Playing ArchiMate Models

Jos Groenewegen[1], Stijn Hoppenbrouwers[1], and Erik Proper[1,2]

[1] Radboud University Nijmegen, Toernooiveld 135, Nijmegen, the Netherlands
[2] Public Research Centre Henri Tudor, 29 avenue John F. Kennedy
Luxembourg-Kirchberg, Luxembourg

jos.groenewegen@gmail.com, stijnh@cs.ru.nl, e.proper@acm.org

Abstract. This paper concerns the application of a gaming approach to the validation of ArchiMate models, with the aim of enhancing validation, by non-architects, beyond mere reading of the model. The game offers a guided process for systematic exploration of ArchiMate models, and for systematically raising questions about them. The development process and the design principles behind the game are discussed, as well as the information transformation involved in creating a model-specific game from an ArchiMate model. The game has been evaluated through application in a small real life case. We discuss the influence of our approach to model understanding by the players, and the conceptual merits and flaws of the game.

Keywords: enterprise architecture, architecture models, validation, games

1 Introduction: Games for Enterprise Architecture?

This paper addresses the problem of validating the completeness and correctness of architecture models. It aims to do this by providing a proof of concept of a game (or 'game-like procedure') that helps validate architectures. We show that a game-based approach to architecture model validation has a fair amount of merit in helping to create a good understanding of an architecture model, a basic requirement for validation.

The concept of 'architecture' is broad and various definitions exist. We refrain from further discussion of the term here. ArchiMate models [1] are now the chosen architecture representation standard of The Open Group [2]; we assume this is enough reason to take them as a valid subject of our game. However, we do believe our approach could in principle be generalized to other types of (architecture) models. Exploration of such extension, however, is not within the scope of this paper.

Basic issues underlying our effort are what a game is, and what our specific requirements for a model validation game are. Definitions of "game" are as varied as the games that are out there [3]; as is the case with "architecture", no undisputable definition of "game" exists. Some literature is available on what makes games work [3,4,5,6]. An essential part of games, being interactive systems, is that there has to be some kind of interaction between at least one actor and the game; also, in multi player games, there is interaction between actors (players).

Games have a set of rules within boundaries of which the actors can operate and interact. These interactions should lead to fulfilment of some clear end goal. This can be both a goal in the game or, in the case of 'serious' games, a goal outside of the game (in our case, validation of a model: a 'utility goal' [7]).

In this paper, we follow [6, p15]: *"A game is a system in which players voluntarily engage in a goal-oriented, artificial conflict, that results in a quantifiable outcome. The activity takes the form of a process which is defined by rules, yet offers freedom of action."* This is a very workable definition in our case. Even validating an architecture model is a form of artificial conflict. The players need to overcome an artificial challenge: to provide structured information concerning the validity of the model, thereby increasing the chance that the model is actually valid. Within this scope a game can be developed that suits our utility goal.

Our game design is restricted by only a fairly rough and limited working set of requirements: besides aiding the validation of ArchiMate models, the game has to be learnable relatively quickly. Furthermore, the pre-existing knowledge required for playing the game needs to be fairly minimal.

2 Method

The main aim of our small project was showing the possible strength of a game based approach to validating ArchiMate models. We set out to create a playable game. The general methodological frame was design science [8]. We developed a prototype, tried it, and evaluated it, improving it as we went along. Fortunately, only a few cycles were required to create the final version presented here.

As discussed in [9], there is no existing set of comparable games. Thus, not only an objective measure of the game's quality was outside our reach, but so was even a simple comparison with another game. At the same time there is no good measure for the quality of architecture models [10,1]. Some work has been done on quality of modelling [11,12], but it cannot be readily applied to a game approach. The best we could do to show the game's value was to try it out and report on our experiences (and those of the players).

However, this does not mean we designed the game without basing it on some ideas and principles from the literature. Indeed, the first step was a literature study, which was undertaken in co-operation with the Netherlands Architecture Forum (NAF) Workgroup on Games and Architecture. [3] and [9] were taken as broad guidelines. After the literature study on existing games, the constructive part of the project began. The process of game creation was based on known methods for game construction [3,4,5], inevitably along with a substantial dose of creativity.

Testing a game is not trivial [3], especially in a field such as architecture. To achieve proof of concept we first applied the game in a "dry run" on the Archisurance case [13] to see if it fitted within the conceptual framework underlying ArchiMate. Next, we applied it in a small real life case to observe the effectiveness of the game. Interviews with game participants were held before and after the sessions to find out about the players' understanding of the architectural model and the correctness of the model. Although this by no means guarantees a perfect game it is a reasonable measure of the merit of the game concept, which enables us to find out if it warrants further research (in view of [14,15,16,17]).

3 Architecture Game Types

Before focussing on the design of a specific game, we addressed the general question of what types of architectural game could be usefully distinguished in the first place, based on their utility. This work was done in collaboration with a group of professional architects (plus one game design professional) at the NAF workgroup mentioned earlier. The categorization had to cover the full scope of how games could possibly be applied.

We identified the following four possible categories for games-for-architecting:

1. Convincing people of the added value of the concept of architecture through a game. An existing game in this vein would be the Ordina Alignment Game [14]. A simplified game simulation could work here, mirroring the problems architecture is supposed to remedy, and creating an experience as to how the remedy works.
2. Creating architecture: a game, or a set of integrated games, that support the creation of (representations of) architecture. No such games are known to us, nor is it clear what such a game might look like.
3. Analyzing and validating architecture representations. Again, no examples or existing ideas about how to go about creating such a game were found.
4. Creating awareness of a completed architecture among the stakeholders. Some examples are known [9]; a simulation game might also work here.
5. The categories have proven robust and seem to cover all relevant situations in the field.

As mentioned, some games of type 1 and 4 already exist, but not so for 2 and 3. Our research interest was thus more keenly raised by the latter types. The second category of game is rather ambitious and would need to support complex structured conceptualization. Though similar games have been considered for some other forms of modelling [18], for architecture we felt it would overstretch our current capacities. The third category of games, however, seemed promising for an initial attempt. It depends on an architecture (or architecture model) already being available, so the questions raised 'only' involve whether the architecture is a good one. The process to be followed can be quite different than one for the creation of an architecture. When an architecture is created one has to go through an elaborate creative process and generally come up with, and reach consensus on, 'new' knowledge. Evaluating a representation that has already been made seems much less taxing for what is, after all, only an initial attempt. Based on these considerations the choice was made to develop a game of the third type.

4 The Information Covered by ArchiMate Models

To find what information is typically expressed through ArchiMate models, we started by studying the ideas behind the ArchiMate language, as discussed in [13]. Several strict separations within the ArchiMate language are indicated. The first split is between the layers distinguished in the enterprise: the business layer, the application layer, and the technology layer. The three layers all reflect dynamic systems, that can be framed in one meta-model. For more information on this meta-model, see [13], fig 7.

However, more information underlies ArchiMate models than can be captured by means of its meta-model. This concerns 'the stories behind the actors and objects'. These stories provide the reasons why, for example, actors are placed in the model. The idea behind ArchiMate models is that every concept should clearly contribute conceptually, both in the modelling language and in its application in an actual architecture model [10].

For dynamic systems, ArchiMate further splits the components into three categories: the active structure concepts, the behavioral concepts, and the passive structure concepts. A precise definition is the following, taken from [13]:

> *"Active structure concepts are concepts concerned with the execution of behaviour; e.g. (human) actors, software applications or devices that display actual behaviour. The behavioural concepts represent the actual behaviour, i.e. the processes and activities that are performed. The active structure concepts can be assigned to behavioural concepts, to show who (or what) performs the behaviour. The passive structure concepts are the concepts upon which behaviour is performed."*[13, page 9]

In other words: *who* does things, *what* do they do, and *on what* do they do them [10,13]. A clear relation exists with basic semantic roles in natural language sentences (subject, predicate, object).

A further split in ArchiMate concerns the three kinds of components on the one hand (active structure, behaviour, passive structure), and the *relations* between these components on the other.

We observe that what is missing is some *context-related information*. As mentioned, the split into three components is based on the structure of natural language and the way we reason about active processes [13]. As discussed in [15,17,16], more information is needed for this. The 'who'-'what'-'on-what' split that ArchiMate makes is often seen in representations of action. However, to have a full understanding of action, a fourth parameter has to be added: the *why* of the action. There is some reason behind the action being undertaken; no action is taken for sake of itself. If there is a reason for existence of every part (concept) in an architectural model, then the architect should have thought about this at least in principle. And yet, such information is not captured through the meta-model. Thus, the validation of ArchiMate models should benefit at a fairly fundamental level from taking such information into account.

As a foundation of the validation game we will use the three types of component: active structure elements, behavioural elements and passive elements. Because they are also basic concepts in natural language, humans can understand these fairly well [13,15,16,17]. For full understanding, however, we will need a fourth type of component: the reasons underlying action/behaviour as modelled. We work under the assumption that the model (explicit knowledge) and the architect's tacit knowledge about the choices made are somehow available to players of the game, if only after questioning. With this knowledge we seem to have everything we need to validate both the architecture model and (some of the) rationales behind it [2].

5 A Game for Architecture Model Validation

In this section, which is the core of the paper, we describe the game design and its underlying principles.

5.1 Aim of the game

It is clear that a formal validation of an ArchiMate model should not be the aim. We are primarily concerned with a qualitative, human-knowledge-based type of validation, with a certain degree of subjectivity. This is inherent in all stakeholder-based validation. The second aim of the game is accessibility. The whole idea of a game to aid validation is that it does not typically involve the creators of the model (architects), but stakeholders with no special modelling skills.

Please note that the game is merely a proof of concept. It does not need to be optimized in terms of effort needed to prepare, or actual execution of the game. It has to be good enough to show that the approach is viable.

5.2 Basic design choices

The first design choice is that we want to make sure that all relevant knowledge is available through the players involved. No constraints are put on who can play, as long as all knowledge required is covered within the set of players.

The second choice is that we will base each instance of the game on a specific, existing ArchiMate model. This means the relation between the model and the game should be clear at all times. Whatever final form the game takes, we should be able to trace it back to the original ArchiMate model in such a way that the model can directly benefit from the game.

Next, the reasons behind the interactions between the elements of the ArchiMate model have to be incorporated (as explained in section 4).

5.3 The game

Now that the core principles of the design are clear we arrive at a major creative step. As discussed, a main lack of knowledge required for validation lies in the implicitness of knowledge of why the architect puts forward “actors acting on elements”. Such knowledge is largely tacit, process-related, and hard to make explicit. It involves an intuition (of the people doing the work) of how things work and how the process flows, which the architect has to distil a model of [19]. The first step in the creative process was the realization that dealing with intuition is a major problem in traditional model validation, which could be better tackled through a game approach allowing for a not too strictly regimented process. This should lead to better understanding of the architect’s intent and rationales.

Active components display behaviour and they do this on passive structure elements. The possible behaviour that is displayed, and by whom and on whom, is all captured in the architecture model. If we create the possibility of “walking through” the architecture in some sort of logical order, a client can become an actor on another actor, which implies a temporal dependency.

From here we went to the insight that every interaction captured in an architecture model entails a similar dependency: an actor acts in a certain way on a client, for a reason. In ArchiMate, such interactions are always depicted by means of minimally two elements. In other words, they are never ‘hidden’ in one element. Because of this it is possible to see every action taken, and in this way step through the model. Thus, one can incorporate a temporal aspect into the relations.

Every active structure element has a starting point in the architecture, a ‘path of influence’ through the architecture, and an ending point where the original actor ends up. The path the actor takes through the architecture, and the actions it takes, can thus be seen, followed, and understood. This concept lies at the core of the game. The players need to be forced to think actively about the behaviour the actors display. The chosen way to do this is to have the players play the actors associated with their specific domain expertise. Every actor has a starting point and an end point in the architecture (not necessarily different ones), and paths may be shared. The players’ choices at every possible step represent the behaviour of the actor, and should coincide with the behaviour the architect had in mind when creating the model.

In the current context, we assume that if an architect has a correct idea of the actual situation in the domain, the ensuing model is produced correctly. In other words, we assume (naively) that architects are fluent and flawless in ArchiMate and are also perfect in performing the preparation. Admittedly, the architect may influence the outcome of the game though the way the game is set up by him. Also, one may wonder whether third-party validation, divorcing the architect form the process, would be preferable. However, cutting out the architect form the validation process would require for someone else to do the game preparation. Since this requires in-depth knowledge ‘behind the model’, and since automated game generation is not an option at this stage, we cannot but acknowledge these points but have to simply assume they do not arise.

It is time to take a look at what all this leads to. After presenting the game rules we will discuss how an ArchiMate model can be transformed into a playable game.

5.4 Basic (generic) game instructions

We will now discuss how the game is set up in general, excluding mode-specific aspects. For a discussion of the preparations for this phase see 5.5.

Setting up the game board

Place different coloured pawns (or marked pawns) at each starting location marked by the game master (typically, the Architect). Each marked start location has one or more designated end locations. The map is laid out, meaning the assignment points are laid down as well as the routes between them (which are still locked).

Start

Each player moves in turn, starting with the youngest player. A player goes through the following steps (some of which are not always performed, depending on the situation):

1. Move a pawn to any 'reachable' assignment, and try to complete the assignment.
2. Pick up available items on routes or in squares, to assist with or as required for assignments.
3. Trade items with another pawn it can reach.
4. Drop available items on a tile or on a route it can reach, for other pawns to pick up.

A player can go through steps 2 to 4 as often as he likes. Step 1 may be performed at most 3 times in a row before another player gets a turn (who might skip it if he wants).

Game goal

Pawns start at their designated start location and can move across any explored path as far as they want. They have one or more designated end points. When an unexplored path is entered, the pawn has to 'open' the associated card (provided by the game master) and complete the assignment in order to continue. If the pawn withdraws after reading the assignment, the path stays closed. Opened paths can generate passive structure elements in their own or other tiles (as described on the card). Once a pawn reaches an end point, then its route (every step in which it undertook action) is 'marked' and a new pawn is generated at the start location.

The game ends when a line can be drawn, across marked squares and routes, between all end locations.

5.5 Preparations by the Game Master

The basic instructions are simple but the preparation for an actual game is more complex. Within the model to be evaluated, all active structure elements need to be marked. Any active structure element that comes from an earlier 'interaction', or involves an actor of an earlier interaction, is marked as an assignment point. If there are no further routes to take for an actor from an assignment point, it is also marked as an end location. Any active structure element with no previous location is marked as a starting location. The passive structure elements are incorporated into the game as items that can be carried along. The behavioural elements are the choices that force people down a route. Furthermore, they can be 'nodes' where several routes cross. A node can be the creator of passive structure elements, an assignment, or merely a crossroad.

To help clarify this, we provide an example game transformation for the archisurance example. The archisurance model shown in fig. 1 is taken from [2].

Within the model we first need to mark the components. Thin squares lines designate active structure elements. A thicker line marks start or end points, depending on ingoing and outgoing lines. Circles mean behavioural elements. A cross is a passive structure element. Some liberty has been taken with respect to the intent and meaning behind this model, for explanatory reasons. Thus we arrive at the marked model shown in fig 2.

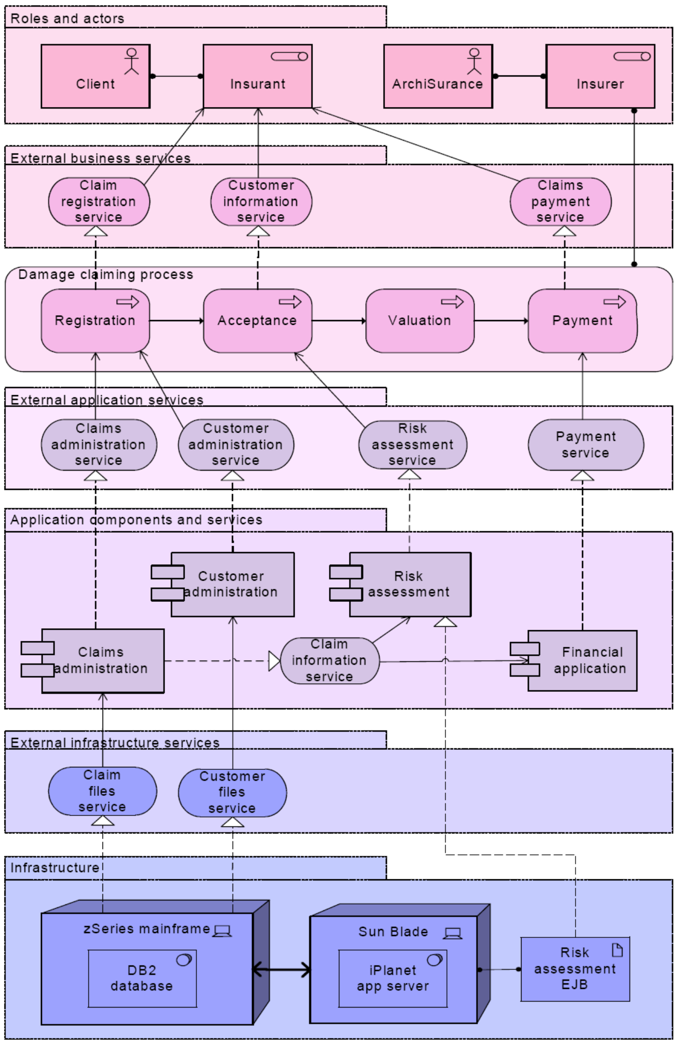

Fig. 1: a targeted ArchiMate model

For the sake of workability in this example we will not add all the in-depth options further expanding the model. The marked model given is still incomplete. Where is the architect's tacit knowledge about the process flow?

The preparations to make this explicit can be quite extensive as the implicit knowledge is written down in game form through requirements for opening up nodes and possible transformations. Let us see what it would look like when filled in for the given archisurance example. First we will give the options at each node or point of behaviour. Then we will make explicit demands for any routes that exist.

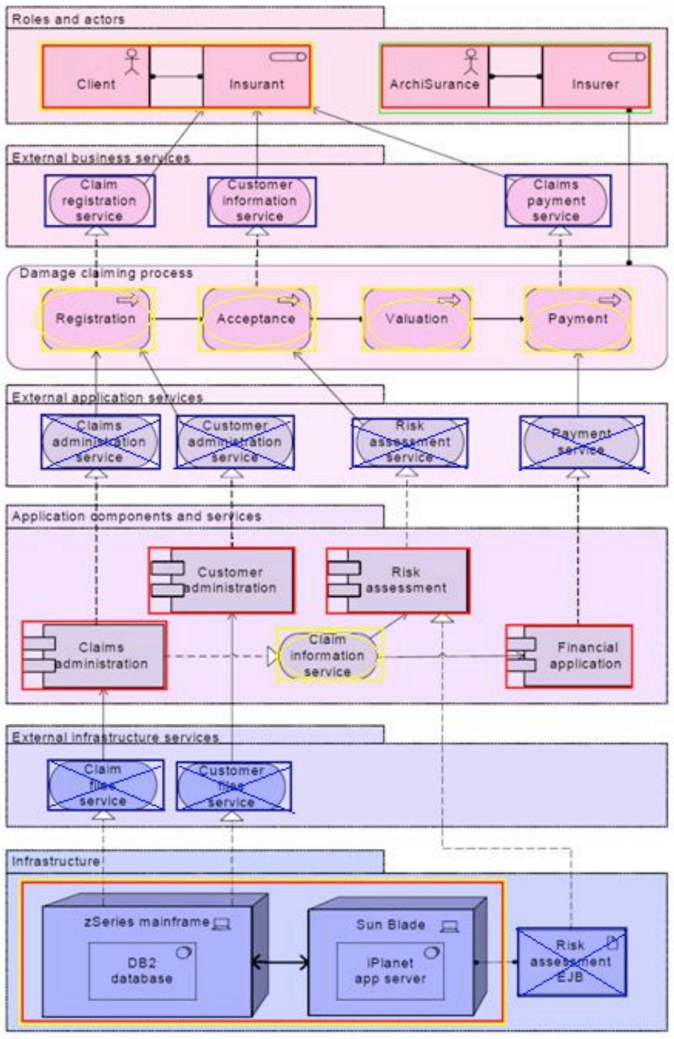

Fig. 2: an ArchiMate model marked as a "game board"

1. Client/Insurant
2. Registration a. With claim registration service and claims administration service and customer administration service create filled claim registration service
3. Acceptance a. With customer information service and risk assessment service create filled customer information service
4. Valuation a. Create cost item
5. Payment a. With cost item, payment service and claims payment service create filled claims payment service
6. Insurer a. Be end point
7. Claims administration a. With filled claim registration service create fulfilled claims administration service
8. Customer administration a. With filled claim registration service create fulfilled customer administration service
9. Risk assessment a. With filled customer information service and risk assessment service and risk assessment EJB create fulfilled risk assessment service
10. Claim information service a. Choose to go to risk assessment or financial applications
11. Financial administration a. With payment service and filled claims payment service create fulfilled payment service
12. DB2 Database

Route demands (if none stated none exist):

1. Registration -> acceptance requires filled claim registration service
2. Acceptance -> valuation requires filled customer information service
3. Valuation -> payment requires cost item
4. Payment -> insurer requires filled claims payment service && if player = brown -> fulfilled payment service
5. Claims administration -> claims information service requires fulfilled claims administration service
6. DB2 database -> Risk assessment requires claims administration -> claims information service = open

The somewhat intimidating amount of information here reflects the amount of information contained in architecture models. The important advantage of our approach is that this information is given to players in an explicit format in the form of cards and clear goals. The abstract layers and width of the model are replaced by understandable goals and paths.

6 Evaluation

A ‘real-life’ example model was played and evaluated using an architecture model of the small organisation the players were part of.

6.1 Main Findings

What went well?

The preparations were found to be time consuming, but no inherent difficulties were encountered going through this step. As the game itself unfolded, results were promising. Through the game, the staff members participating indeed developed a better understanding / feel for the model than they had with the original model, and the sense of model complexity was reduced.

Points of improvement

When players were found to miss out on information from the model it was difficult to note the exact knowledge gap. Although the area of missing knowledge was easy to find, the precise knowledge gap could not be identified. In other words: seeing that a model is not valid is easy, but pinpointing why the model is not valid is more difficult.

Lessons learned

Not unexpectedly, we found that it is very important to ensure the knowledge of the players fits the domain they are playing. If knowledge is missing flaws in the model can be missed.

A second lesson learned concerns the ‘after-game’ phase. To maximize understanding of what the players noted, and what they went through, debriefing the players after the game is important. Although doubts about the model should in fact be expressed during the game, people tend not to do so, especially in a social context with colleagues present. Furthermore, debriefing helps consolidate the concepts of the architecture, and the understanding a player has of her own work processes.

6.2 Model understanding by the players

Through playing the game the players achieved a considerably better understanding of the contents of the model than by merely looking at the model. They were far more capable to see the relations between elements after they actually experienced them. Furthermore, due to the explicit and mandatory constraints between moves (see “game preparations”), the actual relations fitted better with their expectations and their experience of them in reality. However, no new knowledge of the actual organisation was attained, while some organisational knowledge was felt to be missing for certain people. Post-game evaluation remedied this, as seven interviews before and after the game were included in the total setup.

6.3 Investments required

The initial investment to get the architecture into a ‘game’ mode is considerable. It requires making tacit knowledge explicit and writing down what the ideas behind all parts of the architecture are. However, it does not seem unreasonable that this should in some way be part of routine architecture documentation. We estimate that the transformation takes roughly same amount of time as the creation / writing down of the architectural model (assuming the knowledge required is available).

The investment of playing the actual game is estimated to range up from 30 minutes for small models to several hours for larger models, especially for those that require a significant increase in player numbers to have all the necessary knowledge available in the game.

The time needed to process the results varies depending on the results. If no flaws are found the results can be processed in several minutes, plus evaluation time with players. However, if flaws are found in the model it can take a significant time investment to pinpoint the exact flaw and correct it.

7 Conclusion

Let us look back at our original goal: “To provide a proof of concept for a game-based approach to validating ArchiMate models.” Indeed, a game-based approach to validating ArchiMate models can work. It does not give 100% certainty the model is valid, but it seems to provide real added value. It introduces a new and different, systematic way of validating a model. It has different requirements from most methods, most notably the need to make explicit tacit knowledge in a multi-dimensional space, based on location, internal relations, and time. Preparing the game could in fact be seen as an additional highly useful activity in model validation. The game may well open up model validation to people who before could not be involved. So, the approach has merit. Further research will be necessary to show the extent of this merit in more complex cases, and more extensive real-life situations. In addition, some further study concerning the detailed interactions taking place as the game proceeds could render good insights as well as input for improvement of the game’s playability [7].

References

[1] Lankhorst, M.M. (edt.): Enterprise Architecture at Work: Modelling, Communication and Analysis. Berlin: Springer. 2005.
[2] Iacob, M.-E. and Jonkers, H. and Lankhorst, M.M. and Proper, H.A.: ArchiMate 1.0 Specification. The Open Group, 2009
[3] Léon de Caluwé, Gert Jan Hofstede, Vincent Peters: Why do Games Work?. Kluwer, Deventer (2008)

[4] R. Hunicke, M. LeBlanc, and R. Zubek: MDA: A Formal Approach to Game Design and Game Research. In: Proceedings of the AAAI-04 Workshop on Challenges in Game AI, pp. 1–5. (2004)

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

[6] Ilona Wilmont.: A gaming approach to collaborative modelling. Master Thesis in Information Science, Radboud Universiteit Nijmegen, Thesis Number 91-IK (2009)

[7] S.J.B.A. Hoppenbrouwers, H. Weigand, and E.A.J.A. Rouwette: Setting Rules of Play for Collaborative Modelling. In: P. Rittgen, edt., International Journal of e-Collaboration 5 (4), p37-52. Special Issue on Collaborative Business Information System Development. 2009.. IGI Publishing, USA.

[8] Hevner, A. R., March, S. T., Park, J. & Ram, S.: Design Science in Information Systems Research. In: MIS Quarterly, vol 28, pp. 75-106 (2004)

[9] Herman van der Bij.: Lexicon Architectuurspellen, Landelijk Architectuur Congres 2009. 27 november (2009); in Dutch.

[10] Proper, H., Verrijn–Stuart, A., Hoppenbrouwers, S.: Towards Utility–based Selection of Architecture–Modelling Concepts. In: Hartmann, S., Stumptner, M., eds.: Proceedings of the Second Asia–Pacific Conference on Conceptual Modelling (APCCM2005). Volume 42 of Conferences in Research and Practice in Information Technology Series. Sydney, New South Wales, Australia, Australian Computer Society 25–36 (2005)

[11] P. van Bommel, S.J.B.A. (Stijn) Hoppenbrouwers, H.A. (Erik) Proper, and J. (Jeroen) Roelofs: Concepts and Strategies for Quality of Modelling. In: T. Halpin, J. Krogstie, and E. Proper: Innovations in Information Systems Modelling, Methods and Best Practices; Chapter IX, p167-89. Advances in Database Research series, IGI Global Publishing, USA. (2008)

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

[13] Lankhorst, M., Proper, H., and Jonkers, H. .: The Architecture of the ArchiMate Language. Enterprise. In: Proceedings of the EMMSAD 2009, held at CAiSE 2009, Amsterdam, the Netherlands. p367. LNBIP vol 29. Berlin: Springer.

[14] Ordina-allignment game http://www.ordina.nl/Downloadcentrum/~/media/Files/Onze%20dienstverlening/Consulting/The%20Alignment%20Game.ashx?forcedownload=1

[15] Taylor, J.: The "rational" organization reconsidered: An explanation of some of the organizational implications of self-organizing. In: *Communication Theory*, 11(2), 137–177 (2001)

[16] Drew, P. & Heritage, J.: *Talk at work*. Cambridge University Press. (1992)

[17] Garcia-Lorenzo, L, Nolas, S-M & de Zeeuw, G.: ‘Telling stories and the practice of collaboration’. In: *International Journal of Sociology and Social Policy,* vol. 28, nos. 1–2, pp. 9-19 (2008)

[18] S.J.B.A Hoppenbrouwers and B. Schotten: A Game Prototype for Basic Process Model Elicitation. In: Anne Persson, Janis Stirna (eds): The Practice of Enterprise Modeling, 2nd IFIP WG8.1 Working Conference, PoEM 2009. Springer, LNBIP series vol. 39

[19] John A. Zachman.: A framework for information systems architecture. In: IBM Systems Journal, v.26 n.3, p.276-292, (1987)