A Generic Architecture for Quickly-Deployable, Flexible, Scenario-Oriented Serious Games

Jan Willem Veeningen, Nick B. Szirbik, Marco P. Blokhuis

A Generic Architecture for Quickly-deployable, Flexible, Scenario-oriented Serious Games

J.W. Veeningen, N.B. Szirbik, and M.P. Blokhuis

University of Groningen, the Netherlands
{j.w.veeningen@student.rug.nl;n.b.szirbik@rug.nl;m.p.blokhuis@student.rug.nl}

Abstract. Serious gaming can be used in system engineering design processes, in the pre-inception phase, where investment scenarios are explored in game sessions with stakeholders. However, scant knowledge exist in the literature about how to design this kind of games. Based on the experience of three consecutive phases of game development for infrastructure projects in biogas, local wind energy and LNG refuelling infrastructures, a generic architecture was developed and made explicit as design knowledge. This paper outlines the architecture, offers insights about its application, and explains how the validation of this knowledge takes place.

Keywords: serious gaming, scenario discovery, generic design knowledge

1 Introduction

Serious games are typically used for educational and training purposes. Some games help various stakeholders (owners, users, customers) to understand better the working of complex systems. However, there are scant research results for using serious gaming as a tool to discover potential development scenarios in the design of complex scalable systems. The generic architecture presented in this paper is potentially useful knowledge for stakeholders who are willing use games in the design and development of complex systems. Such a kind of games has shown the potential to discover unforeseen scenarios based on the decisions that the players made in the game.

The architecture proposes a board game, partially supported by software - to allow players (namely the stakeholders) make calculations and decisions, and to allow the game master to give players an idea about the quantitative aspects on the board. The software's role is also to monitor and record the the decisions of the players. The board allows stakeholders to have an intuitive visual feedback on the status of the game, which changes over time as the players make decisions, mimicking the growth of their target system. The decisions made in the game entail investments into physical elements of the system to be developed. On the board, physical playing pieces are placed to represent these investments, recorded by the software.

Results with games of this kind show that solutions not thought of before the game sessions started can be found by stakeholders - providing them with a stronger incentive to commit for development of the system to be. For example, in a setting of an energy infrastructure where producers and consumers of energy were operating individually, they did not realize that they can work together to gain a more optimal energy price for both parties. Through serious game sessions, those stakeholders can come together and play the game to discover what they can accomplish when they cooperate and understand each other better. Similar early findings were presented in the work of Tan [4], which investigates multi-stakeholder games used for urban development.

The paper is structured as follows. In section 2 the research goal and the methodology used are described. Next, in chapter 3 the generic architecture is presented and what is the novelty about this architecture is outlined. Section 4 shows how the architecture can be applied for new contexts and validated further. Section 5 concludes the paper and presents shortly the immediate next steps for future research.

2 The Research Goal and the Methodology

There is gap in the literature about serious games - that is, there are not yet any development frameworks for games that can be deployed very quickly. Moreover, the games are not flexible and adaptable from one gaming session to the other (or even within the same session). Finally, there is no knowledge to develop games that are scenario-discovery oriented. By studying the development of specific games for scenario discovery, the authors (who were sometimes involved directly in the development) gathered knowledge with the intention to build a framework (a generic architecture with its guidelines for application) for future game development that has as a main purpose the scenario discovery.

As a consequence of this goal, the methodology chosen in this research is design-science research [1]. This research method indicates that an artifact needed in practice is designed within the process, it is validated (in this case by successfully using the game), and generalizable contributions are made for future designs. Here, the artifacts are a series of serious games where consumers of energy, producers of energy and investors can together discover future potential scenarios for their energy infrastructure (e.g. biogas grids, locally distributed win-produced electricity, or an LNG-based infrastructure for road freight). This artifact is specifically designed to solve the real problem of mismatch between goals and perspectives on doing investments in energy infrastructure.

The extant design knowledge used to design the initial versions of the aforementioned games is mostly taken from the game design professional literature [2]. Next to this body of knowledge, the research framework (FASD) proposed by Velthuizen [3] after the development of the first game (named GasBoard) was used to position the next games and evaluate their design process. In the end, after developing three playable games, in order to contribute to literature, it has been decided that a generic architecture (accompanied by guidelines of how to use it) should be extracted from the lessons learned during each iteration of game design, development testing, and game validation.

3 The generic structure and process flow of scenario-centric games

The novelty of these games is that they are explicitly deployed to explore investment scenarios in a complex socio-technical context of an early system development. The specific games developed over the years were related to environments where energy producers and consumers were solely dependent on a large monopolistic grid operator (for gas, or electricity, or LNG imports). The serious games let the players (which were playing the role of the stakeholders in these systems) to engage themselves with investment decisions in alternative and local energy infrastructures. They can make investment decisions and long term commitments, erase the slate, start again, discover alternatives for investment and commitments, negotiate, argue, and try to find win-win situation in the longer term. At the end of multiple sessions of game playing, the experience can offer insights to the players (i.e. the stakeholders) how to develop a decentralized energy infrastructure that is more profitable and efficient from their local points of view. As an outcome of the game, the investment decisions and commitments that the players make can be transformed into potential future scenarios for infrastructure development. These future scenarios help the design and development of the complex system envisaged (in these particular situations, the energy infrastructure, but it can be any kind of strategic infrastructure where the stakeholders are the local communities, the local economy, the local authorities, and potential investors and system developers).

Irrespective of the nature of the system to be, over the iterations made to develop these games it has been observed repeatedly that a game of this kind needs always to consist of the following **structural** elements:

1. Game Roles: These are place-holders in the game for the participants who play the game. For example, such a role can be "Investor", and to each role, we can assign a set of possible actions (e.g. "invest" for the Investor role). Some roles can be easily identified by matching the roles of the stakeholders of the system, but other roles (like Investor) can be added. One player can play multiple roles, and in a more advanced setting some roles (those can have simple and predictable behaviour, easy to capture in mathematical models) can be played by soft-bots specially implemented for the task.
2. Game Components: these are mimicking the physical components of the system. For example, in a game that explores the scaling up of a LNG infrastructure, the components could be: refuelling stations, pipes, storage facilities, supply means, etc. The game starts typically with a few elements, and the number of these is increased during the game. A price is attached to each kind of components.

3. Game Board: the components appear in a "geography" context". To mimic this, a map-like support will be used to place the elements on specific locations.
4. Game Rounds: to mimic the passage of time, the game playing will be separated in rounds, each consisting of two parts: first, a mimicking of accelerated passage of time (for a given period like 1 month or 1 year, depending on the nature of the system to be); second, a period of discussion and negotiations between the players, and decision making.
5. Game Rules: because the players can interact informally during the rounds, it is necessary to establish rules for their interaction that constrain what they can do, and what they can produce.
6. Game outcomes: these can be established formally at the beginning, but such a game should be able to discover outcomes (like a new kind of contract or agreement) during a game playing.
7. Scenario: the (time ordered) set of outcomes
8. Means to record the state: besides the elements placed on the map, there should be some sort of recording the quantitative nature of states and outcomes (e.g. how much was invested at a given moment in elements).
9. Means to record the scenario: a more detailed recording mechanism for the outcomes.
10. A realism checking mechanism: an algorithm or set of algorithms that can infer the change of the state (physical and financial attributes) of the system, showing how the system grows from one round to the other.
11. Game Observers: non-player humans that have role to identify the growing scenario (this is not a task of the players normally, the system developers fill this non-player role).
12. Game master: a human who applies the rules, directs the rounds, mediates, and helps the players during a game.

Another part of the architecture is represented by the description of the **process**-related elements (activities) of playing the game. It has been repeatedly observed that the flow of activities is always similar to the following sequence of steps:

1. The players are instructed in the game playing and they are choosing their roles.
2. The game starts with a given situation of the board (some elements are set up mimicking the small-scale state of the system).
3. A round starts: first, the players must interact: discuss, negotiate, and make decisions (like contractual agreements, investments, etc. according with the rules). These are observed and recorded.
4. The rounds will end by applying the decisions on the board and in the realism checking mechanism, making an accelerated "jum" in time. The state of the system (in scale and output) changes accordingly. This step can be considered also the first part of the next round.
5. The players are informed about the new status, and a new round of negotiations starts from step 3.

6. The game master decides when to stop the game, after several rounds most probably when a scenario has been found, and/or the system to be mimicked in the game is reaching a sustainable magnitude.
7. Post-mortem analysis: the scenario is discussed and refined, and agreements for an eventual new game session are made.

The game master is in the control of the game, its software, and it is responsible to inform the players about the status and evolution of the game from round to round. He is checking the time, forcing rounds to end, and players should ask the game master permission for actions that are not specified a priori, and he should be able to detect easily illegal actions (which violate the rules).

Besides and after the post-mortem analysis (which is centred on the scenario), at the very end of each gaming session, a formal discussion takes place on how players experienced the game and what they thought was positive and what could be improved in the game itself. The observers can record and analyse the results of these discussions, and for the next gaming session new roles, game components, rules, or outcomes can be implemented in a new version of the game - which can involve also changes in the realism checking mechanism and the software tool. It is important for a game development team to constantly have in mind that playing the game multiple times is mainly the way to gather valuable feedback about how the game could be refined and improved. For example, in a situation when the game was played successively, and the game itself evolved, the following feedback was recurring:

– About inter-player communication: communication about the deals were not supposed to be intercepted by other players, as this might contain valuable information. Therefore, it is wise change the layout of the game space, in order to allow pairs of players to negotiate "securely", in spaces (like separate cubicles or even rooms).
– About the tools used during the game: the tools were initially to difficult to use or did not allow for all the moves that were allowed by the existing rules of the game. A simple solution is to have a training round in the beginning of the game.
– About the number of players: it was always argued during the first sessions that adding more players can increase the number of interesting scenarios. A simple solution is to start with the highest number of players possible.
– About player roles and rules: some roles appeared to expand during the games, the investors for the example, in the initial rounds were only allowed to finance investments, but soon turned out to become asset owners and operators of elements in the infrastructure.

4 How to use the generic architecture and validate it further via new games

The most important insight related to this kind of games is that they should be focused on stakeholders' human behaviour. It was noted ([6] and [7]) that in the domain of networked infrastructures, the complex, multi-stakeholder, large-scale system design is perceived as a strategic process. However, this is not its main characteristic. It is neither predominantly financial, neither technological, albeit it has aspects related to these dimensions of analysis. In the end, what all complex system designs have in common is a dominant human dimension, given by the behaviour of various stakeholders, all competing to resolve their (sometimes opposite) goals, understand each other's different backgrounds, biases and prejudices. This is obvious for any large-scale and long term infrastructure project in energy, communication, transportation, defence, landscape transformation, or space exploration. Hence, the study of complex system design is also the study of stakeholders' social interaction and behaviours. All complex system designs (like the Delta project in The Netherlands for example) are part to particular contexts, to the societies and economies which source the resources to implement them, and are sometimes emotionally related to the technologies that are well-established and sustained in those societies (for example, currently nuclear technology is not supported by society in Sweden or Germany). A study of the process of complex system design, albeit it has to be technologically grounded, has to study the process on a behavioural, deeply human dimension. In such a project, each stakeholder tends to project its own beliefs about what the other stakeholders think, how they behave, and how they act toward a decision. Most of the time, exactly because of this projection tendency, the process to find a design and especially to find a plan to realize the design ends in failure.

4.1 Guidelines

In the following list of activities to implement a game based on the previously proposed architecture, the **fourth activity in the list is the most important** and should take most of the time of the game development and refinement. The proposed order of activities for implementations is:

1. Identify the physical elements of the game. Normally, these are the main physical components (modules, sub-systems) of the system to be. The elements are implemented as game pieces that can be placed on the board.
2. Identify the roles in the game. Roles may own or influence elements. For example, if similar elements are owned by two different players playing the same kind of role (e.g. an owner of an LNG refuelling station), different colors should be used for similar kinds of elements, to show different ownership (like in most of the board games).
3. Establish the board. Initially, it is advisable to start with a generic map, composed for example of a raster of equal hexagons. Physical attributes like distances can be inferred from the number of raster parts used for a certain placement of elements. Later, real life maps can be used, in conjunction with geographical elements (like rivers) and infrastructure (like cities and roads).
4. Establish a list of regulated interactions between the roles. For each kind of interaction, establish certain rules of conduct, and potential outcomes (like contracts, long term agreements, new regulations to be enacted, if the player is a regulating body).

5. Find ways to record the quantitative aspects of the state (magnitude) of the system to be at a given moment.
6. Find ways to implement the time advancing algorithms and the realism checking mechanism; initially these should not be too realistic, and should focus mostly on the financial aspects (revenue, losses and profit). Later, more physical aspects can be included in the realism enactment. It is recommended that each player should have a separate view about its own status, and everybody should have some sort of general idea of the evolution of the system to be - this sheet can be projected on a board visible for all players to see.
7. Find ways to capture the outcomes and the overall scenario that emerges.

The game should always exhibit the advancement of the game situation via the decision making of the players and not via stochastic mechanisms like dice or randomly extracted cards - for some degree of uncertainty, these can be used, but with parsimony. The outcome should be clearly a result of human interaction and not a mixture of chance and player skill like in entertainment games. There is no single winner in such games, but ultimately, what is sought after is a satisfying long term win-win outcome for all stakeholders.

4.2 Validation of the generic knowledge

In a design science approach, the validation of a generic architecture for a specific class of artifacts in is a gradual process. Basically, each successful implementation of an artifact in a given context that is using the knowledge embodied in the generic architecture is a validation step. More successful implementations in different contexts means a stronger degree of acceptance of the generic architecture and a growing perception in the professional community using it that this generic architecture is validated design knowledge.

In our years-long process of developing these games, the first game (GasBoard) was implemented by using extant knowledge about board games, and the game was gradually adapted to the requirements of the stakeholders from the biogas industry who were involved in the development project. When the game was mature, and validated itself as a successful artifact, the developers considered that the knowledge acquired about the development process of the game should be captured in some form. This is why the FASD [3] was created at that time. However, this did not represent an explicit form of a generic architecture, albeit the developers had at the time a clear implicit idea about such an architecture.

The next game (FromEnergy2Synergy, [8]), has used FASD as input, was successfully validated by playing it many times, and added to the design knowledge by adding guidelines for implementation and deployment of the game. However, only after the development of the third game (investments in the LNG-refueling infrastructure), the developers decided to explicitly build a generic architecture, which is to be validated separately. The first step towards validation was to enact a validation workshop [5], where the explicit architecture was discussed by the various developers of the three games. The net result was that the generic architecture was refined and better structured.

Finally, the generic architecture was recently (May-June 2017) used in an experiment with students, who developed games in a project. Three development teams were provided with this generic architecture (in the form of a manual) and three teams were not provided. Currently, the results of this experiment are analysed and will be published in a subsequent paper. Preliminary results show a clear advantage for the teams who used the generic architecture. For example, one of the teams, who was late in the development, managed to implement a playable, very immersive and engaging game in only three days. These developers, in a discussion after the last gaming session, considered the generic architecture document to be a crucial element in the speed of the development.

5 Conclusions

To our knowledge, this is the first attempt to build a generic architecture for scenario-centric games. These games have to be quickly enacted for the stakeholders in complex system projects. The game playing takes place very early in the development process (pre-inception), and the games have to be highly adaptable to changing requirements.

The next step is implement more of this kind of games that are using this generic architecture as design knowledge, and based on the success of these implementations, validate and refine further the architecture. The games can be either developed by students, for research projects, or by industrial partners interested in local infrastructural developments where "chicken and egg" situations appeared. These findings have to be reported and the generic architecture should be made available as open source for everyone who wants to develop such games. Any new successful implementations should be described along with the open source, as it validates further the architecture.

References

1. Hevner, A.R.: Design science in information systems research. MIS quarterly (2004)
2. Schell, J.: The art of game design. Morgan Kaufmann, San Francisco (2008)
3. Velthuizen, V.: Introducing FASD: A Functional Architecture for Serious game Development. BSc thesis, University of Groningen (2016)
4. Tan, E.: Negotiation and Design for the Self-Organizing City: Gaming as a Method for Urban Design. PhD thesis, Technical University Delft (2014)
5. De Freitas, S.: How can exploratory learning with games and simulations within the curriculum be most effectively evaluated? Computers & Education, Elsevier (2006)
6. Courtney, H., Kirkland, J., and Viguerie, P.: Strategy Under Uncertainty. Harvard Business Review, 11 (1997)
7. Weijnen, M., Herder, P., and Bouwmans, I.: Designing Complex Systems - A Contradiction in Terms, Research in Design Series, vol. 3, pp 235-254, IOS Press (2008).
8. Veeningen, J.W.: Design Science research issues in developing and applying serious games in the uncertain world of long term energy investments. MSc Thesis, University of Groningen (2016).