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.
This dissertation set out to develop formal tools for the hard task of designing games. Games, as rule based systems, often display emergent behavior; one of the core aspects of games, gameplay, is an emergent property of game rules and mechanics. Many games try to create gameplay with a relative simple set of rules that offer much variation and freedom to players, although not all games do so in equal measure. For games that aim to deliver a structured story-like game experience, level design usually is more critical to the final gameplay than mechanics that build emergent gameplay are. However, these days, games that build on a pure structure of level progression are quite rare; while emergence plays a role in virtually every game.

As rule based systems, games share kinship with simulations. However, where the purpose of simulation is to accurately model (at least certain parts of) a source system, games are often considered stylized representations of systems that might, or might not, exist outside the game. Games can afford many more liberties in order to entertain, to persuade, or to educate. The relation between game rules and the systems they represent, goes beyond accurate, iconic simulation. For games this means that they can use relative simple means to represent dynamic and complex systems. In fact, there are some arguments to suggest that this is where the real power of games lies: games allow us to play with useful shortcuts in understanding complex systems. Yet, as even relative simple games can display complex behavior, games naturally utilize those structural features of rule systems that give rise to the complex behavior in the first place.

Game studies is still a young discipline. Over the years there have been many attempts to create more generic and formal approaches to games and the process to design them. So far, none of these approaches has grown into a standard that spans the game industry and academia. Game design theories face several problems. Perhaps the most prevalent among them is the little trust the game industry puts in the approaches that have been developed over the years. This lack of trust finds its roots in a poor applicability of many of these theories for actual design, especially when one considers that, in general, the investment required to master these theories is quite high. Many theories, frameworks, vocabularies and abstract design tools are more successful as analytical tools.
than they are as tools that might help to engineer games. Another problem is that designers can be quite skeptical towards the entire enterprise of developing design methodology; these designers usually dismiss any theory that tries to formalize (and thereby ‘steal the soul from’) the creative process of designing games.

With these concerns in mind I have presented two frameworks and outlined the possibilities to develop automated design tools that are firmly grounded in design practice and aim to assist designers in creating quality games effectively and efficiently. The first framework, Machinations, focuses on discrete game mechanics and formalizes a perspective on game rules and emergent behavior in games up until the point where games can be simulated using the Machinations tool before they are build. The Mission/Space framework focuses on level design. Perhaps it is less innovative than the Machinations framework, but it helps to structure the process of designing game levels, nonetheless. What is more, combining both frameworks sheds new light on integrating emergence and progression structures in games that many treat as conflicting ways of creating game challenges. The outline for automated design tools builds on the idea that game design can be framed as a series of model transformations. By using formal grammars to describe the models involved in this process and designing rewrite systems to govern transformations between them, a highly flexible yet formal approach to game design has been created. This approach has been made concrete in a series of prototypes that can generate game content automatically or assist human designers in creating game content.

In this final, concluding chapter I will formulate an answer to this dissertation’s research question, and discuss how all frameworks and prototypes have been evaluated and validated. Finally I will look ahead what research and developments they might inspire in the near future.

8.1 Structural Qualities of Games

The central research question of this dissertation is: what structural qualities of game rules and game levels can be used in the creation of applied theory and game design tools to assist the design of emergent gameplay?

The Machinations framework has been developed to formulate one part of the answer. For games with discrete mechanics, feedback loops within the internal economy of the game play a critical role in quality of the emerging gameplay. A game’s internal economy is formed by the production and consumption cycles that involve the game’s most important abstract and tangible resources. Depending on the game, these resources might be anything: from weapons, ammunition and health in a shooter game, to food, status and safety in a city building game. Feedback is created when the accumulation, production or consumption of these resources directly or indirectly affect its accumulation, production or consumption in the near future. Feedback can have many different characteristics. Traditionally, positive and negative feedback are distinguished, but for games it is equally relevant whether or not the effect of the feedback is constructive or destructive, at what speed it operates and how durable it is, among other
things.

The Machinations diagrams are designed to represent a game’s internal economy and to foreground feedback structures within. From the analyses of a number of games it became apparent that several feedback structures are recurrent and can be found in many games. In addition, most games that display interesting, emergent gameplay feature multiple, but not too many, feedback loops.

To explain the structural qualities of games that are level-driven, internal economy and feedback loops do not help us much. Levels have their own structures that are independent of, yet often also related to, these mechanics. In general level design has not been studied as extensively as game mechanics have been. From a review of a number of level design typologies it has become apparent that one of the problems is that in levels two structures operate that are often only poorly distinguished. To this end, the Mission/Space framework separates missions from space. In this way the framework helps to create a clear perspective on level design. A mission is a series of related tasks players must perform in order to complete a particular level. A space describes the topographic or geographic layout of a level. Missions and spaces are related, but in contrast to what some design strategies suggest, they need not be isomorph. Often missions are laid out in space in such a way that players are assisted in setting exploration goals, have a fair sense of where they need to go, are rewarded for formulating intentions and plans, and experience the growth of their own, or their characters’, abilities.

Lock and key mechanisms play an important role in many games that are level-driven. Lock and key mechanisms create flexibility in the way missions might be mapped to game spaces, breaking away from levels in which mission and space have to be isomorph in order to create a coherent game experience.

Distinguishing between mission and space also helps to identify more clearly the difficulties in designing levels that allow for a more articulate interactive experience. Space and physics have evolved much faster during computer games’ relative short history than mechanics to control progression have. By applying the lessons learned from the Machinations framework, more interesting mechanisms to control player progression through a mission can be created. Feedback mechanisms that traverse between internal economy and level design can be created in order to push the emergent behavior of games towards areas that hitherto have been relatively unexplored: it helps designing games in which progression is an emergent property of the underlying system.

Although the Machinations and Mission/Space frameworks can be used as analytical tools, they are set up as design tools first and foremost. They help to formalize existing design knowledge and experience, they allow designers to express and discuss designs, but mostly importantly they make tangible certain aspects of game design that normally are quite intangible. Machinations diagrams give shape to a system that otherwise is largely invisible, the Mission/Space framework untangles two superimposed structures that co-exist in level design and creates a clear perspective on both. Both frameworks are fairly easy to learn, they do not rely on extensive vocabularies or pattern catalogs. Yet, despite their simplicity they are quite expressive, and can capture a large
variety of different game structures. I hope that the many examples I have used throughout this dissertation are ample illustration of this.

The concerns about formalizing game design, as expressed by some game designers, cannot be taken away by these frameworks themselves. In a manner, this dissertation does exactly what some of them oppose: it seeks to objectify quality in game design. By pointing out that certain creative aspects within game design cannot be objectified they dismiss any formal approach to their craft. However, the game industry must acknowledge that quality is not equally distributed over all games. We are able to compare games and discriminate between good and bad games. The frameworks presented are designed to help designers identify, create and discuss these qualities in games. These frameworks are not designed to replace creativity, rather, they are designed to support it.

8.2 Validation

In this respect, one important question remains unanswered: how successful is the applied theory presented in this dissertation; does it really support game designers in their labor? Clearly, it has not emerged as an industry standard, but nobody can expect that anything could have been developed into an industry standard within the scope of writing this dissertation. The theories have been used by people in the industry and by students, but I have not gathered quantitative data to evaluate its effectiveness. How, then, can the frameworks and the prototypes be validated? How can I be sure that what I have presented is of any real value to a game designer?

I have tried to validate the applied theory of this dissertation in four ways. These ways are described below briefly. In the sections that follow they are discussed in more detail.

1. Implementing many of the theories in software tools to create diagrams, simulate games and generate content, has made these theories much more concrete and has validated them to some extent. On many occasions implementing the theory has led to changes, and vice-versa. In the end I have managed to implement (nearly) all of the theories somehow. The work on visualizing game mechanics eventually has led to the implementation of the Machinations tool, the work on level design has triggered a series of level generation prototypes for action-adventure games that later has grown into a more generic tool for generation of game content: Ludoscope.

2. During the period of this research I have actively sought opportunities to present results to the game development community. In practice this led to a number of talks and workshops. The responses from these workshops were usually positive. Although, I did also meet people who were less enthusiastic for reasons I have discussed in Chapter 3. A number of companies and professional designers have used some of the tools here, and continue to do so.

3. I have presented much of the material as peer-reviewed conference papers
and two journal papers. Many of the chapters in this dissertation are the result of several iterations. For example the Machinations framework started out as a diagrammatic notation based on UML and was initially never intended to be interactive. Many fellow academics, but also people from the industry, have commented on earlier work. Their suggestions led to many improvements and shaped the frameworks into the form presented here.

4. Last, but not least, as I was in the fortunate position to be involved in the development of several game design courses for the Hogeschool van Amsterdam (the Amsterdam University of Applied Sciences) I have had the opportunity to use many of these theories to structure courses and train students.

8.3 Teaching Game Design

Over the past five years I have been involved with the development of several game design courses at the Hogeschool van Amsterdam. In February 2007, my colleagues and I started with a half-year minor program on game design for third year students in the departments of computer science and interactive media. This minor was to complement the already existing game technology minor that, at the time, had been running for two years. This was followed by a serious games course that started in 2008, and a full, four-year game development program starting from September 2009.

My responsibility in these courses have been many, but most of them resolved around setting up and teaching design courses where the students learn how to build rule systems in order create interesting, or meaningful gameplay experiences. Common industry practices have always informed these courses. This includes the MDA framework (mechanics, dynamics and aesthetics), play-centric design, physical prototyping techniques, heuristic evaluations and play-testing strategies (see chapter 3 for a discussion of some of these). But much of the material developed as part of this research also quickly found its way into the course material, starting with the Machinations framework and its precursors.

I have always been a strong advocate of designing games ‘inside-out’. By this I mean that it makes sense to start with designing mechanics, and build the game up from there. The MDA framework suggests something similar. Unfortunately, mechanics and rule systems are not the easiest points of departure for novice designers. It takes experience to appreciate the full effects of small changes to the rules. Lacking experience and an accurate vocabulary to express these nuances, the Machinations diagrams proved to be a useful educational tool. A recurrent assignment was to have students create Machinations diagrams for existing games but also for prototypes they were working on. The first assignment was designed for students to become familiar with the framework, the second assignment to help them get a clear perspective on their own work. These assignments are not easy, for many student it takes time to understand the subtleties of the Machinations diagrams. However, even when students made diagrams that were
poor representations of their games, I found that these diagrams were always a good starting point to discuss their designs with great precision and accuracy.

Some students became very skilled in creating these diagrams and kept using them for other assignments as well. These diagrams sometimes turned out to be very detailed and complex (see figures 8.1 and 8.2). But it illustrates that students were able to use them and that at least some of them saw the benefits of keeping to use them. This way, the framework has had an impact beyond the individual workshops and was instrumental in helping students to create a professional perspective on the design of game mechanics.

The workshop that introduces the Machinations framework has evolved into a more or less fixed format. I have used the workshop or variations on it at several occasions. Obviously, it was used during regular classes at the Hogeschool van Amsterdam, but I also used it for workshops I hosted at the Willem de Koning art academy in Rotterdam, at the Game in the City industry event in Amersfoort, workshops at the T-Xchange lab in Enschede, and at Paladin Studios in Leiden (all of these locations are in the Netherlands). During these workshops, participants start with designing a simple board game mechanism that allows them to move pieces on a board, but that also involves some sort of resource.
Figure 8.2: A student Machinations diagram by Rik Ruiten. This diagram (made for Machinations version 2.0) represents a fully functional prototype for a real time strategy game.
Next, I explain the basic idea of a game’s internal economy and introduce the Machinations diagrams to model this economy. After the participants created diagrams to represent their mechanics in this way, I introduce the notion of feedback and stimulate the participants to explore the concept in their own design. Finally I challenge them to redesign their mechanics in such way that it includes multiple feedback loops. This format is an effective introduction to the framework, and often led to quite interesting designs. Most of the participants were able to create mechanics that were interesting, even when the point of departure was rather simple.

Another, more advanced workshop focuses on the use of recurrent design patterns. This workshop started with a short lecture to explain the framework and then challenged participants to create designs based on a random selection of patterns. Workshops like this were held at a meeting of the Dutch chapter of the Digital Games Research Association in Utrecht in 2008, at the INTETAIN conference in Amsterdam in 2009, and at the DIGRA conference in Hilversum in 2011 (all of these locations are in the Netherlands). This workshop is more advanced and suitable for participants with more experience and expertise.

I was not the only teacher involved in these workshops. My colleagues taught similar classes and workshops. They have similar experiences and point out the advantages of having an online, dynamic implementation visualizing these dynamics. In the near future, we plan to capitalize on this advantage more by expanding our workshop repertoire utilizing this property of Machinations.

The Mission/Space framework was developed later than the Machinations framework. As a result it has not been used as extensively in courses as the Machinations diagrams have been. Yet it has also been instrumental in structuring design courses. The game design course for the second year students in the game development program, was heavily influenced by the ideas developed as part of this framework. The focus of this course was on building prototypes. Every two weeks students needed to finish a new prototype. During each of these two weeks there was a central subject. Mission was one of these subjects, and space was another. During lectures the individual perspectives of missions and spaces were introduced and discussed. For students the identification of lock and key mechanism proved to be a practical and highly applicable perspective that helped structure their designs. Over the years I have received multiple reports that clearly indicate that students are able to work with these concepts, both in analyses and in designs (see figure 8.3).

Framing the game design process as a series of model transformations and procedural content generation has not been the subjects of courses. These subjects are quite advanced, probably too advanced for most of the bachelor students I have worked with. At the moment of writing, the most advanced students are at the end of their second year, and the courses these students will take in their final two years are still under development. Despite the fact that some of these students have expressed a direct interest in content generation, it is unlikely that much of this material will make it into the program. At best, it might be the subject of some of these students bachelor theses, or play a role in the research program of the game lab that has been set up at the Hogeschool van Amsterdam.
Chapter 8 | Conclusions and Validation

8.4 Building Prototypes

The working examples of the prototypes (many of which are available online) should validate the theory to some extent. Validation always was one of the reasons for the creation of these prototypes. The initial level generation prototypes, in particular, were created with the idea that if a formalized perspective of level design principles would lead to an interesting implementation, then the perspective must be of some value. This remains hard to test, and the success of the levels generated by these prototypes were never subjected to an objective measure. However, general reception of demonstrations of several prototypes has always been quite positive.

Initially, I did not intend Machinations diagrams to be interactive, but I am very happy that I did take that step early. This has led to many improvements in the theory, and it also has led to the creation of a very useful design tool. The fact that it can be used to simulate games is very important in this respect. Representations of games in the Machinations tools can, to a certain extent, be played. The fact that in this form the representation retains many of the same dynamic behaviors, is a strong indication that the structures these diagrams foreground indeed play a critical role in creating emergent gameplay.

The development of Ludoscope also proved to be an important test bed for the design theory presented in this dissertation. The theory and the tool evolved in parallel. There were many iterations for both and with each iteration one kept improving the development of the other. This process went on into the very last stages of writing this dissertation, and possibly beyond. Over time, the Mission/Space framework grew more formal, and every time it did, the implementation of the framework made the implementation of Ludoscope leaner.
and more powerful. At the same time, the implementation of the framework as an automated design tool kept opening up new avenues for exploration. For example, with mission graphs implemented in Ludoscope it became very easy to add features to check for mission and space inconsistencies. This process is not entirely finished. Ludoscope is still under development. Without doubt, this development will lead to new theoretical insights that will in turn strengthen the development process.

In the end, developing theory and developing software almost became one continuous process. I feel very fortunate that I was in the position to combine these two methods of developing theoretical and applied tools. One might assume that writing and coding was a tough combination but the truth is that the synergy between the two gave me more energy to develop both to a much higher level than I would have been able to do when working on each project individually.

The research also led to the development of a few experimental games. Most of which are never released as they were only rough prototypes. Some of these evolved further, such as the Infinite Boulder Dash example discussed in Chapter 7. In other cases, the game projects I have been involved in served as an excellent testbed for the theory and tools presented here. For example, for Get H2O I created Machinations diagrams to help design the game’s economy. Games such as Seasons benefited from my level design research. These examples are discussed in this dissertation. Other projects include Ascent, a small platform game that was an experiment with abstract resources; Dungeon Run, Flix and Flix 2 were (at least partly) experiments in level design; Campagne, a satirical card game that benefited from applying non-iconic reduction and creating a symbolic simulation of Dutch politics; and Bewbees that benefited from the Machinations framework in order to create emergent gameplay.

Another experimental game has been built by three students at the Hogeschool van Amsterdam under my direction. For this game, LKE (figure 8.4), I asked the students to experiment with a lock and key mechanism that utilizes a consumable resource as the key and implements multiple feedback mechanisms. In this game the player collects ‘key energy’ that is consumed in different measures by different doors. In addition, the more energy players have, the more they can see of their environment. At the same time, enemies become more aggressive when the player has much energy. This means that players need to balance their energy level carefully in order to progress through a level safely. From this experiment we learned that this type of construction works well, especially when the ‘key energy’ also taps into mechanisms, such as the visibility of the environment. The game is still under development and we are still experimenting with more mechanism that consume or are affected by ‘key energy’. For this experiment the students built a Machinations diagram in order to grasp the game system, and they found it very useful to balance the system but also to guide further development (see figure 8.5).
Figure 8.4: The student game LKE experiments with feedback mechanisms for locks and key. Game was built by Tim van Densen, Jerry Gomez and Martijn de Jong.

Figure 8.5: The machinations diagram for LKE, made by Tim van Densen.
8.5 Academic and Industry Reception

The theory presented in this dissertation has grown over the last three years and during that period I reported my progress through the publication and presentation of ten peer-reviewed academic papers. In particular, Chapter 2 is based on a conference paper that I later turned into a journal paper (Dormans, 2008a, 2011a). The Machinations framework and its predecessors were discussed in three conference papers (Dormans, 2008b, 2009, 2011d). The Mission/Space framework evolved while I was working on procedural content and was introduced in a paper on that work (Dormans, 2010). The strategy for integrating structures of emergence and progression outlined in Chapter 6 has been accepted as a conference paper (Dormans, 2011b). The material presented in Chapter 7 was earlier presented in three conference papers and one journal article (Dormans, 2010; Bakkes & Dormans, 2010; Dormans & Bakkes, 2011; Dormans, 2011c).

I also actively sought out people in the game industry and academia to test my ideas through other channels. In some cases these people approached me because they found my work on the Internet or were pointed in my direction by others already familiar with this work. An important outlet for my theories were Machinations design workshops I hosted at conferences and at different companies. These workshops were not very different from workshops I hosted as part of the game development courses at the Hogeschool van Amsterdam, although the pace in these workshops was much higher. The participants of these workshops needed some time to grasp the diagrams, but most of them quickly understood what it was these diagrams try to convey and did see the value of them. Many participants reported that they enjoyed the workshop and that it brought into focus an aspect of game design that normally is not very articulated. Derk de Geus, CEO of Paladin Studios states: “The Machinations framework has been an invaluable tool to visualize our game’s mechanics. Using Machinations’ systems thinking, we have been able to outline complex game systems and identify the strong and weak points in the mechanics. I highly recommend Machinations for designing games and other interactive systems. As an example, I’m seriously considering to use it as a tool for business process modeling” (2011, from personal correspondence).

The person that most strongly influenced the development of the Machinations tool is Stéphane Bura. He contacted me in early 2009 about my first attempts to visualize game mechanics. As Bura himself has an interest in the matter and contributed to the debate himself (2006, also see Chapter 3), he was immediately enthusiastic about the premise of creating visual diagrams to represent internal economies of games. We met shortly afterwards to discuss our work and he immediately encouraged me to develop a tool to express and to execute the diagrams that later would become the Machinations diagrams. After each iteration of the tool and framework Bura was one of the first people to comment and to push me to develop features that would actually help designers to test out designs. This resulted in the features that allow designers to gather data from many simulated runs of which Bura stated: “This is exactly the kind of
Chapter 8 | Conclusions and Validation

tool I need to balance the economies of the games I’m working on” (2010, from personal correspondence).

Experienced developers frequently see the educational value of the framework as its strongest point. As is made clear by Isaac Jeppsen, a system design engineer at Aptima, Inc. who wrote: “I was ecstatic to find your paper on Machinations. I’ve spent the past year at Aptima (a human factors engineering company) trying to explain to them exactly what you illustrated so clearly in your paper. After being a producer at a game development shop, I was continually frustrated by the lack of good tools to explain to my new academic colleagues the essence of a game as the underlying systems that drive it” (2011, from personal correspondence). This would indicate that the framework is more useful for students and novice designers. In a way I agree with this sentiment. Experienced designers need not rely on aids like these frameworks as much as inexperienced designers do, just as programming paradigms and design patterns seem second nature to experienced software developers. Although, I do like to point out that adding the option to simulate games using these frameworks is a more recent and deliberate attempt to increase the value of the frameworks for more experienced designers.

Finally, during the 2010 G-Ameland game jam I had the opportunity to show and discuss my work with game design veteran Ernest Adams. His books on game design have had a strong influence on the Machinations framework, as the notion of internal economy that lies at the heart of the Machinations framework was conceptualized by Adams in the first place. Adams immediately saw the potential of the diagrams: “Machinations is the best practical design and testing tool for game mechanics that I’ve ever seen. It’s much more convenient and intuitive than using spreadsheets or writing code” (2011, from personal correspondence).

8.6 Omissions

The Machinations framework and the Mission/Space framework do not cover all aspects of games. There is more to games than just mechanics. Art, artificial intelligence, and interactive control schemes are some of the areas that have not been discussed here, but that are relevant for game design. This dissertation focused on mechanics, but not even all mechanics have been discussed in detail. As indicated in chapter 1, this dissertation has zoomed in on the mechanics that build a game’s internal economy and control level progression. Mechanics that deal with physics, maneuvering, and social interaction are beyond the scope of this dissertation. However, it is important to consider these omissions at this point, as all of these type of mechanics can be the cause of interesting and emergent gameplay.

The mechanics that govern a game’s physics are of a different nature than those governing a game’s economy. Because physics mostly deal with continuous rules and simulation, the discrete representation of rules and mechanics in the Machinations framework are a poor match. Likewise the graphs of the Mission/Space framework do not match the continuous space made possible by
the physics. To involve a game’s physics on the same level of abstraction as the Machinations and Mission/Space framework, a new framework should be found or created that can deal with that sort of rules easily and effectively. What structural qualities such a framework should foreground I cannot tell. I suspect feedback structures will also affect physics. After all, continuous, emergent behavior such as the flocking of birds also involves feedback. A formal framework for game physics is a tantalizing subject in and by itself, that requires considerable research.

Something similar can be said of mechanics that govern maneuvering and social interaction. Both comprise large fields of research, yet I do think that in these areas great strides can be made by trying to identify those structural qualities that lead to emergent gameplay. As it stands now, the theory presented in this dissertation does little to explain the quality of a game like Go that seems to derive its quality from maneuvering almost exclusively. I have tried, several times, to create Machinations diagrams for Go but was never successful. Games that rely a lot on maneuvering probably benefit from an approach based on cellular automata. Most of these games have many similar functioning units that do respond to their immediate surroundings.

There are also some blind spots within the frameworks themselves. For example, failure is something neither really takes into account. However, failure is a very common occurrence in games, even though it is often neglected in many contemporary games (Juul, 2010). The frameworks, and especially mission graphs, can be changed so that failure to perform certain task becomes an explicit option and integral part of the Mission/Space framework. As was briefly mentioned in a footnote, relations that specify how failure, or inhibition of crucial tasks, might be undone could be a great addition to the framework. This way design strategies for games that better accommodate failure might be developed.

8.7 Future Research

So, what is next? The development of the Machinations and Mission/Space work with their associated tools and prototypes is not finished. I see six opportunities for future research and development:

1. The Machinations diagrams might be quite complete, but the research into recurrent design patterns utilizing the framework is not. The patterns described in appendix B are just the beginning. Many more patterns might be found, developed and described. I would advocate that the total number of patterns be kept low. This means that existing patterns in the library should be reexamined from time to time. I already dismissed and merged some patterns.

2. The Machinations tool might be expanded to allow users to embed diagrams in user defined elements to be reused in other diagrams. This would expand the expressive power of the tool extensively. However, there are some risks involved in exploring this direction. One of the strengths of
the current framework is the level of abstraction it enforces. By allowing embedded diagrams, the focus on the larger structure might be lost.\textsuperscript{1} On the other hand, it would enhance the capacity of the current diagrams into something closer to a visual programming language specific for the design of games. This might present opportunities for developing prototyping tools and programming environments dedicated to the generation of games.

3. I already mentioned that I have used the frameworks and theory in the development of game prototypes. Unfortunately, none of those games is finished at the moment of writing. But that is because of a lack of time, not a lack of applicability of these frameworks. I know I will continue to work on these projects, but I also hope to inspire others to make use of these techniques to create games with adaptable, dynamic gameplay from which story-like progression emerges naturally.

4. Ludoscope, the automated design tool outlined in chapter 7, remains a prototype. I do think that Ludoscope can be developed into a fully-functional, generic game design tool to assist game designers. This would require much work, and plenty of research, but I do foresee that such a tool can be very beneficial to the effectiveness and enjoyability of game design.

5. As suggested in the previous section, formal frameworks might be extended with the development of frameworks for the mechanics of physics, maneuvering, and social interaction. This would allow us to approach more types of games and tap into more sources to engineer emergent gameplay.

6. More work could be made of validating and refining the results of this research. This can be done by further exploring how the applied theory presented here can be used to develop more advanced game design courses. The Mission/Space framework has not seen as much action as the Machinations framework. I do not expect that these frameworks have been exploited entirely. In addition, more could be done to present these theories to a wider, industry audience. That way, these frameworks might gather more support and might evolve further to better reflect the industry’s needs.

8.8 Final Conclusions

Designing games remains a hard task, but gameplay can be engineered. The applied theory presented in this dissertation can help designers to get a better understanding of those structures in games that contribute to the creation of emergent gameplay. It can assist them in their labor to design games, but it can never relieve them of the responsibility that comes from the creative effort required. I do not wish to prescribe the way designers should use this material.

\textsuperscript{1}This was pointed out to me by Chris Lewis in a personal conversation.
Since theories, frameworks, and automated design tools are only as effective as their users, the real challenge lies still ahead.

The applied theory on game mechanics and level design aims to elevate game development to a higher level of abstraction using clearly defined, formal perspectives. Assisted by the frameworks and the tools outlined in this dissertation, designers can focus more on those aspects of the process that require their creativity and ingenuity. In the end, I like to think that I have supplied designers with new, more powerful tools, with which their work might become more effective, but also more enjoyable.