Method Engineering as Game Design: an Emerging HCI Perspective on Methods and CASE Tools

Stijn Hoppenbrouwers[1], Patrick van Bommel[1], and Aki Järvinen[2]

[1] Radboud University Nijmegen, the Netherlands
[2] University of Tampere, Finland

Abstract. In the last half decade, there has been increasing interest in the process of information systems modeling, mostly focusing on improvement of quality, efficiency, and control. While generic guidelines and phasing have been explored to a considerable extent, we are still a long way removed from in-depth understanding, full support, and adequate tooling with respect to operational modeling processes. Building on existing work in modeling methods and method engineering, we propose to expand the scope of modeling process research by taking a Human-Computer Interaction approach, viewing (situational) methods and their tool support, in combination with participants in operational process enactment, as an operational interaction system. More in particular, we discuss the merits of using the Game Metaphor as a clarifying and goal-setting view on the design of such interaction systems. Thus we approach Method Engineering as Game Design.

1 Introduction

Modeling (i.e. conceptual modeling, but also other types) is a central activity in Information Systems engineering and a key topic in IS research. The promise of model driven systems engineering and Model Driven Architecture (MDA [1]) reflects a tantalizing vision of better and easier representation, visualization, analysis, and manipulation of conceptual structures in information and knowledge systems development, and far reaching advantages that result from it. Still, much is left to be desired concerning the scale and quality of modeling in practice. Industrial use of models is widespread, but mostly relatively primitive in view of the advanced techniques that are in fact increasingly available. In addition, even simple, standard models are often not realized because doing so, justifiably or not, has a low priority or is considered too hard or expensive. Meanwhile, developers of advanced model-dependent techniques (related to the semantic web, knowledge-based systems, business engineering, etc.) often *assume* models to be available, whereas in practice, creation and management of good models is not taking place on a sufficient scale. This *operational modeling bottle neck* increasingly hampers advance in IT and Business-IT Alignment.

If the promise of model driven systems engineering is to be lived up to, more needs to be done in order to further the operationalization of modeling. Modeling

will have to become more efficient, easier (open to people with less modeling expertise), rendering evident return on investment, and take place (where relevant) in a truly collaborative setting. Tooling will be a large part of the answer. Human factors and Human-Computer Interaction issues will have to be systematically addressed, going beyond aspects usually associated with modeling, i.e. include operational process management and control, usability of modeling methods and tools, communication aspects, ergonomic aspects, and even motivational factors like enjoyment (or aggravation) and challenge (or boredom). In general, we have to ask ourselves, and find answers to, the question: *why does modeling not take off on the scale needed to fulfill its promises, and what can we do about it?*

There is wide consensus (for example in the field of method engineering; see next section) that if we look at modeling, this should involve modeling languages as well as modeling processes. In industry and academia alike, extensive attention has been paid to the former, perhaps excessively so; we refer to the YAMA Syndrome (Yet Another Modeling Approach; [2]). Far less research concerns the process (activity) of modeling. We observe that in investigating operational modeling, the *process* is of paramount importance. As argued in various previous papers, a number of process aspects are key to understanding and improving operational modeling [3]. Some aspects of quality, it seems, can be better achieved through a good modeling process than by just imposing requirements on the end product and introducing a standard feedback cycle. This holds in particular (though not exclusively) for matters of validation and grounding in a socio-political context (as in collaborative modeling). Also, if we want CASE tools to develop beyond mere model editors, a more process-oriented approach seems promising. As suggested, operationalization (in particular if supported by advanced tools, which we believe to be crucial) also requires a full-blown, dedicated investigation of usability and interaction-oriented functionality (HCI).

This paper mostly concerns a discussion of a research paradigm and design metaphor that we believe to be rather helpful in taking the direction suggested above. Put simply, we propose to approach method and tool design as *game design*. This is not so much an entirely new direction as a helpful extension and integration of existing directions. We will discuss parallels between game design and method engineering, and the merits of the proposed approach.

2 Related work in methods and method engineering

As indicated, perhaps the most immediately relevant field with respect to "method engineering as game design" is (Situational) Method Engineering (SME). This field has an open eye to both language and process aspects of methods, and to useful ways of structuring and representing methods ("method modeling"); see for example [4–6]. It also explicitly aims to study method engineering in context of tool support and operationalization. Our current work can indeed be seen as part of method engineering. We believe, however, that our particular focus is not mainstream in SME because for now it is less on "situational" aspects of methods, and also because our focus strictly concerns operational *modeling*

methods, viewed in terms of *model-oriented "interaction systems"*, suggesting a truly interdisciplinary approach also borrowing from IS analysis and design, HCI, process management, systems dynamics, CSCW, and so on.

A goal-driven approach to operational method engineering almost inevitably involves the notion of "quality of models", as the operationalization of "good modeling" will have to be based on a setting of clear modeling goals. Substantial work has been done on this; for overviews, see [7, 9]. Much less work has been done, however, on quality of the process of modeling. Roughly speaking, two mains directions have been proposed: identifying *courses of action* (strategies) that lead to quality models [12, 11] and identifying *metrics and variables* for measuring process quality [10, 13, 14]. Also relevant are [15] concering competencies of participants in modeling processes, and [16], on a practice-based overview of key concepts for analysis of modeling processes.

Related to the metrics-and-variables approach, but much more rooted in management science and process optimization, are some rare efforts to apply Systems Dynamics (SD) to modeling-related processes, for example [17, 18]. We are hopeful that applying SD to processes for modeling, based on variables provided by the quality metrics approach rooted in conceptual modeling, will help analysis and operational optimization of processes for modeling.

A further direction focuses on collaborative aspects of modeling. Relevant fields include CSCW [19] and collaboration engineering [20]. For underlying theory, the Language Action Perspective approach includes a number of relevant contributions (for example, [21]), involving disciplines like semiotics, speech act theory, argumentation theory, and discourse analysis. As part of the collaborative modeling direction, increasingly attention is paid to negotiation [22].

Given the highly interdisciplinary nature of the subject, exploration of relevant literature, which is as fragmented as it is diverse, is an ongoing process. We do not claim the above overview to be comprehensive.

3 The interaction system approach to operational modeling

As mentioned, Method Engineering as a branch of the Information Systems field is related to issues like operational modeling (i.e. practice-oriented, actual modeling processes) as well as tool support for such processes (mostly, CASE tool design). However, the ME tradition still centers mostly on conceptual aspects of modeling languages, and requirements of methods. Processes (procedures for modeling) are dealt with mostly at the level of phasing; analysis and support of detailed interaction between modelers and between modelers and models is not at the core of the field. This focus under emphasizes a number of factors with respect to successful operationalization of modeling processes.

We advocate a different, more holistic view on operational modeling: that of an "interaction system" (in the socio-technical sense, i.e. including various actors and relevant inter-actor interaction) which *brings forth models*. This notion relates to some hitherto minor motives in the IS literature: Interactive Models

[8] and Second Order Information Systems (i.e. information systems that bring forth information systems) [23]. The basic idea is that formal or semi-formal models (including conceptual models, business process models, workflow models, business rules, and so on, and various *views* or *visualizations* thereof) are increasingly the objects that form the primary input for business-oriented IT development (this is in line with MDA etc.). More and more often (and going beyond mainstream MDA) software is generated or parametrized strictly on the basis of such models. This means that interaction with the models changes the system, directly or indirectly. Thus, model manipulation and the human-human interactions involved in it can be safely claimed to be an increasingly important form of Human Computer Interaction. This triggers us to expand our scope on method engineering to encompass, at least in principle, the whole operational socio-technical system that brings forth or manipulates models. The full scala of available (human-centered) IS design and evaluation techniques is waiting to be unleashed on such systems (including the method aspects embedded in them), involving factors like usability, user acceptance, motivation, but also efficiency, effectiveness, etc.

Clearly, CASE-tool-like software is to be an integral part of model-oriented interaction systems, but we believe that a next generation of CASE tools will go way beyond the advanced *editors of representations* they now often are, and will be much more explicitly *process oriented* and *collaborative*, showing properties currently associated with CSCW systems, negotiation systems, decision support systems, dialog systems, etc. This is not to say that no such properties are at all present in current, advanced CASE tools, just that they should become *primary*. As is good practice in IS design, they should be based on extensive study of *interaction patterns*, *activities*, *tasks*, *regulations*, and *strategies* that occur in (collaborative) modeling practice, and of improvement and support thereof. Research as reported in for example [22, 8, 12, 10] moves in this direction, but we feel it could benefit from a more holistic, more operations-oriented (practical!), and also more ambitious context.

Without going as far as providing a generically viable architecture for "model-oriented interaction systems" here, we suggest (based on papers mentioned, including our own work) that such systems could be analyzed and engineered along the lines of following layers or components:

Interface: templates, views, communication. This is where interaction with the models, as well as any *digitized* interaction between participants in the modeling process, takes place.

Procedures. This is a workflow-like component that helps guide and coordinate the interactions *where required*. In many cases, this should not entail detailed prescription of steps in task execution; in some, it may. Still, some assistance in coordinating the tasks and interaction in time is highly desirable.

Goals and Tasks. We believe model-oriented interaction systems should be largely goal-driven. This requires the definition of goals, linked to some conception of which tasks are to be done to achieve the goals [10].

Basic (inter)actions. Though actual interactions take place through the interface, we intend to also keep track of them at a much more generic, basic, analytical level. Interactions will be analyzed at the level of speech acts (discourse analysis), reflecting the detailed and essential communication between participants and the system. This should allow us to abstract from specific views and templates and study (and perhaps guide) generic patterns in conceptualization, collaboration, negotiation, etc. [21, 22].

Rules for Deliverables and their Context. Linked to Goals and Tasks, we need a mechanism (probably rule based) that enables checking of goals and sub-goals achieved against a record of actions performed and the state/status of deliverables. Such rules mostly concern static requirements like syntactic restrictions on models, dependencies between deliverables (for example, "if you have X you must also have Y"), and contextual requirements (for example, "agreement must exist between such-and-such stakeholders on item X") [10].

Repository. This is where all explicit information on deliverables, participants, interactions, context, and process metrics is represented and kept. It is the operational dynamic model repository and a log of the interactions throughout the process; it doubles as a research database.

Please note that the above "components" are not interrelated strictly on the basis of the "stack" presented above. They are to be reflected in an architecture with multiple interrelations. Also, the component sketch above is not definitive.

4 Game design: a brief anatomy

In the wake of the enormous growth of the Video Game industry, but also building on existing academic work in the study of games (not just video games but any imaginable form of game, from systemic, communicational, design, and cultural perspectives), a new field is emerging called *Game Design Theory* (GDT). Let us point out forthwith that we do not refer to Game Theory here (though some links exist between the fields). In view of the current discussion, one could say that Game Theory helps analyze strategies for playing (and winning) formalized games, whereas Game Design Theory helps analyze and design the rules etc. of games without including the behavior of (human) players. GDT leaves the actual playing to humans. In view of method engineering, this is crucial.

We refer to two publications on GDT that we found most useful, and that are quite comprehensive: Järvinen (2007) [24] and Salen and Zimmerman (2004) [25]. While both texts provide a wealth of concepts and methods for in-depth analysis and design of game systems, in this article we use [24] as a reference point for a "brief anatomy of game design". We can of course only give a superficial overview. Note that we aim to compare Game Design and Method Engineering in the next section, and that in view of this we elaborate more on some game elements than on others.

Järvinen's framework for game analysis and design provides nine categories of key "game elements":

1. **Components**: objects that the player is able to manipulate and possess in the course of the game. For example: a deck of cards, pieces in a board game, a football, a character such as Pac-Man.

2. **Rule set**: rules produce each individual possibility and constraint that a game has to offer for its players. At its core, designing games equals designing rules, i.e. constructing a rule set, or implementing existing rule sets for new games.

In general the rule set defines relationships between the other game elements, but there are two main functions of rules that are particularly central in game design. First, rule sets define *goals*. For example: Guess 7 correct, Score more points than your opponent, Be first in goal, Save the Princess.

Second, rule sets state *procedures*. For example: the dealing of cards or a throw of a dice, game system or referee actions, game component behavior by artificial intelligence (AI) in digital games. In Järvinen's framework, procedures are defined as actions that the game system (rather than players) takes. It takesactions with the following purposes: 1) for assigning value to different game states and outcomes by handing out rewards or penalties, and 2) for governing the interrelations of game elements, or their attributes. Both types of procedure handle information within the system.

The distinction between system actions (i.e. procedures) and player actions (i.e. game mechanics, see 4. below) is useful because players often perform game mechanics such as running, jumping, etc. whereas game systems do not perform, but rather, they instantiate actions, such as measuring procedures (e.g. keeping time for runners, or keeping scores).

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, Placing Dominoes, Maneouvring in Gran Turismo, Guessing in Lottery games, etc. Usually, game mechanics have qualitative aspect to them, i.e. whether the player manages to perform the game mechanics is uncertain, and this is part of what creates excitement and unpredictability as to how the game plays out.

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 (World War II in the Axis Allies or Medal of Honor game series), etc. Theme has a metaphorical function: it helps the player understand the system of rules as a game in terms of a particular subject matter rather than the rules in themselves.

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 com-

ponent attributes such as value or number. Different types of information are stored in game systems. It can be categorized according to the structure it is organized into, i.e. according to games as "micro-worlds" with agents, events, and objects:

- Information about events: outcome information, e.g. success and consequences of game mechanics.
- Information about agents: player roles, standing and location in the game, also concerning system-operated players (AI).
- Information about objects: the attributes of components.
- Information about the game system: information in the form of procedures stated in the rule set and information about game states, the complexity of which is defined by the configuration of game elements, i.e. the complexity of the parts of the system as a dynamic whole.

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.

Why is it that all games require at least a rule set, information, components, and mechanics? The answer is that otherwise the players would have no motivation (goals) nor means (mechanics) to play the game, nor objects (components) to focus their actions towards, nor feedback from the game system regarding their actions (information and rule set procedures, including a score system). Themes and interfaces are elements that are genre-specific or technology-specific in nature, but nevertheless common enough to warrant their own element types.

5 Operational method engineering as game design

We trust the reader has already spotted some strong parallels between our own summary of recent process-oriented work in Method Engineering (section 3)

and Järvinen's framework for Game Design. We will proceed to systematically point out those similarities, which go beyond being merely interesting and create opportunities for actual implementation, testing, and evolution of operational modeling methods (i.e. model-oriented interaction systems) in the form of games. We start out by exploring what game elements (conform sections 3 and 4) would be typically expected to feature in the design of "games for modeling".

Components. In an "operational modeling game" these would very likely be the various *intermediary deliverables* and *end deliverables*, and elements thereof. One could for example encounter components like objects, relations, processes, definitions, proofs, textual descriptions, scenarios, grammatical analyses, lexical items, and so on.

Rule set. As can be expected in relation to a rule-based approach to method modeling, the rule set is key in defining games for modeling. In line with [24], we distinguish between two rule sets:

Goals. This has a direct parallel with the "Goals and Tasks" component of the interaction system sketch in section 3. Ultimately, the goal of a modeling game would of course be "make such-and-such model", with underlying sub-goals conform further requirements (quality aspects, including syntax, semantics, and pragmatics) of the model. However, in a process-oriented method approach, intermediary steps and deliverables (for example, a first, sketchy type analysis or a rough draft of a process flow) may be defined as sub-goals, and even be assigned "sub games". Finally, process-oriented goals like timely delivery or effective iteration may also play a role.

Procedures. Järvinen's interpretation of "procedures" differs significantly from the one in method engineering (section 2). The latter refers to patterns for participant behavior: advised or prescribed ways for the participants to structure and fulfill their tasks. Järvinen, on the other hand, explicitly refers to procedures as not applying to players, but to the game system. In Järvinen's framework, procedures-for-players are reflected by game mechanics. Though in that framework it is possible to combine elementary game mechanics into more complex mechanics [24, p377], a workflow-like interpretation of procedures (with the "workflow" coordinating tasks in order to achieve goals) does not readily fit Järvinen's game elements framework.

So how *do* task coordination and strategies relate to that framework? Tasks (either broken down into or directly corresponding to game mechanics) may be directly linked to the creation of or iteration between deliverables (intermediate or end), and also to socio-pragmatic aspects (reach agreement about something, safeguard mutual understanding, validate). Note that they are ultimately meant to fulfill goals (in line with [10]). At an advanced level, then, workflow-like rules may reflect or trigger strategies to be applied in certain situations. This means they guide player behavior. As a consequence, strategies rather belong to the realm of Game Theory, not Game *Design* Theory.

Strategies are for the players to think up and execute (so they indeed are *not* part of game design). Alternatively, however, strategies may be built into the

game as stages or activities (with intermediary deliverables), actively helping the players to work toward some end goal. Such stages are of course part of a specific game design, but indeed have no place as key elements in the game design framework (a meta-model).

As for the interpretation of "procedure" that does fit Järvinen's terminology (i.e. the rules-for-the-game-system interpretation): such game system actions also play an important role in the design of games for modeling. They define the score system (rewards and penalties), and the interrelationships between game elements (not unlike a conceptual analysis of the game system, for example by using an ER or ORM diagram or a UML Class Diagram, and/or rules/constraints). Once again, this confirms a strong overlap between our rule-based approach to operational method engineering, and Järvinen's version of Game Design Theory.

We conclude that despite the terminological confusion concerning "procedures", our existing approach to method engineering (section 3) and Järvinen's framework (section 4) are quite compatible with respect to player action, game system action, and gaming strategies.

Environment. The "stage for gameplay" in a modeling context is not readily defined. Creativity will be called for here, unless we stick to straightforward solutions like "a two-dimensional field in which a diagram is drawn" or other environments already present in current model editors. In any case, the templates, views, visualizations etc. in model-oriented interaction systems (section 3) are for a considerable part reflected in the game environment; the game mechanics (see below) are acted out in this environment.

Game Mechanics. Based on the mechanics library in [24, p378], the interaction types applying to modeling could for example be mechanics like allocation, arranging, browsing, building, choosing, contracting, conversing, discarding, expressing, information-seeking, sequencing, storytelling, submitting, substituting, taking, voting. Interestingly, the majority of the game mechanics listed in [24] seem fit for possible application in a modeling game. In addition, there seems to be a potential relation between at least some of the mechanics above and the "basic interactions" at conversation level referred to in section 3 and in several existing publications in method analysis (e.g. proposing, rejecting, arguing, agreeing, etc.). In addition (as discussed above), more complex, compound game mechanics may be specified that amount to "tasks" (which the player can choose to perform as part of the gameplay), but we should take care not to confuse game mechanics and strategies.

Theme. This is perhaps the most alien of the game elements in view of games for modeling. Putting specific games in context of the domains being modeled does not seem an appropriate way of dealing with themes. A more fruitful approach may be to present the modeler with a metaphorical assignment that helps focus the mode of communication. For example, a theme might be used distantly resembling the "telephone metaphor" as used in conceptual modeling: "imagine you you have to tell a totally literally-minded Martian/genie/gnome

exactly what domain X (or a certain view thereof) looks like. What would you tell it?".

Information. As we view our interaction systems primarily as information systems both supporting and embodying modeling, there is a clear over-all parallel here. Information about modeling events, agents (participants), objects (deliverables and elements thereof), and the system (rules, scores, states, etc.) is present all around, and is administrated by means of a repository.

Interface. This is of course also a primary component of the envisaged operational interaction system (section 3). The objects (deliverables) need to be acted upon, thus operationalizing the game mechanics (basic interactions). Once more, we can start out designing game interfaces similar to existing CASE tool interfaces, but a game context may inspire more innovative forms of interfacing (and thereby of visualization of views), for example dynamic touch screens, three dimensional manipulation (even by means of data gloves and goggles), speech recognition, and so on. Results from mainstream HCI research would have to be consulted and integrated here.

Players. Obviously, these are the participants (possibly also stakeholders in a wider sense). Relevant aspects include preferences, behavior, competencies, expertise, etc. of various roles/stakeholders in modeling. Tuning the gameplay to the players (or vice versa!) is a key part of operational and SME, and one of its greatest challenges. Note that under this item, the players as such are denoted; information about players is in part related to "context" and kept under "information".

Contexts. These are as diverse and multi-faceted in modeling as in gaming in general. Parallel gaming, multi-player gaming, realistic versus educational contexts, virtual environments versus board games or settings similar to management games: many options are open for exploration. Specific information on the backgrounds, capacities etc. of players belongs here as well.

In addition to the above analysis based on game elements, we would like to point out some further issues in "modeling as a game".

Rule-based method modeling

Various approaches in method modeling, including our own, to some degree use rules (or schemas that can be transformed to rules) to represent methods. This is particularly useful if methods (method support) are to be implemented in a CASE tool, as it enables techniques like automated model checking (consistency, syntactic correctness, etc.), customization of languages (rules parametrize the modeling language), but also "context checking" (meta-data on the model that links it to its social context and the requirements on agreement; pragmatic quality, etc. [8]) and even situational procedural regulation (process checking and guidance, e.g. by means of a dynamic "modeling agenda", as we suggested in previous work [12]).

Score systems. Score systems tell players how well they are doing, and scores usually are central in victory and end conditions of games. A central idea in our game-based vision for method engineering is to devise score systems linked to operational quality metrics. In theory this should enable us to encourage and help the players to achieve highest possible quality without prescribing them what to do in too much detail. Designing such score systems is a research question rather than a solution, especially since it requires the development of quality metrics first. And yet, it seems to provide at least a possible answer to the urgent question how to guide participants toward high quality models without overly prescribing them what to do (detailed procedures). It also points toward a certain way of shaping and deploying quality metrics. Notably, the idea of using a quality score system is not all new; for example, an existing tool for rule management called RuleXpress (a business rule editor: see www.rulexpress.com) already successfully includes an elaborate quality score system, yet without using a further game-like context.

Playability and emotive factors. A crucial aspect in game design that has been mostly neglected in this paper is the link between the game system and the emotions and experiences it creates in the players. In view of the "utility of games", it can of course be argued that whereas to invoke entertainment, excitement, challenge etc. are primary goals in many games, in "serious gaming" (including military and technical simulations, management games, and so on), goals like education, training, and even controlled execution of real tasks are more important. However, from a HCI perspective, emotive and motivational factors cannot be ignored. It would certainly not hurt to make modeling fun, or at least make it a pleasant challenge. Game Design Theory has much to offer here. But even if we discard the evocation of positive emotions, there still is the more urgent issue of avoiding negative emotions (frustration, inadequacy, boredom). Linking this to preferences, tastes, interests, and capabilities of specific players, emotive analysis may well help us design the right game (method) for the right player.

Figure 1 presents an overview of relations between key elements of "model-oriented interaction systems" and "games for modeling". Note that several element under "Interactive Modeling System Components" are not explicitly related to "Game Elements Applied to Games for Modeling". First, *procedures* (left column). As discussed above, such procedures (workflow, strategies) are either reflected in some particular game design (as stages or sub-games), or are left for the players to enact. Also, *domain specific context* and *participants/stakeholders* (middle column) stand alone. The domain specific context is left implicit in previous work on operational modeling, but could indeed be fruitfully introduced. The participants/stakeholders are of course part of interaction systems as well but again were not explicitly included in the interaction system framework. This illustrates that using the game system approach provides a comprehensive view on interaction systems.

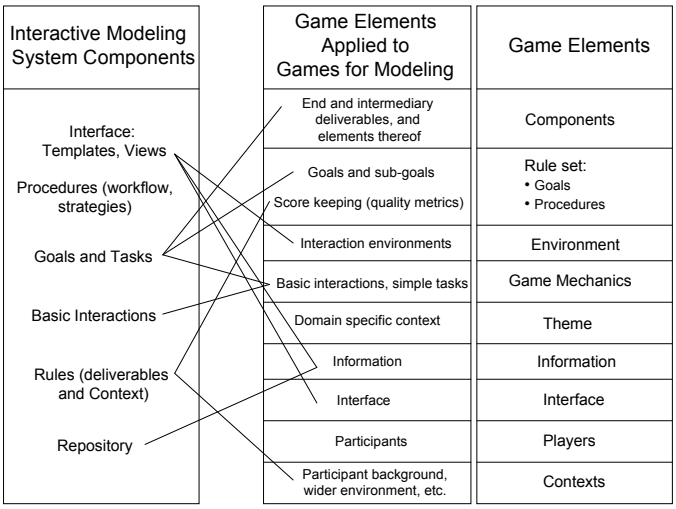

Fig. 1. Relationships between game elements and interaction system components

6 Conclusions and future directions

We have argued in favor of applying the game metaphor to operational method engineering (with a focus on modeling): to approach the analysis and design of systems embodying and supporting the creation of models as if it concerns the design of games. We have discussed recent work in method engineering, in particular process-oriented work (some of it our own) which emphasizes modeling-related human-human and human-computer interaction. We also presented the basics of a state-of-the-art, comprehensive framework for game design, and compared concepts from process-oriented method engineering with that framework. We concluded that basic elements of Game Design Theory match the elements present in process-oriented method engineering to a very large degree. In addition, we put forward the following arguments in favor of approaching operational Method Engineering as Game Design:

- Implementation of "games for modeling" as digital games is a clear option, which is supported by Game Design Theory and experiences in video game development, and also by the great similarity between "interaction systems" and "game systems". This suggests a promising direction of creating CASE tools (virtual collaborative modeling environments) with game characteristics. "Playability" of the game is then analogous with "usability" of the interaction system as well as "workability" of the method.
- Shaping modeling methods as games, digitally or not, creates good opportunities for controllable, repeatable experiments concerning operational mod-

eling. Data gathering is greatly helped by the game information structure that is inherently in place. Also, gameplay requires clearly defined goals. The gaming metaphor makes various aspects of experiment design and method engineering clearer and forces us to address all matters relevant to the operationalization of methods for modeling. In sum, the gaming approach provides an ideal context for operational method evaluation and research.
- In a similar vein, the design of score systems that reflect quality metrics (still very much a research question!) provides opportunities for informing modelers about how they are doing as the game is played, and thereby guide them without imposing restrictive procedures. Thus we hope to design environments that optimize human creativity and ingenuity in solving modeling problems while also safeguarding formal and pragmatic quality requirements.
- Though not explored in any depth in this paper, Game Design Theory offers possibilities to analyze and design emotive and motivational factors in modeling, which play a role in the operationalization of methods.
- Whereas we emphasized the distinction between game theory and game *design* theory, shaping methods as games does improve possibilities to explore game theoretical approaches to finding the best strategies for optimizing scores in games for modeling. Note, however, that we believe that reasonably successful game design and implementation *with human players* will be required before serious, empirically backed up explorations of applying game theory may become possible. This may eventually lead to partial or even full automation of modeling tasks (bringing modeling into the realm of Multi Agent Systems). We hope to have made it clear that we want to study and further actual, collaborative human-human and human-model interaction before engaging in automation of modeling. Still, support of interactive modeling can benefit greatly from AI techniques; our rule-based approach may be an enabling factor here.

The work to be done is reasonably clear. We will design some initial pilot games (in fact we are in the process of doing so), run those games, gather data, and start exploring and improving them. We aim primarily at games for enterprise engineering and business modeling, but do not rule out other areas in which low-threshold formal modeling is an issue. For the games, we will set goals that reflect real life goals rather than academic ones (for example, we prefer to use industrial modeling languages and formats). We will explore game types and specific game designs, borrowing from existing games where possible and attempting some out-of-the-box thinking about modeling and modeling support. Cooperation with CASE tool developers will be sought.

We have mostly referred to operational methods and games as if these are to be applied in real-life modeling, i.e. in the execution of actual industrial modeling tasks. While this is indeed our long term vision, we do recognize that for various reasons, this is not a realistic option in the shorter term. This still leaves good opportunities for applying the game metaphor in research and education. Also, it is important to note that application of the game metaphor to methods and CASE tools need not entail full-blown video game play. Just the introduction

of clear rules and a score system, or other characteristics of games, in methods or virtual modeling environments may also be helpful. Finally, we refer to the flourishing field of serious games and simulations, and point out that to introduce game-like ways of supporting formal and semi-formal modeling in that field may lead to interesting cross-fertilization.

References

1. Mellor, S.J., Scott, K., Uhl, A., and Weise, D.(2002): "Model–Driven Architecture". *OOIS 2002 Workshops*, 290–297. Springer Verlag.
2. Oei, J.H.L., van Hemmen, L.J.G.T., Falkenberg, E.D., and Brinkkemper, S. (1992): *The meta-model hierarchy, a Framework for Information Systems Concepts and Techniques*. Technical report 92-17, Dept. of Information Systems, University of Nijmegen.
3. Hoppenbrouwers, S.J.B.A., Proper, H.A., and van der Weide, Th.P. (2005): "A Fundamental View on the Process of Conceptual Modeling". In: *Conceptual Modeling – ER 2005 – 24th International Conference on Conceptual Modeling*, Klagenfurt. Springer: Lecture Notes in Computer Science 3716.
4. Mirbel, I., Ralyté, J. (2006). "Situational method engineering: combining assembly-based and roadmap-driven approaches". In: *Requirements Engineering*, 11, 58-78.
5. Van de Weerd, I., Brinkkemper, S., Souer, J., & Versendaal, J. (2006). "A situational implementation method for web-based content management system-applications: method engineering and validation in practice. IEEE, Special issue of *Software Process: Improvement and Practice*.
6. Ralyté, J., Brinkkemper, S., and Henderson-Sellers, B.: *Situational Method Engineering: Fundamentals and Experiences*, pp. 313-327. IFIP vol 244, 2007. Boston: Springer.
7. Moody, D.L. Theoretical and practical issues in evaluating the quality of conceptual models: current state and future directions. Data and Knowledge Engineering, (55):243- 276, 2006.
8. 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. Olive, M. Piat- tine, G. Poels, J. Roddick, K. Siau, M. Yoshikawa, and E.S.K. Yu, editors, 21st Interna- tional Conference on Conceptual Modeling (ER 2002), volume 2503 of Lecture Notes in Computer Science, pages 351-363, Berlin, Germany, EU, 2002. Springer.
9. Poels, G., Nelson, J., Genero, M., and Piattini, M. (2003): "Quality in Conceptual Modelling – New Research Directions". In: A. Olivé (Eds.): ER 2003 Ws, LNCS 2784, pp. 243-250. Springer.
10. van Bommel, P., Hoppenbrouwers, S.J.B.A., and Proper, H.A. (2007): "QoMo: A Modelling Process Quality Framework based on SEQUAL". In: Proceedings of EMMSAD'07, Trondheim.
11. Anthony, S., Batra, D., and Santhanam, R. (2005): "The Use of a knowledge-based system cino nceptual data modelling". In: *Decision Support Systems*, 41:176-190.
12. van Bommel, P., S.J.B.A. Hoppenbrouwers, H.A. (Erik) Proper, and Th.P. van der Weide (2006): Exploring Modelling Strategies in a Meta-modelling Context. In R. Meersman, Z. Tari, and P. Herrero: proceedings of OTM Workshops: ORM'06, pp1128-1137, Springer, 2006.

13. Nelson, H.J., and Monarchi, D.E. (2007): "Ensuring the Quality of Conceptual Representations". In: *Software Quality Journal*, 15:213-233. Springer.
14. Moody, D.L. (1998): "Metrics for Evaluating the Quality of Entity Relationship Models". In: T.W. Ting, S. Ram,and M.L. Lee (Eds): *ER'98*, LNCS 1507, pp211-225. Springer.
15. Frederiks, P.J.M., and Th.P. van der Weide (2006): "Information Modeling: the process and the required competencies of its participants". In: *Data & Knowledge Engineering*, Nr: 1, Vol: 58, Pages: 4-20.
16. Hoppenbrouwers, S.J.B.A. L. Lindeman, and H.A. (Erik) Proper (2006): "Capturing Modeling Processes - Towards the MoDial Modeling Laboratory". In: R. Meersman, Z. Tari, and P. Herrero: proceedings of OTM Workshops: ORM'06, pp1128-1137, Springer, 2006.
17. Luna-Reyes, L. F., Andersen, D. F., Richardson, G. P., Pardo, T. A., and Cresswell, A. M. (2007): "Emergence of the governance structure for information integration across governmental agencies: a system dynamics approach". In: Proceedings of the 8th Annual international Conference on Digital Government Research: Bridging Disciplines & Domains (Philadelphia, Pennsylvania, May 20 - 23, 2007). ACM International Conference Proceeding Series, vol. 228. Digital Government Research Center, 47-56.
18. Williams, D. (2004): "Leveraging the Benefits Of System Dynamics To Deliver Requirements Projects Faster, Better And Cheaper". In: *Proceedings of the 2nd International Systems Engineering Research Conference*, Los Angels, USA 14 15 April 2004, International Council on Systems Engineering (INCOSE).
19. de Moor, A. (1999): *Empowering Communities: A Method for the Legitimate User-driven Specification of Network Information Systems.* PhD Thesis, University of Tilburg.
20. Nabukenya, J., P. van Bommel, and H.A. Proper (2007): "Collaborative IT Policy-making as a means of achieving Business-IT Alignment". In: B. Pernici and J.A. Gulla, editors, *Proceedings of the Workshop on Business/IT Alignment and Interoperability*(BUSITAL'07), held in conjunction with the 19th Conference on Advanced Information Systems (CAiSE'07), Trondheim, Norway, pages 461-468. Tapir Academic Press, Trondheim, Norway, 2007.
21. Hoppenbrouwers, S.J.B.A., H.A. Proper, and Th.P. van der Weide (2005): Formal Modelling as a Grounded Conversation. In G. Goldkuhl, M. Lind, and S. Haraldson, editors, Proceedings of the 8th International Working Conference on t he Language Action Perspective on Communication Modelling (LAP'05), pages 139-155, Kiruna, Sweden, EU, June 2005. Linkpings Universitet and Hogskolan I Boras, Linkping, Sweden, EU.
22. 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. Springer Verlag.
23. Hoppenbrouwers, S.J.B.A., and H.A. Proper (2004): "A Communicative Perspective on Second Order Information Systems". In G.E. Lasker, editor, *Proceedings of the 16th International Conference on System Research, Informatics and Cybernetics*, Baden-Baden, Germany. IIAS, 2004.
24. Järvinen, A. (2007): *Games without Frontiers, Theories and Methods for Game Studies and Design.* PhD Thesis, University of Tampere, Finland.
25. Salen, K. and Zimmerman, E. (2004): *Rules of Play, Game Design Fundamentals.* Cambridge, MA: MIT Press.