The Conceptual Relationship Model
Understanding patterns and mechanics in game design

Carl Magnus Olsson

Malmö University
Department of Computer Science
205 06 Malmö, Sweden
+46 40-665 75 02
carl.magnus.olsson@mah.se

Staffan Björk[1], Steve Dahlskog[2]

University of Gothenburg[1], Malmö University[2]
Department of Applied IT[1], Department of Computer Science[2]
412 96 Göteborg[1], 205 06 Malmö, Sweden[2]
+46 31-772 10 39, +46 40-665 71 27
staffan.bjork@chalmers.se, steve.dahlskog@mah.se

ABSTRACT

Rooted in the complexity of purposeful design, this paper embraces a phenomenological perspective of design as both a process and artifact. We use this perspective to interpret why the conceptualization and realization of design intentions can be difficult to achieve and why design is often perceived as a so called 'wicked problem'. This paper revisits the concepts of game design patterns and game mechanics, arguing that refactoring these concepts is needed to clarify their relationships and motivations. We outline the separation of concerns between them and suggest that an additional contextualizing layer should be added to the discourse. Using this, we define and reflect upon what we refer to as the conceptual relationship model.

Keywords

Design patterns, Contextualization, Game mechanics, Intentionality, Phenomenology

INTRODUCTION

Discussing design is an inherently complex task. Design has previously been recognized as an example of a 'wicked problem' (Mateas and Stern 2005; Stolterman 2008), i.e. a problem with multiple plausible solutions as well as multiple subjective interpretations of such solutions. This is partially rooted in the common need to make design decisions despite an incomplete understanding of the problem situation at hand. Design is phenomenological by nature (Dourish 2001) as interpretations of needs and suggested designs are subjective. The designs themselves act as a medium for discussion through which designers may reflect upon their interpretations and intentions, and through which users interact with the design in the way they interpret it. Such user interactions are therefore not under the direct control of designers, and may clash with designer intentions and result in unintended consequences as use drifts away from the designed intentionality (Ciborra 1996; Orlikowski 1992).

In order to view the design as a medium for discussion, a shared language between the involved actors must be agreed upon. Within game research, two influential concepts for such discourse are game design patterns (Björk and Holopainen 2004; Holopainen 2011) and game mechanics (Järvinen 2010; Sicart 2008). Game design patterns have be defined as *"semi-formal inter-dependent descriptions of commonly reoccurring parts of the design of a game that concern gameplay"* (Björk and Holopainen 2004) and examples include 'alarms', 'fog of war', and 'grinding'. Meanwhile, game mechanics have been defined as *"methods invoked by agents, designed for interaction with the game state"* (Sicart 2008) and examples include 'climb', 'grab', 'take cover' and 'shoot'. Unfortunately, semantically similar game design patterns and game mechanics – e.g. 'aim and shoot' vs. 'aiming and shooting', 'area control' vs. 'conquer', and 'bidding' vs. 'bidding' – means that the concerns of the concepts are often difficult to separate.

As is common in any theory development, a certain time for trial-and-adaptation is to be expected before concepts settle down in terms of shared and accepted meaning, as well as their relationships with other concepts. The purpose of this paper is therefore to revisit the present discourse on game design patterns and game mechanics and develop a conceptual relationship model that clarifies the separation of concerns between them. As part of this model, we formulate the hypothesis that the lack of an explicit relationship perspective has led to a design language gap forming between game design patterns and game mechanics – a gap which we suggest a contextualization layer to fill.

RELATED WORK

In this section, we start by elaborating on the inherent complexity of design which we referred to in the introduction. We then focus on presenting our two key concepts: game design patterns and game mechanics, including how they have been used and to what purposes.

Design Complexity

This paper started by arguing that design is inherently complex. This begs (at least) two questions: why is design inherently complex, and what ways exist to negotiate the complexity? Wanting to describe the complexity of changing existing practices, Rittel et al. (1973) introduced the concept of wicked problems. These are problems encountered by professionals planning changes of practices and differ from the 'tame' problems of engineering and natural science in several respects: they are each unique, difficult to define formally and may be seen as symptoms of other problems. Solutions to wicked problems cannot be enumerated and the validity of solutions can only be measured subjectively. They do not have natural stopping rules and every test of a solution affects the end users. Finally, as Rittel et al. (1973) argues, the planner has 'no right to be wrong'. Design situations have been described as an example of wicked problems (Stolterman 2008), including specifically game design (Mateas and Stern 2005). One reason for this is that game designers in most cases wish to provide novel experiences and therefore need to be noticeably different from previous games. Another reason, beyond what Mateas and Stern note, is that game design focuses more upon creating activities and experiences than artifacts, which explains why game design has been described as a second-order design problem (Salen and Zimmerman 2004). This implies that while a game designer may have certain activities or experiences as the design goal, these can only indirectly be inscribed into a designed (game) artifact. Therefore, game designers are forced to rely on inscribed cues to encourage players to act in ways that make them experience the intention behind the design.

How can one deal with wicked problems? It may seem natural that Rittel et al. (1973) did not consider general solutions to this, given the nature of wicked problems as not holding generic solutions. Roberts (2000), however, proposes three main strategies for dealing with them: restricting decision makers to a selected few options, creating different groups whose solutions are compared with each other, and making all stakeholders involved in the creation of a solution. Still, these strategies assume that there is a way of communicating about specific design issues (regarding requirements, judgment criteria, and ways of discussing options respectively), meaning this is a fundamental need for working with wicked problems. In the context of software engineering, Rheinfrank and Evenson (1996) describe this as design languages, noting that specific design disciplines develop these based on three defining characteristics: a collection of available design elements, information about how these can be structured or used together, and descriptions of when specific elements or combinations are suitable or problematic.

The need of being able to express game design knowledge explicitly and clearly has been argued from professional game designers (Costikyan 2002; Koster 2004; Spector 1999). The attempts done so far (e.g. Church 1999; Falstein 2002) have however been limited in scope or depth – likely due to these pursuits having to negotiate the designers work with creating games in a commercial setting. The Mechanics-Dynamics-Aesthetics (MDA) model (Hunicke et al. 2004) is probably the most widely adopted industry driven work, but unfortunately has not been developed significantly since 2004 (Sankalia 2013). The MDA model furthermore does not meet all qualifications of a design language, as it is primarily a three-level categorization that argues that game elements give rise to activities which in turn gives rise to experiences. This passes responsibility for development of design languages for games to researchers, a sentiment echoed by Mateas and Stern (2005, p. 8): *"One of the roles of game studies can be to help map game design space, to develop tools, analyses and languages for navigating this space."*

Game Design Patterns
The concept of design patterns originates from architecture where it was developed to empower the people that would live and work in buildings to affect the development of buildings by giving them a language they could share with the architects (Alexander et al. 1977). Focusing on being practical, a pattern *"describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice"* (Alexander et al. 1977, p. x). Importantly, the solution to a pattern typically includes showing the causal relationship between it and one or more other patterns that are already more precisely defined. Design patterns have also been widely adopted in software engineering through the work by Gamma et al. (2001). Defining a pattern in terms of the relation to other patterns is common there as well.

Design patterns for game design were first introduced by Kreimeier (2002) in a commercial setting. This was soon followed by work from research (Björk et al. 2003), including a collection of nearly 300 patterns (Björk and Holopainen 2004). This collection has been the basis for several forms of analysis, e.g. exploring possibilities for anonymity in board games (Linderoth 2011) and upholding cooperation in mediated gameplay (Linderoth et al. 2012). It has since been updated with a specific focus on patterns related to team work (Bergström et al. 2010), characters (Lankoski 2010), game dialogues (Brusk and Björk 2009), and pottering (Lundgren and Björk 2012). Several other, stand-alone collections of game design patterns have also been created, e.g. Hullett

and Whitehead (2010), Lewis et al. 2012, Milam and Seif El Nasr (2010), and Smith et al. (2011).

In the following text, we have opted to refer to game design patterns in the way Björk and Holopainen did in their book from 2004. They in later work have started to refer to their pattern framework as a gameplay design pattern in order to be more precise in what type of patterns they consider. This has been motivated by the game design process needing to consider many types of design patterns, including but not limited to those concerning software development, user interface, and narration. While we rely heavily on Björk and Holopainen, we are using the more established name game design patterns as an indication of our open stance towards other types and collections of design patterns concerning games.

Game Mechanics

Just as with game design patterns, game mechanics are referred to in many different ways. In the academic design discourse, the term has been explored several times, e.g. by Hunicke et al. (2004), Järvinen (2008), Lundgren and Björk (2003), and Sicart (2008). The earlier mentioned MDA model was proposed as a formal approach to apply to a complete design process for games. In this model, game mechanics are seen as *"actions, behaviors and control mechanisms afforded to the player within a game context"* (Hunicke et al. 2004, p. 3). Meanwhile, Lundgren and Björk (2003, p. 48) define game mechanics as *"any part of the rule system of a game that covers one, and only one, possible kind of interaction that takes place during the game, be it general or specific"*. Two central aspects of this form of definition have been highlighted by others. Järvinen (2008) notes the central role that interaction plays, while Sicart (2008) and Jorgensen (2014) emphasize the relationship with rules. Järvinen (2008) and Sicart (2008) both view game mechanics as a tool for analyzing games, while Järvinen specifically suggests mechanics are actions for the player and that they thus ideally are described as verbs rooted in these rules. However, Järvinen (2008, p. 254) maintains that mechanics should also be working as *"means to guide the player and the game into particular behaviour by constraining the space of possible plans to attain goals"*.

Although Sicart (2008) recognizes Järvinen's approach as thorough, he argues that it is too deterministic in nature, as Järvinen appears to suggest that mechanics exist in order for goals to be achieved. Sicart contests that game mechanics do not necessarily need to be deterministic, and instead presents what he argues as a more open definition: *"[G]ame mechanics are methods invoked by agents, designed for interaction with the game state"*. He elaborates on this to suggest that mechanics are available to any entity acting as an agent in the game, including agents controlled by AI. Mechanics are therefore present-at-hand to interact with in both planned (i.e. goal driven) and unplanned ways. Expanding on an idea by Salen and Zimmerman (2004), Sicart (2008) also discusses compound mechanics. Sicart defines these as *"a set of related game mechanics that function together within one delimited agent interaction mode. These modes are defined by the interaction of these different modalities: as such, the driving compound mechanic is composed by a set of mechanics interrelated to provide a relatively accurate model of driving."*

Game Design Patterns vs. Game Mechanics

At this point, we may start to compare the two core constructs of this theoretical reflection. It is notable that simply looking at examples of patterns and mechanics, e.g. the patterns collection by Björk and Holopainen (2004) and the mechanics collection by

Järvinen (2008), does not provide an easy way of separating the concerns of the concepts. Although one can find patterns on high levels of abstraction from an actual concrete design (e.g. 'luck' and 'spectacular failure enjoyment') which are not present in Järvinen's collection, there are still many very similar elements, e.g. 'aim and shoot' vs. 'aiming and shooting', 'area control' vs. 'conquer', 'bidding' vs. 'bidding', 'gain ownership' vs. 'controlling', and 'herd' vs. 'herding'. The collections thus significantly overlap semantically and simply trying to create a dichotomy based on level of abstraction thus does not work.

Järvinen (2008, p. 384) argues that there is a question of intention that separates the concepts as *"the design patterns approach is a design-oriented problem-solving method, whereas the mechanics approach introduced here is analysis-oriented – with design consequences if one so desires"*. Although this may originally have been the case, documented use of gameplay design patterns in research publications show that they have primarily been to analyze games (cf. Bergström et al. 2010; Hullett and Whitehead 2010; Lankoski 2010; Lewis et al. 2012; Milam and Seif El Nasr 2010; Smith et al. 2011), while Järvinen has presented design insight using mechanics in game development communities (e.g. Järvinen 2010). It therefore seems more appropriate to describe that both patterns and mechanics can be used for design as well as analysis. Being possible to use something for the latter when one can do so for the former is within design a natural and important part of reflection-in-action (Schön 1983) which is implicit to the iterative design and evaluation process that is common in game design (Fullerton 2008).

Järvinen (2008) also proposes a second difference between patterns and mechanics. Here he differentiates based on what context they are intended to be used in, stating that *"[w]hereas in the design patterns thinking games are analysed with the purpose of detecting patterns within the game dynamics, and formalising them in order to create tools for designing certain kind of gameplay (i.e. dynamics), analysing mechanics focuses on detecting specific combinations of game elements and the combinations' consequences during game dynamics"* (Järvinen 2008, p. 383). This is similar to the previous argument regarding intention of designing or analyzing, but an important difference lies in the identification of patterns as something which de-contextualizes them, while the focus of mechanics is how they function in a specific context. That patterns are de-contextualized should however not be interpreted as being independent from each other. In fact, Björk and Holopainen (2004) describe five different types of relations between patterns. Identified relations have been used to drive the argument in several research papers (e.g. Bergström et al. 2010; Lundgren and Björk 2012). The relations between patterns are probably best understood as general consequences or possible combinations, rather than hard links. While these relations may occur in the mechanics that instantiate the patterns, the relations are not guaranteed to be seen by players. In addition, many of the relations found between specific mechanics in a game do not necessarily translate into relations in their pattern counterparts, since the enacted relations between mechanics depend on specific combinations of game elements beyond the descriptiveness that patterns allow.

Before Järvinen (2008) and Sicart (2008) had published their definitions of mechanics, Lundgren and Björk (2003) argued that patterns may be more useful than game mechanics as mechanics lack descriptions of relations and consequences were problematic. Both the work of Järvinen and Sicart refute this, however. Järvinen instead stresses detecting specific combinations of game elements in order to find mechanics, while the definition by Sicart suggests they act as a tool for the discovery, description, and interrelation between game mechanics in games. This implies that an important

aspect of game mechanics is the relations they have to each other. Sicart's emphasis on interrelation can further be interpreted as viewing mechanics as the relationship between game elements. It is also noteworthy how Järvinen and Sicart both stress the context or specificity of game elements in identifying game mechanics. A strict interpretation of this suggests that mechanics found in a game, and their relations, only exist in that game. Other games may, however, have mechanics similar enough that we recognize them as close. However, given that even a minor change in a game design may have considerable effects on the gameplay experience it is problematic to assume that mechanics found in two different games would have similar relations. This leads to the conclusion that mechanics mentioned out of context of specific games, as for example done in Järvinen (2008), must be treated with care as they refer to contextually different mechanics that have a family resemblance in a more Wittgenstein (1953) sense rather than literally. Or, in other terms, as the mechanics become abstracted out of their context, their impact on gameplay and relational relevance loses value as the rich meaning tied to the specific context is lost.

TOWARDS A CONCEPTUAL RELATIONSHIP MODEL

In this section we present three parts, starting with the introduction of a contextualization layer to act as a mediating surface between game design patterns and game mechanics. We then move on by considering the areas of concern for players, designers, developers, and researchers on our path towards a conceptual relationship model. This model is presented in the final part of this section, but one thing is worthwhile noting before reading on: We have chosen to present this section with an undertone of something *being designed*, but the conceptual relationship model – and each part in it – is also feasible to use as tools for analysis and reflection of *already completed designs*.

The Need for Contextualization

Outside game research, design patterns have been widely discussed particularly within software engineering, largely based on the seminal work by Gamma et al. (2001). The prescriptive stance of such *software* design patterns have, however, been contested as not useful for professional game designers (Dormans 2013). This is rooted in software design patterns dealing with the structure of code, rather than striving to capture the design elements and *in situ* effect of them during use. This leaves us with game design patterns and game mechanics as our primary tools for a game design discourse. As discussed earlier, this discourse presently lacks a fundamental model for the conceptual relationship between them, aside from their shared goal to identify a generalizable description language for game design. An effect of striving towards generalization and description of the design – and the *in situ* effect of the design – is that neither design language captures intentionality behind the design choices. Intentionality relates not only to the designer's intentionality, however, but also the developer's interpretation of this intentionality and expression of this in terms of code, as well as the player's interpretation of intentionality during gameplay. If we want to contribute with best practice examples, we need to expand our design language to also capture intentionality.

From a phenomenological perspective, Brentano (1874) describes intentionality as capturing the direction of meaning, and represents that which distinguishes conscious thought from physical or mechanical acts in the world (Dourish 2001). Brentano positions that intentionality captures meaning from the relationship between two entities (such as two agents, but could also be thoughts, expressions, or objects), as this relationship is reflected on in the mental mind. Dourish raises an interesting question however, regarding how some things can carry great social meaning if intentionality is a purely

mental phenomenon of the individual? To deal with this, Dourish proposes that two forms of intentionality exist. First, original intentionality expresses the ability of conscious creatures to mean things with their actions, while the second – derived intentionality – is reached through the interpretation of some action performed by another. Derived intentionality, which Dourish as a researcher of design is more concerned with, is the product of interpretations that a third party makes based on what this third party has observed. This means that Dourish takes inspiration from Dennett (1987), who argues that all intentionality is derived. Intentionality therefore impacts game design as it offers an explanation as to why design decisions – which we earlier explained were difficult to make based on the wicked problem situations a designer often operate in – are also inherently difficult to interpret by others.

This means that those that base their work on design decisions that others have made must derive their own meaning from the decisions, and then attempt to act upon them. For developers, such action is expressed through code, which requires a shift in terms of language also, with the goal to promote the same intentionality to the player. Implemented design cues are thus acts where intentionality – in a somewhat more everyday meaning – is purposefully inscribed into the design, hoping that the same intentionality – in the more phenomenological meaning – is also consciously understood, or at least subconsciously enacted within the spirit of the design intentionality, during use. Implicitly, intentionality also suggests that design and development is likely to drift and that it is impossible to counteract this through formalization and ideals of complete specification since both these are constructs of intentionality. This does not, however, imply that one should simply give up attempts to collaboratively design and develop. Without purposeful reflection on interpretations and a continuous design dialogue with related parties about these interpretations of intentionality, designs would simply fall apart. In fact, we subscribe to the notion of emergent design proposed by Gasson (1997), i.e. fusing the idea of emergent strategy (Mintzberg and Waters 1985) with the argument that design problems and goals emerge from actively engaging in design activity (cf. Hutchins 1990; 1995) rather than as a part of planning.

Active engagement between several different participants in a design process requires means of communication. Design languages can support this, but do not directly help inter-discipline communication since these use different design languages with only partial overlap. Game design patterns and game mechanics are examples of partly overlapping concepts of design languages. Designers and developers arguably use these even if it is implicitly through their tacit understanding of them. Furthermore, as players, reviewers, and game researchers also use these, we see an opportunity to suggest a model that explicates the relationship between design patterns and mechanics, through the artifact as it is implemented in code, from an intentionality perspective. This provides stakeholders with a model that bridges their areas of concern while at the same time allows them to maintain their established design languages. The foundation for this bridge is the dependency on intentionality while designing as well as analyzing designs.

We argue that design intentionality is best understood by adding a layer of contextualization to the design language of game research (see Table 1). The contextualization layer acts as a lens through which generalized patterns are situated in new designs. During contextualization, scenarios are formed to show intended use. Ideally, such scenarios are representative samples of the full context and therefore contain a mixed focus between situation specific intended use as well as shared underlying motivations between the situation specific scenarios (or simply put, why this is a good

idea for the game). Viewed from this perspective, we in practice view the scenarios as a pattern contextualization tool.

At the same time, the contextualization layer also acts as a lens through which developers understand the intentionality behind the outlined game mechanics. The layer may therefore in practice also be viewed as a tool to analyze mechanics and judge if they successfully support design goals. The duality this creates is the motivation for why we view contextualization as a separate layer in itself, but we do so while underlining that the value of the layer is tied closely to both the generalized patterns and the instanced mechanics. In fact, describing contextualization without patterns or mechanics becomes an exercise in futility as context by default includes the perspective from which it is viewed and a description of what this 'it' is.

		Content	Stance
Highest	**Patterns**	Descriptive	Neutral language
	Contextualization	Design goals	Situation specific design direction
	Mechanics	Interaction realm	Actionable components
Lowest	**Code**	Game implementation	Implemented functionality

(Abstraction — arrow pointing upward on left side)

Table 1: Content and stance for different abstraction levels of game design.

Influenced by the rich (and phenomenological) notion of context held by for instance Dourish (2001), we understand context as concerned with space, place, and locales. This implies that the separation between space and place is essentially between the physical and the social, while locales – according to Giddens (1984) – refers to the use of space as providing the setting for interaction. Locales subsequently may hint towards physical properties as the primary way to describe them, but this does not capture the emerging social elements involved in human interaction with others and the physical space they occupy. Locales are subsequently where space and place meet in (and over) time. The effect of our view on contextualization as locales leads us to a chance to redefine game design patterns and game mechanics, in terms of space and place. This means that gameplay interaction may be viewed as expressions of enacted game design patterns, or in the words of Giddens, the social context formed by the 'place' metaphor. Similarly, game mechanics may then be viewed as describing gameplay interaction through the objects that the game mechanics define. This implies game mechanics as providing the physical context, i.e. the space metaphor. In our discussion – and through the conceptual relationship model we present below – we will return to and expand upon the impact this has on the present research stream.

In relation to software design patterns (Gamma et al. 2001), the closest approximation to a software patterns perspective of Table 1 would be to say that software design patterns consist of vertical slices from each layer – starting at the top and cutting all the way down into specific code (or best practice examples of code, to be more precise). Furthermore, software design patterns are typically identified through review of a larger code base – commonly spanning several related products – to identify similarities between components. These similarities are then reviewed together with the related mechanics, contextualization and generalized category that this slice belongs to, in order to identify best practice suggestions for future design and development. Contextualization, on the other hand, deals with descriptions of the intentionality behind a specific design. A future

research opportunity could be to explore the potential for software patterns as a driver for appropriately inscribing design cue intentionality.

Area of Concern and Basis for Use

A relevant question to ask at this point is: how does the suggested view on patterns, contextualization, mechanics, and code affect the primary (human) actors? For the scope of this paper, we view players, designers, developers, and researchers as the primary actors, but the review could of course be expanded to add further actors. In Table 2, we present a mapping in terms of focus and frequency of use for each actor, as well as highlight the primary (darker color) and secondary concerns (lighter color) that the mapping suggests. Tertiary concerns (white color), while not as central as the primary or secondary, still play an important role for each actor, as will be elaborated on below.

	Player		Designer		Developer		Researcher	
	Focus	Frequency	Focus	Frequency	Focus	Frequency	Focus	Frequency
Patterns	Implicit	Low	Explicit	Varies	Implicit	Low	Explicit	High
Contextualization	Explicit	Varies	Explicit	High	Explicit	Varies	Explicit	High
Mechanics	Explicit	High	Explicit	High	Explicit	High	Explicit	Varies
Code	Implicit	High	Implicit	Low	Explicit	High	Implicit	Low

Table 2: Areas of concern for different actors related to game design.

The player concern is primarily task-oriented and subsequently holds mechanics as central. However, there is interplay within a specific game between the game results – as dictated by the implemented code – and the contextualized expectations of intention. In the case of new games and novel game features that emerge during gameplay, players use game design patterns primarily as a tool for exploration and interpretation of the gameplay in relation to their previous experiences with games. Based on the varying degree of understanding and awareness for the mental models of patterns that players have, we view their focus on patterns as implicit rather than explicit (as that would imply a more structured and conscious approach). Due to the important role of first impressions, gameplay design patterns and the impact of these are of vital importance to games as a design artifact. Without a favorable first impression, it is likely that players simply stop playing the game and move on to another which they interpret as more rewarding, despite not fully exploring the game.

The designer concern is primarily focused on qualitative interpretations of intentionality when moving from envisioned design goals to a (sufficiently concrete) descriptive level which allows developers to translate mechanics into code. Skillful design is subsequently as much about the ability to creatively design, as it is about ensuring that developers' intentionality are in line with the design goals. Contrary to players, designers explicitly study and break down other games to identify the patterns they are made up of. In practice, such study serves as the frame of reference for the design that is being developed – as well as for inspiration to new games and the patterns they include – in order to ensure that the design does not drift from the envisioned product without a purposeful decision to do so. Typically, designers deal with code only implicitly, through mechanics and contextualization rather than with actual code impact. They are, however, likely to be among the first in internal playtesting of key mechanics – and greatly interested in the outcome of systematic playtesting by dedicated playtesters – once the

mechanics have reach a testable stage. This implies only an implicit relationship with code, and even then on a somewhat lower frequency.

Hardly surprising, the developer concern lies primarily in explicitly working with game mechanics to translate these into appropriate and effective code. Their work is not without interpretation however, as the mental models of designers and developers (as well as eventual end-users) may vary greatly. In itself, this is not unique to game development as it is well-established in other domains of software design and development also (cf. Olsson 2011; Orlikowski 1992). An argument can be made that this is the central reason for why a shared design language is relevant to cultivate. A shared language does not automatically solve this problem however, as mental models – from a phenomenological perspective – are different for each individual based on their ontological understanding of the world (Heidegger 1927). This implies that the solution is not in *having* a language, but rather in *using it as a foundation for communication to achieve a shared understanding*.

The study of shared meaning can be traced back to Schutz (1932), who suggested intersubjectivity as a focus-point for shared meaning creation. Intersubjectivity, Schutz argues, contains two parts: Firstly, intersubjectivity is the medium through which the communication takes place, where the medium in the design language we have used in this paper may be the use scenarios, prototypes, or similar artifacts for understanding the intentionality this elicits, as well as the format that is used (speech, writing, drawing, gestures, etc.). Secondly, intersubjectivity is the shared ways of using the medium that emerges during communication. This second point is in fact why we continuously stress the word 'shared' in this paper as this is a purposeful clue towards our reliance on Schutz. The challenge of developing shared meaning has received extensive attention in research. Nevertheless, assumptions and unintended consequences of designs (as embodied in the developed artifacts), due to breakdowns in communication between designers, developers, and users, remains to this day one of the most complex issues in software product development.

As a guide to interpretations, and to lend direction as to which assumptions are likely to be appropriate to make, developers lean primarily upon descriptions of contextualization where the mechanics may be viewed in terms of their use scenarios. Implicitly, game design patterns (from previous development as well as experience as a player of other games) inevitably affect how developers make sense of the use scenarios and where no use scenarios exist for a mechanic. This means that developers – however much they may argue that they 'implemented what they are told to' – always add their own assumptions and interpretations for how to effectively convey, through coded design cues, the contextualized intentionality. In this regard, an argument can be made that developers need to be at least as skilled in critical reflection of their mental models and the impact these have on their code, as designers have to be in order to suggest new designs that are (possible to implement and) rewarding to players.

The design researcher concern is primarily in the borderline between generalizable patterns and relationships between such patterns in much the same way as has been outlined by Björk and Holopainen (2004). Granted, this is largely tied to the way in which players start interpreting new designs through the game design patterns they already know and then gradually move towards a lower abstraction layer, such as the contextualization within a specific product and impact from this on the gameplay perception. We subsequently assume that design researchers by default are interested in

the impact on gameplay – as perceived by players – and not only the design process on the product development side. Subsequently, driven by ideals of generalizable results, we fully understand that the discourse amongst design researchers is largely dominated by game design patterns, but we want to recognize that work such as Järvinen (2008) and Sicart (2008) indicates that a game mechanics focus is also valid. As the game mechanics focus has not yet become as established in the game design field, we have noted this as 'varied' in frequency of use (i.e. not because we do not believe it to have potentially as great of an impact as game design patterns have had).

Contextualization, as we have defined it using Giddens (1984), is likely to serve a useful role for design researchers as a bridge between game design patterns and game mechanics. Implicitly, we have already suggested that such research should make explicit in which way intentionality is viewed, given the central role of intentionality of design during contextualization (in the everyday sense, as well as phenomenological sense). In our case, intentionality has been defined by Brentano (1874) and expanded by Dourish (2001) through his reliance on Dennett (1987), and has been included in part due to the phenomenological perspective of our paper embraces, and in part the complementing relationship between intentionality and the work by Heidegger (1927) and Schutz (1932) that we also rely upon.

Even a purely software engineering driven interest in the design process is likely to use player interpretations as (at least) a point of reference to ensure that any process optimization does not negatively impact perceived product quality. We could furthermore see that a computer science researcher may hold a center of concern around the mechanics and possibly even the actual code paradigms and tools that support developers (cf. Dahlskog and Togelius 2012). However, as the focus throughout this paper has been on more traditional design research, we view these special cases as outside the scope of what we are discussing and therefore did not include these in Table 2.

Conceptual Relationship Model

In terms of the relationships between game design patterns, contextualization, game mechanics and code, Table 1 and Table 2 may be used to review the emerging suggestion for a conceptual relationship model. In this model (Figure 1), the generalized game design patterns are instantiated by contextualization to lend specific meaning and direction to a design. This contextualization is then broken down to mechanics that act as the components through which interaction with human and nonhuman agents take place. These mechanics are then implemented into code by developers, using the contextualization to guide their implementation decisions.
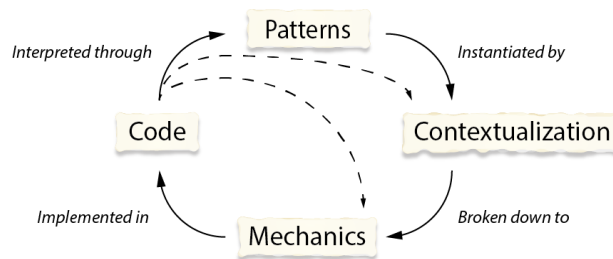

Figure 1: The conceptual relationship model.

Finally, the finished code – which constitutes the end product (or prototypes along the way) – become interpreted through the specific game design patterns that individual players have experience with as they explore the game. Signified in the figure with dashed lines, as this exploration goes deeper, the player is likely to increasingly become more specific with such comparisons and relate to specific games (i.e. interpret through a more contextualized lens) or even go as far as comparing with the specific mechanics that another game holds. Even if such comparisons on a mechanics level may yield different interpretations of the impact on the gameplay experience as a whole (due to the differences in surrounding related mechanics when switching game context) interpretations on a mechanics level are inevitable when it comes to for instance mastering the timing of complex mechanics such as the compounded mechanics that Sicart (2008) define.

An elaborated example of this may be seen in the impact on gameplay that different generations of the compounded mechanic 'quickscoping' from the Call of Duty series (primarily Modern Warfare 1 and 2). Quickscoping relies on the autoaim mechanic and timing together with the power of sniper rifles as weapons to allow one-shot-one-kill. The timing itself is a product of the mechanics 'raise weapon', 'aim weapon', and 'drop weapon'. The effectiveness of quickscoping depends greatly on how well this tactics stands up to comparisons with other tactics to kill opponents. In other words, while a mechanic may be very similar in two different games, it is the context in which they are used that defines if they are interpreted as useful by players, i.e. as a measure of the contextuality that the involved (and implemented in code) mechanics provide.

Within the FPS genre, quickscoping is heavily debated in terms of contextualization and impact on gameplay. While some debate the realism of the mechanic, the discussion is typically held at a lower level of abstraction, instead focusing on if this is a tactic that requires real skill, how much skill it requires, or if it is simply a matter of repetitive training and probability (i.e. the 'anyone can do it' metaphor, hinting at all snipers should learn and use this tactic). Some players view the mechanic as an exploit, particularly as it is related to the autoaim mechanic which was introduced for a completely different purpose. Autoaim was introduced on consoles first to counteract the increased difficulty of aiming when using hand controllers, rather than mouse-and-keyboard as traditional computers use. In other words, autoaim is a mechanic intended to counter challenges that are rooted in the hardware. Put into the context of quickscoping, the mechanic clearly has a considerable impact on the overall gameplay of the series. Due to the widespread gameplay impact it has – together with the debate on skill-level and realism – it is perhaps no surprise that some players view it as a cheap tactic. The tactic has furthermore resulted in those who use traditional sniper tactics (fully zooming in and shooting from a distance) sometimes ridiculed and dubbed as 'hardscopers' despite playing within the original intentionality of the game.

DISCUSSION OF RESEARCH IMPLICATIONS

We have argued for the introduction of contextualization as an added layer in the design language to the more neutral descriptions of game design patterns and actionable components of game mechanics. Looking at Table 1, we cannot help stressing how important it is that contextualization – and the underlying intentionality – receives the attention it deserves by designers and developers alike. While having a design language to use is well-established as vital at this point (cf. Barwood and Falstein 2010; Linderoth 2011; Linderoth et al. 2012; Lundgren and Björk 2012), contextualization itself is directional with a semantic meaning that implies purposeful use, despite context being in

a constant state of change (Dourish 2001). It is therefore not a concept that lends itself to be broken down into shared elements of description in the way patterns and mechanics may be. We therefore believe that contextualization is better viewed as a process of communication than attempt to deconstruct it into a template for specification. At best, we may understand and expressively describe contextualization through the intentionality cues that we share with language, gestures, illustrations, and other artifacts of communication. To this end, we have proposed a conceptual separation of concerns for patterns, contextualization, and mechanics based on Giddens' notion of locales (Giddens 1984). With the proposed separation comes a redefined core for game design patterns as describing places for the social interaction context, and for game mechanics as describing the physical context space for interaction. This leads locales – and therefore contextualization – to become any description of intentionality that includes both game design patterns and game mechanics.

Viewing contextualization as the layer where patterns and mechanics meet and intentionality is elicited, provides research opportunities for further exploration. First, we want to recognize the need for state-of-practice and state-of-research perspectives as relevant together and on their own. Only through a discourse that moves through all levels of abstraction and stakeholders will our design understanding and contributions have the full impact. Second, design practices that embody both game design patterns and game mechanics through their scenario description techniques need to be identified, assessed, and further developed. As we have previously argued, the goal for these descriptions is not to strive for complete specification but rather for capturing the directional stance. This is the point where the original intentionality (Brentano 1874) is exposed to the derived intentionality (Dourish 2001) of other actors in the design chain (or gameplay situation, if the goal is to analyze rather than design). Subsequently, this is where the main argument for why contextualization is best viewed as a process of communication arises, as the effectiveness of the form for description is entirely contextual (implied: ever changing) and therefore dependent on the effect it has in the present, rather than the effect it 'should have', or 'previously had'.

Reflecting over the areas of concern for players, designers, developers, and researchers that emerged in Table 2, two additional things stand out. First, the player is positioned between the designer and developer in terms of primary concern. This may be interpreted as a clue to how important it is for developers of games to also be active players of games for first-hand experience with how their code affects gameplay. It also hints at the importance for developers to continuously keep up with gameplay analysis on all levels – from initial impressions on the gameplay design pattern level to the nuanced enacting of complex compound mechanics in the hands of highly experienced (and even professional) players. Second, the player position also hints towards one of the dangers that follow game researchers. The abstraction level – while natural for researchers that strive for generalizability and theory development that lasts over time – is dangerously far from the player concerns, and similarly relatively far from the developer concerns. This is in part a product of our focus on design oriented researchers which means that the primary proximity may be expected to be with designers. Relying largely on the game design patterns as expressions of gameplay interactions (Björk and Holopainen 2004), has been our way of negotiating some of this risk. As hinted by the conceptual relationship model, understanding the interpretational processes of players as they become more proficient in a game (i.e. has explored and mastered a greater portion of it) is also of need in order to understand the borderland between design researcher and developer concerns.

CONCLUSIONS

This paper has addressed the complexity and wicked problem nature of design, and the impact this has on game design in particular. We have done so by developing a conceptual relationship model that proposes a separation of concerns for game design patterns and game mechanics, something which extant research has left largely implicit. While doing so, we further introduced and argued for contextualization to be added as a layer between game design patterns and game mechanics. The contextualization layer is where intentionality stands as foci for communication, and contains scenario descriptions of game mechanics and the game design patterns they are intended to instantiate. Additionally, by considering the focus and frequency of players, designers, developers, and researchers in terms of patterns, contextualization, mechanics, and code (i.e. the executable game), we furthermore identified main areas of concerns for these actors.

Finally, reflecting on the research implications of this study, we outline four directions for future research. These directions exemplify in which context the conceptual relationship model is relevant from the perspective of designing as well as the perspective of analyzing games.

BIBLIOGRAPHY

Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I., and Angel, S. A Pattern Language: Towns, Buildings, Construction. Oxford University Press, 1977.

Barwood, H. and Falstein, N. "The 400 project," in Proceedings of the Game Developers Conference '01, 2001.

Bergström, K., Björk, S. and Lundgren, S. "Exploring aesthetical gameplay design patterns: camaraderie in four games," In Proceedings of MindTrek '10 (Tampere, Finland 2010), ACM Press, pp. 17-24.

Björk, S. and Holopainen, J. Patterns in Game Design. Charles River Media, 2004.

Björk, S., Lundgren, S. and Holopainen, J. "Game design patterns," in Proceedings of DiGRA (Utrecht, Netherlands 2003).

Brentano, F. Psychology From an Empirical Standpoint. English translation 1995, New York, NY: Routledge, 1874.

Brusk, J. and Björk, S. "Gameplay design patterns for game dialogues," paper presentation at DiGRA '09 (London, UK, 2009).

Ciborra, C. "Introduction: What does groupware mean for the organizations hosting it?" In Groupware and Teamwork, C. Ciborra (Ed.), New York: John Wiley and Sons, pp. 1-19, 1996.

Church, D. (1999) Formal Abstract Design Tools. Available at http://www.gamasutra.com/view/feature/131764/formal_abstract_design_tools. php (accessed Jan. 2014)

Costikyan, G. "I have no words and I must design," in Proceedings of Computer Games and Digital Cultures '02 (Tampere, Finland, 2002), pp. 9-33.

Dahlskog, S. and Togelius, J. "Patterns and procedural content generation," in Proceedings of the Workshop on Design Patterns in Games, DPG '12, in Association with Foundations of Digital Games, FDG '12 (Raleigh NC, 2012).

Dennett, D. The Intentional Stance. Cambridge, Mass.: MIT Press, 1987.

Dormans, J. "Making design patterns work," in Proceedings of the Second Workshop on Design Patterns in Games, DPG '13, in Association with Foundations of Digital Games, FDG '13 (Chania, Greece, 2013).

Dourish, P. Where the Action Is: The Foundations of Embodied Interaction. Cambridge, MA: MIT Press, 2001.

Falstein, N. "Better by design: The 400 project," in Game Developer Magazine, vol. 9, issue 3 (March 2002), p. 26.

Fullerton, T. Game Design Workshop: A Playcentric Approach to Creating Innovative Games (2nd ed.). Morgan Kaufmann (ed.), Elsevier, Burlington: MA, 2008.

Gamma, E., Helm, R., Johnson, R. and Vlissides, J. Design Patterns – Elements of Reusable Object-Oriented Software. Addison-Wesley, 2001.

Gasson, S. Co-operative Information System Design: How Multi-domain Information System Design Takes Place in UK Organisations. PhD thesis, University of Warwick, Warwick Business School, 1997.

Giddens, A. The Constitution of Society: Outline of the Theory of Structuration. Berkeley: University of California Press, 1984.

Heidegger, M. Being and time. English translation 1962, New York: Harper and Row, 1927.

Holopainen, J. Foundations of Gameplay. PhD thesis at School of Computing, Blekinge Institute of Technology. Sweden, 2010.

Huizinga, J. Homo Ludens: A Study of the Play-element in Culture. Boston: Beacon Press, 1955.

Hullett, K. and Whitehead, J. "Design patterns in FPS levels," in Proceedings of Foundations of Digital Games, FDG '10 (Monterey CA, 2010).

Hunicke, R., LeBlanc, M. and Zubek, R. "MDA: A formal approach to game design and game research," In Proceedings of the AAAI Workshop on Challenges in Game AI (San Jose CA, 2004), pp. 1-5.

Hutchins, E. "The technology of team navigation," in Intellectual Teamwork: Social and Technological Foundations of Cooperative Work, Galagher, J., Kraut, R. E., and Egido, C. (Eds.), Hillsdale, NJ: Lawrence Erlbaum Associates, 1990, pp. 191-220.

Hutchins, E. Cognition in the Wild. MIT Press, 1995.

Jørgensen, K. Gameworld Interfaces. Cambridge: MIT Press, 2014 (to be published).

Järvinen, A. Games without Frontiers: Theories and Methods for Game Studies and Design. PhD thesis, University of Tampere, Finland, 2008.

Järvinen, A. "Viral mechanics uncovered," in Proceedings of Game Developers Conference Europe, GDCE '10 (Cologne, Germany, 2010).

Koster. R. Theory of Fun for Game Design. Paraglyph Press, 2004.

Kreimeier, B. (2002) The Case For Game Design Patterns. Available at http://www.gamasutra.com/view/feature/132649/the_case_for_game_design_patterns.php (accessed Jan. 2014)

Lankoski, P. Character-Driven Game Design - A Design Approach and Its Foundations in Character Engagement. PhD thesis at Aalto University, Finland, 2010.

Lewis, C., Wardrip-Fruin, N. and Whitehead, J. "Motivational game design patterns of 'ville games," in Proceedings of the Foundations of Digital Games, FDG '12 (Raleigh NC, 2012), pp. 172-179.

Linderoth, J. "Exploring anonymity in cooperative board games," in Proceedings of Think Design Play: DiGRA '11 (Utrecht, Netherlands, 2011).

Linderoth, J., Björk, S. and Olsson, C. "Should I stay or should I go? Boundary maintaining mechanisms in Left 4 Dead 2," in Proceedings of Nordic DiGRA '12 (Tampere, Finland, 2012).

Lundgren, S. and Björk, S. "Game mechanics: Describing computer-augmented games in terms of interaction," in Proceedings of Technologies for Interactive Digital Storytelling and Entertainment, TIDSE '03 (Darmstadt, Germany, 2003).

Lundgren, S. and Björk, S. "Neither playing nor gaming: Pottering in games," in Proceedings of the Foundations of Digital Games, FDG '12 (Raleigh NC, 2012).

Mateas, M. and Stern, A. "Build it to understand it: Ludology meets narratology in game design space," in Proceedings of DiGRA '05 (Vancouver, Canada, 2005).

Milam, D. and Seif El Nasr, M. "Analysis of level design 'push and pull' within 21 games," in Proceedings of the Foundations of Digital Games, FDG '10 (Monterey CA, 2010).

Mintzberg, H. and Waters, J. H. "Of strategies, deliberate and emergent," in Strategic Management Journal, vol. 6, 1985, pp. 257-272.

Olsson, C.M. Developing a Mediation Framework for Context-Aware Applications: An Exploratory Action Research Approach. PhD thesis, University of Limerick, Ireland, 2011.

Orlikowski, W. J. "The duality of technology: Rethinking the concept of technology in organizations," in Organization Science, vol. 3, no. 3, 1992, pp. 398-427.

Rheinfrank, J. and Evenson, S. "Design languages," in Bringing Design to Software, Terry Winograd (ed.), ACM, New York: NY, 1996, pp. 63-85.

Rittel, H.W.J. and Webber, M.M. "Dilemmas in a general theory of planning," in Policy Sciences, Vol. 4, 1973, pp. 155–169.

Roberts, N.C. "Wicked problems and network approaches to resolution," in International Public Management Review, vol. 1, no. 1, 2000, pp. 1-19.

Sankalia, J. (2013) Layers of Player Understanding. Available at http://www.gamasutra.com/view/feature/195320/layers_of_player_understanding.php (accessed Jan. 2014)

Salen, K. and Zimmerman, E. Rules of Play: Game Design Fundamentals. MIT Press, 2004.

Schön, D. A. The Reflective Practitioner: How Professionals Think in Action. New York, NY: Basic Books, 1983.

Schutz, A. The Phenomenology of the Social World. Northwestern University Press, Evanston: IL, 1932.

Sicart, M. "Defining game mechanics," in Game Studies, vol. 8, no. 2, (December 2008).

Smith, G., Anderson, R., Kopleck, B., Lindblad, Z., Scott, L., Wardell, A., Whitehead, J., and Mateas, M. "Situating quests: Design patterns for quest and level design in role-playing games," in Proceedings of the International Conference on Interactive Digital Storytelling, ICIDS '11 (Vancouver, Canada, 2011).

Spector, W. (1999) Remodeling RPGs for the New Millennium. Available at http://www.gamasutra.com/view/feature/131716/remodeling_rpgs_for_the_new_.php (accessed Jan. 2014)

Stolterman, E. "The nature of design practice and implications for interaction design research," in International Journal of Design, vol. 2, 2008, pp. 55-65.

Zagal, J.P. and Mateas, M. "Time in videogames: A survey and analysis," in Simulation and Gaming, vol. 41, no. 6, 2010, pp. 844-868.

Wittgenstein, L. Philosophical Investigations, 4th edition. Original English translation 2001 (4th editition 2009). Blackwell Publishing, 1953.