Game design tools: Time to evaluate

Katharine Neil
Department of Screen & Media, Flinders University
Sturt Rd, Bedford Park – SA 5042, Australia
and
CEDRIC, Conservatoire Nationale des Arts et Métiers
292 rue Saint Martin – 75003 Paris, France
katharine.neil@gmail.com

ABSTRACT
The art form of the video game has a very idiosyncratic reliance on the process and practice of its designers. We work with creative and computational problems that form a web of deep complexity. And yet, as I have noticed in my professional practice as a game designer, we do not use tools to support our design process. For more than a decade, designers and researchers have argued for the development and use of both conceptual and concrete tools. To this end, formal and semi-formal game design models have been proposed and, more recently, experimental software-based tools have been developed by the research community. To date, however, none of these tools or models have been adopted into mainstream practice within the game design community.

In this paper I argue that it is difficult, if not methodologically flawed, to assess the work in the field of game design support without more qualitative data on how such tools fare in actual game design practice. Evaluation research would be an essential contribution towards answering the question of whether – and if so, how - these experimental formal models and tools can support and improve the game design process.

Keywords
Game design, design tools, ludocore, machinations, game atoms, game diagramming

INTRODUCTION
Game designers, unlike most other design practitioners, typically do not use tools to design. Dating back more than a decade, key game industry figures and game studies researchers have sporadically but consistently identified this as a problem. They have argued that formal, abstract tools for designing gameplay (be they conceptual models or software) must be developed if the craft of video game design is to attain desired levels of sophistication and creative expression.

Towards this goal, theoretical work has been undertaken to formalise and abstract game design techniques into formal models, to create taxonomies and to develop graphical notation systems (Björk and Holopainen 2005; Koster 2005; Araújo and Roque 2009; Natkin and Vega 2004; Bura 2006; Sicart 2008; Reyno and Cubel 2009; Cook 2007). Very recently there have been a few attempts by researchers to concretise some of these ideas into software tools, notably Machinations (Dormans 2009a) Ludocore (Smith, Nelson, and Mateas 2010), and Sketch-It-Up (Karakaya et al. 2009).

Proceedings of DiGRA Nordic 2012 Conference: Local and Global – Games in Culture and Society.

Whether and how these conceptual or concrete design tools can support the practice of game design, however, is not yet satisfactorily determined. Practical evaluation of this work must be undertaken if we are to move forward on the question of tools for game design.

GAME DESIGN WITHOUT TOOLS

The art form of the video game has a very idiosyncratic reliance on the process and practice of its designers; they work with creative and computational problems that form a web of deep complexity. This is both a blessing and a curse to game designers: the real-time algorithmic and interactive dynamics of play are often too complex to be modelled or evaluated successfully on paper, or even in the mind of a designer. While designers in other creative fields - be they film-writers who write a script, or composers who use a piano keyboard to approximate their orchestrations – arguably have meaningful ways with which to abstract, communicate and evaluate their ideas before taking them into production, game designers currently do not. Unlike a film, which can find analogues of itself in other linear media, a game is (at its core) a system; attempts to model and understand a system using a linear form (text, for example) are bound to be less fruitful.

Video game design process remains relatively underdeveloped compared to the sophistication of video game designs being produced today. As Stefan Grünvogel puts it:

> Game design as a craft has created a vast diversity of methodologies to balance interaction, game mechanics and audio-visual presentation for different game genres and players. But there are only very few attempts to support this process by using formal methods (Grünvogel 2005) .

Game developers themselves have made similar observations:

> Compared with the vast body of operational knowledge found in the world of filmmaking, the game design community is just beginning to articulate the concepts and techniques specific to our medium in order to establish methods of game design (Kreimeier 2003).

An academic and industry discourse has developed to highlight this absence in the field of game design, and specifically the lack of concrete and conceptual tools for game designers. Researchers and designers have noted that we lack game design support in the form of computer-aided design (as opposed to production) software, and formal or semi-formal design models and concepts (as distinct from heuristic approaches) that can support game design tasks.

The game designer's method: from word processor to prototype

In this section I describe and problematise the game design process as practised by professional practitioners, and survey how a need for tools and formal design methods has been expressed in the game design and research communities.

In the literature I am surveying, and within my own professional milieu within the mainstream game industry, 'game design' is used as a shorthand to mean the crafting of the core player experience ('gameplay' or 'rules'), rather than the visual, narrative or software components of a game. A 'game designer' can find themselves designing game missions (as a 'level designer'), writing narrative (as a 'game writer' or 'narrative designer'), designing control schemes and menu flow, and a variety of other tasks that

depend on the genre of game being made. There is, however, a core role that has to be performed no matter what kind of game the designer is attempting to create, and this is often simply referred to as 'game design': comprising the tasks of the conception, analysis, and balancing of gameplay.

Typically (i.e. in a standard commercial game development context) game designers perform these tasks using a combination of natural-language-based documentation followed by (or concurrent with) prototyping a playable version of selected elements of their design.

Documentation is considered to be game designer's primary task and manifestation of anything that could be called a design "method"[1]. A game design document (a 'GDD') or suite of documents is authored, which can run to hundreds of pages in length. This is created using word processing, spreadsheet and data flow (e.g. Microsoft Visio) software packages. Diagramming is most often used to map out certain details about user interfaces, narrative flow, maps, screen wireframes and statistics. While there are no standard models or visual vocabulary to express core gameplay concepts, individual designers sometimes create their own ways of diagramming or sketching their ideas visually 'on the fly' (Salen and Zimmerman 2003), often improvising different ways to express their ideas for different projects. Predominantly, however, the game mechanics, (the 'rules') are expressed in natural language.

Following documentation, a software prototype of the design is usually created by production staff (programmers, artists) under the direction of designers, who as designers lack the production skills required to effectively develop the prototype themselves. Game developers argue that prototyping is a critical part of the game development process, because it is considered to be the only reliable means to evaluate the quality of design ideas. Eric Zimmerman & Katie Salen, in their foundational game design text *Rules of Play*, assert that important questions such as "is the game accomplishing its design goals?" and "are [players] having fun?" can never be answered through conception and design alone; they can only be answered by play (Salen and Zimmerman 2003). In other words, games are too unpredictable to be imagined by the designer until they are created. As LeBlanc points out, games are what scientists know as complex systems (LeBlanc 1999). A complex system is a system that exhibits 'emergent' behaviour– behaviour that cannot be simply inferred from the system's rules. This means that the rules of a game alone are inadequate to describe the way the game works when played.

Yet the building of a game prototype is a process far removed from the exploratory, concept development stage of design where the designer is working alone noting her ideas down in a word processor. Software prototypes not only take time and cost resources to produce; prototyping is a process that is removed from the direct control of designers themselves. This leap from writing a document to building a prototype, therefore, is a significant and problematic one. While nobody would expect an architect, for example, to design a building with Microsoft Word, using natural language to communicate the layout of a building prior to construction, game designers are required to do something somewhat analogous to this. Or to use the analogy of game designer Raph Koster, «building a game off of a game design document is like trying to film a movie off of the director's commentary» (Sheffield 2007).

At least three game studies researchers or research teams have premised their work on the view that something is missing at the design stage of game development. One group has

claimed that a lack of methods and tools to help game designers support the 'ideational' stage of game creation has contributed to a lack of innovation in commercial game design. They explain this problem as stemming from the fact that the video game industry is "a production oriented industry and thus the majority of the technology in this industry has focused on production tools. Very little attention has been paid to the ideation stage in the game industry" (Agustin et al. 2007). This is echoed by Mark Nelson, a researcher whose PhD work is based on a similar observation that game designers "have no tools for reasoning about and visualizing systems of game mechanics" (Nelson and Mateas 2009).

Alternative methods

Damning the standard GDD-to-prototype process almost by implication, alternative prototyping methods have been advocated by game development educators and some designers. Their primary aim is to make the creation of game prototypes accessible to designers. These methods remain, however, imperfect solutions to the design problem.

One such method is paper and physical prototyping, inspired by the methods of board game designers. It usually involves using cheap materials like cardboard and plastic tokens as abstract representations of game elements to play out a 'real-world' prototype of the game where humans perform the role of the computer as well as the players. This 'analogue' style prototyping is encouraged as best practise by the major game design texts (Fullerton 2008; Salen and Zimmerman 2003). Publisher and developer Electronic Arts gives its internal designers workshops on physical prototyping methods (Fullerton 2008: 20).

Even the strongest advocates of physical prototyping, however, admit that it is a design method suited only to certain styles of games. Experienced paper prototyper Tyler Sigman observes that:

> Although there are some terrific reasons to make an analog prototype of a digital game, there are also some inherent limitations to such a process. The first, and biggest, is that there are some games for which analog prototyping just doesn't make sense. Case in point, games where a real-time action component is the sole mechanic (Sigman 2005).

Another designer-accessible prototyping method that has been made possible (within the independent and amateur sectors of the game industry in particular) is the use of simplified production tools, some of which have removed the need for the user to know any form of programming to create a simple game. Based on my own experience with these tools I have observed an unavoidable cost: the more these tools make game mechanics easier to implement, the more they sacrifice the user's, i.e. game designer's, creativity. This is because the compromises made to render a production tool easier to use render it less open or flexible, biasing it to a smaller subset of design choices. Nelson and Mateas also consider designer-accessible production tools to be an inadequate solution, observing that there are tools such as GameMaker and Alice to ease implementing games, but not any to help with designing game mechanics (M.J. Nelson & M. Mateas, 2008a).

Both of the above methods represent a kind of "making do" without the aid of programmers or software tools which could almost be seen more as a collection of survival strategies that designers have evolved, rather than a fit-for-purpose solution. Miguel Sicart observed – in reference to design documentation practices – that "most of [the game design literature that advises best-practice methods for documentation] is based

on tradition or a set of common practices more than on a research-based approach to the formal elements of games" (Sicart, 2008) . I would argue that physical prototyping methods also have legacy issues; like game design documentation, physical prototyping also evolved from out of a tradition - turn-based board games.

Arguably, the use of paper prototyping and simplified production tools by designers serves to further highlight rather than resolve the gap in the design tool chain that game designers have to deal with.

EXPLAINING THE LACK OF DESIGN TOOLS

The game industry has no shortage of software-based tools. Game artists, game programmers, level designers, quality assurance testers and project managers all use specific software – sometimes developed 'in-house' for the purposes of a single game project – tailored for doing their job. In an industry that has a strong tradition of developing bespoke tools, one has to wonder why software tools to support game design tasks are not built and widely used.

Seeking an explanation for this leads one to deeper problems within the field of game design. Game designers are not yet applying even purely conceptual "on paper" design tools or graphical notation systems to their design work. These conceptual tools and systems, evolved and confirmed in practise, would form the basis for any computer-aided design support software. (If architects, for example, had not yet devised a way to draft on paper then there would be little point in developing CAD technologies.) Given this poor state of disciplinary evolution, it is no surprise that the primary tool of a game designer is a word processor. Outside of playable contexts (a prototype, for example), the only means we have of modelling and communicating gameplay concepts has been natural language; we do not yet have a shared and commonly understood framework for designing games with anything other than words.

It could be argued that for certain elements of a game – narrative, character design, high-level concepts, for example – descriptive prose and illustrations (storyboarding, for instance) may be serviceable. For a key component of a game design, however, it is not. Defining this key component requires breaking down games themselves into their component parts. Salen and Zimmerman offer three sets of schema that can be used to frame games: rules, play and culture. Rules are the formal elements; "the inner, essential structures that constitute the real world objects known as games (Salen & Zimmerman, 2003: 80). Of all possible schema, the formal, systemic, structural aspects of a game design are arguably the least well served by natural language. This is problematic given that these are the aspects leading theorists consider[2] the defining characteristics that set video games apart from other media. "A game is a system … defined by rules..." (Salen & Zimmerman, 2003: 80).

As academic game studies as a discipline has developed an orientation to games as systems, it has begun to notice that the lack of a formal means of communicating the mechanics of these systems hinders both game analysis and design:

> Ludology, the study of games in general and videogames in particular, has pointed out the need to create models in order to explain the mechanics of games. This lack of a notation to precisely define games and game mechanics has been a traditional game design problem (Reyno & Cubel, 2009).

Even the language designers use has been criticised for not being as formal, standardised and precise as it could be. In 1999 designer Doug Church complained that we lack even a common vocabulary of terms to describe game concepts (Church, 1999). He added that this is a serious problem if we want to pass down and build upon knowledge from generation to generation of game designers, a lack of a common design vocabulary being "the primary inhibitor of design evolution" (Church, 1999). Educator Tracy Fullerton, in her game design textbook *Game Design Workshop*, also complained of this some years later, calling the lack of a single vocabulary "one of the largest problems facing the game industry today" (Fullerton, 2008: 44).

But even the design concepts and techniques that would form the building blocks of any such language are still in the process of being formalised, standardised and shared. Game design has for a long time been a kind of "dark art". Designer Dan Cook goes as far as to label past game design achievements as "accidental successes". He writes: "We currently build games through habit, guesswork and slavish devotion to pre-existing form." (Cook, 2007) A 2003 article by Kreimeier surveying the state of the art of game design method criticised game design texts of the time. These, it was argued, were so informal as to warrant being called "a kind of 'Discourse by Anecdote'", in which "game design experience is presented as a narrative, e.g., as a series of anecdotes and invented dialogs, sometimes as recommendations derived from interviews, or simply as annotated transcript" (Kreimeier, 2003).

Raph Koster is yet another high-profile designer who has argued for formalisation. In an influential presentation at the Game Developers' Conference (the industry's principal conference for developers) in 2005 entitled "A Grammar of Gameplay" (Koster, 2005), Koster highlighted the imprecision of natural language as a tool for designing gameplay and urged designers to develop a graphical notation system for game design. Two years later he repeated this call in an interview, saying "we want it, because god damn do design documents suck as a means of communicating game design" (Sheffield, 2007).

Designer Ben Cousins complained in 2004 that compared to other design disciplines, game designers lack formal training in their craft:

> the only difference here between other media and games is that every moviemaker, songwriter, painter and novelist is acutely aware, and often trained in, the application of the appropriate primary units. Game designers have not yet moved into that phase (Cousins, 2004).

Given this lack of expressing design ideas with any level of formality or abstraction, the goal of developing a software tool for game design seems akin to that of developing music notation tools for composers who cannot read music.

FORMALISATION TOWARDS MODELS AND TOOLS

Over the last ten to fifteen years the "discourse by anecdote" has begun to be replaced by some attempts towards comprehensively defining, analysing and describing technical game design concepts. General, foundational game design texts have been published that have attempted to systematically distil and refine game design theory and method: Tracy Fullerton's *Design Workshop*, Jesse Schell's *The Art of Game Design: A Book of Lenses*, but most notably Salen and Zimmerman's *Rules of Play*. Jarvinen (2008: 48) summarises and reflects upon some of these attempts, and their limitations:

> …in my experience most of the literature functions at its best on an inspirational level (e.g. Koster 2005), or is strongly design-orientated (Salen & Zimmerman 2004; Fullerton et al. 2004). These are important contributions as such, but they rely quite a lot on the reader's personal ability and experience to find practices and methods to transform the inspiration into concrete results – especially considering 'close analyses' of games (a term borrowed from study of literature and the arts).

Further to the limits Jarvinen notes, I would add that this work is more concerned with analysis or design wisdom, than with formal or semi-formal design methods.

Alongside these texts intended for general consumption, researchers and designers have also attempted more experimental work in the form of proposals for formalised, and sometimes visual methods of modelling and describing gameplay. Some have been explicitly conceived to be used as conceptual tools for game design.

> From Doug Church's 'Formal Abstract Design Tools' (1999) to Björk and Halopainen's 'Game Design Patterns' (2004) we have a wealth of frameworks that seek to develop a unified discourse among designers, to promote clarity, better game design, and a clearer procedural structure for designers in the creation of their games (Bojin, 2010).

These frameworks range from "semi-formal approaches" (Grünvogel, 2005) or "textual interpretations of game design practices" (Koster, 2005) to proposals for graphical modeling systems.

Approaches developed by researchers include *Patterns in Game Design* (Björk & Holopainen, 2005), Jarvinen's "library of game mechanics" (Jarvinen 2008), Hunicke, Leblanc, and Zubek's "Mechanics, Dynamics, Aesthetics" framework (Hunicke, Leblanc, & Zubek, 2004). Even Sicart suggests that his definition of game mechanics – summarised as "methods invoked by agents, designed for interaction with the game state" - be used practically "as a formal tool for describing and modifying mechanics in a coherent and comprehensive way" (Sicart 2008).

The game design community has developed a discourse around unit-based models and notation systems. Several designers believe that we need a 'grammar' or graphical notation system for modeling game design: to find the building blocks for this grammar we need to analytically break down games to their core units at the lowest level of structural granularity. These core elements are moment-to-moment player decisions and actions, which have been described by various theorists using metaphors that reference linguistics and chemistry: "verbs" (Crawford 2002: 62); "ludemes"; "atoms" (Cousins); and "choice molecules", with the last of these defined by Salen & Zimmerman as "action > outcome unit" which is "at the heart of interactive meaning", and from out of which larger interactive structures are built (Salen & Zimmerman, 2003: 63). Designer Dan Cook extends the game atoms idea towards modelling the elements of a game system from the point of view of the player's experience, arguing that "to accurately describe games, we need a working psychological model of the player". Cook's concept of a 'skill atom' "describes how the player gains a new skill"(Cook 2007); i.e. skill atoms describe the skills a player must progressively learn in order to master the game. Using diagramming, Cook suggests linking these atoms into "skill chains", structures that represent the order and context in which learning moments for new skills occur.

Researchers have also made calls, and in a few cases concrete attempts, to develop notation and graphical modelling systems to aid game design and analysis. This work has drawn on existing paradigms such as UML (Sicart 2008) and Petri nets (Araújo and Roque 2009; Natkin and Vega 2004; Bura, 2006).

Software-based design support

Very recently, a few researchers have embarked upon projects aiming to develop software tools to support game design.

Created as part of a post-graduate research project at Carnegie Mellon University, "Sketch-It-Up" is a set of processes and technologies designed to be used at the ideation stage of game design. Sketch-It-Up! builds upon the GameSketching system conceived by Dr. John Buchanan and his team in 2007 (Agustin, Chuang, Delgado, Ortega, Seaver, and Buchanan 2007a). Sketch-It-Up! allows a designer to model and rehearse the game flow at a high level: the number, order and difficulty of game interactions and what rewards are awarded to the player.(Karakaya et al. 2009) Its purpose, therefore, is to provide design support for linear, narrative-based games (Karakaya et al. 2009).

Joris Dormans tool Machinations gives designers a means of communicating and modelling "the structure of game systems and patterns that might be found in these structures" (Dormans 2009). It extends Rollings' resource flow model (Rollings and Adams 2003), implementing it in the form of a Petri-nets inspired real-time modelling and simulation environment. Its graphical editor allows a designer to model their game system and 'run' the simulation in real-time or faster than real time, revealing emergent dynamics of the system over time.

LUDOCORE (Smith, Nelson, and Mateas 2010) takes a similar approach, also aimed at giving designers the ability to model and simulating the dynamic behaviour of game systems. Its artificial intelligence based system also performs player modelling, and provides designers with the ability to query potential consequences of rule interactions.

Where are the game designers?

The involvement of respected practising game designers in this push towards developing models and formal approaches to game design suggests that designers themselves would, in theory, find such approaches useful to support their practice. But this should not be automatically assumed. It is important to recognise who these game designers are and who they represent: an elite minority of the game design community who meet and discuss their ideas regularly at events such as the industry's international Game Developers Conference (GDC) but also at exclusive invitation-only events such as "Project Horseshoe", an annual gathering of game designers dedicated to solving current problems in game design. Just as 'best practise' methods like paper prototyping may appear in game design textbooks but not in the workplace, the design ideas and methods described by what could be described as a kind of game design elite should not be taken as representative of the typical methods used by the average professional game designer.

If techniques proposed by these vanguard designers – such as Raph Koster with his "game grammar" and Dan Cook with "skill atoms"- are not filtering through into common design practise, we have also to consider why. One explanation could be that the ideas of the elite are more advanced than those of the majority, but we must also consider other reasons. It may be, for example, that the experimental, risk-taking concept work of creative directors like Raph Koster calls for quite different design methods to the

everyday craft work undertaken by a typical game designer. In 2008, Project Horseshoe published a report in which they appeared to consider this possibility, worrying over questions such as "are skill atoms a pragmatic tool for working designers?" and expressing a concern that although "in recent years a new set of tools for building games has emerged", they still remain hobbled by the problem that "most descriptions are highly theoretical and only considered useful to pointy headed academics or their mad inventors"(Blinn et al. 2008).

It is hard to measure the influence of frameworks suggested by these game designers and researchers upon practising game designers. Certainly in my own experience, while I have spoken to some designers who are aware of these theories I am not aware of any of my peers who use them in their practise. Even Raph Koster admits that he "can't yet picture designing a game" with his own notation system (Koster, 2005). The exception is perhaps the Mechanics, Dynamics Aesthetics theory of game design, which forms the basis of Mark LeBlanc's annual Game Design Workshop at the Game Developers' Conference. But no formal methods based on this analytical framework have been published to date, and I have not yet found written accounts by designers describing their use of the MDA framework in their practice.

Some argue that the design models proposed are as yet too underdeveloped to be used by designers. Designer Stephane Bura, who proposes a solution to what he describes as the "game diagramming problem", qualifies his contribution as an early attempt that needs further work before it can be of practical use. "Barring [additional work]," says Bura, "this grammar will remain a simple descriptive tool instead of an analytical or even a design tool" (Bura 2006).

Joris Dormans, while observing that none of the attempts to introduce formal models for game design have been so successful that they have become an industry or academic standard, attributes this to the fact that they "tend either to be too mathematical for the diverse population of game designers and scholars, or were not explored or presented with enough detail". He adds that, most importantly, they require designers to make an investment by learning a new paradigm – an investment unlikely to be made unless there is an "obvious return" in the form of a better, more efficient design process (Dormans 2009b). Given, however, the lack of published accounts of designers' experience with design models and tools, designers would have little idea about whether there would be a return on any investment in using them.

Dormans' own work, however, has enjoyed promising feedback so far. Though his work is very recent (his doctoral dissertation was published this year) he reports that there are professional developers who have noticed his work and have used and continue to use some of the tools he has created (Dormans 2012: 214). And in his dissertation he documents and describes some of the diagrams and game prototypes created by himself, his students and workshop participants using the toolset that he developed. These experiences sound very promising I am intrigued to know more: how are the tools used in a studio context, at what stages of design and production have they been useful, and so on. Now is surely the time for us to evaluate, explore and share the possibilities of Dormans' approach, and other approaches as well, before yet another tool project is launched that hopes to address this question.

THE NEED FOR EVALUATION RESEARCH

Coming to this research area as a game designer, my first instinct is to ask: will design support tools and methods aid our design work? If so, which tools work best for which of my design tasks? From a practitioner's perspective, evidence of how these theorised tools and methods fare when applied to real game design problems is crucial.

Even from a researcher's perspective it could be argued that work produced in this field cannot be comprehensively analysed, evaluated and compared through reading and thinking alone; research of this kind aims to address a practical problem, and therefore should be primarily assessed in relation to how it practically operates on that problem. A key motivation for this field of enquiry is, after all, a belief that formal design support tools and methods will improve design practice.

And yet we do not yet know if they will, nor if any one approach to this problem is more fruitful, in practice, than any other. This is because very little published work documents the experiences of practicing designers with these tools and formal techniques, a few of which have been public for over a decade. Game designers are not publishing their experiences in trade journals, nor have researchers yet embarked on thorough-going evaluation research of experimental design techniques and tools. Joris Dormans has said in relation to his Machinations design tool that "it remains to be determined how easy it is for designers" (Dormans 2011). And while the developers of "Sketch-It-Up", a tool developed until 2009 at Carnegie Mellon University, workshopped their tool with children, there is no discussion of how the tool fares in the professional design contexts it was built to support (Karakaya et al. 2009).

In all, we have little upon which to base real insight into these tools from a practitioner's point of view, let alone enough empirical support to help resolve the question of whether formalising the game design process with tools and formal models really would aid or improve game design practice. At least one influential designer has expressed doubts as to the potential usefulness of formalisation and abstraction for game design (Schell, 2008: 145), and thus far we can furnish little evidence to counter such scepticism.

Even at a purely theoretical level, the field lacks work that provides comprehensive comparative analysis. While all these proposed design support solutions tend to share core foundational elements – a view, for example, that conceptual design models or concrete tools be formal, abstract and/or graphical, and that they approach games as complex systems and reveal emergent dynamics, and so on – their approaches and results differ. Disappointingly, there is insufficient evidence of dialogue and debate, either between practitioners and researchers or between researchers themselves.

CONCLUSION

Research in the area of game design support has been emerging for over ten years now. Thus far, however, there has been little discussion or comparative critique of this work – a fact that is unsurprising, given that it would be very hard for anyone to develop an honest critique of design models and tools without first undertaking the not insignificant task of evaluating their function in practice. Even the question of how such evaluation might be undertaken opens up difficult and interesting terrain for discussion.

It even seems somewhat pointless for the research community to make further attempts at devising new design tools and models without ensuring practical evaluation the work to date. Without evaluation, we cannot assess our progress and move forward. We need

real data to work with: published evaluation research that can inform future advances in this emerging field of game design support. Published evaluation research would also be important step towards the potential adoption of the results of game design support research by the wider game design community itself. These tools, in some form, could one day change the way we design games, but we will never really know until we pick them up and use them.

ENDNOTES

1 Possibly the earliest attempt to define a game design method is the game design document (Kreimeier 2003).

2 It is worth noting, however, that such definitions have become controversial and been characterised as symptomatic of what critics label 'proceduralism'(Sicart 2011)

BIBLIOGRAPHY

Agustin, Michael, Gina Chuang, Albith Delgado, Anthony Ortega, Josh Seaver, and John W. Buchanan. 2007. "Game sketching." *Proceedings of the 2nd international conference on Digital interactive media in entertainment and arts - DIMEA '07*: 36. doi:10.1145/1306813.1306829.

Araújo, Manuel, and Roque. 2009. Modeling Games with Petri Nets. In *Breaking New Ground Innovation in Games Play Practice and Theory Proceedings of the 2009 Digital Games Research Association Conference*, ed. Barry Atkins, Helen Kennedy, and Tanya Krzywinska. Brunel University.

Blinn, Scott, Stephane Bura, Nicholas Fortugno, Scott Brodie, Daniel Cook, and Victor Jimenez. 2008. *Group Report: Multiplayer Game Atoms*. *The Third Annual Game Design Think Tank Project Horseshoe 2008*. The Avallone Media Group.

Bura, Stephane. 2006. Game Diagrams. http://users.skynet.be/bura/diagrams/.

Cook, Dan. 2007. Gamasutra - Features - The Chemistry Of Game Design. *Gamasutra*. http://www.gamasutra.com/view/feature/1524/the_chemistry_of_game_design.php.

Dormans, Joris. 2009a. "Machinations: Elemental Feedback Patterns for Game Design." Ed. Joseph Saur and Margret Loper. *North* (2009): 1-7.

———. 2009b. Machinations: Elemental feedback structures for game design. In *Proceedings of the GAMEON-NA Conference*, 20:33–40.

———. 2011. "Simulating Mechanics to Study Emergence in Games." *Artificial Intelligence*: 2-7.

———. 2012. Engineering emergence: applied theory for game design. PhD diss., University of Amsterdam.

Fullerton, Tracy. 2008. *Game Design Workshop: A Playcentric Approach to Creating Innovative Games*. Elsevier Morgan Kaufmann.

Grünvogel, Stefan M. 2005. "Formal models and game design." *Game Studies* 5 (1): 1-9.

Jarvinen, A. 2008. "Games Without Frontiers: Theories and Methods for Game Studies and Design." *presented as PhD thesis in University of Tampere*, 7.

Karakaya, Bulut, C. Garcia, Daniel Rodriguez, M. Nityanandam, N. Labeikovsky, and T. Al Tamimi. 2009. "Sketch-It-Up! Demo." *Entertainment Computing–ICEC 2009*: 313–314.

Kreimeier, Bernd. 2003. Game Design Methods: A 2003 Survey. *Gamasutra*. http://www.gamasutra.com/view/feature/2892/game_design_methods_a_2003_survey.php#1.
LeBlanc, Matt. 1999. Formal Design Tools: Feedback Systems and the Dramatic Structure of Competition. In *Computer Game Developers' Conference*.
Natkin, S, and L Vega. 2004. "A petri net model for computer games analysis." *International Journal of Intelligent Games Simulation* 3 (1): 37-44.
Nelson, M.J., and M. Mateas. 2009. A requirements analysis for videogame design support tools. In *Proceedings of the 4th International Conference on Foundations of Digital Games*, 137–144. ACM.
Rollings, Andrew, and Ernest Adams. 2003. *Andrew Rollings and Ernest Adams on Game Design*. New Riders Games.
Salen, K, and E Zimmerman. 2003. *Rules of Play: Game Design Fundamentals*. *MIT Press*. MIT Press.
Schell, J. 2008. *The Art of Game Design: A book of lenses*. *Annals of Physics*. Vol. 54. Morgan Kaufmann.
Sheffield, Brandon. 2007. Defining Games: Raph Koster's Game Grammar. *Gamasutra*. http://www.gamasutra.com/view/feature/1979/defining_games_raph_kosters_game_.php.
Sicart, Miguel. 2008. "Defining Game Mechanics." *Game Studies* 8 (2): 1-18.
———. 2011. "Game Studies - Against Procedurality." *Game Studies* 11 (3).
Sigman, Tyler. 2005. The Siren Song of the Paper Cutter: Tips and Tricks from the Trenches of Paper Prototyping. *Gamasutra*. http://www.gamasutra.com/view/feature/2403/the_siren_song_of_the_paper_.php.
Smith, A.M., M.J. Nelson, and M. Mateas. 2010. LUDOCORE: A logical game engine for modeling videogames. In *Computational Intelligence and Games (CIG), 2010 IEEE Symposium on*, 91–98. IEEE. doi:10.1109/ITW.2010.5593368.