Engineering emergence: applied theory for game design
Dormans, J.

Citation for published version (APA):
Dormans, J. (2012). Engineering emergence: applied theory for game design Amsterdam: Creative Commons

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: http://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (http://dare.uva.nl)
Summary

Game design theories aim to assist game developers in creating and understanding games. Applied theory for game design supports creativity, speeds up the design process, and increases quality of games. This dissertation presents two theoretical frameworks for designing games and discusses how designers can use them to refine the game development process.

Currently there are several design theories for games. However, none of these theories has surfaced as a widely accepted standard within the game industry. Most existing (often academic) design theories have a poor track record. There are two reasons for this. One: they require a considerable investment to learn, and two:, they are more successful as analytical tools than as actual development tools. It should come as no surprise that this has caused considerable skepticism towards design theory within the game industry.

With these concerns in mind, the frameworks presented in this dissertation were designed to be practicable and applicable. The first framework frames games as rule-based systems. It helps to identify and understand the structure of game rules in order to facilitate design. The second framework helps designers to create open, dynamic worlds that to provide players with a smooth gameplay experience. The two frameworks are not completely separate: in a good game, game worlds and rules combine into a complex, carefully balanced whole. One cannot do without the other.

The first framework that focuses on game rules is called Machinations and it utilizes dynamic, interactive diagrams. These diagrams model a game’s internal economy that consists of the flow of vital, and sometimes abstract, resources. Resources can be of many types, for example: points, money or ammunition.

The most important structural features these diagrams capture are feedback patterns. Feedback is generally recognized to play an important role in emergent behavior in complex systems in general, and games in particular. Feedback occurs when state changes in a game element affect other elements and ultimately feedback to affect the first element. In games, feedback often creates an upward spiral. For example, in the classic board game Monopoly investing money in property generates more money to invest.

The Machinations framework distinguishes many different types of feedback: feedback can be positive or negative, constructive or destructive, fast or slow, etcetera. For designers it is important to understand the nature of the feedback loops that operate in their games. What is more, from the study of several games and their internal economy recurrent patterns emerge. These patterns codify general solutions to commonly encountered design challenges. Machinations diagrams are an excellent vehicle to identify and reason about these patterns.

Game designers can create Machinations diagrams with the software developed for this study. The main advantage of this software is that these diagrams can be run: like the games they model, Machinations diagrams are dynamic and interactive. This allows designers to simulate games easily and efficiently; it allows them to balance and tweak a game’s internal economy for particular effects long before a prototype is built.
The second framework, Mission/Space, focuses on level design and the mechanics that control player progression through a game. Unlike existing models of level design, the Mission/Space framework capitalizes on the idea that in a game level two different structures exists. Mission graphs are used to formalize the structure of the tasks and challenges laid out for players, while space graphs are used to formalize the geometric construction of game levels. These two constructions are related, but they are not the same. Rather, the different ways missions might be mapped onto a particular space play an important role in the resulting player experience. Mechanism for locks and keys, which are often used to prevent players from reaching certain areas in a level too quickly, are an common tool to create mission/space structures.

As with the Machinations framework, the Mission/Space framework was used to develop software tools to assist designers in creating levels. In this case, the program Ludoscope allows designers to implement mission graphs and space graphs and simulate a player’s progress through a level. This way Ludoscope helps to identify potential deadlock, or other flaws, in their design.

The Machinations framework focuses on structures of emergence in games and the the Mission/Space framework focuses on structures of progression. Combining the two frameworks provides leverage to investigate how emergence and progression can be combined. Many games combine elements of both. However, there also are many games where there is a considerable mismatch between the sophistication of the mechanics in their economy, space and physics on the one hand, and the mechanics to control progression on the other. By combining the Machinations framework with the Mission/Space framework, two new strategies for integrating emergence and progression surface.

The first strategy zooms in on lock and key mechanisms that rarely include feedback mechanisms. Machinations diagrams can be used to explore more sophisticated and emergent mechanics to create locks and keys, and to control player progress. In the second strategy progress itself is modeled as a resource allowing it to become part of feedback mechanisms that operate on the scale of the entire level or game.

The Machinations framework and the Mission/Space framework formalize different perspectives on games. The notion of ‘model driven engineering’, taken from software engineering, can unify these different perspectives into one formal approach to game development. This perspective ultimately allows us to automate certain aspects of creating games. Using formal grammars and rewrite systems it is possible to codify transformations that allow designers to generate spaces from mission, mechanics from spaces, or missions from mechanics. By dividing the development process into a series of small transformations, flexible tools can be created that can (help to) automatically generate entire game levels, and, to a certain extent, entire games. Another application of these techniques is the development of games that adapt to player performance.

To implement these techniques, the Ludoscope prototype was extended to allow designers to define such grammars and rewrite systems. This creates a potentially very powerful, generic game development tool that allows games to be designed at a high level of abstraction.
The various tools developed as part of this research are one way to validate the applicability of the theory developed in this dissertation, but it is not the only one. Through various presentations and workshop, academy peers and industry veterans were able to comment on and influence the development of the frameworks and tools. In addition, the frameworks, and Machinations in particular, were used in the game development courses taught at the Hogeschool van Amsterdam. We found that these frameworks help students to understand games and improve their designs. Mainly because they formalize important aspects of game design that are not immediately apparent or tangible, and because they facilitate students to simulate and experiment with these aspects.

In the end, designing games remains a tough challenge. The tools and theory presented in this dissertation assist game designers by codifying design lore, simulating games before they are built, and even by automating certain aspects of the design, but they can never replace the human creativity and ingenuity that goes into their design. However, I do think that these tools and theory will assist them and can elevate the art of game design to the next level.