A Systematic Review of Game Design Methods and Tools

Marcos Silvano Orita Almeida[1] and Flávio Soares Corrêa da Silva[2]

[1] Federal Technologial University Paraná, Campo Mourão, PR, Brazil
marcossilvano@utfpr.edu.br
[2] Institute of Mathematics and Statistics, University of São Paulo, São Paulo, SP, Brazil
fcs@ime.usp.br

Abstract. The game designers craft is very young if compared to film-making and software development. The knowledge base and formal techniques of these areas is far more comprehensive. Even after decades of evolution of the games production software, the range of design centered techniques and tools is still limited, as observed by many authors. Thereby, efforts have been made towards the establishment of game design formal methods. This paper presents a sys-tematization over the contributions of researchers and designers towards con-ceptual and concrete tools. These efforts converge to two approaches: the build of a shared design vocabulary and a game design modeling language. While valuable, the existing implementations of these approaches are not mature enough to gain industry adepts, serving only as reference to future works. Moreover, it is needed to discover the designers particular methods, which may contribute to-wards the constitution of a unified design toolbox.

Keywords: Game design, game design methods, game design tools.

1 Introduction

The game design is one of the key-areas to the digital entertainment industry. With the emergence of new technologies, production tools have evolved considerably, but little has been made to provide support to game design. Designers still use the same instruments from the beginning of the area and, although the industry has seen con-secutive successes in sales, researchers and professionals agree that the lack of tools, whether conceptual or software, imposes a barrier to any standardization attempt and hinders the knowledge transfer between generations of designers. They proposed conceptual and concrete tools that could complement or replace the design document, aiming improvements to the games creation process.

This paper presents a systematization of these efforts through a chronological overview of the main approaches and their implementations, in order to map them within the context of the state of art of design tools. Few studies have

J.C. Anacleto et al. (Eds.): ICEC 2013, LNCS 8215, pp. 17–29, 2013.

been published with a similar purpose. Kreimeier (2003) summarized the main methods proposed until 2003, analyzing them from the designer's perspective. Neil (2012) presented a discussion over the need to evaluate the proposed design tools.

2 The Game Design and Its Tools

The process of creating games has an intrinsic reliance on the designers creative skills. Books, movies, music and real world facts and culture serve as inspiration for games. Furthermore, existing games typically act as a foundation for new ideas. Ex-perimentation is an essential part of this process, which can be summarized in three steps: design, documentation and prototyping. The game design document is the main artifact produced by a game designer. It is used to communicate the designer's vision to the development team. It also acts as a guide to the whole development process and considered by many as a production method (Kreimeier, 2003).

As a key part of the game design activities, the design document has been subject of some efforts trying to establish more standardized content and format. However, no concrete results have been achieved. According to Dormans (2012), the lack of standards in design documentations inhibits the creation of a universal design methodology. Kreimeier (2003) points that while some designers highlight the need for the document others deny its practice. For Keith (2010), the document is rarely used by developers in later development stages, serving only as a contractual object. The size and format of the document can be factors that lead to such practice. Even with the use of visual aids, such as sketches and storyboards, Costykian (1994) points such tools are not enough to describe the design. Its static nature is also highlighted as a problem. Dormans (2012) points out that changes in the game concept are common and updating the document to reflect them becomes unproductive as the project progresses. Librande (2010) emphasizes that although the document has the purpose of communicating the designers vision throughout the development team, those involved in production feel little motivation to read it. The author emphasizes the need for visual languages, more expressive and compact.

Typically in a game development project, testing prototypes are produced in pre-liminary stages after the design document, with intense focus on gameplay and usually disregarding artistic presentation. They are commonly used as a concept proof tool and experimentation environment to evaluate and evolve the gameplay initially planned in the design document (Salen and Zimmerman, 2003). Game designers have unanimous agreement about the value of experimentation through prototyping, regarded a critical part of the game development process and considered the only reliable method of verifying the design quality (Neil, 2012).

With rare exceptions, designers are not able to build the game prototypes. Game creation toolkits are creatively too restrictive and analog prototypes, like board games, are ineffective for games with interaction mechanisms heavily based

on real time actions (Sigman, 2005). Therefore, software prototypes are usually built by people involved in the production, like software developers and graphic artists, based on the documentation and guidance provided by the designers. This creates a gap between the conception of game ideas and the experimentation process that could attest them (Neil, 2012), turning prototyping into a costly and slow process, away from the direct control of the designer. Moreover, the prototype is commonly constructed after the completion of a significant part of the design document, making it impossible for designers to do instant gameplay experiments during the game conception phase.

Generally speaking, designers aim a tool that would allow them to build experi-mental prototypes directly from the definition of a set of game characteristics. This could provide a way to perform instant proofs of concepts while creating the design documentation, through an iterative process of design and testing of gameplay. In this environment, the ability to reuse existing elements of games as a design vocabulary would be an essential resource. A study of requirements gathering for design tools conduced by Nelson and Mateas (2009) confirms this direction. However, software tools are just facilitators for the use of practical and consolidated conceptual tools. The architecture and software engineering are examples of this scenario: the CAD software have been developed only to aid in the use of conceptual tools that already existed and were of common use. In this way, the first logical step would be the development of a conceptual tool box aimed at supporting the game design, which has been the main target of many studies in the area.

3 Early Discussions towards New Design Tools

Although widely adopted as the mainstream tool in the industry, there is a consid-erable lack of formalization and standardization in design document. Despite the visible success of the industry, researchers and professionals agree that the lack of tools, whether conceptual or software tools, hinders the knowledge transfer between generations of designers and the evolution of the design process itself (Neil, 2012). However, this is not a recent issue.

In 1994, Costikyan (1994) published a discussion about the need for greater for-malism on game design, suggesting the search of a common vocabulary that would allow designers to analyze and describe games. For him, designers should have a way to analyze games, understand them and identify the elements that make them good or bad. Therefore, Costikyan emphasizes the need for the creation of a shared language for game design. His discourse has been directly or indirectly echoed by many researchers and designers. For Church (1999), the main inhibitor of the design methods evolution lies in the lack of a common design vocabulary to describe games concepts, hindering the transfer of knowledge between generations of designers. Later, Fullerton (2008) also expressed the same opinion, pointing out the lack of a common vocabulary as one of the biggest problems faced by the games production industry of its time. Like Costikyan, Church advocates the creation of a vocabulary, an analysis and design tool that

would allow an univocally identification of the elements that constitutes games, aiming to establish a collection of reusable design concepts. Through such vocabulary, Church hopes to be able to dissect a game, identify and separate its forming components, understand how they fit and balance together, and analyze which ones benefit or harm certain games or game genres.

Over the years, practitioners and researchers have identified the need for formal models and tools to support game design (Neil, 2012). In this context, "formal" does not refer to mathematical models, but to organized, standardized and structured models and tools to aid the game design methods. This paper presents a sys-tematization of the efforts made to date to build such models and tools, mapping them into the universe of approaches of game design tools and highlighting their specific contributions. This systematization is presented in the form of a state of art map of game design tools (Figure 1), which organizes the existing tools into two main groups: shared design vocabularies and visual design languages.

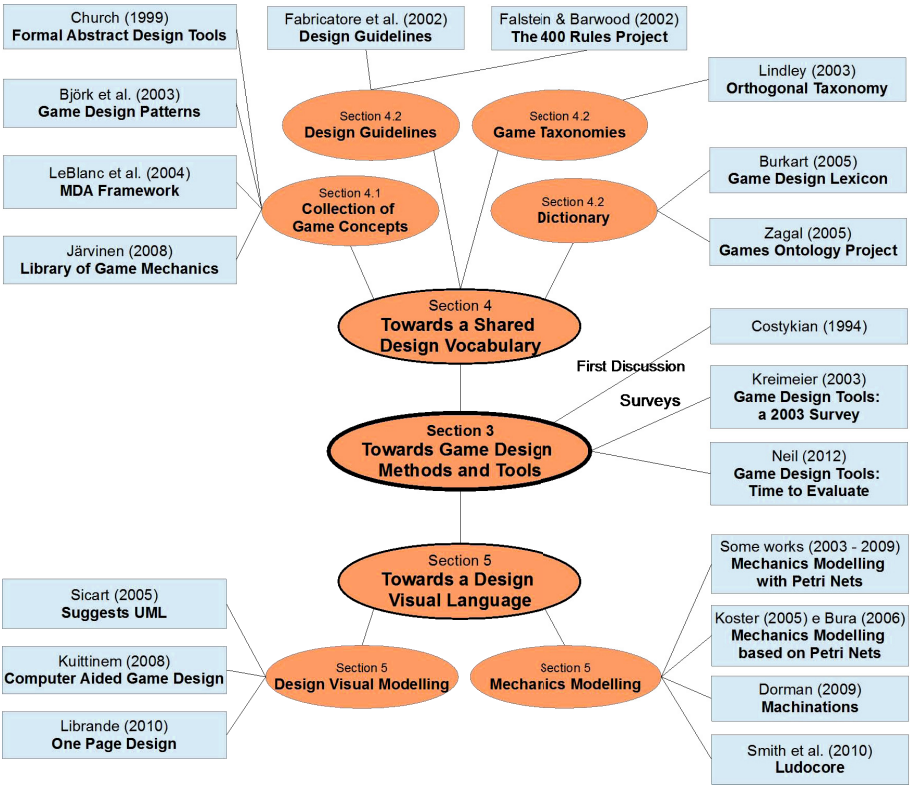

Fig. 1. State of art map of game design methods and tools

4 Towards Better Communication through a Unified Vocabulary

Many researchers and designers consider that a shared and unified vocabulary can bring significant benefits to the area (Neil, 2012). The fours approaches towards this accomplishment are discussed through this section (upper segment of Fig. 1. map).

4.1 Collections of Design Concepts

In 1999, Church (1999) proposed the creation of Formal Abstract Design Tools (FADT). They comprise a vocabulary of recurring game design concepts. Each term is presented by name and description. For the author, a FADT must be precise, unam-biguous and applicable to as many situations as possible. However, although Church has aimed to dissect the games in its forming parts the FADT do not define games components, but rather, abstract concepts and features. This can be noticed in one of only three terms defined by the author: "Perceivable Consequence: a clear reaction from the game world to the action of the player." In general, the model is too simple and relationships with games or genres are not covered. It also lacks information about the use implications of each FADT in a game. Its application relies too much on the abstraction ability of each designer. Kreimeier (2003) pointed out that in the end, only 25 FADT were documented and the work was discontinued. Dormans (2012) reported that between 1999 and 2002, the Gamasutra[1] site hosted a forum where people discussed and expanded Church's tool, although there are no reports of its use on real world projects. Although simple and short lived, Church's work repre-sented the first attempt towards a standardized collection of game design concepts.

In 2002, Kreimeier suggested the adoption of an approach similar to the Software Design Patterns of Gamma (1994) to document recurring concepts of game design. Following the same line, Björk et al (2003) proposed a model of Game Design Pat-terns (GDP) which, though inspired in Gamma, does not follow the approach of pairs "problem-solution" when structuring patterns. The GDP document recurring concepts of game design. The patterns represent games mechanics as well as transversal char-acteristics, like Church's FADT. Each design pattern is documented by a well-defined structure, which includes name, definition, usage examples, application instructions, narrative aspects, consequences of use, and relationships with other patterns. Accord-ing to the authors, the GDP aim to provide a tool for analysis and design of games (Björk et al. 2003). The project has a wiki[2], in which there are 386 patterns collabora-tively documented, which shows the intention of making it a tool of practical use.

As a design technique, the Game Design Patterns represent an evolution over the FADT and the most significant contribution in trying to set up a database of design concepts. Its design patterns inspired structure allows highlighting the

[1] `http://www.gamasutra.com/` (visited on 2013/03/13)
[2] `http://gdp2.tii.se/index.php/Main_Page` (visited on 2013/03/13)

relevant aspects of each pattern and their interrelationships. On the other hand, the project has short-comings that discourage its use. There is not enough correspondence between patterns and the games or genres that use them to allow a games-centered analysis. The wiki's documentation and navigation simplicity hinders both the understanding and the use of the patterns. It is common to find incomplete and contradictory docu-mentation on patterns, with disagreements between title, definition and usage exam-ples. Finally, there is a lack of graphical models to facilitate the patterns understanding and to allow the visualization of the hierarchy and relationships between patterns and games. These characteristics transform the GDP in a little intuitive scattered collection that requires a high learning curve.

LeBlanc et al. (2004) highlight the importance of considering both the perspective of the designer (producer) and the player (consumer) when designing games. Their work proposes a framework to analyze games in three different components: mechanics, that describe the static rules of the game system, described in the design document; dynamics, that represent the run-time behavior of the mechanics on a play session, present in games and prototypes, and; aesthetics, which comprehends the desired emotional responses of players, obtained from tests and experimentations with the prototype or game. This separation of components in rules, run-time and emotional responses highlights the focus on gameplay experiences when designing the game rules, and with it, the importance of building software prototypes for testing the outcome of the designed concepts. Moreover, it ratifies a very common practice of designers: the investigation of the emotional result of mechanics implemented in the existing games. Thus, as part of their work, designers must constantly experience a vast amount of games in an analytical way, which is quite costly. Thereby, the importance of building a database of design concepts drawn from existing games becomes evident, though neither of the existing implementations has been successful.

Similar approaches to the ones of Church (1999) and Bjork et al. (2003) were dis-cussed in the work of Järvinen's mechanics library (Järvinen, 2008) and the definition of game mechanics inspired by the object-orientated paradigm of Sicart (2008). There is also a collection of game concepts documented in the GiantBomb[3] site, which although presents an informal approach and a focus clearly aimed at end users, without the apparent intention of establishing a tool for designers, features a simple and functional solution based on collaborative construction. On the site's database, the understanding of the concepts is facilitated through the use of illustrations. Moreover, it has a considerable range over the library of available games, relating them to the concepts used. Although devoid of formalism, this collection of game concepts presents functional characteristics that may inspire future works in the area.

[3] `http://www.giantbomb.com/concepts/` (visited on 2013/03/13)

4.2 Other Approaches and Their Implementations

Unlike the attempts focused on building a collection of design concepts, Falstein and Barwood initiated the "The 400 Rules Project" in 2002 (Falstein, 2002). Its aim was to identify, record and share practical experiences of designers indicating direc-tions to be taken or avoided on a game project. From the initial planning of 400 rules, only 112 were documented on the project's website[4] through the collaboration of various designers. As in the Game Design Patterns project, this work demonstrated a clear intention to provide an artifact of practical use. In a similar approach, Fabricatore, Nussbaum and Roses (2002) proposed design guidelines from an analysis of the influence of gameplay mechanisms in player's motivations. Regardless of the obtaining method, the documentation of design guidelines can assist in the knowledge transfer between generations of professionals, especially during the training of new designers. Moreover, it has an intrinsic practical nature, considering that portrays the concrete learning of the designer's craft. On the other hand, guidelines cannot constitute a tool that actually supports the idiosyncratic crea-tive process of game design.

Burkart (2005) and Zagal et at. (2005) started distinct projects aiming to create shared design dictionaries in order to provide unambiguous definitions for terms commonly used in game design documents. These dictionaries would be available in a digital encyclopedia for online access and stored in a database to be embedded in documents through markup languages, such as XML. Of these, the Games Ontology Project of Zagal et al. (2005) was published in a wiki[5], where 179 definitions of spe-cific game design terms are documented. Although dictionaries alone cannot become design tools, Kreimeier (2003) states they are necessary complements to any method or conceptual tool. For him, even if recurring design concepts patterns, mechanics and games components do define part of a vocabulary, not every term defines a design concept but must be uniquely defined. Thus, dictionaries of terms can be seen as a lower abstraction layer, serving as a foundation for other approaches. Much like the design guidelines discussed above, a shared dictionary of terms has a concrete value of practical application in the designer's everyday, especially when writing doc-uments and communicating, but cannot surface as a design tool by itself.

A common language practiced by the industry and academia is resultant of the games taxonomy (Järvinen, 2008). Age, target audience, purpose and genre are examples of criteria that lead to different classifications. Of these, the genres classification is perhaps the most concrete example of value and practical application of a games vocabulary, structuring games with similar interaction characteristics under the same group. This classification helps to define the typical elements of each game genre and to understand how the mixing between genres occurs. Some authors suggest that not all games can be framed in a hierarchical system of categories and subcategories (Björk, 2003). Among them, Lindley (2003) proposed an orthogonal taxonomy for games that defines a spatial system of characteristics, where games and genres are positioned according

[4] `http://www.finitearts.com/Pages/400page.html` (visited on 2013/03/13)
[5] `http://www.gameontology.com/index.php/Main_Page` (visited on 2013/03/13)

to their proximity to these characteristics. As an example, the author presents spaces of one, two or three dimensions applied to various characteristics such as simulation, ludology and narratology. Further studies could relate the taxonomies in games and the collections of design concepts in order to investigate which quantitative and qualitative relationships can be drawn between elements, market and user preferences. The outcome of these studies could suggest which elements are desirable for a particular game project that fit into a classification.

This section presented four different approaches and their implementations at-tempts to build a shared design vocabulary: collections of design concepts, design guidelines, dictionaries of terms and taxonomies in games (as seen in the upper seg-ment of the Fig. 1 map). Of these, the first has promising applications as a supporting tool for game analysis and design, allowing designers to create new games from an accumulated knowledge base. However, the existing implementations still do not allow this. The other approaches dictionaries, taxonomies and guidelines have an intrinsically practical nature and although they cannot surface as tools, they are significant contributions to the conception and adoption of a common vocabulary. Overall, all approaches show an evident lack of maturity and computational support for adoption or even for experimentation in real world scenarios.

5 Towards Better Communication through Visual Game Design Modeling

Although of predominantly textual content, designers commonly use visual artifacts as auxiliary tools in the design document (Neil, 2012). Game events and charac-ters behaviors are often expressed by diagrams and story boards. Conceptual illustra-tions are used to preview the game graphics. Furthermore, it is common for designers to create their own visual notations to express part of the game design in which they are working (Salen and Zimmerman 2003). The ¨One Page Design¨ schemes of Lib-rande (2012) are an example of this practice.

Designers have found in the visual modeling a strong ally to communicate their vi-sion of the game to other team members. Librande emphasizes that visual models are more synthetic, naturally communicative and scale better. For him, the diagramming practice forces the designers to extract the essence of the gameplay in few visual elements. Also, designers are driven to specify the relationships between the components of gameplay and to adopt a top-down approach, breaking larger problems into smaller pieces. In general, visual schemes create a game design map to the team, facilitating its understanding and update. However, the game design profession lacks a standard visual language and efforts have been made towards this approach. Along with the attempts to develop a common design vocabulary, the demand for visual design languages comprises the second main approach in which projects have followed towards design formal methods and tools (lower segment of Fig. 1). With this objective, two branches can be differentiated: the search for visual languages of design and languages for game mechanics modeling.

The use of visual languages for representing the design resembles the requirements and design modeling on the software engineering. This similarity has been noted by some authors, who discussed the use of UML diagrams to create design maps. Sicart (2008) presented a definition for game design based on the concept of object orienta-tion, suggesting the use of UML (Fowler, 2004) for modeling, although no examples were shown. Demachy (2003) discussed the use of Extreme Programming, an agile software development method, in the production of games. He also refers to UML as a design tool that can be used to describe game elements and the gameplay. Blumen-thal (2005) designed the Space Invaders game as a pedagogical demonstration, using UML to capture the requirements and design of the game. In general, though textual content documents are frequently used in software specification, visual languages are heavily used to enhance communication inside and outside of the development team. Likewise, the UML can be a convenient tool to communicate the design in a games project. However, further studies of UML application must be performed to make it suitably adapted to the needs of game design.

In order to generate design models based on Game Design Patterns of Björk (2003), Kuittinen (2008) implemented the software CAGE Computer-Aided Game Design one of the few initiatives already made to build concrete tools. His goal is to allow designers to select the patterns that will constitute the game concept and visual-ize their inter-relationships in a diagram. The descriptions of the used patterns are then integrated into the design document. Although simple and academic, this work represents an interesting attempt to apply a conceptual model and to integrate it to the method currently used in industry. In an opposite direction, the software Sketch-It-Up! (Karakaya, 2009) provides a visualization of the design narrative. Although not related to the design modeling, the software represents another attempt to bring computational support to game design. The central idea of the tool is to enable designers and developers to build a global view of the game through animatics (ani-mated story boards). In a brainstorming session, participants interact simultaneously by the software, controlling drafts of actors and objects in the game to create animated sequences of representations of the gameplay.

In contrast to the design modeling, academic approaches has focused on a lower level of abstraction, trying to put together a vocabulary of logical constructs that allow representation of game mechanics, a process that has been done textually in the design documents. Adaptations of Petri Nets for this purpose have been addressed by several authors, such as Brom & Abonyi (2006), Roque & Araujo (2009) and Natkin & Vega (2003). Koster (2005) and Bura (2006) also presented adaptations of Petri nets with several own customizations. Going a little farther, Joris Dorman (2009) built a software tool for mechanics modeling and simulation in real-time called Machinations. Using a visual language, designers specify the mechanics of the game. Once completed, the rules in the diagrams can be executed, allowing a ¨game simulation¨. Ludocore, a tool created by Smith, Nelson and Mateas (2010) follows a similar approach. However, instead of graphic constructions, designers use a textual programming language

to create mechanics models, which can also be simulated in the environment. Although mechanics execution is apparently promising, the both tools use unknown and little intuitive languages. This is a fact that deserves attention, since the designers themselves use flowcharts to assist in the textual descriptions of mechanics. Additionally, the "game simulation" performed by these tools do not represents the player perspective, preventing the experimentation of the game aesthetic components.

The approaches presented in this section were distinguished into two groups: the constitution of languages for description and simulation of game mechanics, and the visual representation of the game concept through design modeling. In both, the goal is to act as a facilitator agent of communication in development teams and between generations of designers. In general, they represent the adoption of consolidated engi-neering approaches in game design. Although designers who heavily base their work on narrative techniques can provide resistance to the use of visual compositions, they represent a trend and a need already highlighted by researchers and practitioners.

6 Final Thoughts

The Game design lacks a shared tool box containing both broad application solu-tions and specific to certain genres of games. On one hand, professionals and compa-nies crave for tools that allow them to improve production and reduce the risks of investment in new projects. On the other, independent developers and beginners seek for tools that can bring them productivity and directions. Furthermore, the use of standardized tools can bring industry and academia closer, contributing to build a universal knowledge base of game design. The design document, the main tool cur-rently used, is considered by many authors as inappropriate to such aspirations.

In general, designers crave for productive and standardized tools and techniques that do not sacrifice the freedom and creativity inherent to their craft. Several attempts to set up these tools have been made and can be generalized under two approaches: the building of a design vocabulary and the development of visual languages for design modeling. Under the first group, researchers and designers have addressed topics such as dictionaries of design terms, taxonomies in games, design guidelines and collections of design concepts. The first three approaches can not be considered tools. Dictionaries are elementary, serving as a basis for communication between professionals and for project documents. Design guidelines help to transfer experiences to new generations of designers. Taxonomies help to organize the vast library of existing games and to compose a design vocabulary, besides essential to market studies. On the other hand, the collections of design concepts, addressed in the fourth topic, represent a promising possibility of a design toolbox and an important component in building a universal knowledge base of design.

Designers commonly build their work over the previous results of other professionals, through experiments and analysis of existing games. As a source of

inspira-tion, designers constantly use commercial and independent games in order to raise requirements to the project they are working on. This practice also allows them to increase their knowledge about game mechanics and the gameplay experiences they generate (their aesthetic components). Moreover, this process of research and experi-mentation acts as a proof of concepts to the desired mechanics. In general, the goal of this practice is to extract the elements that can lead games to success or failure, understand how they fit and balance, and assist designers in the decision to include these elements into their projects. On the other hand, the existing attempts of implementing this approach of collection of design concepts FADT (Church, 1999), Game Design Patterns (Börjk, 2003), Library of Mechanics (Järvinen, 2008) and Library of Concepts (GiantBomb[6]) do not meet the criteria raised. Together, these findings create a strong incentive for new attempts to build a collection of recurring design elements drawn from existing games. Moreover, this approach is aligned with the needs raised by Costikyan (1994) and Church (1999).

From the perspective of its development, games can be considered as the combination of filmmaking and software. So it would be natural that the tools coming from these areas were borrowed for both development and design of games. From the filmmaking, the loan occurs: designers use story boards, sketches and cameras and narration techniques to design and document game concepts. However, the software development tools have very little influence over the designer work. While software design and programming techniques are useless to the designer, requirements analysis instruments can benefit them. From the perspective of requirements capture and analysis, game designers have a very similar role to the system analysts, once they must understand and document the needs of users by their perspectives. Thus, it would be expected that visual languages heavily used by analysts, like UML, were adapted for game design, which does not occur in practice. Furthermore, although several authors agree with this perspective and others highlight the benefits of using diagrams in the game design, little has been done.

Even criticizing the model of designing games as a whole, designers and research-ers have been focusing solely on the creation of environments and languages for modeling and simulation of game mechanics, such as machinations (Dormans, 2009) and Ludocore (Smith, Nelson and Mateas, 2010). In these implementations, the use of unfamiliar and counter intuitive languages, the high learning curve required and the uncertainty of practical gains, keep the designers away. Moreover, these tools have been validated in academic environments, applied to restricted work groups, usually composed by academics and beginners, which does not reflect the area. Even its authors are uncertain about the gains that such tools will bring to other designers and highlight the need to evaluate them on the field. Neil (2012) proposed to carry out such evaluation.

Last but not least, in addition to recognizing the designers needs, it is needed to know their particular methods, those not widely documented. The commercial success of the industry cannot be overlooked, a fact that certainly contributes to certify these methods. In particular, it is wanted to find out what are the tools

[6] `http://www.giantbomb.com/concepts/` (visited on 2013/03/13)

the own designers create and use to work on particularities of their projects. It is possible that some significant importance tools have already been developed and abandoned at the end of a project. The design schemes on a single page of Librande (2012) are examples of an instrument carved from his needs. Becoming aware of such tools, created and used in specific design situations, can contribute with the approaches discussed in this paper to advance towards the constitution of a design toolbox. The fact that industry and academia agree about the needs of game designers indicates that both know what must be done, but the rejection of using the conceptual and physical implementations available, makes it clear they yet do not know how to do it.

References

1. Arajo, M., Roque, L.: Modeling Games with Petri Nets. Proceedings of 2009 Digital Games Research Association Conference (DiGRA). Brunel University (2009) **Brunel University**
2. Björk, S., Lundgren, S., Holopainen, J.: Game Design Patterns. In: Level Up - Proceedings of Digital Games Research Conference (DiGRA). Utrecht University (2003)
3. Blumenthal, R.: Space Invaders: A UML Case Study. Regis University, class notes (2005)
4. Bura, S.: A Game Grammar (2006), `http://www.stephanebura.com/diagrams/`
5. Brom, C., Abonyi, A.: Petri-Nets for Game Plot. In: Proceedings of AISB, vol. 3 (2006)
6. Burkart, P.: Discovering a lexicon for video games: New research on structured vocabu-laries. International Digital Media and Arts Association Journal 2(1), 18–24 (2005)
7. Church, D.: Formal Abstract Design Tools. Gamasutra (July 1999), `http://www.gamasutra.com/view/feature/3357/formal_abstract_design_tools.php`
8. Costikyan, G.: I Have No Words & I Must Design. Interactive Fantasy, Eng. (2) (1994)
9. Demachy, T.: Extreme Game Development: Right on Time. Every Time, `http://www.gamasutra.com/view/feature/2827/extreme_game_development_right_on_.php`
10. Dormans, J.: Engineering Emergence: Applied Theory for Game Design. Teste de Dou-torado. Amsterdam University of Applied Sciences (2012)
11. Gamma, E., et al.: Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley (1994)
12. Fabricatore, C., Nussbaum, M., Rosas, R.: Playability in Action Videogames: A Qualitative Design Model. Proceedings of Human-Computer Interaction 17(4), 311–368 (2002)
13. Falstein, N., Barwood, H.: More of the 400: Discovering Design Rules. Presentation at GDC (2002), `http://www.gdconf.com/archives/2002/hal_barwood.ppt`
14. Fowler, M.: UML Distilled: A Brief Guide to the Standard Object Modeling Language, 3rd edn. Addison-Wesley, Boston (2004)
15. Fullerton, T., Swain, C., Hoffman, S.: Game Design Workshop: Designing, Prototyping, and Playtesting Games. CMP Books, San Francisco (2004)

16. Järvinen, A.: Games Without Frontiers: Theories and Methods for Game Studies and Design. Tese de Doutorado. University of Tampere (2008)
17. Karakaya, B., Garcia, C., Rodriguez, D., Nityanandam, M., Labeikovsky, N., Al Tamimi, T.: Sketch-It-Up! Demo. In: Natkin, S., Dupire, J. (eds.) ICEC 2009. LNCS, vol. 5709, pp. 313–314. Springer, Heidelberg (2009)
18. Koster, R.: A Grammar of Gameplay. Presentation at GDC 2005, San Francisco CA (March 2005), http://www.raphkoster.com/gaming/atof/grammarofgameplay.pdf
19. Kreimeier, B.: Game Design Methods: A 2003 Survey. Gamasutra (March 2003), http://www.gamasutra.com/view/feature/2892/game_design_methods_a_2003_survey.php
20. Kuittinen, J.M.: Computer-Aided Game Design. Master's Thesis in Information Technology. University of Jyväskylä (January 19, 2008)
21. LeBlanc, M.: Formal Design Tools: Feedback Systems and the Dramatic Structure of Competition. In: Computer Game Developers' Conference (1999)
22. LeBlanc, M., Hunicke, R., Zubek, R.: MDA: A formal approach to game design and game research. In: Proceedings of the AAAI 2004 Workshop on Challenges (2004)
23. Librande, S.: One-Page Designs. Presentation at GDC 2010, San Francisco, CA (March 2010), http://stonetronix.com/gdc-2010/
24. Lindley, C.: Game Taxonomies: A High Level Framework for Game Analysis and Design (2003), http://www.gamasutra.com/view/feature/2796/game_taxonomies_a_high_level_.php
25. Natkin, S., Vega, L.: A petri net model for computer games analysis. International Journal of Intelligent Games Simulation 3(1), 37–44 (2004)
26. Neil, K.: Game Design Tools: Time to Evaluate. In: Proceedings of DiGRA Nordic 2012 Conference: Local and Global Games in Culture and Society (2012)
27. Nelson, J.M., Mateas, M.: A Requirements Analysis for Videogame Design Support Tools. In: Proceedings of the 4th IC-FDG 2009, Orlando, Florida, USA, April 26-30 (2009)
28. Salen, K., Zimmerman, E.: Rules of Play: Game Design Fundamentals. MIT Press (2003)
29. Sicart, M.: Defining Game Mechanics. Game Studies. The International Journal of Computer Game Research 8(2) (December 2008), http://gamestudies.org/0802/articles/sicart
30. Smith, A.M., Nelson, M.J., Mateas, M.: LUDOCORE: A logical game engine for modeling videogames. In: CIG 2010 IEEE Symposium, pp. 91–98 (2010)
31. Tremblay, J., Schneider, K., Cheston, G.: Game Case Study, ch. 17. Software Development in an Object-Oriented Domain, University of Saskatchewan (2010)
32. Zagal, J.P., Mateas, M., Fernadez-Vara, C., Hochhalter, B., Lichti, N.: Towards an Onto-logical Language for Game Analysis. In: Proceedings of the DiGRA, Canada (June 2005)