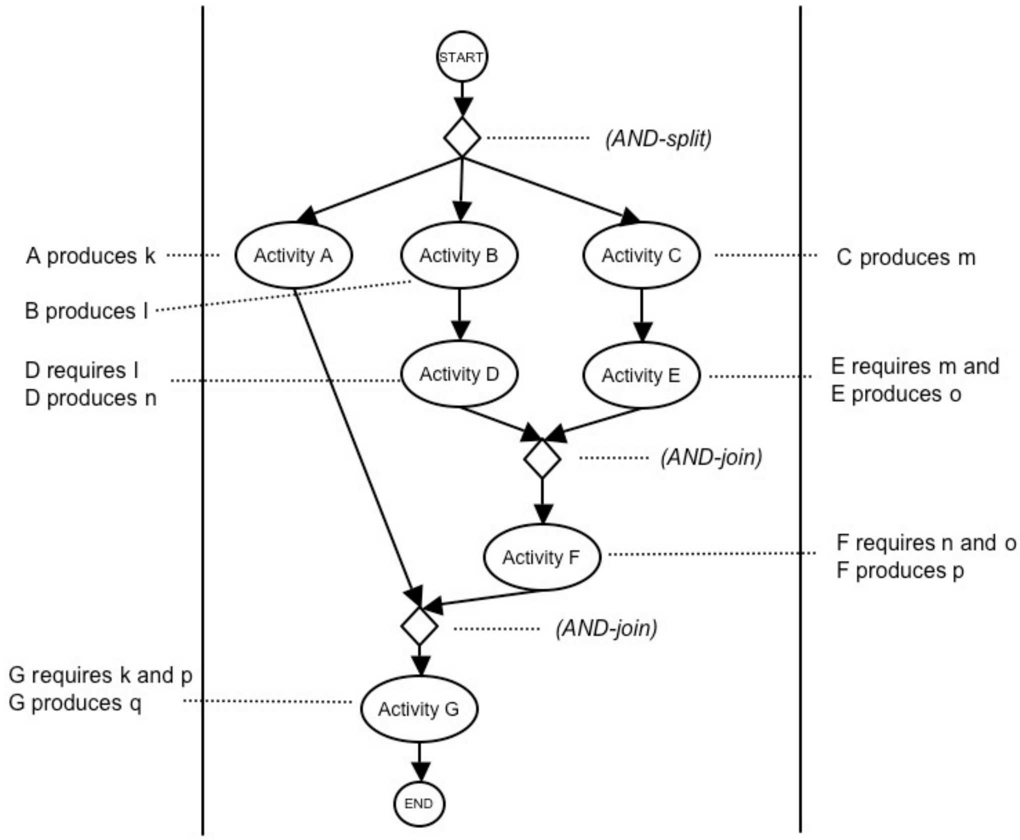

Figure 1: Basic BPMN diagram and underlying dependencies

Indeed, beyond "model checking" (verifying explicit meta-concepts and syntactic constraints, i.e. the grammar of the modeling language, and possibly some other static analysis of the emerging model), very little help is available to the modeler (either an Organization Participant or an ICT Developer) in creating "good models", i.e. models that adequately represent some aspect of the Organization *and* are useful as artifacts for rational engineering. In other words, the analytical aspect of *the act of modeling* is largely ignored. It is left to the inherent talents, and capabilities (largely intuitive) of the modeler to perform the analytical part of model construction: it is art rather than science, and this artfulness is the mainstay of what makes (formal) modeling highly specialized, expert work.

require formal models. However, these are not the cases we focus on here.

3. Quality and Goals in Modeling

In the previous section we repeatedly mentioned the importance of "good models", but of course this raises the urgent question of what a good model is. Part of the answer depends on the *context of creation and use* of the model, whereas some other aspects (typically the more universally rational ones) can be more generically applied as principles of model quality. Fortunately, the international information systems community has brought forth a framework for model evaluation that attempts to combine the many aspects involved in a reasonably fundamental manner. We refer to the SEQUAL framework (Krogstie and Jorgensen, 2002; Krogstie et al., 2006) that is rooted in semiotics[3] and pragmatics. While not being an operational (industrial) framework for model evaluation, it is a good basis of an integrated approach to model quality that fits our current perspective on modeling. Below, we briefly present the main concepts of the model.

The SEQUAL quality framework

There are two main categories of concept in the framework: presumed (sets of) entities, abstract or not, that constitute the model *and its context* (including its use), and relations between such (sets of) entities that reflect aspects of model quality.

- **G:** goals of modeling (normally organizationally defined).
- **L:** language extension; set of all statements that are syntactically correct in the modeling 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 modeling (i.e. of the audience at large).

[3] The link between semiotics and information systems dates was emphasized in the influential publication of (Falkenberg et al., 1998).

- **K_m:** a subset of K_s; the knowledge of only those stakeholders actively involved in modeling.
- **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 modeling tools (i.e. "machine readability").

SEQUAL quality definitions:

- **Physical quality:** how well the model is physically represented and available to stakeholders; a matter of *medium*.
- **Empirical quality:** how well the model comes across in terms of *cognitive ergonomics*, e.g. layout for graphs and readability indexes for text.
- **Syntactic quality:** how well the model conforms to the syntax of the modeling language, involving L.
- **Semantic quality:** how well M reflects K_s.
- **Ideal descriptive semantic quality:** Validity: $M/D=\varnothing$; Completeness: $D/M=\varnothing$.
- **Ideal prescriptive semantic quality:** Validity: $M/D^O=\varnothing$; Completeness: $D^O/M=\varnothing$.
- **Domain quality:** how well the domain fits some desired situation: D compared with D^O.
- **Quality of socio-cognitive interpretation:** how well 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 modelers.
- **Quality of technical interpretation:** similarly, how well a tool or group of tools interprets the model, i.e. how T matches M.
- **Pragmatic quality -actability:** how well the model, or the act of modeling, influences the actability of the organization. Note that this enables description of the effect of the modeling process even in case the model as such is discarded.
- **Pragmatic quality -learning:** how well the modeling 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. It has proven useful to distinguish between communal (shared) *understanding of*, *consent about,* and *commitment to* models.

Note that in the framework as presented, there is no clear coverage of the analytical aspect discussed in section 2, confirming that even this advanced analysis of model quality in the field of information systems does not seriously consider “rationales behind modeling decisions” as a criterion for model quality. This does not mean that such rationales as such have never been considered in the literature, or that they are deemed unimportant *per se*. I suspect they are excluded because modeling methodologists have simply not looked beyond the models as end products, with their clear and focused utility. This keeps sound, explicit argumentation underlying modeling decisions in the margin of methods, as a kind of luxury item. Changing our view, and putting rationales central as a *source* of good models points towards a shift in focus: not on the quality of the end product, but (also) on the *process of thought, communication, and action that produces the end product in a stepwise fashion*.

We can thus add the following concepts to the SEQUAL framework as a refinement:

- **M_R:** the explicitly formulated rationales (argumentations) behind the statements in someone’s model of part of the perceived reality, including all the concepts that are needed to express the rationales.
- **Analytical quality:** how well the statements constituting model M are backed up by rational argumentation.

Crucially, I include analytical quality as directly related to the model M, not to semantic quality (though an indirect relation of course exists). This emphasizes the explication of M_R as a task to be performed by the modeler: it is not “part of reality” (the domain, D). Also, it implies that since M_R is a subset of M, all quality aspects related to M (for example, quality of interpretation or quality of learning) in principle also apply to M_R. The analytic view on M is thus added to the language view on M that was already firmly embedded in the model.

Quality-related goals for modeling

The quality analysis along the lines of SEQUAL is product-oriented. In (van Bommel et al. 2008) we suggest a more process oriented version that translates the various quality aspects to *goals for modeling*, in particular:

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* and *validity* as defined under "Ideal descriptive/prescriptive semantic quality". Note that "Completeness" in an operational sense would in fact be defined as $K_s/M = \emptyset$ (Krogstie et al., 2006 has it as $M/D = \emptyset$). Validity would then be $M/K_s = \emptyset$. 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 on (parts of) the model *among some or all actors involved*. 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: Clearly, these relate to M_R and the notion of Analytical Quality. In earlier papers, we categorized Argumentation goals as an extension of Validation goals, but in view of the discussion above I suggest they may in fact be more fundamentally linked with creation goals. This is in line with M being the main concept for creation goals.

Grammar goals: Language (L) related: concerns syntactic quality, i.e. compliance to syntactic rules of the modeling language.

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 still a poorly understood 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. We suspect there is a role to play here for *relevance* of the model (or parts of it) in view of both its utility and the worldview of the modeling participants. In addition, argumentations and, again, the perceived relevance thereof will be involved. Reduction of complexity and information seems another possible factor in determining "good abstraction", but we should be careful not to confuse *views* on the model with the integrated model to which these views provide focused windows.

The modeling goals listed above can be viewed in context of two sets of more global process goals:

Usage goals (including actability and knowledge goals): they represent the *why* of modeling, where above modeling goals represent the *what*. In SEQUAL, the usage goals are generically covered by G, but less explicitly also by the Pragmatic qualities (both learning and actability) and, related to the former, Knowledge quality.

Efficiency goals are part and parcel of process-oriented views, typically entailing cost-benefit ratios. If enough explicit information on available resources, desired deployment thereof, and also of the utilities of the model (both primary and secondary) is available, such cost-benefit analysis may be possible not only in hindsight, but also as an aid in making methodological and process choices. SEQUAL ignores efficiency goals (possibly because they are not deemed typical "quality goals" and are too much process-oriented), but QoMo takes them on board.

From goals to structured processes

Goals, sub-goals, and combinations of goals can be set for concrete sessions or activities in modeling, involving one or (often) more participants. *End goals* may be worked towards via *intermediary goals*. Strategies and techniques can be selected and deployed to achieve specific goals for concrete situations (van Bommel et al., 2008), in line with goals set but also with resources available and capacities and attitudes of participants, which are highly community-specific as well as situation-specific (again we emphasize the link with the field of Situational Method Engineering here).

At this point, let us emphasize that in parallel with, for example, the creation goals and grammar goals that reflect the need for formal models as end products, some other goals are often present (validation goals, interpretation goals), which are of a much more social nature, and are typically seen as problematic yet uninteresting (if not irrelevant) in the classic, formalist systems modeling tradition. In view of our comprehensive approach to quality of modeling, we seek to effectively combine social and formal aspects as equal forces. Common understanding, consent, and commitment with respect to some model (Hoppenbrouwers et al., 2005) are *products* that are complementary to the delivery of actual model representations, and their realization should be on the agenda in formal modeling just as in more traditional forms of group decision making (for example, policy making). Communication between stakeholders and analysts, and the often lengthy process of reaching common understanding, consent, and commitment amongst them, can only be achieved if the participants adequately enact the whole process. Such social utility of models is not easily bought and cannot be automated or cut short. It has more to do with the creation[4] of common reality and language than with the engineering of formal structures.

4. Collaborative Interactive Modeling

Goal-driven modeling implies the deployment strategies for modeling; strategies can be explicitly imposed by methodology (phases in an approach, even cookbook-like procedures) or left to the modelers (as is usually the case). For relatively detailed guidance of modeling (as inevitably required in case non-experts are to do the job), setting the goal of "creating a good model about domain D in language L" is clearly not enough: more concrete, fine grained goals and procedures are needed. However, in the mean time creative thinking and intuitive choices should not be stifled by an overly restrictive stepwise process. Very experienced modelers typically *do* want to be left free to "do with a modeling language as they please". There needs to be a situational balance in this respect, and achieving it is far from trivial (van Bommel et al., 2008).

As is known from process management and regulations modeling, a possible (somewhat technical) solution to this problem is to formulate rules to constrain actions and products as far as required, and to leave the rest of the actions to participants. Setting the rules so that the process runs smoothly and effectively is much more a research question than contemporary practice, and it is quite central to my approach of studying and designing modeling tools and procedures.

The whole conglomerate of participants, with their many goals and activities and procedures and conventions, and their languages and conversations and texts (indeed, models are, of course, texts; Taylor, 1993), comprises an *interactive system*. Interaction takes place between participants, but also between participants and the models, views, documents, products, or whatever classes of artifact we might distinguish (also see Krogstie and Jorgensen, 2002). In case these artifacts are embodied in

---

[4] I deliberately avoid the term "construction" here, though the "construction of social reality" (Searle, 1995) is of course a closely related topic

computational environments kitted out with some form of artificial intelligence, the models actually "talk back" (this is already the case with, for example, automated model checkers, which report on errors and anomalies found in the models).

The knowledge and expressions of the individual participants are as much part of the system as the end products of the processes which that system enacts. Study of such interactive *social* systems is urgently required, and in fact is now increasingly on the research agenda (Rittgen, 2007; Hoppenbrouwers et al., 2005b).

However, the engineering angle to model-oriented interaction systems also demands a more utilitarian approach: the design of tools and techniques for the operational design and support of *socio-technical* model-oriented interaction systems. It will be no surprise to the reader that we advocate the goal-driven, communication-oriented concepts introduced in the previous sections as a primary basis for such design. Fortunately, we can also employ a wide range of well-known, well-developed methods and techniques from the field of information systems in order to create a supporting environment for our interactive systems. This includes formal techniques such as: database engineering; automated model checking and other forms of rule-based analysis; workflow management; but also "softer" techniques like collaboration (CSCW) and negotiation support (this is not a comprehensive list).

Conversations and interaction

Despite all the technical hullabaloo involved in implementations, the essential concepts in our view on model-oriented interaction systems very much concern *conversation*, and hence relate to such topics as language, discourse, text, pragmatics, and semiotics. Indeed, ideas as held in Language Action Perspective and Pragmatic Web communities are an important theoretical foundation for our global view on modeling.

Our view of systems development and formal modeling as a conversation dates back to (Veldhuijzen van Zanten et al., 2004; Hoppenbrouwers et al., 2005a). In the first of these publications, we already emphasized the required *rationality* of conversations for modeling, not just at a conceptual level but also in terms of efficient (cost/benefit related) decisions made concerning the depth and amount of conversation required, and techniques selected.

Recently, the conversation view on the activity of modeling has been refined and applied by (Rittgen, 2007), who concluded from an empirical study of collaborative process modeling conversations that the conversational interaction patterns and collaborative decision patterns are very similar to those in *negotiation*. Put bluntly, collaborative modeling *is* negotiation.

Part of such negotiation (especially in a cooperative setting) is *argumentation*, which directly connects to our discussion of the analysis issue in section 2 and analytical quality in section 3. Indeed, we view the finding of creative ways of guiding non-expert modelers towards easy-to-grasp argumentations (intermediary products) underlying complex models (end products) as a key approach to simplifying and supporting modeling processes. In view of some exploratory technical experimentation we have done, we expect our approach will benefit greatly from automated reasoning to help derive the target models from the conceptualizations and argumentations as systematically elicited by/within the interaction system. Such a highly rational method of course needs to be tightly interwoven with collaborative aspects, as discussed.

Setting the rules for language games

One aspect of modeling and interaction has yet been underexposed in this paper: the terminological/concept issue. This is immediately related to the "interpretation goals" as mentioned briefly in section 3. High quality collaborative modeling involves a tremendous lot of information exchange, discussion, and in unlucky cases downright bickering about the meaning of language (mostly terms). This may concern formal concepts, but usually the subject of debate is simply an informal definition that participants have to agree on. Terminological definitions, and also more formal (relational) conceptual models (often called "domain models" or "domain ontologies"), are fairly standard items in organizational modeling contexts (which does not mean they are always used, or appropriately used). The dynamics of concepts and language in a systems development context, and the management of concepts and categories in communities, is an issue in itself (Bowker and Starr, 1999; Hoppenbrouwers, 2003).

Most immediately relevant to the study of interactive modeling in a community context is the fundamental importance of conceptualization and language for any effort to create domain descriptions. Elementary concepts are omnipresent and disagreement or lack of knowledge about them can seriously undermine both formal and informal modeling. Consequently, conversation concerning elementary concepts (linguistic meta-communication; Hoppenbrouwers, 2003) will in many cases be a necessary activity in interaction systems. Interestingly, such interaction reminds us of Wittgenstein's Language Games (Biletzki and Matar, 2006; Wittgenstein, 1953)

However, the original philosophical notion of language game applies more to the *use* of language than to conversation about construction of language (though of course, such conversations are then “language games about language games”). Possibly, then (and leaving behind the strictly metaphorical origin of the notion of “language games”), conceptual meta-communication can be positioned as the activity of challenging and creating common rules for *operational* language games. These probably are no longer Wittgenstein’s language games –yet they are still games played with language as a topic.

This brief and terrifically superficial excursion to philosophy of language provides us with a nice bridge to the discussion of “modeling as a game”: it relates to the broad notion of games as a rules governing social action (in this case, *language action*). If methods (in particular, methods for modeling) are task-specific yet situation-generic prescriptions for goal-driven action, then it makes sense to view methods as games –on a metaphorical level, yes, but why not also on a practical level?

5. Systems Modeling as a Game

Finally we end our substantial run-up and return to the Main Theme of games and gaming. Clearly, the kind of gaming we are talking about here belongs to the field of “serious gaming”[5], which covers quite a broad range of games. Training and education are currently the chief areas for serious gaming (often with a strong simulation component).

Games, whether designed for entertainment or for some other purpose, are “hot”. Homo Ludens lives (Huizinga, 1998). Homo Ludens also dies, occasionally: game addiction among on-line role playing gamers is gradually getting out of hand[6]. The power of gaming is taking hold worldwide (in particular video gaming, but not exclusively so), and indeed I too have been inspired not just by the great similarities between operational modeling methods and games (see below), but also by the prospect of making operational methods and tools for modeling more accessible, more stimulating, and yes, I admit it, more “sexy”.

Lois von Ahn[7] demonstrates that offering simple but compelling classification games on the internet can lure huge numbers of people into

[5] See, for example, the home page of the Serious Games Initiative at http://www.seriousgames.org

[6] See http://en.wikipedia.org/wiki/Video_game_addiction

[7] See http://www.cs.cmu.edu/~biglou/

generously contributing time and effort to otherwise infeasible classification tasks (for example, of images on the internet), thus harnessing human intelligence for "human computing" by means of games (Von Ahn, 2006; Von Ahn and Dabbish, 2004).

Among the countless technical works on game design and implementation, there are a few that systematically discuss the fundamentals of game design (not just of video games but of any imaginable kind of game), and thus concern *Game Design Theory*: Salen and Zimmerman (2004) and Järvinen (2008). Reading these seminal works, it becomes clear that much purposeful thought has gone into ways of making games captivating; indeed, game designers have now become so good at this that serious addiction is sometimes the result. Still, if we allow ourselves to run with the devil for at least a few yards, we might learn something about how to make dull or hard tasks (including collaborative ones) more pleasantly challenging, more easily learnable and doable, and generally more effective.

But the usefulness of game design theory for method and tool engineering goes much further. Järvinen's comprehensive theoretical framework in particular provides clear concepts for analyzing and designing games that help greatly in doing game design (and therefore also aid method and tool engineering in a gaming context). Game elements, according to Järvinen, include:

1. Components: objects that the player is able to manipulate and possess in the course of the game.

2. Rule set: rules produce each individual possibility and constraint that a game has to offer for its players, including *set goals* and *procedures*.

3. Environment: the stage for game play. For example: a board, a field, or a virtual environment in a digital game.

4. Game mechanics: describe possible means with which the player can interact with game elements as she is trying to influence game states in order to complete a goal. For example: throwing in basketball, hitting in Tennis; in more verbal games (and more relevant to our sort of gaming), proposing, asking, rejecting, and so on.

5. Theme: game theme is the subject matter that is used in contextualizing the rule set and its game elements to other meanings than those which the game system as an information system requires. For example: real-estate market in Monopoly, or a fictional context, or a historical event.

6. Information: what the system and players need to know; the game state communicated. For example, a score board, or a screen display, and/or component attributes such as value or number.

7. Interface: the tools to access game elements via game mechanics when direct access is impossible. For example, game pads, dance mats, mouse, steering wheels, etc.

8. Player(s): the human factor in the game; their behavior, mood, abilities and skills, relationship with games, game tastes.

9. Contexts: the physical location of the game, the time, players personal histories, and other informal, external aspects to the game system that possibly affect the experience of playing the game.

At least the following elements are minimally required to design a game: a) components complemented with rules governing their behavior, b) an information structure to store the game states and component attributes and relations, c) at least one game mechanic to give players something to do, and d) a goal that the mechanics are designed to help completing, combined with an end or victory condition. Whatever the set of elements in an individual game is, the players interact with the elements via game mechanics. Game mechanics are compounds for game elements: minimally, they include the player(s) in the game system, as they give them the opportunity to play through performing according to their abilities and skills.

As reported in detail in (Hoppenbrouwers et al., 2008), there are great similarities between game systems and model-oriented interaction systems. For example, the game mechanics cover speech-acts (like proposing, questioning, rejecting, etc.) and conversational modes that are key to our existing conversational analysis of the modeling process as discussed in section 4.

Games are by nature extremely well suited for digitization —as is convincingly shown by the gaming industry. Large scale collaborative as well as competitive multi-player setups are among the most popular of games. Can collaborative work be shaped as a game? Can game aspects make creative but constrained work more pleasant or more productive?

As far as I am aware, using games for (formal) modeling (in particular, community-based and collaborative modeling) is a new branch in serious gaming. This also means we are still in the pioneering phase.

Arguments for approaching operational modeling as gaming

My chief arguments for taking the gaming approach to method engineering in ICT development and modeling are summarized below.

Improve motivation of modelers

Close to Von Ahn's brilliant idea, it would be very helpful from both a methodological and a productivity (industrial) point of view to make modeling more attractive, and thereby boost actual modeling in order to answer the needs of bringing AI and ICT to their full potential. A somewhat monomaniacal view for sure, but I am not above that.

Improve quality of modeling

Much more in line with common objectives in the field of conceptual modeling (especially in context of information systems and business engineering), a gaming setup may help improve the quality of the products of modeling, both *textual* (the models as such) as *contextual* (knowledge, understanding, agreement etc. across communities involved with models and modeling). As discussed, useful strategies for modeling can be built into gameplay (e.g. as sub-games, tasks, challenges) or be left to participants (i.e. pose a challenge), as best fits the situation. In case strategies are left open to the players, score systems may still help focus and guide players, in specific game instances (tactical adjustments) but also in the longer run (self-improvement; improving strategic insight and skills). People are very good at optimizing their behavior in fulfilling repeated tasks/skills: at improving their performance in view of certain clear indicators (the score system).

Make formal modeling available to non-modelers

This is perhaps the most ambitious reason for working on games for modeling. It is, however, also an urgent one –that is, if we view integration of AI and ICT in society as a blessing, not a curse. As argued in section 1, if business-IT alignment, or even "human-IT alignment", is to really take off, large scale and low-threshold formal modeling *will* be required. A difficult but obvious way to proceed is to create "Modeling Wizards": software applications that make creation of required models as painless and efficient as possible. Modeling for the masses is conditional to empowering communities in the user-driven specification of the information systems they need (de Moor, 1999).

Tooling: virtual environments for collaborative modeling

The relation between digital tools and environments for modeling and digital games is also obvious. Completely virtual work environments may not be accepted on a large scale yet, but completely virtual multi-player games most certainly are. It is quite possible that the systems development and modeling tools and environments of the future feature serious game characteristics.

Research approach: improving performance by improving game design

Moving from operational modeling to research and development, the game *metaphor* as well as the application of game design theoretical concepts will help greatly in asking the right research questions concerning model-oriented interaction systems and duly constrained modeling behavior and goals. Games can be tried and tested, providing readily available structures and data on conversations and results, and therefore offering an empirical hold on modeling processes that otherwise would be much harder to obtain in large volumes. An evolutionary cycle for improving methods/tools based on analysis of actual game interactions and results may well be of seriously benefit to focused situational method engineering.

Enable and justify study of human factors underlying (the design of) model-oriented interaction systems

Finally, on the level of scientific politics and communities, the interaction system and gaming approach to method engineering and modeling justifies a shift in focus towards human-computer studies and the modeling process, thus moving away from the technically/mathematically oriented systems modeling tradition that still puts formal languages and texts central. Human factors and matters like communication, thinking processes, and motivation then become primary citizens in the world of methods for modeling.

6. Future Directions

Obviously, I will have to put our money where my mouth is. Though I have chosen not to add half-baked examples of games for modeling to this already lengthy paper, me and some of my co-workers are in the process of developing first versions of several modeling games: one for low-threshold elicitation of basic formal process models, one for collaborative and competitive value chain modeling. We definitely work towards the creation of digital modeling games, but an important phase in game design concerns paper prototyping, in other words we also (first) develop board game test versions of our games. In the mean time, we push the setting of rules of play beyond regular natural language form and into formality, since this is required in order to computerize the board games, but also because this adds an axiomatic flavor to the rules that is useful for our more theoretical work in method and tool engineering.

We aim to first create some games for relatively standard, generically applicable activities such as process modeling, information modeling (basic concepts), value chain modeling, and business rule modeling (though the latter is a particularly tough one). Much empirical work and evolutionary development will ensue once we have our first workable prototypes.

Another interesting angle to the games-for-modeling approach is not to focus on specific games for specific modeling tasks, but on more generic gaming *aspects* to be built into digital environments for systems modeling and development. We are engaged in an analysis of possibilities for introducing game elements in an existing, industrial tool for a Dutch business engineering and ICT solutions company. Prominent ideas include the introduction of a score system linked to quality metrics, the introduction of collaborative modeling support linked to group task setting (goal driven group workflow: challenge task forces to tackle a mission), and also experimental use of our “Business Process Modeling game” as an add-on to the core environment. The company in question is convinced that with a new, “digital native” generation arriving on the job market within a few years, providing fully virtual collaborative environments with gaming characteristics may give them a real competitive advantage.

With the same company, we are negotiating the possibility of getting access to logs of the interaction patterns and mutations of models as occur in real life use of their operational modeling environment (modeling tool). This would create good opportunities for empirical study of interaction patterns as they occur in actual, operational modeling, which in turn may advance game design/method engineering.

As I write this, the idea of taking the gaming approach in methods and tools for formal modeling is approximately half a year old. Despite good intentions, preliminary research, and actual game design being on its way, we have no concrete results yet. I hope to show you a working prototype of at least one game at the actual event. In the mean time I hope the ideas presented will at least fire your imagination and perhaps initiate some fundamental discussion concerning my own little corner of organizational engineering and communication.

REFERENCES

Berners-Lee, T., J. Hendler, O. Lassila (2001): The Semantic Web. In: Scientific American Magazine - May, 2001.

Biletzki, A. and A. Matar (2006): Ludwig Wittgenstein. In: The Stanford Encyclopedia of Philosophy (Winter 2006 Edition), Edward N. Zalta (ed.), URL = <http://plato.stanford.edu/archives/win2006/entries/wittgenstein/>.

Bommel, P. van, S.J.B.A. Hoppenbrouwers, H.A. Proper, and J. Roelofs (2008): Concepts and Strategies for Quality of Modeling. In: Halpin, T., Proper. H.A., and Krogstie, J.: Innovations in Information Systems Modeling: Methods and Best Practices. Advances in Database Research series. Hershey, USA: IGI Global.

Bowker, G.C., and Star, S.L. (1999): Sorting Things Out - Classification and Its Consequences. Cambridge, MA: MIT Press.

Dietz, J.L.G. (2006), Enterprise Ontology - theory and methodology. Heidelberg, Berlin, New York: Springer-Verlag.

Elgood, C (1993): Handbook of Management Games, 5th ed. Aldershot: Gower Publishing.

Falkenberg E.D., A.A. Verrijn-Stuart, K. Voss, W. Hesse, P. Lindgreen, B.E. Nilsson, J.L.H. Oei, C. Rolland, and R.K. and Stamper, R (1998): A Framework of Information Systems Concepts. IFIP WG 8.1 Task Group FRISCO, IFIP, Laxenburg, Austria.

Hoppenbrouwers, S.J.B.A. (2003): Freezing Language – Conceptualisation Processes across ICT-Supported Organisations. PhD Thesis, Radboud University Nijmegen.

Hoppenbrouwers, S.J.B.A., H.A. Proper, and Th.P. van der Weide (2005a): Formal Modelling as a Grounded Conversation. In G. Goldkuhl, M. Lind, and S. Haraldson, editors, Proceedings of the 10th International Working Conference on the Language Action Perspective on Communication Modelling (LAP'05), pages 139–155, Kiruna, Sweden, EU, June 2005. Linköpings Universitet and Hogskolan I Boras, Linköping, Sweden, EU.

Hoppenbrouwers, S.J.B.A., H.A. (Erik) Proper, and Th.P. van der Weide (2005b): A Fundamental View on the Process of Conceptual Modeling. In L. Delcambre, C. Kop, H.C. Mayr, J. Mylopoulos, and O. Pastor, editors, Conceptual Modeling – ER 2005 – 24 International Conference on Conceptual Modeling, Klagenfurt, Austria, EU. Volume 3716 of Lecture Notes in Computer Science, pages 128–143. Berlin: Springer Verlag.

Hoppenbrouwers, S.J.B.A., P. van Bommel, and Aki Järvinen (2008): Method Engineering as Game Design: an Emerging HCI Perspective on Methods and CASE Tools. Submitted to EMMSAD'08.

Huizinga, J. (1998): Homo Ludens: A Study of the Play-Element in Culture. Translated in English from the 1939 Dutch text. London: Routledge.

Järvinen, A. (2008): Games without Frontiers, Theories and Methods for Game Studies and Design. PhD Thesis, University of Tampere, Finland.

Kecheng L., Clarke, R.J., Andersen, P.B., Stamper, R.K., and Abou-Zeid, E.-S. (Eds.): Organizational Semiotics -Evolving a Science of Information

Systems. International Federation for Information Processing publications, Vol. 94. Berlin: Springer Verlag.

Krogstie, J. and Jorgensen H.D. (2002): 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: Springer Verlag.

Krogstie, J., G. Sindre, and H. Jorgensen (2006). Process models representing knowledge for action: a revised quality framework. European Journal of Information Systems, 15:91-102.

Moor, A. de (1999): Empowering Communities: a Method for the Legitimate User-Driven Specification of Network Information Systems. PhD Thesis, University of Tilburg.

Nonaka, I. and Takeuchi, H. (1995): The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation. Oxford University Press.

OMG (2006): BPMN 1.0: OMG Final Adopted Specification, 2006.

Ralyté, J., Brinkkemper, S., and Henderson-Sellers, B. (2007): Situational Method Engineering, Fundamentals and Experiences. IFIP vol. 244, , pp. 313-327. Boston: Springer.

Rittgen, P. (2006): Negotiating Models. In: J. Krogstie, A.L. Opdahl, and G. Sindre, Advanced Information Systems Engineering, Proceedings of the 19th International Conference, CAiSE 2007, Trondheim, Norway, June 11-15, 2007. Lecture Notes in Computer Science 4495. Berlin: Springer Verlag.

Rasmusen, E. (2006): Games and Information: An Introduction to Game Theory (4th ed.). London: Wiley-Blackwell

Salen, K. and Zimmerman, E. (2004): Rules of Play, Game Design Fundamentals. Cambridge, MA: MIT Press.

Schoop, M., A. de Moor, and J.L.G. Dietz (2006): The Pragmatic Web: a Manifesto. In: Communications of the ACM, 49, 5, (May 2006).

Searle, J.R. (1995): The Construction of Social Reality. London: Penguin Press.

Taylor, J. R. (1993): Rethinking the theory of organizational communication: how to read an organisation. New Jersey: Ablex Publishing.

Veldhuijzen van Zanten, G.E., S.J.B.A. Hoppenbrouwers, and H.A. Proper (2004): System Development as a Rational Communicative Process. In: Journal of Systemics, Cybernetics and Informatics, Nr: 4, Vol: 2, International Institute of Informatics and Systemics (IIIS), 2004.

Von Ahn, L. (2006): Games With a Purpose. In: Computer 39, 6 (Jun. 2006), 92-94.

Von Ahn, L. and Dabbish, L. (2004): Labeling images with a computer game. In: Proceedings of the ACM CHI, 2004.

Weigand, H. (2003): The language/action perspective. In: Data & Knowledge Engineering (47:3) 2003, pp 299-300.

Wittgenstein, L. (1953): Philosophical Investigations. G.E.M. Anscombe and R. Rhees (eds.), G.E.M. Anscombe (trans.), Oxford: Blackwell.