CHALMERS | GÖTEBORG UNIVERSITY

Facets of Fun: On The Design of Computer Augmented Entertainment Artifacts

SUS LUNDGREN

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
GÖTEBORG UNIVERSITY
Göteborg, Sweden 2006

THESIS FOR THE DEGREE OF LICENTIATE OF ENGINEERING

FACETS OF FUN

On the Design of Computer Augmented Entertainment Artifacts

Sus Lundgren

Department of Computer Science and Engineering
Chalmers University of Technology and Göteborg University
SE-412 96 Göteborg, Sweden
Telephone +46 (0)31 7721000

Göteborg, 2006

Facets of Fun:
On the Design of Computer Augmented Entertainment Artifacts

Sus Lundgren

Technical report 29L
ISSN 1652-876X
Department of Computer Science and Engineering
Research group: IDC | Interaction Design Collegium
Department of Computer Science and Engineering
Chalmers University of Technology and Göteborg University
SE-412 96 Göteborg, Sweden
Telephone: +46 (0)31 772 10 00
Printed at Chalmers Reproservice, Göteborg, 2006

This work is dedicated to a person whom I have never met, and whose name I do not know, but whose generosity saved my life. A part of you still lives inside me, keeping me up and running day after day after day. Thank you... You will not be forgotten.

ABSTRACT

This thesis discusses the design of interactive entertaining artifacts such as computer games and computer augmented toys, games and objects. The aim has been to find and/or develop design tools – i.e. methods and approaches – that can be used when designing such artifacts in general, and when aiming for an emotional response or tempting challenge in particular. Another aim is to inspire game designers, interaction designers, product designers, artists and everyone else interested by describing actual projects where interactive entertainment artifacts have been created. Five tools are being presented, each with an accompanying case. They are:

Slow technology; an approach to create a challenging object by using ambiguity, thus promoting reflection. This is being exemplified by the Interactive Quilt, which is an enigmatic combination of a jukebox and a quilt.

Animal Expression Transfer; an approach that can be used to create emotional responses to an artifact by transferring animal traits onto an object, as in the case of the Iron Horse, a computer augmented bike making horse-sounds.

Game Mechanics for Computer Augmented Board Games; a set of building stones for computer augmented board games, suitable for increasing, decreasing or regulating the challenge in a game. They widen the design space of the traditional board game by enabling complex functionality that does not burden the user, but enrichens the game experience. The projects MultiMonsterMania, a collectible and interactive card game with the accompanying game The Hatchery exemplify this.

Game Design Patterns; a pattern language for games that can be used to discuss, analyze and design games of any kind, amongst them also computer augmented games. How this can be done is being demonstrated in the description of a workshop.

The Forces-Clashes-Remnants-model; a design model for iterative multidisciplinary game design. It is being exemplified by describing the design process of myTHeme, a computer augmented storytelling game for non-storytellers.

Keywords: Interactive entertainment artifacts, design tools, animal expression transfer, slow technology, game mechanics, Game Design Patterns, forces, clashes, remnants, the Forces-Clashes-Remnants model, the Interactive Quilt, the Iron Horse, myTHeme, MultiMonsterMania, The Hatchery.

REFERAT

Den här avhandlingen handlar om hur formgivningen av interaktiva och underhållande artefakter, som exempelvis dataspel eller datorförstärkta leksaker och dito spel. Syftet har varit att hitta eller att utveckla designverktyg – metoder eller angreppssätt – som kan användas när man skapar sådana artefakter i allmänhet, eller när man är ute efter att skapa ett emotionellt gensvar eller en utmaning i synnerhet. Ett annat syfte är att inspirera speldesigners, interaktionsdesigners, produktdesigners, konstnärer och alla andra genom att beskriva några faktiska projekt där sådana artefakter har skapats. Fem designverktyg med ett tillhörande exempel beskrivs och de är som följer:

Slow Technology; ett angreppssätt med vilket man kan skapa spännande och utmanande artefakter genom att använda tvetydigheter och uppmuntra till reflektion. Ett exempel på detta är The Interactive Quilt som är en gåtfull blandning mellan ett lapptäcke och en jukebox.

Animal Expression Logic; ett angreppssätt som kan användas för att skapa en emotionell koppling till ett föremål, i det att man överför ett djurs drag och beteenden till ett objekt. Det tillhörande exemplet är Järnhästen (”The Iron Horse”) – en cykel som låter som en häst.

Game mechanics för datorförstärkta spel; en uppsättning byggstenar som man kan använda för att öka, minska eller reglera utmaningen i datorförstärkta brädspel. Dessutom ökar de designrymden för brädspel genom att tillhandahålla komplex, men ändå dold funktionalitet som inte belastar spelarna. Det datorförstärkta samlarkortspelet MultiMonsterMania med det tillhörande spelet The Hatchery exemplifierar detta.

Game Design Patterns; designmönster för spel. De kan användas för att diskutera, analysera och designa alla typer av spel, alltså även dataspel och datorförstärkta spel. Hur de kan användas demonstreras i beskrivningen av en workshop.

The Forces-Clashes-Remnants-model; en modell för iterativ flerdisciplinär speldesign. Hur den används visas i exemplet myTHeme som är ett datorförstärkt kortspel för sagoberättare utan fantasi.

ACKNOWLEDGEMENTS

On a professional note...

Lars Hallnäs, my supervisor – for believing in me, teaching me, and supervising me without telling me what to do. You're the best!
Staffan Björk – for once hiring me to PLAY, working with me and sparring with me; arguing, agreeing and playing.
Hanna Landin – for working on the Iron Horse with me way back, and for walking by my side along the PhD path, always ready for discussion, feedback, opinions and good advice – and lunch.
My colleagues – the entire Interaction Design Collegium at Chalmers and my former colleagues at PLAY, for discussions, feedback and help.
All my project mates; no thesis without you – Sara Johansson, Pär Stenberg, Paula Thorin & Fredrik Nilsson; Hanna Landin, Johannes Prison & Magnus Johansson; Staffan Björk & Jussi Holopainen; Staffan Björk, Jennica Falk, Karl Petter Åkesson & Jussi Holopainen.
Frank Nestel, game designer – for valuable comments on the art of game design.
Karl Petter Åkesson for sanity-checking my incoherent babbling about sensors etc. If any mistakes still can be found there, they are mine.

On a personal note...

Magnus Lundgren – for being the best husband ever, and for patiently answering questions about software design and development. :)
Bengt Gustafsson, the ladies at Pol and the rest of the personnel at wards 20 & 33 at **Sahlgrenska** University Hospital, for keeping me alive and well in general.
My family – The Henrikson clan, the Lundgren clan and the Sjöberg clan for love, support and food. :)
Petra, **Fredrik** and **Patrik.**
Jürgen Kappe for getting me out of bed and into the office for some two weeks in October so that this thing could finally be ready for print!

And last but not least **Pyret** – for ehhh… kicking me, having me go to the bathroom every 45 minutes and just invading my body in general… bringing so much joy, excitement, confusion and happiness. :)

CONTENTS

List of Figures

All photographs, drawings and illustrations made by Sus Lundgren unless stated otherwise.

PART I:
SETTING THE STAGE

INTRODUCTION

This is an attempt to inspire interaction designers, product designers, game designers and others to create richer interactive entertainment artifacts, artifacts perhaps being a tad more entertaining or emotive than before, by being inspired by the projects described, or by using the design tools – various methods, approaches and techniques – that were used when creating them. In addition, I want to point out how computer augmentation of an object can help in making it more interesting, useful or simply more fun!

Note however that I leave the analysis, exploration and definition of "fun" and "entertainment" to others; this is not within the scope of the thesis. Neither do I provide a fail-proof way to design it; there is most likely no such way, since we all are different and it is impossible to please all. Instead I highlight a few tools that are fit to use when trying to design something entertaining. Some are high-level, some are low-level, some are abstract and some very practical, some are concerned with game design and some are not; but all of them can be seen as a few facets through which we can explore the design of entertainment and fun – hence the title "Facets of Fun".

"Entertainment" is still a wide definition though, and since this is a work in interaction design and computer augmentation, the rest of the thesis will "only" discuss the realm of *interactive entertainment*.

What is Interactive Entertainment?

As early as 1995 Wired Magazine (Platt 1995) featured an article entitled "Interactive Entertainment: Who writes it? Who reads it? Who needs it?" in which many computer game concepts focusing primarily on narratives are presented. In the article, Platt quotes Janet Murray[1] who comments on the three "key pleasures" she claims are characteristic for electronic media. They are *immersion* (the sense of actually "being" in the game), *rapture* (the trance players can fall into, gaming for hours and hours) and *agency* ("the player's delight in having an effect on the electronic world,").

Even if those three terms can be used about any children's game, e.g. playing Cowboys and Indians, the term "interactive entertainment" seems to be tightly intertwined with digital games, typically computer and video games, both at

[1] In 1995 Jane Murray was director of the Laboratory for Advanced Technology in the Humanities at MIT. She is the author of "Hamlet on the Holodeck: The Future of Narrative in Cyberspace" (Murray 1997).

conferences[2] and in branch organizations[3]. My work is however based on a somewhat wider definition, made by Fjellman & Sjögren (2000) in a Swedish research report. The report deals entirely with different types of computer and video games, but the authors' definition is wider; they define interactive entertainment as *"…any kind of entertainment that is experienced using digital technology and demands that the user participates actively"*. According to this passively watching a DVD-film is not interactive entertainment; even if there is digital technology involved, the interactivity is lacking. Likewise, playing an ordinary board game is not interactive entertainment in this sense – it is truly interactive but does not involve any digital technology.

Note that even if the "digital technology" seems to be a key factor, there is actually nothing that says that "an ordinary computer" or a game console with their standard input devices (e.g. a joystick) and a screen need to be present; only "digital technology", i.e. computational technology of any kind. This computational technology can actually be anywhere, for instance embedded in an object – any object. Why not a board game, a card game, a bike or a quilt?

Defining the Interactive Entertainment Artifact

The main part of this thesis are five chapters, or *facets*, describing a case and a tool related to the design of such computer augmented artifacts. I prefer to call them artifacts to point out the fact that many of them are actually computer augmented *physical* objects as opposed to residing within a digital/computational system. The reason that I want to make this distinction is because it widens the design space; we leave the digital realm and suddenly we can use – and need to consider – lots of physical qualities like the texture of a fabric, the physical experience of riding a bike or shuffling a set of cards in you hand. In this, creating the interactive entertainment artifact is a mixture of computing science and handicraft.

Then again there is another view on what this design space actually contains. Saying that it can incorporate almost any design discipline isn't exactly a way to define it, even if it is an important distinction. However it is not the key boundary. You will notice that I will discuss properties like "emotional response" and "intellectual challenge" in some facets. To me, these two aspects are important when creating an

[2] E.g. IE (The Australasian Conference on Interactive Entertainment) and AIDE (Artificial Intelligence and Interactive Entertainment) and DIEC (Digital Interactive Entertainment Conference).

[3] E.g. the Interactive Entertainment Association of Australia is an association for computer and video game companies.

entertainment artifact; it needs to grab my interest either by proposing me with an intriguing and challenging task, or by encouraging or enrichening my experience of it by triggering an emotional response – otherwise it lacks entertainment value.

My rationale for this is that without sufficient challenge an activity can be perceived as boring, or soothing, or calming or "nice" but hardly entertaining. Then again, "challenge" is to be interpreted widely. Also the level of challenge must suit the user and the situation; whereas a three-year-old will be perfectly well entertained by trying to put the round pieces through the round holes and see them disappear, this task is not challenging enough for an adult. The adult may be more entertained by trying to solve a cross-word, however not when being for instance tired or nervous about some upcoming event at work. In order to be entertaining, the challenge has to be tempting to that user at that moment.

Moreover, an artifact can be perceived as being more entertaining if the emotional response to it can be made stronger using the emotional response that comes from admiring, playing, caring and identifying with an object. Here, especially, computer augmentation can be very helpful since it can provide an interactive behavior – a seemingly emotional or physical response to the user's interaction with it.

Thus, my definition of an interactive entertainment artifact – which has slowly unfolded during this work – is as follows:

An interactive entertainment artifact uses computer augmentation to entertain and engage an active user by providing a tempting challenge and/or evoking a strong emotional response.

Research Question

All the design projects that have been carried through in order to create this thesis have evolved around the tools and principles of designing the artifacts per se, but have also dealt with how to enhance the artifacts with aspects of emotional response or tempting challenge in order to make them entertaining. Thus, they all have revolved around one central question, investigating different aspects – facets – of it, namely:

How can we go about to create interactive entertainment artifacts in general, and how do we design for emotional response and/or tempting challenge in particular?

The result is five facets, five excursions within the design space of creating entertainment and emotion, using computer augmentation of physical objects. All of these will be described and discussed in Part II: Design Tools and Cases.

How to Read this Book

The intention is that each of the chapters – the facets – that describe a tool can be read separately and independently of the others. However, the concepts explained in the background part are a basis for understanding the rest so for a deeper understanding it might be a good idea to skim the background headlines and stop and read about unfamiliar topics. The tools are discussed in depth in their own facets in Part II, but in addition there is a deeper discussion on the design space and the tools in relation to each other in Part III.

BACKGROUND

This thesis deals with the design of *interactive entertainment* which per definition involves computer augmentation. Now, there are a set of topics that are relevant in relation to this, and they will be briefly introduced below. Firstly, the qualities of *computational technology as a design material*, since no interactive entertainment artifact can be made without using computational technology as an essential part. Secondly, the discipline that uses computational technology as one of its most prominent materials; *interaction design*. Thirdly, the technologies we can use to computer augment things; an area called *ubiquitous computing*.

Computational Technology as a Design Material

Traditional design materials are for instance wood, clay, metal, fabrics, paper etc, but also newer materials like plastic. Lately, computational technology too, has become a common part of many everyday things; toys, watches, cameras, music players. cell phones, etc. Thus, computational technology can be said to be a design material as well (Löwgren & Stolterman 1997 and 2004, Dunne 1999, Hallnäs et al 2001, Hallnäs & Redström 2002). Explicitly seeing computational technology as a design material does acknowledge that it can be deliberately used to create expressions and behavior rather than just simply carrying out a technical solution according to given specifications.

Unlike many other design materials, the computational technology in itself isn't actually physical. It is shaped by non-physical tools like text commands, algorithms and the like, and even if it can be said to reside within physical objects like a processor, or a wire, or a hard disk, these objects do not *express* the material. The visible physical appearance of a hard disk does not change depending on what data it carries, neither does a cable change its color or position depending on what data it transfers.

This non-physical nature of computational technology is part of its flexibility; the only things that can be said for sure about it are that it needs electricity to work, and that it is built upon a digital logic. It is *"a material without properties"* (Löwgren & Stolterman 1997, p.2).

In addition, computational technology in essence is not spatial. It is temporal, existing only when in action – when being used, accessed – and in this it manifests itself through the behavior of the object that has been computer augmented. It is a

fragile, flexible, volatile and very powerful design material that cannot exist without other materials expressing it. Those other materials are typically a combination of other, non-physical materials (e.g. electricity, light and sound) as well as more traditional materials and constructions. In turn, the properties of the chosen expression materials define, denominate and delimit the impact and expression possibilities of the computational technology.

In many ways, computational technology has the same properties as does music; in both cases we have an abstract material that only exists in action, by performance, carried by things like instruments or computer screens. In both cases a code language is used to define the design, e.g. some sheets of music define the piece, and some kilobytes of program code define a certain behavior of an object. Still, the score isn't the music, as the code isn't the behavior. Even if we can train ourselves to "hear" the music when we read the score, it is still actually not there until it is being played, existing only in action.

In the same manner one can bluntly state that when looking on code one could conclude what it does. This may be true for very simple code describing a more or less fixed course of events, but as soon as the program becomes more complex or requires the unforeseeable input from a user, it is impossible to get an understanding of the behavior of that, what the code implements, simply by looking only at the code. This is no different from the fact that a piece of music will sound different every time it is performed live – depending on the mood of the musicians, the response from the audience, the acoustics in the room etc – and therefore is impossible to predict completely correctly.

This flexibility is the most important quality that computational technology can bring to an artifact. Few other materials can provide this; once an object is cast in metal, carved in wood or sculptured into stone it cannot change. But an object that gets some of its expressions from computational technology can be active, flexible, and have a complex behavior. It can *react* on the user's input and change accordingly – in a more complex way than just turning off or on a light. It is *interactive* – which brings us directly to the next topic!

Interaction Design

Terry Winograd (1997) has pointed out that even if the first computers were actually used to *compute*, computers today are typically used to *communicate*. According to Winograd we have left the machine-focus, not seeing our computers as machines, but as computational environments (or *"habitats"* as he puts it) in which we work, talk and play. He foresees an ascendancy of interaction design:

> *"In the next fifty years, the increasing importance of designing spaces for human communication and interaction will lead to expansion in those aspects of computing that are focused on people, rather than machinery. The methods, skills, and techniques concerning these human aspects are generally foreign to those of mainstream computer science, and it is likely that they will detach (at least partially) from their historical roots to create a new field of 'interaction design.'"*
>
> *– Terry Winograd,*
> *In: "From Computing Machinery to Interaction Design" (1997)*

According to Wingrad, the interaction designer is for the design of software what the architect is for the construction of a house; a professional with knowledge in both engineering and human aspects. The interaction designer shifts from seeing the machinery to seeing the lives and the needs of the people who use it. In analogy with this, the aim is not to create an interface, but an *"interspace"*.

Richard Buchanan (1999) also comments on this shift, claiming there are four orders of design focusing on *symbols* (e.g. graphic design), *things* (e.g. industrial design), *thought* (e.g. environmental design) and *action*, the latter being the order of interaction design. Seeing communications and constructions as forms of action is a basis for the design domain of interaction design, in which the designer is *"…focusing on how human beings relate to other human beings through the mediating influence of products"* (Buchanan 1999, p. 11). Buchanan also claims that interaction design does not have to be concerned with the digital medium, an opinion that is not shared by everyone, including myself.

Looking at action and interaction, Chris Crawford (2002, p. 5 - 10) elaborates on *interactivity*, claiming that successful interaction requires *"listening"* (as for instance picking up input, if being a computer), *"thinking"* and *"speaking"* and at least two

actors taking turns doing this. Actors can be humans, but might as well be anything having a behavior complex enough to do this, e.g. a dog or even a computer. All of the three activities need to be present according to Crawford; a book for instance isn't interactive since it only "speaks" (according to this definition; there are plenty of people who disagree). Moreover, the *"listening", "thinking"* and *"speaking"* needs to be rather elaborate. I.e. a turn-off switch that gives a feedback buzz isn't interactive enough, even if it does "listen" (notices that it has been pressed), "thinks" (sends a signal) and "speaks" (sees to it that the sound is played). Why not? Well, firstly, because this is *all* it can do. Secondly because it will do the same thing regardless of who presses it and how and why. In line with this Crawford summarizes interactivity design, or interaction design as the craft that *"…addresses the entire interaction between the user and the computer."* (p. 10). The word *entire* is a keyword here, since Crawford means that this puts an emphasis on the "thinking" part of the interactivity.

A similar definition is made by Alan Cooper[4] and Robert Reimann (2003, p xxix - xxx):

> *"Simply put, interaction design is the definition and design of the behavior of artifacts, environments and systems, as well as the formal elements that communicate that behavior"*
>
> *– Alan Cooper and Robert Reimann*
> *In: About Face 2.0: The Essentials of Interaction Design (2003, p. xxix)*

Personally, I agree with this definition although I would like to add that we do not only design the behavior of objects – in a sense we try to design the behavior of the users as well. In doing this we create a system based on communication, action and reaction, which is pretty much what Crawford means with *"listening"*, *"thinking"* and *"speaking"*. In achieving this, the computational technology is our most important design material.

Interaction design is thus a new discipline of its own, lurking between the boundaries of closely related fields like industrial design, product design, interface design and human-computer interaction. All of these design disciplines are similar as they share many visions, views, tools and methods; within one design project one can

[4] Alan Cooper is the designer of Visual Basic and has also written the bestseller "The Inmates are running the asylum" on the software industry and on the design of user interfaces.

slowly shift from one discipline to another without noticing. Hence, there is a lot of disagreement on where the borders between these intertwined subjects actually are or should be drawn, and whether some are subsets of others, or perhaps vice versa(!) etc. Since the design of interactive objects or systems typically involves many disciplines (e.g. programming, graphic design, material science, electronic engineering, sound design), the interaction designer seldom works alone, but in a group of other professionals. Typically, the interaction designer is highly concerned with the interaction with the product, including social and cultural aspects. What happens if many users use it; how does this affect users, society and culture?

Also, the interaction designer must not necessarily focus upon creating products whose most important property is its usability. Instead, the design can be targeted towards making the product fun, entertaining, beautiful, puzzling or whatever other wanted quality, rather than useful.

Thus, the interaction designer's goal can be said to be: *To create an interactive object that with its responses affect the use, the user and the context in the intended way.*

Ubiquitous Computing as a Basis for Computer Augmentation and Interactivity

If the interaction design resides within the computer, so to speak, the ways for it to manifest itself would be via the screen and via the sound the computer makes, via the ways input can be made and via the numerous functionalities that the standard desktop computer can provide. However, computational power can also be embedded into other objects, in this way computer augmenting them. This concept, called *ubiquitous computing*, (sometimes referred to as *pervasive computing*) was coined by Mark Weiser in 1988. At the time he was working at Xerox PARC where most of the first products using ubiquitous computing were created (Weiser 1993, Abowd & Mynatt 2000), and in doing this he foresaw an upcoming change in the relations between computers and humans. According to Weiser the human-computer relations can be divided into three main eras; the mainframe era with very few computers and many users per computer, followed by the personal computing era where the typical computer is owned and use by one person. Currently, we are in the shift between this era and the one Weiser foresaw, namely the era of the ubiquitous computers, where each user owns, uses and accesses many computers, computers that are sometimes owned and used by this person only (like a laptop) or shared by many others (like for instance an Internet server). Weiser's main thesis is that the

computers will "disappear" when computational power becomes as common and invisible as electricity. Then it will be everywhere, embedded in many things, and we take its existence and its functions for granted; we use it without giving it any thought. (Weiser and Brown 1996). In Weiser's vision computers would disappear totally, becoming an extension of the human mind. Computational power should be everywhere; quiet, calm, disappearing, automatically within reach when needed:

> *"We will dwell with these computers, whose presence we will ignore most of the time, and they will provide us with constant clues about our environment, our loved ones, our own past, the objects around us and the world beyond our home. Computers will act like books, windows, walks around the block, phone calls to relatives. They won't replace these, but augment them, make them easier, more fun. […] Ubiquitous computing just might help to free our minds from unnecessary work, and connect us to the fundamental challenge that humans have always had: to understand the patterns in the universe and ourselves within them."*
>
> *– Mark Weiser and John Seely Brown*
> *In: "The coming Age of Calm Technology" (1996)*

Even if we are just at the dawn of this era, the use of Internet, cell phones and local networks can give a clue on what such a future can be like. Interestingly the rise of the ubiquitous computing era has brought about an increased interest in relations between humans and computers in terms of usage, ease of use and user approach, leading to the origin of the discipline of interaction design, which differs from human-computer-interaction in that that it deals with *any* computer augmented object, i.e. any object using ubiquitous computing. Thus, these two subjects are closely intertwined and co-dependent.

In line with this, Abowd and Mynatt (2000) listed the following three themes of ubiquitous computing research of today as being the most important ones:

1. To create *natural interfaces* between humans and computers, i.e. interfaces where natural interaction forms like speech, hand writing and drawing can be used.
2. To create applications that are *context-aware* and can adapt their behavior and functionality to the current user and context, preferably by itself collecting information from its physical and computational environment.

3. To create applications that *automate* the capture of life and provide easy access to those experiences.

The first two themes are the most interesting ones when designing interactive entertainment artifacts, since automatization however nice it is, isn't very engaging by nature.

Tangible Interfaces – computer augmenting objects

The idea to create natural interfaces between humans and computers, interfaces who are designed to suit *human* types of interaction has lead to a research area called Tangible (i.e. touchable, concrete) Interfaces – just as Abowd and Mynatt (2000) proposed. Here, one notion is that bits – the smallest data unit, having either the value 1 or 0 – shall be represented by atoms, thus creating "tangible bits" (Ishii & Ullmer 1997). This means that physical object shall be used to "carry" and manipulate data, e.g. when computer augmenting game pieces. Another, very illuminating example, of this is the i-LAND project (Streitz et al 1999). The system consists of interactive office furniture; an electronic wall that can be used as a super-whiteboard on which documents can be shown, moved, updated etc, a similar table and chairs with built-in laptops or docking facilities for laptops. All of these are set in one room, and the idea is to support cooperative and very intense work. All the items are connected via a Local Area Network and can exchange information through normal computer operations, but another way to move data from one piece of furniture to another is to use a sub application called Passage, which provides the possibility to carry data "in" physical objects. Each piece of furniture has a Bridge embedded next to its border. The actual document is drawn to the margin, and a physical object – any object; keys, glasses, a pen, whatever is unique – is placed on the bridge in order to annotate the data to it. The object can then be carried to another Bridge, and the corresponding document will show up on the screen next to it. The Bridges use the exact weight of the object as an ID coupled to the data. This provides a very easy way to move a document from one environment to another without fussing with mounted hard disks, navigating the file system to find the right file etc. Also note how the objects carrying the data are not altered or computer augmented in any way.

Suitable technologies for computer augmentation

Below is a rough introduction to the world of microcontrollers, input and output devices and a few examples on ways to combine and use them. It is only a brief overview aiming to demonstrate some of the vast possibilities of computer augmentation. Note that the number of devices and technologies increase and improve constantly. The book "Physical Computing: Sensing and Controlling the Physical World with Computers" by Tom Igoe and Dan O'Sullivan (2004) can be recommended for anyone wanting to know more.

Microcontrollers

A microcontroller is a kind of one-chip computer (Igoe & O'Sullivan 2004), typically very small. They are common in many everyday appliances, and in addition they are often used for automation. In practice, the microcontroller is a highly integrated chip which includes all or most of the parts needed for a controller. It is of course programmable, even if it cannot support as much or as complex code as a normal computer.

There are several different levels of microcontrollers and microcontroller systems. The simplest ones are small chip-size devices to which all additional electronics have to be connected (e.g. Microchip's PICMicro microcontroller). Others are larger and more elaborate; they are composed of several components and ports for input and output (e.g. BX-24 microcontrollers). Higher level controllers come with a hardware interface to other devices, and a simple programming language (e.g. Teleo). Naturally the latter ones are the most expensive, but they require much less work and knowledge in electronics. Instead, lower level processors are cheaper and more flexible.

Typically the microcontroller works as a hub in the system, collecting input from buttons, sensors or other input devices, processing this data (sometimes with the help of an external, more powerful computer) and finally using it to affect the output devices, which can be lamps, displays or even motors.

Since the microcontroller is a one-chip solution it drastically reduces parts count and design cost, and will most likely be embedded into a computer augmented artifact.

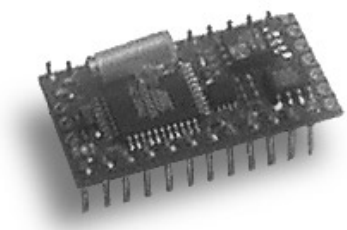


Figure 1: A microcontroller, namely a BX-24 chip, natural size.

Input

Input devices are the devices humans use to communicate with computers. For instance the keyboard I am writing this on is an input device connected to the "computer" in my laptop, and it communicates the letters and signs I want to appear on the screen, just as the mouse communicates how I wish to move the cursor on the screen. But an input device can be anything; the buttons or jog dials on a CD-player, or any knob, switch, handle etc.

All of these input devices are used for explicit input; i.e. there is a user and she or he deliberately inputs information to the computer augmented system. If we want to achieve implicit input instead, sensors can be appropriate to use since the users can then input information to a system without noticing that they are doing this; in this sense the technology really becomes invisible.

Sensors

Most of us are used to sensors that pick up environmental data – a thermometer being one example. There are sensors to measure the presence of, or changes in almost everything, e.g. heat, pressure, tilt, distance, movement, weight, sound, speed, pressure, bend etc. They then turn this into changes in voltage, resistance or capacity, which can easily be transformed into digital information that can be interpreted by a computer.

When using this kind of sensors, it is important to make sure that the sensor is apt for *measuring the actual property*, and the actual range the property will have in the application. If we for instance use a compression sensor to detect if a light wooden piece is put on a tile, then the compression sensor must be able to measure pressures of only a few grams. Second, the sensor should *not influence the measured property*. If for instance a thermometer having room temperature is put in a very small container of very cold liquid, it will itself increase the temperature of the liquid somewhat, obscuring the measurement. Thirdly, the sensor should *not be affected by any other actual property*; it shall measure one property and one property only, and do it well.

Image processing

A camera can be seen as an "image sensor" and with correct (and sometimes very complex!) programming the stream of images can be analyzed and the computer act on the information extracted. An easy technique is to have a background with an even color and simply try to detect how much of the image that is not in that

color, thus finding out if there are many people in front of the camera, or perhaps just one, very near. If comparing several images, conclusions about the movement of this person can be made. This is the basis for gesture recognition. Also, one can look for changes in color or light or of course dig into more complex matters like face recognition etc.

ID-sensors

The sensors mentioned above do not really have the ability to distinguish different object from each other in an easy way. Of course object can be made in different sizes or materials, thus having different weight and if one uses pressure sensors sensible enough, these changes can be detected, i.e. the sensor can help distinguish the king from a pawn in the chess game. But it cannot distinguish the different pawns from each other. Sometimes there is a need for assigning a unique identity to an item, and be able to detect it. There are different sensing techniques for this.

RFID-tags and readers are very commonly used. The abbreviation stands for Radio Frequency ID, and as implied, they use very weak radio signals to communicate. The technology consists of two units, a small *tag* that is embedded into or hidden under the actual object. This tag holds only one piece of information; an unique ID, and it does not need any electricity or anything else to work. It is passive until activated by a *reader*. The reader is flat and larger than the tag; both come in varying sizes. Unlike the tag, the reader needs power from some source, however not much; a battery will do. Whenever a tag is placed on, or near, the reader (this can be modified) the reader activates it and collects the ID, which is then transferred to some computational device. Sometimes this is combined with the reader itself. In that case it normally has a small display built into it. Unfortunately the readers are rather expensive in comparison, whereas the tags cost almost nothing. From an economical point of view it is therefore beneficial to design the system for many tags and very few readers.

Bar-codes can be used as well. Every item is marked with a bar-code that can be read by a bar-code scanner, just as in any shop. Unlike the RFID-tags the barcode has to be visible, which can be a drawback. Also, when reading the code, the bar-code has to face the reader, which may require rotation of the object.

Output

Output devices are those devices that the computer uses to communicate with a user; feedback devices. For instance the screen of a computer is an output device; it

shows what is going on; that the letters I am just writing really *are* appearing in this document. Not that a screen doesn't have to be a "screen" per se, it may just as well be the image of the screen projected on a wall, or on a table. If shadows of players or of player's hands are an issue, the image can be projected from behind, i.e. from underneath a table having an opaque glass surface. Or, the screen needn't be the regular computer screen ; it can be the screen of a cell phone or a Personal Digital Assistant (PDA, like a Palm Pilot for instance).

Apart from a screen and what is happening on it, typical output devices are light emitting diodes (LEDs, the small lamps that show if an appliance is turned on or off), built-in digital displays, vibrators or loudspeakers that give off sound.

Motors are other output devices; instead of displaying information they move something. There are many different kinds; they can be small or big, weak or strong, they can run on direct current (i.e. on batteries) or alternate current (i.e. from the electric mains). Step motors turn their driving shaft in steps, meaning that their rotation can be controlled down to less than a degree, resulting in a very exact motion.

Putting it all together

For people not so apt in electronics, there are ready-made sets of sensors and some kind of reading unit that can be used. One example of this are Phidgets (www.phidgets.com) which can be seen as a set of Lego pieces consisting of various sensors (including RFID) that can be put together in any combinations and that are joined together by a Universal Serial Bus[5] which can be plugged into a computer. A programming interface allows rapid application programming in a variety of languages, e.g. Visual Basic, Java, Delphi, C and C++.

Another possible way is to use more or less ready-made sensors together with Processing, which is an open source programming language and environment based on Java (http://processing.org/). It has been developed by artists and designers and is targeted towards non-professional programmers (e.g. artists, students, hobbyists, researchers) that want to be able to set up relatively simple but yet functioning prototypes. It is free to use and was first released in April 2005. It has evolved from ideas explored at MIT, but the project was initiated by Ben Fry (Broad Institute) and Casey Reas (UCLA Design | Media Arts).

5 The USB-technology is commonly used for connecting anything from printers to cameras to mice to the computer.

PART II:
DESIGN TOOLS
AND CASES

APPROACH

As mentioned, the research question for this entire work has been: *How can we go about to create interactive entertainment artifacts in general, and how do we design for emotional response and/or tempting challenge in particular?* I have attempted to get some insight in this topic by simply designing a series of interactive entertainment artifacts. In some cases, trying out a tool was the objective, in others realizing a design idea was the goal, however by using a certain tool to do this. Thus each of the following chapters describes both a tool and a case where it was used. In this they are five different facets highlighting different aspects of the research question.

The aim with presenting the *cases* is to show what could be done, and how, in this way highlighting the endless opportunities of computer augmentation. For instance, I do not want to discuss solely how to create better computer games, but how to create better games in general, using computer augmentation. If this means that the game created is best suited as the typical console game, fine, but it might just as well be about running around in the city, using PDAs to see clues and reach a certain goal. I also want to point out that we can take an existing artifact, e.g. a bike, and somehow enhance it with computer augmentation, enrichening the experience of riding the bike.

The aim with presenting the *tools* was another, namely to actively answer the research question. It should be noted that some tools, e.g. Slow Technology and the Forces-Clashes-Remnant model might as well be used for designing almost anything, but that doesn't reduce the fact that they can be very useful for purposes similar to those described in the accompanying cases.

The facets are as follows:

Facet No. 1: The Interactive Quilt – puzzling entertainment created using Slow Technology. The Interactive Quilt itself was a cross between a quilt and a jukebox, and the tool, Slow Technology, is a concept claiming that sometimes we do need, and appreciate mental challenge. The focus was on the object, not on the tool which was applied unconsciously. The chapter is partly based on a short paper:

Lundgren, S., Johansson S., Nilsson F., Stenberg P., and Thorin, P. (2003) Mapping Fabrics to Music: Lessons Learned. In: Proceedings of The ninth IFIP TC13 international conference on Human-Computer Interaction (Interact 2003), Zürich, Switzerland.

Facet No. 2: The Iron Horse – an interactive toy created using Animal Expression Transfer. The Iron Horse is simply a bike augmented with the sounds of a horse. In the project we actively used the tool Animal Expression Transfer to select and map suitable audial expressions from horses onto the bike in order to create a new object that sounded and behaved in a way logical to both bikes and horses. The focus was both on the object and the design tool. The chapter is partly based on a short paper published in association with demonstrating it:

Landin H., Lundgren S., Prison J (2002) Aesthetic artifacts: The Iron Horse: a sound ride. In: Proceedings of the second Nordic conference on Human-computer interaction October 2002, Aarhus, Denmark, October 19-23, ACM Press.

Facet No. 3: Joining bits and pieces. This was a series of experiments aimed at computer augmenting board games using the concept of mechanics as a tool. A mechanic can be described as a building block for a game, and it is never a strict rule, just a hint of the gameplay contents, e.g. "trading" or "bidding" or "set collection". The focus was on mechanics as tools, rather than on designing games. The chapter is based on the following publications:

Lundgren, S. (2002) Joining Bits and Pieces - How to make Entirely New Board Games using Embedded Computer Technology. M.Sc. Thesis in Interaction Design at the Department of Computing Science, IT University of Göteborg, Göteborg University and Chalmers University of Technology; supervised by Björk, S.

Lundgren, S. & Björk, S. (2003). Game Mechanics: Describing Computer-Augmented Games in Terms of Interaction. In: Proceedings of the 1st International Conference on. Technologies for Interactive Digital Storytelling and Entertainment (TIDSE 2003), Darmstadt, Germany.

Haahr, M., Lundgren, S. and Reijers, N. (2002) Multi Monster Mania. A part of "Designing Ubiquitous Computing Games - A Report from a Workshop Exploring Ubiquitous Computing Entertainment." by Björk, S., Holopainen, J., Ljungstrand, P. & Åkesson K-P. (2002), in Personal and Ubiquitous Computing January, Volume 6, Issue 5-6, pp. 443 - 458.

Game Design Patterns. These are a way to design, discuss and analyze games, i.e. a tool. The Game Design Patterns were based on a pattern language for gameplay design, which we created, thus the focus was entirely on developing this tool. The chapter is partly based on the following paper:

Björk, S., Lundgren, S. and Holopainen, J. (2003) Game Design Patterns, In: Proceedings of Level Up - 1st International Digital Games Research Conference 2003, Utrecht, the Netherlands.

MyTHeme – a storytelling game for non-storytellers. MyTHeme was a computer augmented card game which created narratives, and when designing it we simultaneously detected a tool, namely an event-driven iterative model for multidisciplinary game design, which we later articulated and called the Forces-Clashes-Remnants model. The focus was primarily on designing the game; designing the model was a bonus. The chapter is partly based on the following short paper:

Lundgren, S. and Björk, S. (2005) Forces, Clashes and Remnants: a Model for Event-Driven Iterative Multidisciplinary Game Design, In: Proceedings of ACM SIGCHI International Conference on Advances in Computer Entertainment Technology (ACE 2005)

Research Path

The first project, the Interactive Quilt was actually a project gone wrong. The aim was to create an intuitive interface, mapping fabrics to music, but due to several layers of ambiguity (e.g. everyone's rather individual conceptions regarding fabrics, music and musical genres) it became a puzzling, intriguing object. In this, it gave an important clue to what people can find entertaining and also pointed out that we had unintentionally created an object fitting the Slow Technology design program. The Interactive Quilt also was inspiring in the sense that it was based very much on a mix of emotional and intellectual reactions and actions.

In the Iron Horse project this exploration of emotional response was taken further, since we experimented with how to best map animal characteristics onto another object, in this case a bike. The tool we used here, the concept of animal expression transfer, i.e. transferring expressions from an animal onto an object, thus augmenting the object with some animal traits, can be used when designing for emotional response.

In none of the above projects, the computer augmentation was a goal per se, just a means to reach the goal, but in the Bits and Pieces-project I wanted to explore this particular aspect of creating interactive entertainment artifacts. Thus I set out to create a few computer augmented board games. In this work I very much relied on the concept of mechanics which are "*a part of a game's rule system that covers one general or specific aspect of the game*" (Lundgren 2002, BoardGameGeek), e.g. the notion "trading" or "bidding". The outcome was a set of very specialized game design tools, namely a set of new mechanics designed especially for computer augmented board games, albeit many of them can be applied to any computer augmented game.

The next logical step seemed to be to explore and extend the game mechanics even further, due to an expressed need for a common game design language and design tool. In the Game Design Patterns-project we therefore created a pattern language for gameplay design.

In parallel we also wanted to keep on experimenting with computer augmented board games, and thus we created myTHeme, a high fidelity prototype of a computer augmented card game. Although the original aim of myTHeme was to explore computer augmentation in combination with interactive narratives, the most interesting outcome was instead the design method we came to use. It is interesting partly because it deals with the multidisciplinary aspect of game design, and partly because, again, there is no common terminology or design approach for game design projects, who currently lend much of their methods from other disciplines.

FACET No. 1:

The Interactive Quilt – Entertainment through Ambiguity

In this chapter it is demonstrated how to create an ambiguous interface and why this can be a feature and not a flaw.

> *The party is just starting; its in that initial, slow, low somewhat embarrassed phase that occurs before people get relaxed enough to start talking, that initial phase when everyone tries hard to come across as being nice and when the guests search for common grounds. After a while the hostess comes by, and places a bowl of peanuts on the table next to an oddly looking framed… something…a quilt perhaps, and when passing it she gives it a good pat and suddenly it starts playing music. Beatles' "Can't Buy me Love" fills the air. You look closer. It seems to be nothing more than a golden frame around nine patches of fabric, one white and fluffy, one black and shiny, one made out of a piece from a pair of Levi's jeans etc. Attached to it are two golden speakers. Being curious, some of you walk over to the item, examining it closer. It turns out that it plays different songs depending on which patch of it you press, and there seems to be some kind of inner logic in which songs are played, but you don't really get it. Happily you keep on experimenting, and after a while some ten people have gathered around the quilt, guessing, trying, discussing and arguing vividly. The party has started.*

The Interactive Quilt is actually just a nine-button interface that chooses a random song from nine corresponding folders of music and plays it, albeit disguised as a framed quilt where each patch covers a button. Initially it was a project about creating the perfect intuitive user interface, but it took a different turn and instead became an *intriguing* user interface. In this, it is illustrating of how ambiguity and slow technology can give an entertainment value to an item[6].

Unlike the other chapters in this book, the design case of The Interactive Quilt is *not* described as an example or design method one could follow in one's own design. It is an entertaining story about mistakes, misconceptions and an unforeseen result since the initial design goals prescribed clarity and an intuitive understanding, not ambiguity. It is still interesting, since the case very clearly illustrates the key features of a slow technology object.

6 The Interactive Quilt project was run by myself, Sara Johansson, Fredrik Nilsson, Pär Stenberg and Paula Thorin, being students at the IT-University. The contents in this chapter are based upon the project report (unpublished) and the paper "Mapping Fabrics to Music: Lessons Learnt" (Lundgren et al 2003).

Topic: On Slow Technology and its Counterparts

In Weiser and Brown's vision of *Calm Technology* (Weiser & Brown 1996) computational power should be so common that it became transparent and taken for granted, pretty much like electricity is today. It shall serve us silently and unnoticed unless asked for, pending between periphery to the center of our attention. In any case it should be easy to use, more or less intuitive and non-disturbing. This is of course an honorable approach, but as any other it has its counter examples. For instance, Lars Hallnäs and Johan Redström have introduced *Slow Technology* (Hallnäs & Redström 2001, Hallnäs et al 2001), which, despite the name, can be seen as an opposite to Calm Technology.

Slow Technology is a design program focusing on the need for mental rest and contemplation rather than efficiency and effectiveness. The authors state:

> *"…we believe that we do not only need to create calm technology, we also need actively to promote moments of reflection and mental rest in a more and more rapidly changing environment."*
>
> *– Lars Hallnäs & Johan Redström*
>
> *In: "Slow Technology; Designing for Reflection" (2001, p. 202)*

A basic principle of Slow Technology design is to "*amplify the presence of things*" making them into something else than just neutral tools (Hallnäs & Redström 2001, p. 209). One example mentioned is an ordinary doorbell as compared with a doorbell that has a different signal every time; the different tunes are part of a much longer melody. In the latter case, the doorbell is "slow"; even if it still works as a doorbell it also has a quality of its own, something that makes us reflect on its own presence and behavior rather than just pressing it to get inside. This is rather the opposite of the standard computer augmented item or program where the aim is efficiency – to save time. Still, it can be slow if:

1. It takes time to learn it
2. It takes time to understand why it works the way it works
3. It takes time to apply
4. It takes time to see what it is
5. It takes time to find out the consequences of using it

In a fast application, these are either flaws or tradeoffs made due to complexity. In any case it can be perceived as being frustrating. In a slow context however, it can be about learning and understanding (1 and 2) or reflecting on presence (3, 4 and 5), which is the way we can relate or react on for instance art (Hallnäs & Redström 2001, p 203). Thus, slow objects can challenge our intellects if we choose to engage, providing the same mental rest as working intensely on a cross word or a sudoku. This is in line with the main purpose of a Slow Technology mentioned in the quite above, i.e. to stimulate reflection, thought and exploration.

Taking temporal aspects into consideration is also important; partly because the qualities of the Slow Technology object may both be present only in use (which means it has to be used for some time in order to get to know it) and/or because it may change only very slowly over time, meaning that what it is and does can not be concluded by one short observation. In any case, getting to know a slow tech object per definition will take some time in that the reflection and perhaps exploration will take some time in itself; otherwise the object is not "slow" for that particular user. In this the slow tech takes time, as opposed to a fast tech object whose efficiency aims to *take away* time.

Hallnäs and Redström (2001) also formulate two basic principles – evolving around an awareness of both material and form – for designing Slow Technology:

> *- Focus on slowness of appearance (materialisation, manifestation) and presence – the slow materialisation and design presence of form (F).*
> *- Focus on aesthetics of material and use simple basic tools of modern technology – the clear and simple design presence of material (M)*
> *– Lars Hallnäs and Johan Redström*
> *In: "Slow Technology; Designing for Reflection" (2001, p. 210)*

In extension to this they also state that technology should be used to bring forth the material instead of hiding it. In a later paper (Hallnäs et al 2001) the guidelines are extended with four design leitmotifs for Slow Technology. They are: *Composing in time* (using the temporal aspects of computational technology); seeing the *computational thing as a display*, using the *ubiquity of information* (anything can be perceived as information, e.g. a person moving through a room, or not moving through it for that matter, and there are ways to pick up this information and other ways to display

it); and finally *aesthetics* (sometimes letting appearance being more important than function, asking what a thing is rather than what it can be used for).

Two years later Gaver et al (2003) introduced a similar line of thought by claiming that ambiguity can be used as a resource for design, at least when creating items that are inspiring an thought provoking, which can be a virtue in itself. "Ambiguity" in this case is *"…a property of the interpretative relationship between people and artifacts."* (Gaver et al 2003, p 3), i.e. not lack of consistency or a built-in fuzziness in the artifact itself, but an unique representation for each individual due to his or her interpretation of the artifact. Such ambiguity can be achieved in three ways they claim. Firstly by *ambiguity of information* – whenever we lack information we struggle to fill in what we do not know, trying to create a conceptual model of whatever we see. The example given is the different interpretations we can make looking at a piece of art; is that cigar just a cigar or does it symbolize something else, and in that case what? Secondly, *ambiguity of context*, either by moving one object from one context to another, thus transmuting its meaning, like Duchamp taking ready-made mass-manufactured objects (e.g. a bottle rack) to art exhibitions claiming they were art, or by changing the context of use, like mothers setting a cell phone alarm off so that it rings, giving it to their baby as toy/distractor/rattle. Thirdly there is *ambiguity of relationship*, when the issue of the object is not what it means or does but if and how and why one would use it, i.e. how one relates to it, e.g. a prayer device.

Gaver et al (2003) also list several ways to increase ambiguity, e.g. increasing ambiguity of information by using imprecise representations, over-interpret some data or cast doubt on the information sources. Ambiguity of context can be created by for instance adding novel or unexpected functionality, i.e. by "mixing" two objects, or by taking out expected functionality. Ambiguity of relationship can be achieved by pointing out certain things without explaining why or adding disturbing side effects to the use of the artifact.

Case: Designing the Interactive Quilt

Designing the Interactive Quilt was a student project, being a part of the course "Ubiquitous computing". This course ran in parallel with a course in Human Computer Interaction (HCI), and thus the latter course too, influenced the design process very much.

Initial design process

Since the project was a student project it had very few boundaries, we could do almost anything as long as we created a computer augmented object and reflected on the issues surrounding this. Although the initial discussions spanned over a variety of different ideas, but eventually the all came to evolve around quilts.

The main idea: the Interactive Quilt

Apparently, quilted items had been used as votive gifts, armor, art, payment, everyday textiles and mourning. They were – and are – not only used for practical use, but also for expressing emotions and creativity. Also, the fabrics used in the quilts reflect when and where they were made. Today, after some 5000 years of quilting, a wide variety of techniques and fabrics have come up. With this in mind, we felt that it was natural to take quilting yet another step by combining the newest fabrics with traditional techniques and brand new technology to create the interactive quilt. As generations of quilters before us we wanted to create something that is both practical and beautiful, using different kinds of fabric to express different emotions.

The interactive quilts first imagined could be used in lots of different contexts in a variety of different ways. Finally we settled for a quilt hung in public spaces such as cafés, coffee shops, waiting rooms and the like. It was to work like a primitive jukebox, each patch symbolizing a musical genre.

Design challenge

Due to the nature of the users, being beginners, and the context, being public, we wanted to make an artifact that the users could understand and use intuitively, without mental effort. Therefore the main problem was how to map fabric and music in a successful way that was intuitive to everyone. This was indeed very inspired by the efficiency focus taught in the HCI-course; we wanted to create the perfect user interface. Thus another issue was how to communicate the functionality of the product. How was it to be understood by the user?

Initial design decisions and goals

The basic design idea was to create a quilt where the patches acted as buttons. Already from the start it was clear that the patches should symbolize different genres of music, and that the songs "behind" them should belong to these genres. Also,

it was important for the quilt to have visible speakers attached to it to signal that it somehow can emerge sound; this would not be as important in a private home where the users can be expected to know what the quilt "does". Starting out, we had the following explicit goals:

1. To successfully map different kinds of fabric with different kinds of music.
2. To evaluate if the kind of interactive quilt we want to make would actually be used.
3. To build a fully functional Interactive Quilt

Implicit goals were to make an object that was aesthetically pleasing and very easy – intuitive – to use.

Mapping fabrics to music

This part of the project started with a brain storming process to try to define what kind of fabrics that were wanted. These were purchased and user tested. The intention of this first test was to see if different people made the same associations to musical genres. Eight people, aged between 20 and 30 years, participated, one at a time. They were shown the fabrics, and where requested to speak freely about their associations to music. They were also encouraged to touch the fabrics. This resulted in associations to genres as well as special songs or occasions.

Some fabrics originated very diverse associations, and thus they were put aside, and were not used in the first prototype. Overall, the answers were more diverse than expected, and we set out to conduct a second user test with a bit different premises and a new set of fabrics. Four people, aged between 20 and 30 years, participated in this second test. The persons that participated were first asked to associate freely, just as in the previous test, but thereafter they were asked to associate the fabrics with given musical genres. The genres were pop, blues, jazz and classical music. From these answers it was pretty clear that even when a pretty homogenous group were fed with genres they wouldn't necessary agree.

The first prototype

The first prototype consisted of the nine fabrics that the users had agreed best upon, which automatically settled the selection of genres. These were: Ambient music,

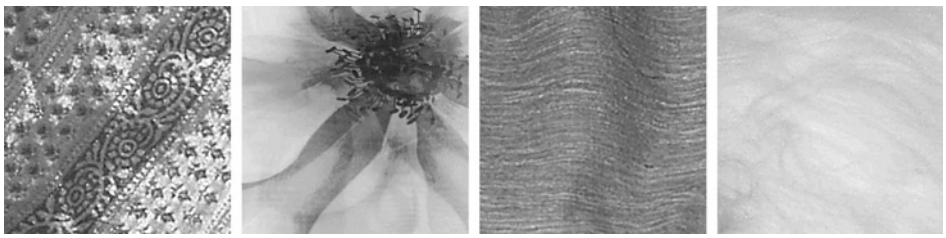


Figure 2: The fabrics tested for the Interactive Quilt differed in color and structure, and in addition some had woven patterns, others printed patterns. In all, eighteen different fabrics were tested.

Figure 3: The first prototype is being tested; computer augmentation is being faked. From the upper left the fabrics symbolize the following genres: Ambient music, Swedish folk ballads, children's songs, romantic ballads, soul, oriental, oldies, rock and country. Also note the loudspeakers placed above the quilt. Project team member Sara Johansson is posing as a tester.

Swedish folk ballads, children's songs, romantic ballads, soul, oriental, oldies, rock and country.

Square pieces of foam rubber were covered with the fabrics, to give an impression of soft resistance and that the fabric was pushed *inwards*. Meanwhile, music (in the form of mp3-files) was collected to each of the genres. The selections of songs was somewhat random, especially for the categories none of us favored. It was also limited by which songs we could find, or already had.

Four people, all between 25 and 35, three female and one male, tested this prototype, one at a time. The pieces were simply taped onto a wall in a square shape. During the first test, the music was played through a computer's loud speakers, but in the following three tests we put external loud speakers next to the quilt. Thus the three last tests had more of a Wizard-of-Oz-character than the first[7]. After the first test we also added a sign saying, "Touch me" above the square.

The users were told to imagine that they were in a public space such as a café, and that the pieces of fabric were sewn together (to prevent them from maybe try rearrange the pieces). After the test they were asked which musical genres the different fabrics represented.

As soon a user touched a piece of fabric a person sitting at the computer quickly and randomly chose a song of that category. A new song was chosen randomly every time the user touched that fabric. If someone touched the same fabric repeatedly, he or she could "zap" through the songs.

The white fluffy fabric seemed to be very inviting. Most users touched it first but it was unclear whether this depended on the fluffiness of the texture or the simple fact that it was the center patch. Most users pressed the fabrics with the entire palm; only one poked with her finger. It didn't seem very likely that the users would actually touch the quilt in a public place. Two of them were actually afraid of what was going to happen when they touched it. One user asked if he'd be electrocuted!

There was great confusion about the connection between musical genres and fabrics. Not all users understood that the fabrics represented genres. Many of them pressed the same fabric a number of times, trying to build a conceptual model of what the quilt was "doing". The only category that "worked" was the country music/ cow fabric. Rock/jeans also worked fairly well, even if some users labeled it pop. The

[7] The Wizard of Oz-method (described by Salber and Coutaz in 1993 and Preece et al 1994) can be used to test the on-going design of a not yet completed computer system. A remote human being ("the wizard") acts out the systems tasks, so that the system appears to be fully functional even if it's not. Typically, the test persons are not aware of the human wizard until after the test. In the test mentioned here, the person playing the appropriate songs on the computer sat in the other end of the room and appeared to be doing something else.

users also vaguely grasped the idea behind the ambient, ballad and soul categories. A very important observation was that the users perceptions and conclusions highly depended on which songs that were randomly played in each category.

Mapping music to genres

Suddenly a totally unforeseen issue had appeared. It came from the observations that the test persons formed their conceptual idea of what each button/fabric did/symbolized after perhaps one or two songs. Then a third song came along that didn't fit their mapping at all and they were confused.

We *had* expected some difficulties with mapping fabrics to music, but were not at all prepared for disagreements and ambiguity in the comprehension of genres. Nevertheless we should have seen this coming since not even we, the members of the design team, could agree on which songs belonged to which category. Should "Hey Jude" by the Beatles be placed in the Rock genre or amongst the Oldies? Or is it perhaps a Romantic ballad? In addition this was made even more confusing than it needed to be, since the category "Romantic ballad" was one of the nine genres. This is a highly subjective category, even more so than rock or blues which actually can be categorized in terms of rhythm, key etc. But any song can be a romantic song if you ask the right person.

Final design decisions

At this point, it was very obvious that the initial goal of creating clear mappings between fabrics and music was out of reach, and that no quick fix was to be found within the timeframe of the project. The final genres chosen were Ambient, Rock, Romantic, Country, Soul, Jazz, Oldies, Blues and Classical music.

Another problem was that the fabrics didn't go well together. It was almost impossible to make good-looking combinations with the range of fabrics that had been appointed. One idea was to use fabrics of only one color, but with different structure, but this would reduce the possibilities of correct mapping, and would also undo all the efforts made by selecting and user testing the current fabrics.

The final solution – found after a long struggle – was to turn the quilt into some kind of art, turning it into a framed picture, and to intentionally make it really vulgar and kitschy with a giant golden frame, and golden loud speakers as well – if it couldn't be made beautiful, why not go with the flow in the opposite direction?

Construction

The Interactive Quilt system consists of three layers. The *interface*, made out of the quilt, the patches being buttons resting on switches, a *middle layer*, consisting of a microcontroller, a BX24-chip processing and forwarding signals from the switches-buttons-patches, and a *back end layer* consisting of a laptop computer receiving signals from the BX24 and playing various songs on the basis of this; through a pair of golden speakers that in turn belong to the interface.

Each patch is a "package" consisting of a flexible plexi glass board, a thin layer of foam rubber, and the fabric. The purpose of the foam rubber was that the patch would appear a bit softer, whereas the plexi glass was needed partly to keep the shape of the patch and create a flat surface for it, but more importantly to divide the pressure so that the switch would be pressed regardless of where on the patch the user applied pressure. This was also supported by the four rubber cubes which also served to keep the patch away from the switch until pressed. In addition the cubes brought with them that some force was needed when pressing a fabric, which gave a nice "feel".

Each patch is mounted on top of a small switch. These were deliberately chosen for the nice clicking sound they made when pressed; this noise gave extra feedback on the fact that the patches were actually conceptual buttons and could be used as such; that they were intended to be pressed at. Each of the switches is wired to a corresponding input on the BX-24 microcontroller. The serial cable of the BX-24 was then hooked to a laptop, programmed to play a random song from the genre the pressed patches represented. If a switch was clicked whilst a song was still playing, a new song was played immediately.

The construction seems to be both robust and functional. The Interactive Quilt was later exhibited for one week straight, being examined by thousands of visitors, without breaking down. It was also set up at an office full of interaction design researchers and ran for several months, which included being shown off to guests and used at entertainments at parties.

Testing the final prototype

The impression of the final prototype was a lot better than expected. It was put in a hallway at the IT-University, and the first person who passed by was requested to come and use it. The music drew more people to the site, and they willingly

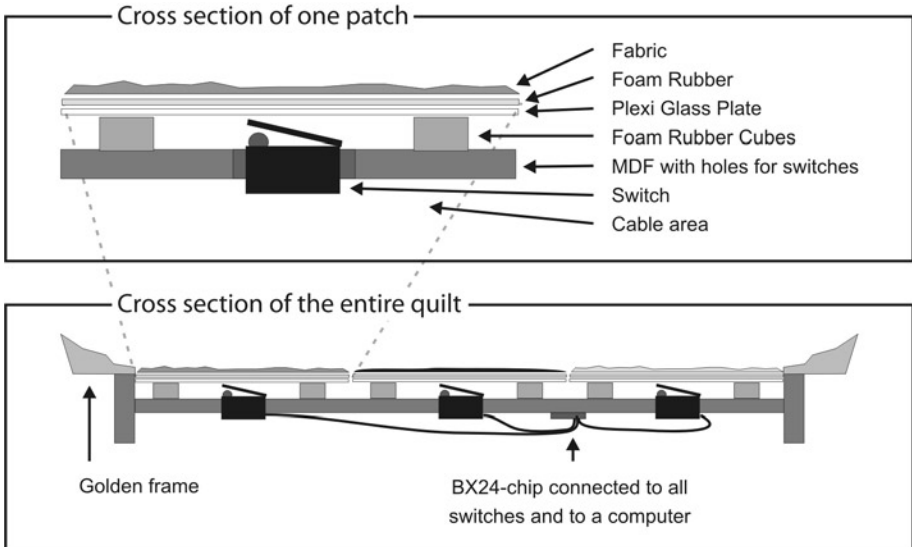


Figure 4: The construction of the Interactive Quilt.

Figure 5: The final (working) prototype is being tried out at the IT-University. The testers are Linda Sjödin and Magnus Nilsson.

interacted with the quilt and had comments about it. It still wasn't obvious to them which fabric was associated to what music, but it didn't seem to matter. Instead they explored it. Of course, one must keep in mind that most of these people are open minded to new technology, and that a more normal range of users would interact more hesitantly. For some reason many users touched the fabric, listened a few seconds to the song, and if it was acceptable to them, backed a few feet and stood listening and looking at the quilt. They hardly ever listened to an entire song; instead they went up to the quilt again and pressed a patch. It was not unusual that they pressed the same patch repeatedly, listening to each song just long enough to recognize it or get an idea about it; this "zapping" was a deliberate attempt to try to unveil the functionality of the quilt.

Thereafter, the Interactive Quilt was tested at a coffee shop in the centre of Gothenburg around noon a busy weekday. The quilt was leaning against the wall, flanked by the loud speakers. The test took about an hour. The users were aged between 2 and 60, and they were filmed as they interacted with the product. The noisy environment made the results less accurate, since the users didn't always hear the songs. A longer testing period and a quieter environment would be acquired for a more accurate result. For example we had to ask people to interact with the quilt which would probably not have been necessary if the quilt was to be left at the café for a longer period of time. However this quick test gave some interesting indications.

The café visitors were looking at the quilt, and seemed interested in it, but would only come up to try it out when we asked them. This could depend on general shyness or unwillingness to be filmed. After a while when some people had come up to interact with the quilt other followed. They were a bit shy but the response was positive. It seemed to be a surprise to the users that the quilt played music. After a while they discovered that each patch contained several songs, which also astonished them. They also noticed that there was a connection between the fabric and the different genres of music. The response to this mapping was good, and they agreed to most of the mappings when told. They were impressed by the functionality of the product, though they did not always like the physical appearance of the quilt. Even if they seemed to enjoy toying with it they didn't stay for more that a couple of minutes. Exactly why they didn't stay longer is not clear; it might be because they felt uncomfortable in the testing situation, because we filmed them, because they lost interest or because they were in a hurry.

Like in the previous test, the users pushed the buttons often to change songs. This was probably because of the test situation, but could also be an indication of curiosity; which song would be played next? The original idea was that people would press it once and than listen to an entire song, but no one did. Later, more informal observations at parties confirm this observation; the quilt isn't used as a jukebox, as we intended, but as a puzzle. Also it seems that once someone has broken the shyness barrier and has started interacting with it, others gladly follow.

Reflections – Designing Towards Slow Technology

Problems

Clearly, the Interactive Quilt did not turn out the intuitive interface we had intended it to be. As is obvious by now, we never succeeded in creating clear mappings between fabrics and music; how and why will be discussed at length below. However it is our opinion that the quilt was liked and could actually be sold, however most likely as a more fancy stereo player in the home. The "shyness" factor might be an issue, preventing the jukebox use in public environments such as a café as we had intended. Then again, the ice-breaking effect that occurs when someone starts using it and others join might be valuable enough to overcome this.

Our tacit goal to create something beautiful wasn't fulfilled either. This however was simply a side effect of finding perfect mappings. This parameter guided the selection of fabrics, so they were chosen without their aesthetical qualities in mind. If we were to build another version of the quilt, this would be re-prioritized since the mapping can only be made arbitrarily anyway.

Lessons learnt

Given that the Interactive Quilt was a student project it was rather well carried through in terms of user tests and design iterations depending on the outcome of those. The focus on user tests came partly from the wish of making the perfect intuitive interface, and both this wish and the way to go about was very much influenced by a HCI-tradition. This approach was important because it very early indicated that the original intention of making the interface intuitive was much

more complicated than foreseen, and it also pointed out the problem with the different comprehensions on music genres. It may be just as useful when deliberately designing an object that shall be ambiguous, since it will guide towards the right level of ambiguity.

In retrospect, being aware of the complexities of mapping fabric to music it would have been better to focus on the aesthetic expression by picking materials that looked and worked better together. Possible paths here would be to choose fabrics from the same range of color, using structure to distinguish them, or to use fabrics with similar structure having carefully chosen colors that work well in combination. Or, the shape and size of the patches could be used as a design element as well.

Overall, when creating an artifact that is to be perceived as being beautiful rather than practical and efficient, a lot more care has to be given to expression and choice of material, than we did in the Interactive Quilt project.

Possibilities

Now, everything stated under "problems" above is related to the initial goals, in which we aimed towards a quilt that was a practical, efficient artifact – a fast artifact, as opposed to slow. Thus, we initially thought we had failed, and in that sense of course we did. But interestingly, the obscured mappings were only seen as a failure by *us*. The *users* didn't mind, they just happily played along and as it turned out this was the key feature of the Interactive Quilt. The ambiguity of the Interactive Quilt did not trigger the user's interest per se, but it *kept them interested*. It gave them something to explore, something to talk about, something to contemplate over, something that intrigued them, challenged them and did not simply function in a simple and foreseeable way. In this sense the Interactive Quilt is a well-working Slow Technology object. In addition, it actually *is* easy to use in a way; you press a button – you hear a song. This means that interacting with the interface is not hindering the exploration of it; almost ideal conditions for anyone wanting to create a puzzling object.

The reason that the Interactive Quilt is "slow" is that it features as many as three highly subjective dimensions: the comprehension of fabrics, of music and of musical genres respectively. Thus the understanding of the quilt's functionality is unique for each person, meaning that the designers' understanding differs from everyone else's,

in turn meaning that for everyone but for the designers the quilt will act more or less unpredictable. Having these subjective dimensions can be a way to achieve what Gaver et al (2003, p 3) mean when they claim that *"by thwarting easy interpretation, ambiguous situations require people to participate in making meaning."* When analyzing the Interactive Quilt according to Gaver et al's premises it features both ambiguity of information – using imprecise representations – and ambiguity of context – it is a mixture of two objects, and it can be either a quilt with unexpected functionality or a music player with removed functionality.

As for the slow qualities that Hallnäs & Redström (2001) point out, the Interactive Quilt is a above all categorized by the fact that it takes time to learn it and to understand why it works the way it works, i.e. the two properties that are related to learning and understanding – which is why it challenged and entertained its users.

Designing the "Slow" Interactive Entertainment Artifact

Earlier we have discussed the definition of the interactive entertainment artifact, which states that computer augmentation is an essential part. To clarify the connection between slow tech objects and interactive entertainment artifacts we must remember that the design guidelines for Slow Technology state that *"simple basic tools of modern technology"* (Hallnäs and Redström 2001, p. 210) shall be used to bring forth material, presence and behavior. This is exactly what computer augmentation should do. It is not a purpose in itself, it is simply supportive, aiding in increasing the design space. Since many slow tech objects are about reacting on use or information, often changing their behavior over time, computational technology is an essential material in any slow tech object – and so is entertainment since they are made to encourage mental challenge in terms of reflection, exploration and play.

When applying Slow Technology as a design approach in order to design mental challenge in the form of entertainment, one must be aware that this brings with it some issues that are inherent in any entertainment artifact.

Firstly, like art, the Slow Technology objects can impossibly be liked by everyone, since humans are different and need different levels and kinds of challenges. What is interestingly puzzling for one individual may appear as too complex for someone else whilst yet another person finds it boring, uninteresting or to "deep". This issue is

always present, regardless of the method or approach used to create an entertainment artifact.

Secondly, the user will at some point either have understood the object, given up on it or lost interest in it, in which case s/he probably will abandon it unless it can actually provide some useful functionality, e.g. like the Interactive Quilt – once figured out it can still serve as a jukebox or music player. Without this feature, the object will be like a puzzle or a riddle; once solved it is no longer of value. This may be inevitable and not that bad in itself, one example being the 1993 game smash hit *Myst*[8], which is essentially a set of riddles and puzzles set in a series of beautiful and mystic places (12 million unit sales total across all platforms and sequels in 2003)[9]. It's a wonderful game, but once finished, it's over; the replayability value is very low. Then again that was never an intention of the designers.

Thirdly, balancing ambiguity and usability is a difficult task. The risk of the slow tech object being perceived as just being weird and useless ought to increase the more complex the interactions are, especially if the interaction with it is very strained, creating a barrier between the user and the object. Unless, of course, what the object is and does is very obvious and the whole challenge is to find out how to interact with it; imagine a computer without keys that takes whistling as input for instance. Anyhow there is a risk that the task of mastering the use of the object is so frustrating that users give up; again the right balance must be found. I believe it is important to separate usage and understanding from each other, making one (preferably the usage?) clear and intuitive, and understanding and hidden meaning complex. Again returning to *Myst*, the puzzles, the riddles of the hidden story and the environment were beautiful and intriguing whereas interaction and navigation was very simple. There were only a few possible actions in the game, actions like turning 90 degrees, moving forward towards something that could be seen (the cursor changed to indicate that a certain path or direction was "allowed") and clicking on objects on wanted to interact with, which was again indicated by the cursor. However one could not choose what to do with the object, it by itself performed its predefined action.

To conclude, I believe that Slow Technology and deliberate ambiguity can clearly be an approach when designing artifacts that shall be mentally challenging and thus

[8] *Myst* by Robyn and Rand Miller et al 1993, published by Cyan Inc.

[9] Data found at http://www.tiscali.co.uk/games/myst/history.html (2006-09-22).

entertaining. Both Hallnäs & Redström's (2001) reflections as well as the ways pointed out by Gaver et al (2003) can be used to steer design processes.

However, in order to satisfy the "usage" part of the artifact, it may be suitable to combine it with more traditional approaches and methods from the Human Computer Interaction and the Interaction Design disciplines. Furthermore, the artistic qualities of the object mustn't be forgotten in that designer's mustn't forget to consider material choices, graphical design etc in order to support the underlying logic that is hidden behind the ambiguity.

FACET No. 2:

The Iron Horse – Creating Entertainment with Emotion

This chapter deals with the concept of expression transfer and how it can be applied in order to create an object with a convincing expression and behavior of an animal, in this way creating a strong emotional bond.

You canter at full speed along the small forest road, and you can smell the grass, hear the regular clippety-clop sound from the hoofs and feel the wind and the sun against your face. It's a perfect summer's day and you feel good! The road is winding uphill and you slow down to a trot. At the top of the rise you decide it's time for a breathing pause, and you slow down even more, to a walking pace, whilst admiring the view. The air is clear and you can see very far; your village looks so small in the distance, and your own house so tiny. You have gone far today! But! Suddenly you hear the sound of other hoofs! Within a few moments your friend Susannah shows up at the ridge, red-cheeked and breathless.

"I saw you!" she says, accompanied by greeting neighs. "Oh boy have we been riding hard to catch up with you! Didn't you hear me call?"

But you didn't; you were too immersed in your own thoughts and the experience of riding through the countryside on a sunny afternoon. With an apologizing smile you shake your head and call out a friendly challenge:

"C'mon! The last one down pays the ice cream!"

The four of you set off, the steep downhill slope triggering your Iron Horses to canter as fast as ever You win the race, and eventually you return home, full of ice cream, sun and play, leaving your Iron Horse in the garage. Tomorrow is another day. Tomorrow you can play with it again.

The Iron Horse was a bike – that sounded like a horse! With attached loudspeakers and sensors measuring speed, tilt and proximity to the owner it mimicked the audial expression of a horse using hoof sounds, neighs and snorts[10]. The main idea came from a wish to create a toy we really would have liked when we were kids (well, some of us at least). The aim was to create something with a flawless and seamless inner and outer logic; all of its expressions (what functions it expressed and what functions it actually had) should match and create a whole. In this way, an extra entertainment value was added to the bike via the extra emotional responses that

[10] The Iron Horse project was a project run by myself, Magnus Johansson, Hanna Landin and Johannes Prison, being students at the IT-University. The contents in this chapter are based on the project report (unpublished) and the paper "The Iron Horse – a sound ride" (Landin et al 2002).

came from the horse and that turned into an entirely new object of dual nature: the Iron Horse. Creating this duality was the most interesting part of this project.Thus, the contents of this chapter is a description of the concept of expression transfer and how to apply it in order to create an object which is a hybrid between an animal and an item. This is interesting since animal traits often evoke emotions and affection, which in turn makes an object richer from an entertainment point of view.

Background: The Family Tree of the Iron Horse

Humans have a tendency to inscribe behavior and emotional response into almost anything, including for instance a computer. This common phenomenon is called anthropomorphism, and it can be described as follows:

> *"We have evolved to interpret even the most subtle of indicators. When we deal with people, this faculty is of huge value. It is even useful with animals. Thus, we can often interpret the affective state of animals – and they can interpret ours. This is possible because we share common origins for facial expression, gesture, and body posture. Similar interpretations of inanimate objects might seem bizarre, but the impulse comes from the same source – our automatic interpretive mechanisms. We interpret everything we experience, much of it in human terms. This is called anthropomorphism, the attribution of human motivations, beliefs, and feelings to animals and inanimate things. "*
>
> *– Donald Norman*
> *In: "Emotional Design: Why We Love (or Hate) Everyday Things," (2005, p 136)*

We say (and think) thinks like "The computer does not *want* to start", "That coffee machine really *hates* me!!!" inscribing intentions and emotions into the most trivial everyday objects. The most common example must be the strong emotional response children express towards toy animals: "Eeyore *is sad* because you didn't let him come to school with me. He says he has *been longing for m*e all day!"

The reason that we care extra much for toy animals is that they, like many Disney cartoon characters, are what we perceive as being "cute". Apparently a baby like-appearance (large eyes low in the face and pudgy cheeks, large head and short limbs)

elicits tenderness and affection towards not only babies but any beings with a similar appearance (Pinker 1997, p. 445). In addition toys are small, soft and lack sharp teeth, claws or anything else that could be perceived as dangerous or threatening.

Donald Norman (2005) too, comments on this appearance in relation to robots and anthropomorphism, saying that robots that should serve human needs, e.g. robot pets should have "…an animal or childlike shape together with appropriate body actions, facial expressions and sounds" (p. 176) in order to interact successfully with people.

Toy ancestors: tamagotchis, Barney, furbies and the AIBO

There are numerous toys that use animal traits as a way to create emotional responses – the average teddy bear being the perfect example – but not as many computer-augmented ones. Four very significant examples are Sony's robot dog AIBO, Hasbro's Furby (a small fuzzy toy), ActiMates Barney and of course the tamagotchis.

Tamagotchis are virtual pets. They first showed up in Japan in 1996 and can be seen as the first digital pets. They live in small boxes, manifesting themselves as pixels on an embedded screen. The twist with the tamagotchis was that they need nurturing in order to stay alive and grow; they do not only need food, but also snacks, medication, and they also want to play and be caressed. All of this is done via various buttons.

Shortly thereafter in 1998-99, ActiMates Barney – a purple dinosaur – showed up on the marked, having been developed by the Microsoft's Interactive Toy Group. Of the mentioned toys, Barney is the only one with a clear educational purpose. In his freestanding mode he can sing, play games, and play peek-a-boo, but when the child is at the computer using a software featuring Barney, he will (via radio link) get a more complex behavior, reacting on what the children do on the screen.

Figure 6: A furby. Note how large the eyes are in proportion to the body.

The concept that a digital being could have its own personality and need training and care was taken a step further in the Furby concept. The Furby is a small furry toy created by Tiger Electronics, now owned by Hasbro. Unlike tamagotchis it doesn't get disappointed if ignored for longer periods if time; if ignored long

enough it just dozes off to sleep. A new idea was that several furbies can interact with each other, unlike Barney and the tamagotchis respectively. Also, they speak their own language, furbish, but do speak more and more English if being played with; one of the first phrases being "I love you"!

Of the toys mentioned, the AIBO robot dog is the only one that tries to mimic an existing animal. The first AIBO was released in 1999, and Sony labeled it "The First Ever Entertainment Robot". Since, four other versions have followed. While the first version was rather dog-like, the latest model (from 2003) can do lots of things dogs can't, like for instance play music and record images, as well as speak to its owner. It even blogs! In addition it has to be trained by its user in order to develop mentally.

Kahn et al (2004) made a study comparing how preschool children related to an AIBO as compared with a similar plush dog, finding that the children to a great extent saw the AIBO as more "alive" than the toy dog. The children were more often engaged in activities concerning reciprocity and apprehension when playing with the AIBO than with the plush dog, and vice versa they much more often mistreated the plush dog. This suggests that the intelligent and interactive behavior of the AIBO perhaps triggers a stronger emotional response. In comparison, Melson, Kahn et al (2005) made yet another investigation, comparing children's responses to an AIBO as compared to a live dog. As – perhaps – expected, children reacted on the AIBO by treating it in quite dog-like ways, however treating the dog even more affectionate, staying more close to the dog than the AIBO and touching it more. Moreover, children were essentially evenly split on whether AIBO was more like a desktop computer or a live dog.

Paro – the robot seal

Wada et al (2005) describe Paro, a robot looking like a baby harp seal. This form was intentionally chosen in order to avoid preconceptions about the animal's nature; seals are normally not as familiar as dogs or cats. It can recognize speech and differ between darkness and daylight, and it also has a built-in tactile sense so that it can react on being touched, which it does by moving its neck, paddles and eyelids.

Paro was built for therapeutic use, and was therefore tested at a pediatric ward, improving children's mood and even triggering an autistic patient to recover his ability to speak within the eleven days the test was run. Paro was also tested at a day service center for elderly for five weeks. Again, Paro improved the patient's

moods and led to significant improvement in the mental state of some. As in other experiments (with the AIBO for instance), the users were aware of the dual nature of the robot. Some liked it very much and treated it much as a child (even if aware that it was a robot), others liked it and considered it being "well done", and one patient claimed not to like it that much (but nevertheless responded physically to it). There wasn't an exact correlation between liking Paro and improving health-wise.

One patient did not like animals, regarding them as being filthy, smelly and dangerous but she liked Paro: "Paro never bite me, and it's very clean. I want this one!". Interestingly, to her the robot features of Paro were positive traits whereas for others, these traits can muddle the impression and emotional response to the robotic toy.

PillO'Mate - a descendant of the Iron Horse

Some six months after the creation of the Iron Horse, another group of students at the IT-University chose to use computational power to apply animal characteristics onto an object; PillO'Mate (McGee & Harup) was a cushion with some of the behaviors of a cat. The main purpose of the project was to create a calm and friendly product, serving as a companion for lonesome people and soothing for people under pressure. Therefore the PillO'Mate is expected to be used pretty much as we use a cushion – and a cat – when in need of comfort, i.e. being held in the lap, perhaps hugged. Hence, the project was oriented very much towards expressions, trying to copy the wished "cat expressions". Thus critical goals was for it to have the same weight as a cat (i.e. heavier than an average cushion), to be able to emerge heat similar to the body temperature of a cat, and the possibility to purr. In addition technology had to be very well hidden in order not to ruin immersion.

Just like a cat, the cushion needed some attention in order to react; it would get warm and purr *only* if held and patted respectively. When user tested there were clear indications that users inscribed a personality into the cushion: "*It has stopped [purring], it must want me to keep stroking it*". Interestingly, people have different notions of a cat's purring; they commented heavily on that particular aspect, wishing for it to be louder, quieter, faster, slower, more varied etc. The other aspects (the size, form, weight and fur) weren't commented as much at all. One explanation to this could be that these were seen more like properties of a cushion – and a cushion can have various shapes and forms – whereas the purring was the most cat-specific trait of the PillO'Mate.

Topic: Expression Transfer

The main issue in the Iron Horse project was to seamlessly integrate the notion of a horse into the bike. Our approach to do this was to use the concept of *animal expression transfer* as our guiding star throughout the project. This tool is based upon concepts like *expression logic* and *epression transfer* in general, as being described below.

On expressions and expression logic

Expressions are manifestations. Normally, the expressions of an object are a combination of expressions from many dimensions: size and shape, color and looks, sound, smell, taste, feeling and last but not least how it acts; what it "does", how it "works". Together, the expressions form what is commonly called the *gestalt* of the object; i.e. *"…an arrangement of parts which appears and functions as a whole that is more than the sum of its parts"* (Monö 1997, p.33).

When studying natural objects, like animals, plants, mountains, weather etc., we can conclude that their appearance and – in applicable cases – their behavior is dictated by a set of "rules". Take a dolphin for example. It has a streamlined shape in order to swim as fast as possible; this is also why it lacks external ears. It whistles, since this sound is well heard under water. If we look at a picture of it we can conclude – without knowing anything about dolphins – that it lives in water since it has fins and no legs, and that its sharp teeth indicate that it is a carnivore, etc. Every detail of the dolphin's expressions makes sense – follows the "rules" – when related to the context dolphins live in. It does not have antlers or a trunk since these features do not fit into its way of life – adding those would be to obscure the logic of the expression it makes. Thus, there is a strong connection between the expressions and the behavior of an object. This connection can only be verified and studied *in action*. We cannot be sure if the object behaves what it expresses without using it. Just looking at a picture of a dolphin does not tell us anything about the social structures of it, how it sounds or in which ocean(s) it lives, or which prey it prefers. To do this we have to study the behavior of living dolphins; we need to see dolphins *in action*.

Human made objects, too, have gestalts, sets of expressions somehow indicating what they are and how they can be used. ***The act of designing is the act of creating the logic that is inscribed into the expressions of the object***. If this underlying logic is

consistent, the expressions of the object will be consistent, at least in the right context of use. A chair can be used to sit upon, just as intended, and just as it expresses. If we try to use a hammer as a chair however, the context is wrong and it will not work; however this is unlikely to happen, since the expressions of a hammer clearly indicate that it shall be used for something else than sitting, preferably hitting something.

Expression Transfer

Based on the discussion on expression logic, we[11] would like to propose the design method of Expression Transfer. This is a design method where a subset of the expressions and behaviors from one object are mapped upon the expressions and behaviors of another object, in order to create a hybrid object. One could argue that such a hybrid object is nothing more than a metaphor, but unlike the metaphor, the purpose of expression transfer is not to clarify, indicate or explain certain functionality. Thus, Expression Transfer is not made with a pedagogical purpose, but to create a new kind of object existing *in its own right*, not representing anything in particular, just being itself. It can be used to augment the existing expressions of an object, or add on new functionality or new expressions to an object with the deliberate purpose of enrichening it somehow.

Animal Expression Transfer

Although we believe that Expression Transfer can be applied to all kinds of objects, we have currently only been working with transferring expressions from animals onto objects. This can be used to add emotional value to an item, since users are more likely to anthropomorphize such an object.

The Feline Car

Imagine a flashy, high-powered car; a Corvette, a Porsche or a Jaguar. Now, imagining transferring some of the traits and behaviors of a big feline, i.e. a tiger onto it. What kind of hybrid object would this result in? Which traits and behaviors are suitable to transfer from the tiger to the car? Well, the visual appearance and physical structure of the both are highly indifferent and probably not suitable to merge. However, there are similarities. Both signal power. Both are part of an invisible and volatile

[11] The initial ideas regarding Expression Transfer were outlined by myself, Hanna Landin and Johannes Prison, and are currently being more elaborated by Hanna Landin and myself. Hanna and I have also invented the concepts of the Feline Car and the Puppy Printer together.

ranking system depending on age, shape, speed and general appearance, and both express their superiority with a roaring sound. Both can be said to have a territory, which they roam. If focusing on power as an aspect of rank, we can transfer the following behaviors:

If two Feline Cars meet, only the one with the highest rank will roar. This means that that car's owner or driver will feel satisfied, but perhaps also that he or she will drive and treat the car differently in order to keep its rank as high as possible. The rank in turn is depending on some set factors like age and model, but can also be increased by being close to home (i.e. in its own territory) or having a sufficient mileage (i.e. experience). It is decreased if the tank almost empty or if the car has recently been taken for a long drive (i.e. is tired).

In addition, the driver can hear the Feline Car purr when it is satisfied (e.g. when it is being tanked or is sun bathing). If possible the motor sounds heard from inside could be muted or increased respectively depending on if the Feline Car is in its own territory or sneaking through someone else's.

The Puppy Printer

Puppies are cute. They are also insecure, attention-seeking and aim to please higher ranked members of the pack. Office printers however are often stored away somewhere far and not eager to please at all – unfortunately. If we combine these we get the Puppy Printer. It is wireless, and roams the corridors seeking for attention and love. It comes on a whistle, and will eagerly greet the person who comes first to work. It tends to linger in the vicinity of larger groups, checking what is going on. If being annoying, it can be shooed away. It sometimes yelps, sighs and whines happily or unhappily, depending on how it is being treated. Like a puppy it also needs to be fed (with paper and ink or toner) every once in a while.

Applying Expression Transfer

How one applies Expression Transfer depends on the design objectives – the goal. In the Feline Car case, the aim is to strengthen one aspect of the car; showing (off) power, and thus another object (animal) with *similar* expressions was chosen and from it a set of suitable expressions were transferred to create a logical whole. In the Puppy Printer case it is the other way around; a wished behavior was taken from the puppy and transferred upon an object *lacking* this behavior. In both cases the expressions of the transferring object (animal) are examined carefully in order to see which ones are suitable to transfer in order to fulfill the goal. Thus the goal has to be

clear, since it shall be used to motivate all the transfers, thus providing some kind of expression logic to the hybrid object (which can be seen as an expression-augmented version of the object the new expressions have been transferred onto).

Case: The Iron Horse

The Iron Horse was a project explicitly exploring the aspects of expression logic and, as it turned out, animal expression transfer.

Design Challenge

The main design challenge in the Iron Horse project was to transfer the expressions from a horse onto a bike in a way that seemed logical and thus consistent. Creating the hardware and the software wasn't really the issue per se, however creating the expression of them together with the bike, was. This was critical, since one of the most significant qualities of games, and of free play is *immersion*, and immersion suffers from, or can be scattered by, inconsistent logic of the interface; it must be easy to use and feel natural. The technology supporting immersion must therefore be transparent, hidden, not obvious, and the interface must be so natural that hardly any thought needs to be used to handle it; it should be an extension of body and mind, not a troublesome layer between the user and the entertainment. Also, immersion can be strengthened by giving the user free hands in creating it; if your mind is free enough you can immerse yourself into a world where a stick can be anything; a dagger, a wand, a ladle or a scroll. This freedom can be achieved by letting the user themselves add certain dimensions to items, i.e. how they look, how they feel or what they can do, or all.

In line with this, we wanted the children to feel that the Iron Horse was a horse in some aspects, and that it had its own personality and behavior; most of which could be inscribed by the users themselves, for instance the physical appearance of the horse part of the Iron Horse, which is solely imaginary. In order to avoid disturbing inconsistencies and unexpected behavior, we had to be very careful when designing the expressions of it – we had to find a *logic of expressions*, coming from horses but suitable to bikes. This was the key factor in the project.

Figure 7: The final version of the Iron Horse, having the visual appearence of a bike with some extra stuff attached to it. By solely looking at the Iron Horse, it is not possible to conclude what it "does" – the "Iron Horse-ness" is hidden in its sounds and only appears in action, when riding it.

Physical limitations and scope of design

In the Iron Horse case the hardware already existed, since we used an existing bike and added some technology to it, rather than designing the whole Iron Horse from scratch. In this sense, the Iron Horse, when not sounding, has the *visual* expressions of a blue, 1985 Crescent bike with a carrier bag and some other stuff attached to it. Obviously the designers of the bike had their own agenda and their own goals when designing the bike. The colors, the materials, the design of the different part were all chosen for various reasons, e.g. usability, durability, beauty, ease of mass production, price etc. All these variables were more or less set.

Thus, our area of design would become to change the expressions of the bike *in action*, in this way turning it into an Iron Horse. Now, since the whole "Iron Horse-ness" was hidden in the sounds it emerged, it to some degree went back to its

bike-state when quiet. However, for anyone who had experienced it for only a short while, anthropomorphism acted, creating a stronger emotional bond, and of course this connotation changed the way these people interpreted the visual expressions of the bike, i.e. still regarding it as an Iron Horse even if all the expressions that turned it into this were currently not present.

Stating the Design Objectives

Before designing anything, i.e. before starting to deal with which set of expressions to transfer, the design objectives (Jones 1992, p 194 - 200) have to be stated. This simply means analyzing or identifying the apparent goals as well as the context of use, expectations from sponsors (if any) and available resources (money, time, skill, material etc.). If some of these are contradictory, adjustments have to be made, but finally there is an overarching specification of what to design; a goal.

Luckily, the context of use was very intertwined with the essential objectives. The main idea with the Iron Horse was, again, to create a toy that we wished we'd had. It was important to us that the (audial) expressions should be as horse-like as possible, but still follow the limitations of the bike (i.e. the expressions of the bike should not collide with or contradict those of the Iron Horse). It should encourage play, and to favor fantasy and creativity. The latter led to an early and very strong design decision: the Iron Horse has no built-in need for training, caretaking or cuddling, thus avoiding any kind of duties or demands, and it has no functions for getting attention or focus in any way. It is up to the user to decide to which degree he or she wishes to tend to his or her Iron Horse. In line with this, we did not want to inscribe any "rules" into it that could somehow steer a game; any application designed for the Iron Horse should only be supporting play, not steering it. Yet another design decisions related to the values of joy and play was to not transfer any aggressive behaviors from the horse to the Iron Horse.

As for limitations, there were some due to the fact that we had a limited amount of time and skill (prohibiting us from putting any effort on creating specially designed loudspeakers and such), and there was an explicit demand that the Iron Horse should be robust enough to be exhibited for a week, during which it was to be explored by thousands of children. In addition we ourselves set the limitation to only use sound to create the Iron Horse. One might argue that the Iron Horse could have become more horse-like with a mane and a tail somehow attached to the bike, but that would have imposed ideas of the color of the Iron Horse, something we wanted to leave to the user to imagine. Perhaps an Iron Horse's color varies from day

to day? In addition, this is something the young bike owners can do by themselves. We also discussed having a moving saddle that would mimic the movements of a horse, when riding the bike. This idea was left out for three reasons; traffic security issues, lack of time and lack of competence in implementing it.

Transferring expressions from horse to Iron Horse

Since we aimed for immersion, we decided to look for similarities between horses and bikes and use them as a basis for which sounds and behaviors to transfer. Thus the relevant issues were: which sounds exist and which of those are suitable for the Iron Horse (i.e. fit both a bike and an Iron Horse), and how and when are those emerged. The latter is important since it creates a link between horse behavior and bike behavior, creating Iron Horse behavior in the intersection of these. According to our way if dividing horse sounds, there are three categories:

- Sounds made by the hooves: gait sounds (e.g. walk, trot, canter), kicks etc.
- Sounds made by the mouth: neighs, whinnies, snorts, puffs, breaths, sighs, chewing, drinking etc.
- Sounds made by rest of the body: Standing up, lying down, scratching, urinating etc.

A bike does not have a body in the sense that it doesn't have a lot of flexible limbs and can change position as it wishes, and it certainly has no digestion system. Thus, all of the "body" sounds were excluded as well as the eating and drinking sounds made by the mouth. As for the hoof sounds, a bike does not have explicit feet, but it *has* wheels that serve the same purpose; to support movement. Therefore, the gait sounds (walk, trot and canter) were kept, and they were simply and naturally mapped to the speed of the bike. However, a bike's wheels don't ever leave the ground as kicking hooves, so the kicking sounds were excluded as well, which was just as good since these were considered to be aggressive and would thus have been excluded anyway.

Moreover, a bike does not have a mouth. Neither does a bike have a personality, but an Iron Horse does, and for various reasons we chose to manifest the Iron Horse's personality by using the sounds being left over for this; neighs, whinnies and snorts.

Horse behavior as a design rationale for Iron Horse sounds

The way the Iron Horse sounds can all be explained by taking a closer look at horse behavior. Wild horses live in family groups, consisting of a stallion, up to ten mares and their offspring. The stallion's task is to protect and cover his mares, and to keep other stallions – including his own sons – away. He herds the group, keeps watch and may occasionally scare off smaller predators that'd otherwise prey on the foals. There is also an alpha mare that can take care of the family if the stallion is dead or away rambling. She is involved in raising the foals, and she can punish them in the worst possible way for a horse – by temporarily expelling them from the group. No wild horse likes being away from its family since the risk of being attacked is much higher. This strong herd instinct affects a lot of horse behavior, and in order to support it, each Iron Horse has a built-in capacity to detect the presence of other Iron Horses and it's own owner (via radio signals).

For instance, one of the most common horse behaviors is to move at the same pace as surrounding horses. If, for instance one horse makes a bolt, it is likely that it takes some others with it, and that those left behind move faster too. In order to transfer this behavior from horse to Iron Horse, the idea is that if a group of Iron Horses are ridden together and one for some reason chooses to slow down or leave the group, there will be a certain delay in the change of the gait sound of one's Iron Horse. It has to move slower than normal before the sound changes from trot to walk.

Horses also greet other horses with neighs and snorts. This behavior too, is implemented in the Iron Horse, since it can recognize other Iron Horses as well as its owner. Also, horses who want to play sometimes indicate this by rearing and neighing, which is why an Iron Horse may also neigh if reared.

Just like horses have their own personalities, Iron Horses do. They are more or less likely to neigh, snort or whinny in various situations (for instance if being prompted to move), and they do not change from walk to trot and from trot to canter (or vice versa) at the exact same speeds as other Iron Horses.

Note also that just in analogy with reality, the gait sounds are the only ones the rider can control. The neighs etc however, are controlled by the Iron Horse itself.

Realizing the Iron Horse – technical setup

In order to support all of the behavior and requirements stated above, the hardware and software system of an Iron Horse had to fulfill the following requirements:

- Keep track of the bike's speed
- Keep track of whether the pedals move or not
- Keep track of the bike's tilt
- Detect presence of owner
- Detect presence of other Iron Horses
- Signal own presence
- Process data collected as above
- Play sounds
    - Hoof sounds
    - Neighs & whinnies
    - Snorts

Keeping track of the bike's speed and if the pedals moved, were both solved using magnets and a magnetic field sensor. In order to measure the speed, two magnets were placed on two opposite spokes of the back wheel. The sensor was then placed on the rear fork of the bike, measuring the frequency at which the magnets passed it, i.e. how fast the wheel spun. Two magnets were used since with only one the intervals were very long at a slow pace and it was hard to detect when the bike had stopped, leading to that gait sounds were still played for a few seconds even if the bike had stopped. Having two magnets solved this. Having more magnets would have made this even more precise, but could not be used, since the sensor had to have a certain range, and having too many magnets would then make it detect magnetic fields all the time above a certain speed, making it impossible to measure differences in speed above this point. In a similar way it was measured whether the pedals moved; one magnet was placed on the pinion, with a sensor on the frame next to it. Keeping track of the tilt was done by placing an accelerometer on the crossbar.

In order to make the Iron Horse context aware, radio signals are used; each Iron Horse has a transmitter and a receiver, and in addition there is a transmitter placed on for instance the bike's key or on something else the owner is likely to carry on him or her. Note that since only one Iron Horse prototype was built, the herd instincts were never implemented, and no radio transmitter was placed in the bike. However it had a working receiver acting on the signals transmitted from the key.

All of the sensor data was collected by a BX24-chip (natural size 34 x 17 mm), processed and sent to a small laptop running a Java program that is used to handle the playing of the sound files. The gait sounds are simply determined by measuring the speed and changing the gait sound whenever a certain speed limit is crossed. When it comes to neighing, whinnying and snorting however, a certain chance is involved. For instance, it was a 70% chance that the Iron Horse would snort if being urged to walk faster, i.e. if the pedals had been still for a while and the user then started threading. Similarly there was a certain chance that the Iron Horse would neigh if reared or if the user approached. Clearly, all of the Iron Horse's behavior is hidden in the software. A lazier Iron Horse can be created by changing the thresholds at which the gait sounds change, or by changing the probabilities for neighing and snorting at certain occasions.

Both the BX-chip, the laptop and a battery were hidden in a carrier bag. The sounds were played through two small loudspeakers, one hidden under the saddle, and one attached to the crossbar, close to the handlebar. Since the battery was enclosed it was possible to ride the Iron Horse prototype freely, but obviously the battery needed recharging every once in a while.

It was hard to find good gait sounds, and it would have been even harder to record ones, but some decent ones were found and used. Unfortunately these were obviously recorded on different types of ground, which somewhat interfered with the expression logic; it is strongly recommended to use longer sound loops with gaits recorded on the same ground. Neighs and snorts were easier to find, and several were used in order to provide a more varied expression.

Applications for the Iron Horse

The Iron Horse concept was explored even further by the creation of possible applications for it. Initial ideas included an idea about the users being cowboys, herding cattle, but where should the cattle come from? Letting Iron Horses temporarily taking the roles of cows, but still with a rider on their back would have severely corrupted the expression logic in several ways – neither horses nor bikes do suddenly change appearance into something completely different, and people don't ride cows. Another idea evolved around the typical game where the players belong to opposite sides and set out to shoot each other. There are several technical solutions that can be used to detect if someone is shot, but again, there are problems,

this time in the immersion dimension. For instance, what happens if an Iron Horse gets shot? Firstly, the rider may get upset. Secondly, a horse will fall if shot, but a bike won't, and there was no way to take care of this discrepancy in a good way.

A very feasible idea was that about a racing application apart from the fact that a special application isn't really necessary; all you need is two Iron Horses and an agreement on the distance and off you go! A similar idea, that of a trotting race was really tempting since a certain skill is now suddenly involved, that to ride as fast as possible without breaking into canter. However, in trotting, the human is driving the horse, sitting behind it in a sulky, and this could not be transferred to an Iron Horse. And again, if the users could accept the fact that they sat on the horse during the race it was the exact same case as for the regular horse race; no special application is needed.

At last, we found an application that could be investigated further: a horse jumping application, as described below. Later, I also came up with a dressage application for the Iron Horse; the concept was interesting to explore, but several aspects were a bit dubious from an expression logic point of view.

The Show Jumping Application

A jumping track consists of several fences that shall be jumped in a defined order, and since the fences are of different height, some of them require that the horse has a higher speed when jumping off, in order to jump cleanly. Albeit it is the horse's job to jump big, brave and fast, it is the rider's job to remember the track, and to ride the best line to each fence in order to save ground and time, however still giving the horse speed and space to jump clean. This implies that the rider must judge how many strides there are between obstacles and then collect or extend the canter in order to position the horse correctly in front of each obstacle. Usually, the track has to be ridden within a time limit; if it is exceeded, there is a penalty (normally ¼ fault per second). Knockdowns are 4 faults each, and so are refusals. In short, the equipage with the least faults wins, with time as a tie breaker, but there are several variants with second or even third rounds on shorter tracks.

Since an Iron Horse can't jump, symbolical poles are laid out on the ground; the more poles the "higher" the fence, demanding a higher speed. By combining the poles in different groups it is possible to create different types of fences, like double-triple or other combinations. In order to jump a fence without knocking any poles down, the Iron Horse must pass over it at a sufficient speed and the rider must sit

Fig 8. To the left, the wrong position of the pedals when passing a fence. To the right, the correct position.

still, i.e. not move the pedals. Strides are mapped to pedal revolutions, so a stride is finished when the pedals are at the same level, right pedal first, meaning that this is how the pedals must be positioned when jumping. The position itself is not arbitrarily chosen either; note how both feet are as far from the ground as possible.

This design decision forces the rider to carefully choose the way between fences in order to have time to treadle enough revolutions to both reach sufficient speed to clear the fence and have time to position the pedals right. This very closely imitates the rider positioning the horse.

Just as in real life there are no hoof sounds heard during the jump, and knockdown results in the sound of falling poles, and perhaps the horse snorts or grunts somewhat between fences. The application consists solely of the poles making out the fences, and some additional sensors that detect the poles, some additional code to process this and the additional sound of falling poles. It does not support timekeeping or keeping track of scores. Firstly because this is not made by the horse. Secondly, because one of the design objectives was not to impose rules on play. The only rules imposed here are directly gotten from the nature of horses and fences; a certain speed and a good position are needed to jump a fence without knocking it down, period. If the users want to count faults according to standard rules – or any other rules – they are free to do so, if not, fine. Thirdly because this would have required a lot of extra technology, like an electric billboard showing the results and a lot of extra interaction. If leaving this out, interaction is very easy and natural in the current design; the poles are laid out and… there's a fence.

This application was not implemented, but can be solved technically. However we did an experiment by laying out poles (made of fabric) on a soccer ground, copying

an actual track, and tried to "jump" it, and this actually required some skill! This was gratifying since we did want it to be rewarding to practice; success is so much sweeter if one can fail. While testing we also stood up during the jumps, since the rider does that when jumping, but we did not see any reason to implement a way to track this, i.e. enforcing it. If users want to do this due to immersion, that is enough.

The Iron Horse on display

The Iron Horse was displayed at the Universeum – Sweden's national science discovery centre – in Gothenburg during a week in May. Universeum's general goal is "to stimulate young people's thirst for knowledge and encourage them to get actively involved in science and technology"[12] and it aims to attract children and teenagers between the ages of 5 and 19.

The Iron Horse was exhibited together with other similar objects created by other students during the same project course. During the week it was exhibited circa 10 000 people, most of them children visited Universeum, and quite many of those passed by the Iron Horse. A few months later, in October 2002, the Iron Horse was exhibited as an Aesthetic Artifact at the NordiCHI conference (Landin et al 2002).

At neither of the exhibitions, any formal user studies were made, since it was not within the scope of the project. Our general, and perhaps highly subjective, impression was however that the horse-like expressions of the Iron Horse amused or amazed most users and annoyed none.

Reflections

Problems

Did we succeed in designing the Iron Horse? We believe so, but we cannot "prove" it. We didn't in any way measure the degree of entertainment or if users experienced any anthropomorphism towards it, since this wasn't really in the scope of the project.

[12] Quote from Universeum's own press information at http://www.universeum.se/.

Nevertheless we did some observations during the first exhibition – albeit casual, informal and spontaneous, i.e. not in a user test like manner. We did notice that the children seemed to like the Iron Horse, and they clearly noticed and appreciated that it was not an ordinary bike. They lined up, waiting for a chance to use it, and some were unwilling to leave it; several little girls were dragged away from it. It must be taken into account that the children visiting the exhibition were excited already, frantically trying everything out but on the other hand they seemed to like the Iron Horse even if their experience of it was muddled because they couldn't ride it freely – in such a crowded place havoc would have been the result. Instead, we had to place the Iron Horse in what is called a trainer; a device enabling the back wheel to spin (however requiring some force) without driving the bike[13].

These observations – albeit very non-hardcore scientific – suggest that the audial augmentation really enhances the experience of the bike; it transfers from a bike to something else: an Iron Horse.

Starting the project we had stated that we wanted to create the toy we would have really liked when we were kids. And in this sense we succeeded. We ourselves saw the Iron Horse as its own thing, with its own behavior and personality, and we grew very fond of it, and of riding it. Even if we ourselves deliberately had created all the triggers for anthropomorphism and were aware of them, we still fell for it.

Lessons Learnt

It must again be pointed out that lots of the immersion and expression of the Iron Horse lies in the sounds and the quality of the sounds. An ideal expansion would be to sense the color of the ground (using a light sensor and making an analysis of light wavelengths) as well as its smoothness (using a microphone and frequency analysis), thus concluding which kind of ground it is (grass, concrete, asphalt, gravel) and play gait sounds accordingly. After all, a horse walking on grass sounds very different from one walking on asphalt. Also, stereo sound would be better; the front speaker playing the sound of the fore legs and the neighs, the back speaker playing the sound of the hind legs and the falling fences. In addition it'd be better if all the neighs, whinnies, snorts, puffs etc were recorded from only one horse, or at least one horse per Iron Horse, so to speak.

[13] Only a few people have actually ridden the Iron Horse freely, but they testify that that experience is significantly stronger than simply riding it in the trainer.

Possibilities

Again, the Iron Horse was designed to encourage fantasy, imagination and free play. One can question whether the sound enhancement is really necessary to create emotional response in order to achieve this; kids have been imagining that their bikes are horses for decades! I remember putting reins on my bike, attaching them to the handlebar and then proudly "riding" down the street on my "horse"[14] so obviously my imagination worked well without any computer augmentation of the bike. However, what the Iron Horse adds to this is the unpredictability and a certain sense of autonomy. It "decides" when it changes its gait from trot to canter; not the rider, even if the rider has a significant influence. It whinnies "at own will", not on the rider's wish or command. This will give it a stronger personality and presence than the imagined horse, and this perceived personality will most likely create a stronger emotional response, just as children rank the robot dog AIBO as being more alive than a plush dog and less alive than a live dog. (Kahn et al 2004, Melson, Kahn et al 2005).

It would definitely be possible to turn the Iron Horse into a product. The components that make up the system are actually quite small; obviously a laptop isn't actually needed; a small embedded computer will work just as well, since it just needs to process some software and play sounds of a reasonable quality, something that today's cell phones already do, and they are small enough. Thus these components could be made to melt into the expression of the bike, and with specially designed, streamlined speakers the Iron Horse would look better. Still, it needs electrical power to run. Batteries could be recharged over night, which is a bit dubious since it perhaps breaks the expression logic – or if is this the Iron Horse's way to "eat" and should it in that case give feedback in form of chewing sounds? After having read about expression logic and how to transfer expressions it throughout the entire chapter, what do *you* think?

Expression transfer as a design approach

Did the expression transfer help to create the personification and immersion that seemed to be evoked by the Iron Horse? This is unclear, since only one prototype was created, and it was not tested immensely. However keeping a keen eye on the

[14] Normally, it was a large, kind, and very fast dapple gray stallion which I called Stardust (Dusty for short), but occasionally it turned into one of the horses described in Walter Farley's books on The Black Stallion. Yes, I still remember. :)

design objectives when transferring expressions gave clear motivations for each and every design decision. Hardly any decisions were made arbitrarily. Thus the Iron Horse "works" in the sense that users interpret its behaviors (sounds) as a natural part of the Iron Horse-ness of the bike; it does not disturb them or confuse them, or requires a lot of thinking; it is just there, strengthening immersion without disturbing it.

Thus, when applying expression transfer, the design objectives must be used as a basis when selecting the expressions to transfer. Therefore, they must be stated first. Thereafter, one can investigate the behaviors of the animal and how they are expressed (in sound, movement, appearance). These must be analyzed in terms of how well they fit with the objectives, and whether they are technically and practically possible to transfer at all. Which expressions make sense to transfer and which collide with the given expressions of the hybrid object (if any)? And, second, are there any other reasons to omit some expressions (for instance avoiding aggressive sounds in the iron Horse case), and is there a logical boundary that can be followed (for instance excluding "body" sounds since the bike does not have an organic body)? Third, which other design decisions may effect the expressions and how can this be countered?

Once a selection has been made, the theoretical part of the expression transfer is accomplished, and focus shifts towards making the chosen expressions coming across as life-like and exact as possible using whatever suitable tools, techniques, means and materials at hand.

Special issues related to Animal Expression Transfer

When transferring animal traits onto other objects, it does not seem necessary to take all traits, or even as many as possible. Both the PillO'Mate and the Iron Horse use rather few traits from a cat and a horse respectively but can still evoke very positive feelings and associations to the animal. The PillO'Mate cushion uses only weight, heat, fur and vibration; no sounds or movements or even a cat-like appearance, the Iron Horse uses solely sounds (and not even all the possible ones).

However, if using Animal Expression Transfer as a deliberate way to create emotional response, a positive and interactive response seems to be the most important design element, and the transferred expressions should thus address this. *All* of the robotic pets or toys mentioned in the background have this strong trait in common: they respond positively to the user. For example the tamagotchi will express happiness if fed or played with, the furby says "I love you", and the Iron

Horse often greets its owner with a neigh. This behavior originates from the human need for positive acknowledgement and affection, and triggers affection in turn. Note that Barney had to be improved with positive, friendly and/or affective, non-dominant phrases like "You're my special friend" in order for the children to like him (Strommen 1998)! It seems that this one quality is very crucial when creating an affective response in humans, and unlike in traditional products where the affective behavior is always constantly displayed (i.e. the imprinted smile on the baby doll's face or the cute face of the plush toy dog) the computer augmented products can create a stronger response since they show their affection actively, as a *result of what the user does*. It seems that this small change in behavior, i.e. actively responding rather than passively displaying, makes a great difference. *Interaction* makes a great difference.

Another interesting question related to this context is whether there is a difference in enhancing familiar objects (bikes, cushions, plush animals) with animal traits, or enhancing (robotic) animals or creatures with the traits of a computer? Remember that of all the listed toys/robots with traits from existing animals (the Iron Horse, the PillO'Mate and Paro) the one with the most computer-like functions, namely the AIBO, was partly perceived as a computer; when comparing it with a live dog children were essentially evenly split on whether AIBO was more like a desktop computer or a live dog (Melson, Kahn et al 2005). This reaction might be directly traced back to the fact that the AIBO's expression logic actually suggests this. It has the general shape of a dog, but nor fur; instead its surface is smooth and shiny, the joints clearly visible. It has never had naturally looking eyes; in the latest version they are a band of spots/lights across the face. In line with this it has non-dog behaviors that are very much those of a computerized object, i.e. taking pictures by itself. However it should be mentioned that the robot seal Paro was also perceived as artificial despite is fur and life-like appearance, but it was judged so by elderly people who perhaps are more skeptical towards computer augmentation.

Figure 9: One of the later AIBO versions.

Another issue related to this is whether an appearance that is non-animal-like reduces the emotional response and the overall expression logic? Perhaps not. Both

Paro and the furbies can be seen as not-so-real versions of an more lively original (albeit that the living furby does not exist) whereas the AIBO, the Iron Horse and PillO'Mate can be experienced as a *combinations* of items; a dog and a robot; a bike and a horse; a cushion and a cat. It may be that this *obvious* dual nature of these objects results in that they do not suffer from the notion that they are lesser versions of something. Instead they are new objects, and as long as their expression logic makes sense and is true to the dual natures within, users will accept them more readily.

To conclude, an Animal Expression Transfer may be suitable if you want to increase the emotional response to an artifact, or to strengthen the immersion when using it.

INTERMEDIATE: DESIGNING GAMES

Playing games is a social activity, a way to loosen up, a way to meet friends under laid back circumstances, a way to compete in a friendly way, a way to stimulate the brain and learn new things – not necessarily all the answers to the *Trivial Pursuit*[15] questions, but tactics, strategy, math and probabilities, and anything that is part of a game's theme. And, last but not least, being a good loser – and a good winner! According to Dutch Historian Johan Huizinga (1938) every aspect of the human society can be described as a game or in terms of playing. Gaming has been important to man as long as history has been recorded and most likely before that too. Lately, the game industry has been shifting towards computer games and online gaming, leading to a need for interaction design as an integral part of game design, just as important as gameplay design, graphic design and all the other disciplines that are needed whenever creating a game.

The upcoming three facets describe ways to design computer augmented games, and although they discuss very different aspects of this, they are all related to the same issues; what is a game, what methods for game development are there, what needs do game designers have? Thus, a brief introduction to these topics are given here.

What is a Game?

In "I have No Words and I Must Design" (1994) Greg Costikyan starts out by defining what a game is *not*. It is not a puzzle, since a puzzle is static and always has the same solution whereas the game's outcome is affected by the players' actions. It is not a toy, since a game has a goal. Neither is it a story, because a story is linear and a game is not.

What is a game then? In Homo Ludens (Huizinga 1938), one of the founding books on play theory, the historian Johan Huizinga simply states that every game **has** its rules. As a paraphrase, game historian and game designer David Parlett (1999) states that "every game **is** its rules" and defines a game as an activity with an agreed set of *components* (such as pieces, board, markers, game money etc) combined with an agreed set of *rules* and being a *contest* in order to be the first one to *achieve a goal* (i.e. "to win"). Costikyan's (1994) definition is the same in it's essence but unlike the

[15] *Trivial Pursuit* by Chris Haney and Scott Abbott 1981, published by Horn Abott.

others he focuses on activities rather than "things", claiming that "*A game is a form of art in which participants, termed players, make decisions in order to manage resources through game tokens in the pursuit of a goal.*"

Wolfgang Kramer, a commercially successful board game designer[16], elaborates on Parlett's definition to give a definition of a board game (Kramer 2001). He too, describes it as a "game with rules", pieces and a goal which players compete to reach first, but he also adds the factor of *unpredictability*, claiming that if a game always takes the exact same course it is not truly a game, but a chain of foreseen events; pretty much like seeing the same movie over and over again, which is very much in line with Costikyan who claims that games are not stories, but non-linear. In a very poetic piece Kramer also pinpoints games as something that we do voluntarily:

> *"Although the game has rules which are like laws, playing a game is voluntary and cannot be forced on the players. Whoever plays a game, voluntarily binds himself to the rules. Where force is involved, there is no game."*
>
> *Wolfgang Kramer*
> *"What Makes a Game Good?" (2001)*

In addition Kramer also lists a series of criteria that he believes every true board game must fulfill; playing it should be a shared experience with the other players (i.e. all players are co-located in time and space), all players shall have equal chances to win, playing means interacting mentally and physically with the game (ruling out very simple games) and the game world must be separated from the real world, not have any consequences for real life (ruling out playing poker about money etc.). These are ethical statements rather than definitions, and most of them fit all games, not just board games.

Current Approaches to Game Design

In the same manner that there is no bullet-proof method to write a "good" book there is no bullet-proof method to create a "good" game. This can be said about all disciplines that have an artistic element. However, depending on discipline, years of praxis has turned into a set of methods or approaches. Notwithstanding the fact that

[16] Wolfgang Kramer has won numerous game design awards – for instance the prestigious German "Spiel de Jahres" (Game of the Year) award no less than five times – and has designed and sold at least 105 different games.

game design is an old discipline per se, it is not a common craft, and this might be why it lacks a rich set of tools in the form of methods, approaches and processes that can be used when designing a game.

Game Design Patterns (Björk & Holopainen 2005) are discussed in depth in Facet No. 4, and can be said to be a development tool for games. It consists of a framework describing the context of game design, and a collection of design patterns focusing on the interaction games provide. It can be a useful tool for developing games and gameplay. Its strength is that it is not tied to any specific type of games . However this also means that it is not at all concerned with any other possible design disciplines involved when creating a game; a weakness.

Computer game design – a new discipline lacking tools

If looking at computer game design, the common approach is to use a mixture of techniques and approaches borrowed from all of the various disciplines involved, i.e. software development, movie making, narration and traditional games. Even if this works, the professional game designers express a lack of a developed design discipline tailor made for designing digital games (c.f. Costikyan 2002, Spector 1999). There are a few examples of professional designers inventing and researching such models for computer game design, e.g. Chris Crawford (1984) and Will Wright (2003), but they are a fraction of all game designers. Due to this lack of tools, game designers either improvise the gameplay design or try to apply the experience of other game developers as describe in media such as the website Gamasutra[17]. Of course, this is a way of looking for inspiration, winging a development process and using one's hard-earned feeling for game design, rather than a formalized method.

Board game design – a question of mechanics vs. theme

As for board game designers they commonly talk about "designing by theme" or "designing by mechanics". The first of course means that the inspiration and foundation for the game design is a theme (i.e. winning the Monte Carlo rally[18], or excavating the ruins of Tikal[19], or develop a healthy breed of amoebas[20] which in turn

[17] Gamasutra: http://www.gamasutra.org

[18] As in *Formula Dé* by Laurent Lavaur & Eric Randall 1991, published by Ludodelire & Descartes Editeur.

[19] As in *Tikal*, by Wolfgang Kramer & Michael Kiesling 1999, published by Rio Grande Games, Abacus and Ravensburger respectively.

[20] As in *Ursuppe* (a.k.a. Primordial Soup) by Frank Nestel & Doris Matthäus 1997, published By Doris & Frank and Z-Man Games respectively.

sets boundaries for what can and can't be done within the game as well as which components and events that are suitable, in turn being a basis – and sometimes a logical constraint – for writing the rules. I.e. developing a better metabolism would be very suitable in the amoeba-game whereas totally useless and out of context in the racing game. Designing by mechanics is a more abstract approach where the designer starts out with a set of mechanics, creates a more or less abstract game and adds the theme onto it in the process. One example of this is are the games *ShowManager*[21] vs. *Atlantic Star*[22] which is essentially the same game with two different themes; setting up musicals vs. setting up cruises across the Atlantic. Now, these are ideological approaches rather than explicit methods, and actually they are just as applicable to any kind of game design as board game design.

Still, it seems to be understood that all board game design is an iterative process. Daniel Cook (2002) describes it as evolutionary, and describes an iterative process consisting of six steps in every iteration:

- Playtesting
- Identifying which changes from the previous version that worked.
- Discussing flaws and possible new changes/improvements with the players
- Evaluation of these suggestions and their consequences
- Choosing which suggestions to carry through
- Carrying trough the suggestions regarding changes and components, e.g. rewriting rules and redesigning cards.

Looking upon gameplay design as being primarily evolutionary may be important since it serves the purpose to communicate that the core of the game is something ever-changing, almost *living*. This view is advantageous when communicating shifting design goals to all design disciplines, but the downside is that it does not help in describing how changes in focus occur and why they occur.

Research on game design

In academia – apart from mathematical game theories – game research has for example evolved around exploring the game from a sociological or cultural point of

[21] *Showmanager*, by Dirk Henn 1997, published by db-Spiele and Queen Games respectively.

[22] *Atlantic Star*, still by Dirk Henn, published by Queen Games 2001.

view (Callois 2001 and Huizinga 1986, respectively). Another common angle is to look at the narrative aspect of the game, thus drawing theories and methods from fields like literature, film and theatre (c.f. Laurel 1993, Murray 1998 and Manovich 2001). The existing research on the design of games (c.f. Fulton 2002, Fullerton et al 2004 and Salen & Zimmerman 2004) investigates all the design disciplines that may occur in a project (e.g. software design, hardware design, game design etc.), advocating an iterative process. However this research does not explain *how* the disciplines put requirements on each other. To aggravate the situation, much of this research has been focusing on aspects of engineering, theatre, sociology, narratology etc. Ironically, the discipline which perceivably is the most important and driving discipline, gameplay design, is the least researched and developed one, even if the emerging area of game studies[23] is growing.

Several research projects on computer augmented games have been carried through (see facet No. 3) but in most research projects the research objectives *per se* are a very distinct driving force, in which these games differ from any other games. This means that the success criteria in meeting research goals and design goals need not be the same; the research questions or hypotheses can be answered without reaching all design goals and vice versa a "good" game can be created without guaranteeing any valid research results. Since the focus is the entire research process in particular, and the game design process in general, normally no consideration is taken to the special problems that occur when the gameplay design is to be integrated with the rest of the designs.

Conclusion: a need for game design methods

Despite the research made on games in general, and the design of computer augmented games and computer games in particular there are still very few useful methods for game design. It seems that such methods should be iterative, following today's praxis, and that there is a need to clarify interrelations between both the involved design disciplines and the various parts (e.g. mechanics/rules) of a computer augmented game. These issues are addressed in the upcoming facets (i.e. Facet No. 3: Joining Bits and Pieces, Facet No. 4: Game Design Patterns and Facet No. 5: myTHeme in terms of forces, clashes and remnants, respectively). It is my hope that these chapters together will help game designers by increasing the number of methods available.

[23] See www.gamestudies.org for the premier journal

FACET No. 3:

Joining Bits and Pieces – using Ubiquitous Computing to Enhance Board Games

This chapter deals with how board games can be enhanced and enrichened via the use of embedded sensors and added computational power. This can provide games with new ways of interaction and behavior, which are impossible without computer augmentation.

Sunday afternoon. March. Outside the weather is gloomy, wet and cold, but inside you gather around the big dining table under the bright light of the two ceiling lamps. Poker time? No, not anymore. You've abandoned poker for more intriguing games. You, Mag, Sheldon and Staffan set up the pieces, Greg plugs in the board and you are ready to go! Heartily you engage in a series of bidding, trading and negotiation, sometimes computer-mediated, sometimes not. Late in the game Sheldon risks a computer-mediated trade and you achieve the last of the stones you need to collect in order to win. Now, all you have to do is to move one of your game pieces to its home city. But – so close but still so far! Suddenly the board announces that another end condition has become true; all players have at least two blue stones each. Ouch! You didn't see that one coming since resources are hidden! Instead of you winning, the game ends with a scoring phase, Mag being the victorious one. For about five minutes he brags with his brilliant strategy of gaining riches while manipulating the rest of you to gain blue stones.

"I want a rematch" Greg claims

So you reset the board and start over again.

Joining bits and pieces was a way to try to dissect and improve board games using the concept of game mechanics. The idea was that by studying existing game mechanics – these can be described as building blocks for games – and then study computer augmented game, new mechanics could be described, and these could then in turn be used to create new, computer augmented, games. This would be a way to take board games yet a step further, widening the space of design and experience[24].

Background: Computer Augmenting Board Games

In April 2006 the No 1. site for board game aficionados, BoardGameGeek[25] listed 201 "electronic games" out of a total of more than 22 000 – and of these 201

[24]This work was conducted by myself only, and most of this chapter is based on my Master Thesis, "Joining Bits and Pieces" (Lundgren 2002).

[25] BoardGameGeek (http://www.boardgamegeek.com) was started in January 2000, and is a board game database that holds descriptions, reviews, articles and session reports on over 22 000 games.

many are labeled electronic since they can play sounds or the like, i.e. they do not contain any advanced electronics or computational power. Thus, *less than one percent* of all commercial board games – ignoring the numerous clones of chess computers – can be said to be truly computer augmented. Clearly, the progress in board game design is not as rapid as for instance in computer games and role playing games – note the interesting merger between these two categories in on-line game worlds like *World of Warcraft* or *EverQuest*. Thus, in terms of evolution, complexity and change the board games are lagging far behind. New features have swept in every once in a while, such as modular boards (i.e. the board changes between game sessions since it is constructed anew every time by combining smaller tiles into a larger board.). These were made popular by two games, *RoboRally*[26] and the seminal *Settlers of Catan*[27], the latter having sold some 2 500 000 copies of the original game worldwide (sequels and expansions not included). Most of these new features come in the form of new *mechanics*, i.e. new rules or basis for rules, hardly ever in the form of new components.

One might argue that there are numerous sites and programs that provide computer versions of board games. For instance a quick Google search on the words *"play Mahjong game online"* resulted in ca 1 750 000 hits, replacing *Mahjong* with *Monopoly* resulted in 4 340 000 hits (of course not all are valid) and so on… Still, the board games *per se* have not changed; they have only reoccurred in another media. A game of *Ludo*[28] is a game of Ludo even if played on a computer instead of on a board; still the same rules and the same gameplay. This is not what I mean by computer augmenting a game; in a computer augmented game computational power is a necessity.

Commercial computer augmented board games

Apart from numerous computer augmented chess games, there are few commercial computer augmented board games. An early example from 1979 is *Stop Thief*[29] where a thief is being chased in a city landscape. The "thief" is played by the game and clues are given in the form of sounds like breaking windows, subway noise,

[26] *RoboRally* by Richard Garfield 1994, published by Wizards of the Coast.

[27] *Die Siedler von Catan*, a.k.a. *The Settlers of Catan* a.k.a. *Les Colons de Katane*, by Klaus Teuber 1995, published by Mayfair Games, Kosmos and Patnik respectively.

[28] *Ludo* is an ancient game also known as *Pachisi* and, in Sweden, *Fia med knuff*.

[29] *Stop Thief,* published by Parker Brothers 1979.

steps etc. A successor was the fantasy game *Dark Tower*[30] where the electronic tower runs the game by keeping track of troops and supplies and managing combat with random monsters etc. It took almost twenty years until the next "big" (or at least known and selling) computer augmented board game showed up in 2003: Reiner Knizia's *King Arthur*[31] . In this game the board players wander in a landscape and run into various encounters like robbers, dragons, friendly villagers, the Lady of the Lake etc. What happens in a certain spot is randomized by the game, or semi-randomized. It also keeps track of the score and certain player-specific information.

These three are the most well-known ones, the bestsellers among computer augmented board games. It seems that most of the game design experiments using computer augmentation can be found within the research community.

Also, a new interesting release is hitting the market in the fall of 2007. It is a game console by Mattel, called HyperScan[32] which is meant to support and enhance new collectible card games. It is hooked up to a television set and with it comes a set of computer augmented, RFID-tagged cards. The console scans the cards and they then appear on screen for the game. After the game, the winning cards are upgraded. Players can trade cards and plan strategies as in any other collectible card game.

Examples and inspiration; Crossing the board game borders

Most computer-augmented games have been developed within academia. Below is a brief overview of such games that use computers but do not conform to the traditional characteristics of computer games. They have been chosen since they exemplify different ways to computer-enhance games, as well as the new game mechanics this technology provides. The examples have also been selected because of their strong focus on creating playable games rather than testing technologies.

Virtual Reality and Mixed Reality games

These games rely on high-precision body tracking and see-through head-mounted displays. The *MIND-WARPING* system (Starner et al 2000) is a fighting game where one player places physical game pieces depicting monsters onto a projected board, whereas the other player plays the game in a VR-environment, fighting

[30] *Dark Tower*, published by Milton Bradley 1981.

[31] *King Arthur* by Rainer Knizia 2003, published by Ravensburger 2003.

[32] Press release at: http://www.shareholder.com/mattel/news/20060720-204512.cfm.

these superimposed monsters with gestures. Other similar games include *AR2 Hockey*, *AquaGauntlet*, and *RV-Border Guards* (all described in by Tamura in 2000) which superimpose computer-rendered game elements on the players' physical environment.

Interactive tiles and interactive boards

Gorbet et al. (1998) created the *Triangles* interface. It consisted of a set of triangular tiles that could be connected with each other along the edges using magnetic connectors. These also transfer electrical power and data within the system. The purpose of the project was to provide a natural interface that could easily be used with two hands, and several applications were built. In all, the triangles were used as input, but the output varied from pure sound to changing web pages or films or presentations etc shown on a screen. One of the applications was *Cinderella*, a storytelling game. In this case the triangles were decorated with figures and events from the Cinderella fairytale, and when children combined them, they could hear different parts of the story.

Mandryk et al (2002) created *False Prophets* which uses a computer augmented board together with augmented pieces and handheld computers. Each tile of the board is equipped with infrared phototransistors, and the downside of the game pieces contained an infrared light emitting diode. In this way the board can keep track of the pieces. What is on the board is being projected by a projector and therefore the board can change dynamically during the game according to what the players do. Some information in the game is private (the players are looking for clues), and this information is shown on each player's handheld computer.

Peitz et al (2006) developed *Wizard's Apprentice*, a game that supports two kinds of player modes; there is one Wizard – perhaps an adult not as interested in the game or busy with something else – and a number of Apprentices, e.g. children or just more enthusiastic players. The Wizard assigns tasks to the Apprentices who set off across the board to solve them, defeating enemies, collecting items etc. The board keeps track of the Apprentice's location using a combination of RFID-technology and light sensors and the game keeps track of what has happened, randomizes monsters etc.

Turning board games into "room games"

Björk et al (2001) have created *Pirates!*, which is a mobile multiplayer game. It is a virtual game set in a physical world. Players use handheld computers to play

(typically PDAs[33]) and on their screens they can see what happens in the game. When moving about in the room the environment in the game changes; the virtual game world consists of an archipelago of islands that can only be found, digitally, by moving to the physical spot where they are located; the game world is fixed and superimposed onto the real world. The players are captains on ships setting out on a number of quests like discovering new island, finding treasures etc. without being killed by monsters or robbed by pirates. If they (physically) meet other players they can interact with them on their screens. All of this is enabled with a server and a wireless network that the PDAs are attached to. Short range radio frequency technology is used in order for the virtual islands to detect any PDAs within range, as well as for the PDAs to recognize each other.

Jay Schneider and Gerd Kortuem (2001) have turned the traditional board game *Clue* into a Live Action Role Playing game called *Pervasive Clue*. Just as in Clue the object is to find out who killed the victim, in what room and with what weapon (*"Professor Plum killed Col Mustard in the library using the DAGGER!!!"*) by collecting clues. There are up to three clues in each room, and they are hidden, e.g. inside books, in boxes, under tables etc. Just like in *Pirates!*, the players use PDAs that can pick up radio signals coming from the hidden clues. However they have to come within a very short range in order to detect them. For immersion, the PDAs are equipped with magnifying glasses so that players get an "alibi" for snooping around with their PDAs, but also this indicates that the PDAs are used to look "closer" at things. If a PDA is placed within the clue's range the clue will be picked up by the PDA and the player can see it. The PDA will keep track of all the clues a player has found, eliminating possible murderers, weapons and rooms.

Biofeedback games

Such games use biological input, typically from biosensors attached to the players' bodies. Two examples are Brainball (Ilstedt Hjelm & Brovall 2000) where players' brainwaves guide a ball towards the opposing goal, and *Relax-to-Win* (Bersak et al 2001) where players need to relax in order to win a race.

Defining a computer augmented board game

In this chapter you will find that the term board game covers a wide range of games, using a definition somewhere in-between Parlett's and Kramer's, since there is no

[33] PDA = Personal Digital Assistant, e.g. a Palm Pilot.

unanimous agreement on what a board game really is, or rather, what fits into the category and not – is dart a board game for instance? Thus, I made up my own definition:

- A board game consists of rules (including a way to win) and components.
- A board game is a played on a board or with components laid out on a flat surface, normally a table.
- A board game may consist of a deck of cards solely, as long at is it not a standard deck of cards[34].
- A board game is played by at least two people who are at the same place, the same time.
- A board game is more about decision-making, tactics and analysis, rather than about trivia knowledge, and every game session differs from any other game session, e.g. the outcome is not fixed.
- A board game is not a sport[35].

However, this text is about computer augmented board games, and this requires an extra add-on to the definition:

- A computer augmented board game is a board game enhanced with embedded computational power and, if needed, embedded sensors etc. in such a way that the enhancement adds *significant qualities* to the game, qualities that could not easily be removed from gameplay or provided in any other way.

I want to point out that a true computer augmented game has to benefit very much from the embedded computing. For instance a game simply playing sounds or always reacting on the same stimulus in the same way is an electronic game, not a computer augmented one. *Computational power provides intelligence and unpredictability to a game*.

[35] This is a limitation based on the fact that the thousand and thousand games that can be played with a deck of cards belong to their own category.

[36] Note that following the other definitions both snooker and table tennis are board games. Still we sense they are not. Many attempts have been made to distinguish games from sports, but regardless the definitions it seems there will be misclassifications. However there seems to be a silent agreement on what games are sports and not.

Topic: Game Mechanics being a Basis for Game Rules

What is a rule? A rule is a very precise instruction on one aspect of a game and how to play it. It can be something like:

> *"When castling, you simultaneously move your hitherto unmoved king, and one of your hitherto unmoved rooks. The king moves two empty squares towards a rook, and that rook moves to the square at the other side of the king. No other pieces must be standing between the king and the rook when castling, the king must not be in check, the square the king crosses must not be attacked by an enemy piece, and the act of castling must not place the king in check." (from the rules of Chess)*

This rule exists only in *Chess* or variants of *Chess*; it is specific for this game. Therefore, when speaking of games and games' rules in general, more generally applicable concepts have to be used. Enter: *the game mechanic*. A game mechanic can be described as *"a part of a game's rule system that covers one general or specific aspect of the game"* (Lundgren 2002, BoardGameGeek). Mechanics are the building blocks of games. They are general description of something that is done during the game, and sometimes they also imply what elements and rules that are needed to achieve this.

A good example is the mechanic "trading". It simply states that something can be traded. It does not clarify what is being traded, or by whom, or when, but it clearly indicates that the game probably has something worth trading (goods for instance) and in this it gives a hint of what the game is about; trading and thus most likely negotiating too. However, if we want to turn the trading mechanic into a rule, we have to elaborate and claim something like:

> *"…a player can trade resources with other players. All trades must involve the player whose turn it is; other players may not trade amongst themselves. Players need not offer an equal number of resource cards. Players are allowed to discuss what they have in their hand with other players to facilitate trading. Once a trade is agreed upon, both players exchange the stated resource card(s); they cannot cheat the other player by giving anything other than what*

was offered. Each trade must contain at least one resource card from each player. Trades can never include Development cards.

In addition to trading with other players, a player can trade resources with the bank. On his turn, a player can trade four of any one resource to the bank in exchange for one of any resource. This is the only type of trade that can be made with the bank unless a player has a settlement or city touching a port hex […].

Ports are indicated by semi-circles on the corners of some water hexes. If a player has a settlement or city built on an intersection that contains a port, that player can make special trades with the bank. These trades may be made immediately upon building the port, even during the same turn. Note that not all corners of port hexes have ports; only those with semi-circles are actually ports. There are two types of ports: generic ports and specific ports. Generic ports have a question mark in the center of the hex. They allow a player to trade three of any one resource for one of any resource, instead of the normal four for one rate. Specific ports have a resource pictured in the center of the hex. They allow a player to trade two of that specific resource for one of any resource. The resource that can be traded at that port is indicated by the icon shown on the port hex. No other resources can be traded at specific ports, but the player can still make normal four for one trades involving other resources."

– Klaus Teuber

In: The rules of "Settlers of Catan", Mayfair Games (1995)

Wow! One word, "trading", just turned into 349! We can see why talking about games in terms of mechanics instead of specific rules can facilitate conversation. *"What do you want to play?" "I'd like to play a trading game today!"*

Over the years the No 1. site for board game aficionados, BoardGameGeek has listed various mechanics. The list expands, contracts and changes from time to time; the list presented here is a merger of the mechanics listed in September 2002 and April 2006 respectively. Note that most of the mechanics relate to some kind of interaction within the game.

List of game mechanics from Boardgamegeek

Acting * / Role-Playing**
Action Point Allowance System
Area Enclosure
Area Movement
Area-Impulse
Auction/Bidding
Betting/Wagering
Campaign/Battle Card Driven
Card Drafting*
Clue-giving**
Chit-Pull System
Co-operative Play
Commodity Speculation
Crayon Rail System
Drawing**
Dice Rolling* (Roll and Move)
Event card interaction**
Hand Management
Hex-and-Counter
Line Drawing*
Matching** / Pattern Building*
Memory
Modular Board
Negotiation**
Open card selection**
Paper-and-Pencil
Partnerships
Pattern Recognition*
Pick-up and Deliver
Point to Point Movement*
Rock-Paper-Scissors
Roll and Move
Secret Unit Deployment
Set Collection
Simulation*
Simultaneous Action Selection
Singing
Stock Holding
Storytelling
Tile Placement
Trading
Trick-taking
Unit Deplyment**
Variable Phase Order*
Variable Player Powers
Voting

* = Listed in 2006 only ** = Listed in 2002 only

An almost complete list of explanations can be found in my M. Sc. Thesis (Lundgren 2002) which can be downloaded at http://www.cs.chalmers.se/~lundsus/lundgren_joining_bits_and_pieces.pdf.

The point is, however, that mechanics are like Lego™ pieces. Alone the mechanics are rather useless, but when put together in a fitting way, they can form a game. Some mechanics fit very well together, some don't, but all can be used as building blocks for a game.

Design Challenge

In the previous text we have concluded that here are very few computer augmented board games, and that the board game evolution is slowly moving forward with the invention of new game mechanics. Perhaps people who work with game design are not into computing or vice versa? Still, embedding computational power into a board game could add a great deal, since it opens up for entirely new kinds of *interactions* and therefore *rules*.

As for the interactions, it is important that they still feel very natural (which is line with the general ideas on ubiquitous computing). E.g. moving pieces around on a game board can be seen as a kind of natural interaction, not using traditional input such as keyboards or pointing devices. Moreover, the logic and the rules of the game shall carry the logic of the computational behavior so that interaction is logical-natural, unnoticed as being interactions with a computer, only noticed as interactions with the game. *"I moved my king to the castle"*, not *"I input data about my movement to the computer"*.

As for the rules, the interesting question is "Which ones of the current mechanics can be improved to be applicable to computer augmented board games? And are there any new ones?" To find this out, was the main goal of the entire project.

Case: MultiMonsterMania

In the years 2002 – 2004 I conducted several game design experiments, some just concepts, some more elaborate prototypes. In some I created games, in others I created game systems together with fitting games. A *game system* is a set of components to which many different rules can be applied, i.e. very many games can be played with the same components. A normal standard deck of cards is the most well-known example; with it you can play thousands of games, from a thrilling, adrenaline-intense game of poker to a more peaceful tournament in Bridge, to a more casual session of *Oh Hell!*[37].

Some of these design experiments were used as inspiration, along with existing games designed by others, to find qualities for computer augmentation in board games, and to illustrate what can be done. All of them are described in my thesis, and in addition you can read about another computer augmented board game

[37] *Oh Hell!* is a game where players bid how many tricks they will take, called *Plump* in Swedish, *Plums* in German.

I've developed, *myTHeme*, in Facet No. 5. Here, however, I only present the most illuminating example: MultiMonsterMania and *The Hatchery*.

MultiMonsterMania is a future game system and *The Hatchery* was a game for it. MultiMonsterMania are collectible, interactive cards, consisting of small screens. Each card carries a monster that has special abilities depending on its number of arms/tentacles etc. Apart from using monsters in games, one can also breed new monsters into blank cards. *The Hatchery* is a future game to be played with the MultiMonsterMania cards. It is about hatching eggs by constantly adapting a changing environment to them.

Design process

The environment in which MultiMonsterMania was created was a research atelier arranged by Staffan Björk, Jussi Holopainen, Peter Ljungstrand and Karl Petter Åkesson. (Björk et al 2002). It took place in February 2002 and thirteen people plus the organizers took part. The aim was *"…to bring together creative and driven people to exchange experiences and ideas about expanding the domain of ubiquitous computing to computer entertainment."* (Björk et al 2002). The atelier started with introducing participants, background research, and available tools and solutions currently used within ubiquitous computing.

In order to free the participants minds from today's technical restrictions the atelier also featured a session focusing on what people would do to entertain themselves in 2010. A set of 16 possible 2010 futures were created and each group of participants chose one of them as inspiration for their game or entertainment concept.

I joined Mads Haahr and Niels Reijers and together we created the MultiMonsterMania concept. We started out by stating some objectives: players should be able to earn money, have fun, express creativity and collect cards. The earning of money came from the fact that the future we designed contained a lot of poor street kids and we wanted to give them an opportunity to get rich, either by being good at playing, or – as we decided later – by breeding valuable or rare monsters or design games or artwork for games. This was partly inspired by the people who today earn a living by selling virtual characters and won items belonging to online game worlds (such as *World of Warcraft*) on for instance eBay for real money[38]. We used various

[38] An example: On the 1st of May 2006 a *"Level 60 Warrior, almost fully Epic equipped, has Lionheart helm and Stronghold Gauntlets"* was on sale for USD 300. This was one of the cheapest characters on sale; there were another 14, ranging from 300 – 700 USD each.

formal or non-formal variants of brainstorming and body storming to come up with the concept. I then got the assignment to come up with an actual game for MultiMonsterMania, which I did together with Jussi Holopainen, again using brainstorming but also by picking some mechanics we wanted to use as well as, again, non-formal brainstorming and body storming/playtesting.

The MultiMonsterMania game system

Multi Monster Mania is a card game where cards have computational power that allows them to be dynamic. Cards also have limited connectivity (via for example Bluetooth) so that they can communicate and interact over short distances. This opens up the possibility for playing a number of different games with the cards; a rule card can be used as a kind of game server running the game and moderating the interactions and actions of the cards.

Empty cards can be bought, and these can be programmed with new games or environments, albeit not with new monsters. However new monsters can be bred into blank cards.

The game concept draws element from tamagotchis in that the cards contain "living" entities, and from collectible card games in that cards can be traded and that a player can choose an appropriate set of cards out of his or her deck, fitting the game as best as possible (pretty much as how a trainer decides the setup of a football team before each game).

Technology

MultiMonsterMania is based on the fact that over the last couple of years Personal Digital Assistants (PDAs) have decreased in size while increased in functionality. Today they come with color touch screens, short range network connectivity (e.g., IR or Bluetooth) and a reasonable battery life as standard. If this trend continues, it is not impossible to imagine a future PDA with the same features however smaller, thinner, lighter and much cheaper, say a few Euros or US Dollars. MultiMonsterMania is designed to reside in such PDAs which in this case can be regarded as interactive cards. Note that Mattel's new game system HyperScan (a game console for collectible card games to be released in the fall of 2007[39]) with its interactive cards could also be used for *The Hatchery*.

[39] Press release at: http://www.shareholder.com/mattel/news/20060720-204512.cfm.

Monster cards

Figure 10: A monster.

A monster "inhibits" one and only one card in terms of where its code resides; however during a game its graphical representation may occasionally move to a location card. The card carries a "biography" of the monster in terms of age, parents, breeder, offspring, games played, endured training etc. Most importantly however, it contains the monsters genome and, based on the genome, its properties (in terms of speed, endurance etc.) I.e. a monster with four legs and a pair of wings is faster than one without two legs. However that slower monster may have lots of arms instead, enabling it to pick up and carry things.

The monster genome consists of a sequence of gene groups. A sequence determines firstly how many instances of something there is (i.e. how many legs, arms, tentacles, eyes etc.) followed by one or more modifier genes (of which one is dominant) defining the type (i.e. human eyes, cat eyes, eyes on sticks etc.). The more complex the monster, the longer the genome.

Thanks to the genome, monsters can be bred using a rather simple algorithm determining the appearance of the offspring. Two monster cards are placed next to each other (any two – monsters do not have genders) and next to a blank card that the new monster can be bred "into". Monsters that are similar in appearance are more likely to breed (following the laws of nature). This is determined by analyzing each monsters genome and comparing the number of genes they have in common with the number of genes they don't. This determines the odds for propagation. The genes of the offspring are determined gene group by gene group. The numbers of instances (e.g. legs) are added and using normal frequency distribution a number of instances for the child is calculated. Thus, if both parents have two legs, the offspring can have anything from zero to four legs, however most likely two. Thereafter the modifying genes are added onto the offspring's genome and the one with the most instances "wins"; if they are equal, the most dominant wins and determines whether in this case the legs are frog legs or hoofed legs.

The design rationale for this is that the reproduction algorithm will produce offspring with an appearance that seems rather natural and expected by the breeder (following some kind of expression logic), however sometimes featuring anomalies running in the family, or other unexpected or surprising outcomes. The design

rationale for having a reproduction algorithm at all is to encourage players lust for experimenting as well as the possibility to breed monsters of a kind that are not for sale when buying standard monster cards. One could also imagine that standard monsters carry a set of inactive (recessive) modifying genes that can suddenly appear after some breeding if they suddenly outnumber more dominant genes.

Other cards

Apart from the monster cards themselves, there are three other kinds of cards. There are Game cards that act as servers for a game, specifying the logic and rules for that particular game. Some games need one or more Location cards (desert, wood, Dracula's castle etc.). Also, there are Modifyer cards which change other cards. Some are applied to rule cards, enforcing a special variant. Others are applied to locations, influencing for instance the weather or time of day, and some are applied to monsters, enhancing them in some way, i.e. making them faster or giving them more stamina.

MultiMonsterMania: Special features made possible via computer augmentation

Since cards are programmable and monsters can be bred the MultiMonsterMania framework can be extended in very many ways. New game cards can be created and programmed as well as new locations. These games can e designed to use the interactivity of the cards, their computational power and their ability to communicate, thus differing them from any card game existing today. Long-time breeding can enable new mutations. New genes can be introduced in new releases of monster cards, genes that can become a part of the gene pool by breeding. This built-in flexibility and interactivity is the strength of the MultiMonsterMania system and it would not be possible without computer augmentation.

The Hatchery

The Hatchery is an example of a game featuring the MultiMonsterMania cards. It was invented by me and Jussi Holopainen, also as a part of the workshop where the MultiMonsterMania concept was created. Below are the entire rules in an excerpt from my Master Thesis (Lundgren 2002). It is only a rough layout; it has never been playtested. The aim is in instead to exemplify how computer augmentation can

benefit to a game. Due to the lack of playtesting, no exact points or costs have been set, and some rules only state what "might" be done.

Object of game

This game is played with teams of monsters. Each team has four eggs in their chamber in an egg-hatching factory. Also, each team has the same setup of eggs; one purple, one pale green, one pink and one turquoise. How fast eggs grow or shrink depends on the environment in the entire factory. This environment is visualized with a color – the closer this is to an egg's own color, the better it grows. Each player gets points for hatching his or her eggs, the sooner the better. Points can also be earned by cooperating with other players.

The color in the factory can be changed by feeding pellets into its heating system, and therefore the game is about the following things;

- Harvesting pellets and feeding them into the factory
- Cooperating to get more pellets and points.
- Saving shrinking eggs by temporarily taking them out of the hatching chamber.

The game ends when all eggs have been hatched, and whoever has the most points wins.

Time, not turns

Note that the game is not turn based. Players move monsters and perform monster actions (such as gathering pellets, saving eggs etc) simultaneously and more or less continuously. Every action in the game (i.e. moving from one place to another, feeding the factory with a pellet, steeling a pellet or exchanging pellets) takes a certain amount of time, sometimes depending on a monster's properties (e.g. monsters with wings move faster than monsters with only legs). The time an action takes and how much of it is left, is shown on a small status bar on the monster card. This signals that the monster is "busy" and cannot be moved. If the monster card is moved anyway (more than 2 mm's) before the action is completed, it is reset. Players have x seconds to move a monster from one location to another. (Moving another player's cards is of course cheating). Thus, The Hatchery is a resource and time management game.

Game pieces

- Each player chooses three monsters each from his or her stock
- As many hatching chambers as there are players (location cards)

- Four pellet fields: one red, one white, one blue, one green (location cards)
- Game field entrance (location card)
- One rule card

Layout of game

The hatching chambers are put next to each other on the middle of the table. By doing this they connect into one large hatchery.

Each player puts his three monsters next to each other to indicate to the game that they are a team. After this, one of the monsters is put next to that team's hatching chamber to claim it as theirs. When doing that, four eggs of different colors appear in that chamber. The monster can be taken away from the chamber again; it isn't stuck there for the rest of the game.

The four pellet fields, the game field entrance and the rule card are also laid out on the table in some appropriate way, just like the monsters.

On colors

Since the game is about changing the color of the hatchery chamber to match the eggs, colors and the change of colors are important in the game. Figure 11 shows how pellets affect the color status in the hatchery.

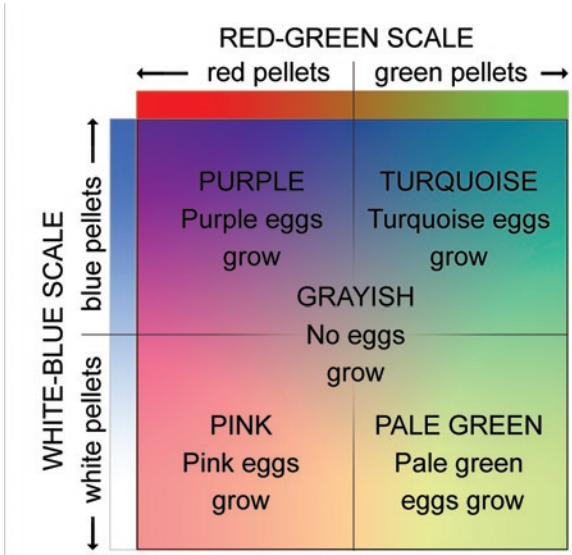


Figure 11: The color matrix of *The Hatchery*. E.g. feeding the hatchery chamber with lots of red an blue pellets will result in a purple color. If white pellets are added the color moves towards pink.

How monster properties affect the game

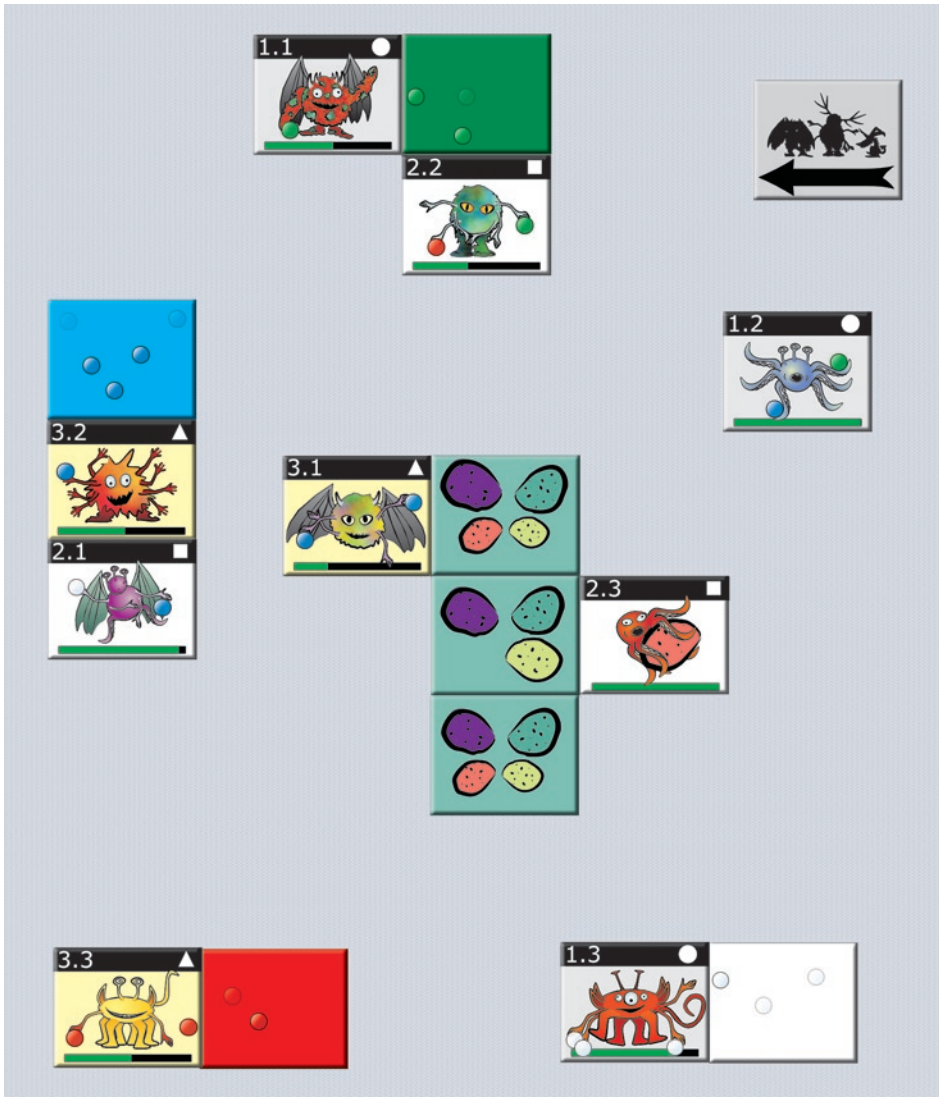
- Monsters with a lot of legs move fast
- Monsters with a lot of wings move even faster (twice as fast?)
- Monsters with a lot of arms can carry a corresponding number of pellets
- Monsters with a lot of arms can also harvest pellets with arms not currently used for holding other pellets
- Monsters with at least two arms or tentacles can hold one and only one egg. When holding one egg, it can not hold any pellets, regardless of how many arms it has; it just clings to the egg with all arms. To pick up an egg all hands must be free (pellets must be given away to other monsters). A monster holding an egg may not move.
- Monsters with tentacles can steal and carry pellets, but not gather them.

Hatching eggs

This section describes what goes on in the egg-hatching factory. The changes of color, growth of eggs and deliverance of pellets is computed and thereafter shown on the cards by themselves.

- The entire factory has the same color. This color depends on the mixture of the colors of the pellets used currently (pellets get used over time, like if they were burned).

  Comment: It might be possible that the color of the pellets fed into the factory is "spread" from wherever it is inserted, meaning that the color tone varies across the factory. If one monster feeds one green pellet into his chamber, this would affect the color in that chamber in the corresponding way. In surrounding chambers, the color should change in a similar way, but calculating with ½ pellet instead of 1. Two chambers away the effect is calculated with ¼ pellet. If this idea is used, it is important that the hatching chambers put on each end of the row are also connected virtually, so that color can spread from one of them to another. In this way there is a difference in how fast eggs grow.
- The factory is fed with color pellets by putting one monster holding one or more pellets next to the player's hatching room and then simply drag and drop the pellet(s) from the monster to the factory. It takes a certain amount of time to deliver the pellet(s). This is shown by a small status bar on the monster card, counting down the time.
- If no pellets are fed to the factory color will move towards gray (see color matrix.



The Hatchery "in action". On the top we can see monsters 1.1 and 2.2 harvesting the green pellet field. Monster 3.1 has just begun feeding the factory with his pellets whereas monster 2.3 is saving Team 2's pink egg from shrinking in the currently hostile environment. To the left, monster 3.2 is harvesting blue pellets, but it looks like one of them has just been stolen by monster 2.1's tentacle. To the right, monster 1.2 is inactive in the void; carrying two pellets. At the bottom, monsters 3.3 and 1.3 are harvesting pellets from the red and white pellet field, respectively. No monster is entering or leaving the field through the field entrance in the upper right corner.

- Eggs develop (= grow) if their color is close to the color in the hatching factory.
- Eggs shrink if the color of the hatching factory is somewhere close to "opposite corner", e.g. pale green eggs shrink in a purple environment.
- Eggs neither grow nor shrink of the color is grayish (in the middle of the color matrix) or in the other end of its column or row.
- Eggs can be taken out of the hatching factory of they are at all times carried by a monster. When they are outside they neither grow nor shrink.
- Once an egg is hatched (has reached the hatching size) it is sort of "done" and will no longer be affected by the color in the factory.

Locations and movement

There are several different, virtual places in the game. A monster is "at" a place if its card is placed next to a location card (such as red field etc). Note that the location card needs to have a free edge where the monster can be placed. Hatching chambers have only two active edges, the short ones, whereas all other areas have four. The possible locations for a monster are:

- Next to the red, white, blue or green pellet fields.
- Next to the hatching chamber (a monster can only be place next to its own hatching chamber and there are two places; the short edges).
- Next to the field entrance
- In void – any place else on the table. Monsters in void can be next to each other in the void if their cards lie edge to edge.

Movement is measured in time – it takes a certain monster a certain time to get to a certain location. From a computers point of view the locations reside in a virtual space where every location is exactly the same distance away from any other location (the cards making up the factory counting as one location), including the void area.

Depending on how many legs or wings a monster has, it will move faster through the virtual space. To practically move a monster you move its card from one location to another so that the monster card touches the location cars. The time it takes for it to actually get there is computed by the card and a small status bar is counting down on the monster card.

- A monster without legs or wings cannot move at all.
- A monster can carry as many pellets with it as it has arms.

- A monster may use all its arms to carry another monster of the same team. The monster being carried can still hold pellets.

Actions and pellets

Action is measured in time – it takes a certain monster a certain time to perform an action. Possible actions – all described in detail below – are to gather, switch, trade cooperate or steal to get pellets, to feed pellets to the factory, to remove eggs from the hatching chamber or put them back, and to switch monsters on the team.

Getting pellets: gathering

Pellets can be gathered at the pellet fields (one field per color). They are gathered by placing a monster card next to the field. The monster starts picking up one pellet per free hand. This takes a certain amount of time. Probably it takes a little longer to pick up three pellets at once, but not at all as not as long as 3 * 1 pellet. A small status bar on the monster card visualizes the time.

Note that tentacles can't be used for gathering pellets, only for holding/carrying. **Note** also that pellets actually *grow* in the field which means that there is not an unlimited supply at all times. It takes a certain amount of time for a pellet to appear, and all this is of course calculated by the pellet field card itself.

Switching pellets

Pellets can be moved from one monster to another by putting the cards next to each other and drag and drop the pellets between the cards. This can be done with any two monsters regardless of team. See trading below

Getting pellets: Trading

Two players can agree to trade pellets with each other. They just place the two monsters carrying the pellets they want to trade next to each other and drag and drop pellets between the cards. It **is** allowed to break a promise by not "paying" for a pellet that has been dragged to your monster.

Getting pellets: Stealing

If a monster with a *tentacle* is placed next to a monster of an opposite team carrying pellet(s) the tentacle monster will steal a random pellet from the other one (maybe one per tentacle, or this my be too powerful?). Only tentacles can steal pellets, not arms.

Getting pellets: Cooperating

At the start of the game each monster will get the special ability to create a pellet of one certain color if fed two pellets of two other determined) colors. For instance the monster Bob would produce exactly two white pellets if fed exactly one red and one green pellet. The monster Michelle would in a similar way produce two blue pellets being fed a red and a white.

However the two monsters feeding the producing monster (which they do by being put next to it and dragging the acquired pellets onto it) have to belong to **two different teams**. Any monster not belonging to the producing monster gets victory points. The producing monster gets the pellets but may share them by dragging them.

Feeding pellets to the factory

Pellets can be fed to the factory by placing one's monster next to the short (free) edge of one's hatching chamber and dragging the pellets onto it.

Removing or putting back eggs

A monster placed next to the hatching chamber may remove an egg from it, or put an egg back into it (by simply dragging the egg between cards). There may be a point in doing this if the factory environment (color) is bad for the egg and causes it to shrink. A monster needs at least two arms or tentacles to hold an egg, and it can only hold one egg at a time. A monster holding an egg cannot move.

Switching monsters on the team

A player may switch a playing monster against a monster from his or her deck by paying either 4 pellets of the same color or 3 pellets, all of different colors, at the game field entrance (location card).

Comment: The sum of pellets paid needs to be validated by playtesting.

End of game

The game ends when all the eggs in all chambers have been hatched. A player is stays in the game even if all his or her eggs have been hatched; he or she can still influence the color in the factory by feeding pellets to it, or earn points by cooperating with other players. The score is kept secret throughout the game – or maybe not.

Winning – getting points

Parameters for calculating score:

- A player scores points whenever his or her eggs hatch (the sooner the better).
- A player scores points whenever he or she has managed to hatch all his or her eggs (the sooner the better).
- A player scores points for cooperating with other teams (se cooperation above).
- A player scores points for number of pellets held at end of game?

The Hatchery: Special features made possible via computer augmentation

The Hatchery cannot be turned into a non-augmented board game, not if the original feel of the game and course of gameplay is to be kept. There are several features that need computational power to work as intended. Others could be transformed to more or less complex and tedious rules, rules that would be playable but not pleasing.

Features brought by computer augmentation

As mentioned, The Hatchery is not turn based. Instead every activity takes a certain amount of time, and this amount of time can vary depending on the abilities of the monster performing it. This enables continuous, coinciding play where all players can be active at the same time. However, this constant computation and keeping track of perhaps a dozen different but yet simultaneous time scapes is impossible to simulate in any appropriate way. One might argue that it is possible to use action points instead of seconds to count the "cost" of an action. In this case, the game splits up in turns, one per second. *"What does this monster do this second?" "It is* ***still*** *moving from the blue pellet fields to the field entrance." "OK. What does that monster do this turn?" "It has arrived at the blue pellet field and will start picking up pellets" "Yeah but there is only one pellet growing there now and it's not ripe"…* and so forth. Humans should not need to bother with these tedious tidbits when computers can.

Of course a player could instead get x action points per turn, spending them to complete any action(s) s/he wishes. In this case some of the logic of the game is lost, since it can be translated to the team of monsters sharing some kind of joint source of energy from which they sometimes benefit and sometimes do not. Furthermore,

in both of these cases the whole simultaneous flow of the game is lost and all actions have been turned into calculation. It is no longer about having an intuitive feel for how long things take, juggling monsters and keeping an eye up for opportunities and smart moves while the clock ticks, but about calculation and planning ahead. In this, it changes the game experience and player style completely.

Other things that cannot be simplified down to human computation are the algorithms keeping track of the color of the factory and how it affects how fast the various eggs grow. Firstly, one have to keep track of how many pellets have been fed lately, and of which color (bear in mind that the effect of the pellets wear off with time; yet another complication that is necessary, otherwise the color would move towards gray). Second, a shade has to be calculated using this information, Third, it must be determined which eggs are within a positive (growing) or negative (shrinking) interval. If, as suggested, the color also should spread a bit unevenly, or if eggs would grow faster if nursed, extra factors are added to the equation.

Features enhanced by computer augmentation

Provided that one throws away the most prominent feature of the game, that it is not turn-based, but time based, i.e. turning each second into a "turn", many features can be simulated with rules on movement, dice rolling etc.

Monster movement in the original game relies on a) the fact that some monsters move faster than others due to their abilities and b) that there is an equal distance between all locations on the board, including "the void". The former could be simulated by translating each monsters movement abilities into a number of movement points ("steps"); the faster the monster is, the more movement points it can spend in a turn, i.e. the more steps it can take. This requires that we set up distances in terms of steps between locations, in the form of tracks (in case monsters don't move the entire distance in a round it is necessary to keep track of how far they've come) to place the monsters on. This requires some kind of board, which again meddles with the original rules. Again, by turning a time-based entity into some kind of points residing in a time-free dimension, we change the game from an intuitive and fast game into one about calculation and planning.

Pellet growth too, could be simulated by either rolling dice or just adding pellets according to some standard rule ("each round each pellet field grows five pellets"). However these simple rules do not take into account that the field may be affected by constant harvesting. More complex rules could take that into account of course, but such rules are note directly concerning the players' first-hand and therefore run the risk of being forgotten or regarded as a boring "maintenance".

Result: New Mechanics

As has been described in the case and in the background, the prime reason for computer augmenting a board game is that this takes the burden of keeping track, scoring, randomizing, calculating and changing items/modes/board layout off the players. They no longer need to meddle with such "maintenance" and can concentrate on playing the game. Of course, board gamers are not more stupid than other people (sic!). But using this technology we can make games more complex, more flexible, and more interactive without making the game harder to play – only more intriguing.

This section summarizes a set of mechanics that can be used for designing, or occur in, a computer augmented game. There are both altered mechanics and entirely new ones; they have been found by analyzing the existing set of mechanics as well as by designing and/or analyzing computer augmented games per se. The numerous examples are taken from my own collection of games.

But, before diving into this, just a short caveat: Just because a mechanic can be enhanced, this does not mean that it should be. Of course, the roll of a die can be substituted with a random number generator, but why? One mustn't forget that there is a significant difference between rolling a die and pressing a button to get a random number. It feels different, the items involved act different etc. Here, we find a strong aspect of interaction design.

Also, strictly speaking, some of the mechanics mentioned below do not actually need computer augmentation. Just as most parts of The Hatchery *could* be run by humans, they can be simulated; players can keep track or count or roll dice. But again, such tasks are tedious, and if accomplished erroneous they can affect the game negatively.

Transferring traditional mechanics into computer augmented ones

Of the traditional set of mechanics, the ones suitable for computer augmentation are the ones that are information-related (Lundgren 2002, p. 70-72); hardly surprising since the strength of ubiquitous computing is gathering, keeping, processing and expressing information. These altered mechanics can all be classified as handling the following:

- The resources or combination of resources a player has.

- Hidden information submitted by players or based on each player's resources.
- Which items are on the board, where they are and if they are allowed to be there, in some cases also the state of them.
- Actual state of various conditions in the game, such as prices of various resources and how they could change.
- Calculating, keeping and displaying information, or parts of it, for instance the score.

Novel computer-augmented mechanics derived from existing mechanics

Active Board

This mechanic is derived from Modular Board, a mechanic that allows the board to change during the game, e.g. by adding or removing tiles.

Definition: The board is (inter)active. By itself it can generate and sometimes show hitherto secret information. In this, the board itself changes its appearance. It can also be active in the sense that it collects information regarding what is going on (e.g. who moved where). Sometimes it displays or reacts on this information.

Explanation: This mechanic is used in The Hatchery, for instance in the hatching factory that changes its color by itself, and in the pellets fields where pellets grow at a certain speed. It is also used by Mandryk et al (2002) in the *False Prophets* game. The board was projected onto the table and changed appearance depending on what the players did and how they moved. With a projected board, it is very easy to change the appearance; any change in pixels will be shown. The projected board in *myTHeme*, showing the ongoing story, the score etc, is also an example.

Technical suggestions: How this should be done is highly dependent on what information is to be displayed when and why. If the entire board is being projected on a tale or on the wall, it can be easy to show changes.

Anonymous Trading

This mechanism is a variant of the traditional trading mechanism.

Definition: Players anonymously propose the trades they are willing to make. The game calculates the offers and matches the trades. The outcome is shown.

Explanation: One significant aspect of trading is actually whom you are trading with. A deal that is perfectly acceptable with one player is unacceptable with another,

since the second player may gain too much of an advantage though the deal, or might even win, whereas it is "safe" to make the deal with the first player. Of course this can be a balancing factor in a game because typically players are reluctant to trade with the leader, but in trading-intense games it can just as well impair an early leader far too much. It could just as well be a feature to provide anonymous trading; this would change gameplay significantly. In the example below, players state (to the computer) which kinds of pieces they are willing to sell and which ones they want to buy and what they will pay for that, e. g:

```
* Blue wants to sell 1 castle for 1 gold
* Blue wants to sell a store for 2 gold
* Blue wants to sell a tower for any price
* Blue wants to buy a farm for 2 gold
```

The computer calculates this, and gives some output in form of text, images on a screen or LEDs, creating an outcome like:

```
* Blue sells 1 tower to Red and pays 1 gold to Green (for 1 castle).
* Red sells 1 store to Green and pays 2 gold to Blue (for 1 tower).
* Green sells 1 castle to Blue and pays 1 gold to Red (for 1 store).
* Yellow didn't trade.
```

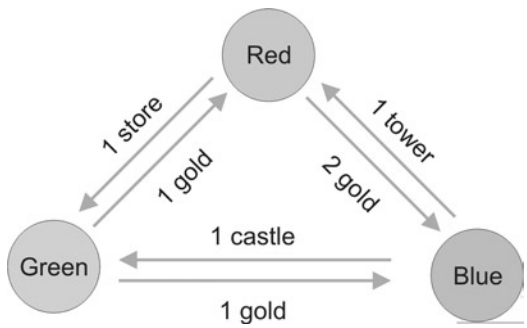


Figure 13: A graphical representation of the trade.

Note that Blue didn't manage to sell the store for 2 gold; Red's store was cheaper and so Green bought it instead. Neither was there any farm for sale for 2 gold. In this case it is quite obvious who traded what with whom and for how much.

However, the trade can be much more secret, if the output is changed to something like:

```
* Blue gets 3 gold and gives 1 tower and 1 castle
* Red gets 1 tower and gives 1 store and 1 gold
* Green gets 1 castle and 1 store and gives 2 gold
```

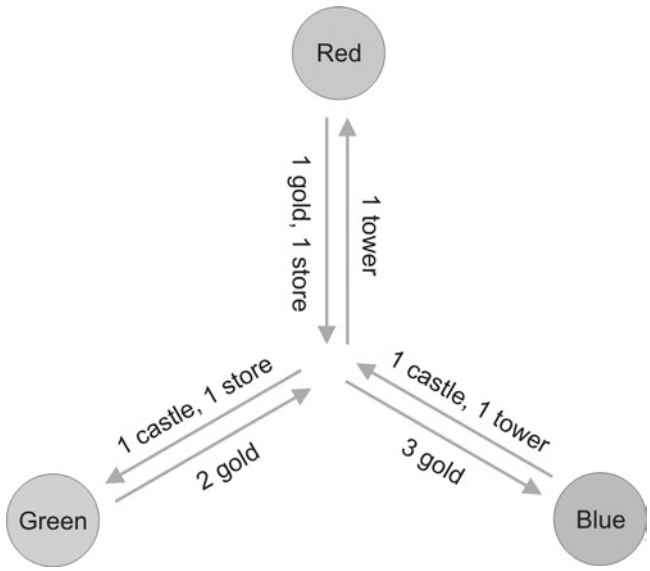


Figure 14: A graphical representation of the trade.

In this case we do not get to know that Red was desperate enough to pay two gold for the tower; we just see the final, balanced outcome. As far as we know, red could have bought the castle too from Blue, selling it on to Green. It is of course up to the game designer to decide how obscured this information should be.

In both cases, haggling has been replaced by second guessing the other players, but also more carefully calculating what one can afford. It also opens up for some unexpected happy outcomes ("*Oh my gosh, I actually got a library!!!*").

Technical suggestions: The simpler the trades, the more suitable for computer augmentation they are. When trading borders negotiation, it is probably best left to humans. Now they key to augmentation this lies in how to enter one's trading wishes. Most likely one would want to state both what one wants to sell and for how

much, even if there are games where only one is stated, e.g. the game *Res Publica*[40]. Also, the computer must of course have some rules on how to perform trades, for instance how to deal with two customers offering the same price for one item.

One way would be to have a "trading display" (see below), either one per player (better but more expensive) or only one that is passed around by the players. It should have a couple of areas where tokens could be placed; e.g. one for what the player primarily wants to buy, and one for his or her second wish, and then one for what he or she would like to pay, or if money isn't involved areas for the items or coins that the player wishes to give away. The tokens could be recognized using RFID technology, weight sensing or some other suitable technology. Each player would also need a special set of trading tokens.

One could also imagine suggesting several deals, e.g. "one castle for one store" or "one tower for one store". In that case, the areas should be divided into "buy-sell"-pairs. In either case, the player also has to put a identity chip somewhere. Also, there has to be a way to tell the computer to calculate the input.

Why not use buttons also for entering the wishes? Well, this doesn't make the interaction as natural, and it would also be necessary to provide feedback regarding which information had been entered.

Complex Commodities

This mechanic is based upon the mechanic Commodity Speculation, combined with Active Board.

Definition: There are complex relations between the prices of different commodities as the supply and demand of them are interdependent. The prices may also depend on other occurrences in the course of the game. Thus the prices are calculated, kept up to date and displayed by the game.

Explanation: This mechanic can be used in stock market games. For instance, the price of meat will rise if the price of hay rises as an effect of a dry summer, whereas the prince on iron remains stable and the price on electricity drops. Such complex behaviors would be tedious to determine with die-rolls and calculations, but easy to simulate with a simple computer program.

Technical suggestions: The board needs to be able to show the prices somehow; if it's projected this it not a problem. The prices could also be shown on built-in digital displays, or as small LEDs (diodes). If changes can be triggered by players

[40] *Res Publica* by Reiner Knizia 2000, published by Avalanche Pres Ltd & Hexagames/Queen Games respectively)

actions (e.g. buying huge amounts of iron, or playing the "War" event card) the game needs to collect this information, or players must be able to input it actively.

Computerized Clues

This is of course a variant of Clue-giving, and it can enable much more interesting clue games.

Definition: There is information in the game that is not common, and that the game distributes occasionally, and not necessary publicly. Different players may get different pieces of information depending on their actions. Players may also get secret information on other players. At certain stages in the game all information may be revealed.

Explanation: This is for instance used by Mandryk et al (2002) in *Pervasive Clue*. An example of a game possible to augment (albeit this will change gameplay) is for instance the game *Dragon's Gold*[41].In this game players cooperate to kill dragons and then negotiate about how the dragons' treasure should be divided amongst them. The treasures are part open, part secret. Using computerized clues, the treasures could be entirely hidden, the game only displaying information like which one of the four treasures that is the largest one, which one of them that contains most green/blue/etc gemstones, which one is worth the most points in terms of gold and silver and so forth.

Figure 15: Dragon's Gold, a dragon with its treasure.

Technical suggestions: The technical solution depends highly on what the clues are and whether they are generated and known by internal code in the game, or if they are based upon what happens on the board.

Secret Bidding

This mechanic is of course inspired by the regular mechanic Bidding.

Definition: Players may bid secretly on an item. Only the player who wins the bid and the sum that is bid is displayed. Therefore, no one will know if and how much the other players bid – or if they bid at all.

Explanation: This mechanic can be useful if the secret bidding is to be kept *really* secret. In many bidding games it is a disadvantage to show that one is eager to buy

[41] *Dragon's Gold*, a.k.a. *L'or des Dragons* a.k.a. *Drachengold*, by Bruno Faidutti 2001, published by Descartes Editeur.

a certain item, especially if there are more for sale. It can be used both for one-bid bidding where all players enter only one bid, or for bidding where players may raise their bid. In the latter case, the game must display the latest bid, but not who bid. This would make it possible for the player who is selling to raise the bid somewhat (in a non-obtrusive way) by bidding himself.

Technical suggestions: For instance**,** each player places his bid (e.g. coin tokens) and an ID-marker ("Blue player" etc) on a special tag reader that is passed around, or each player has his or her own reader. The game announces who won the bet (using a projected board, sound output or whatever) and what he/she is going to pay. The bids of the other players remain secret.

Secret Partnerships

This mechanic is derived from the traditional game mechanic Partnerships. Most likely it will have to be combined with Computerized Clues and/or Active Board to work well.

Definition: Players are divided in teams, without knowing with whom. The combined actions of a team lead to certain consequences in the game (calculated and displayed by the game itself), wherefore it is important to figure out who is on which team.

Explanation: In most games, partnerships are explicit. All players know who is on what team, and they base their decision making on this. Using computer augmentation, a game could certainly have an element of trying to find out who is actually teaming with whom, and who one's own conspirator is.

In it's simplest form, and combined with the mechanic Computerized Clues (see above), each player plays an X or an O, and the computer then gives output regarding the amount of X-s and O-s played by each team, or how one team looks or that none of the players that played X-s in this round are on the same team, etc. Of course the game should be more complex than this, getting the players to secretly maneuver to find out information about the teams, since the configuration of teams affects what happens in the game. The goal in the game itself does not have to be to find out who's on one's side, it might still be to achieve as many points as possible, but cooperating with one's team mates should then increase the chances of getting points.

Technical suggestions: How this should be done is highly dependent on what the rest of the game is about.

Entirely new, computer augmented mechanics

These mechanics are not derived from a certain mechanic, or can be used to enhance many types of mechanics.

Active Dice

Definition: The probability of the outcome of a die roll may be skewed or may depend on the outcome of previous rolls.

Explanation: A die will always provide a certain probability for a certain outcome; using an ordinary six-sided dice, the probability to roll 6 is 1:6 etc. Similarly two or more dice used together will create a bell curve outcome, for instance 7 is the most likely sum to roll with two six-sided dice whereas 2 and 12 are the rarest outcomes. If one wants a skewed probability, this can be hard to obtain with regular dice, even if they are numbered irregularly. Say for instance that one wants to make 2 and 12 as common as 7 without affecting the probability for the other numbers. This is impossible using dice (but can be achieved by using cards). In addition one might also want the probability of a die roll to depend on the outcome of the previous roll, or on the state in the game.

Technical suggestions: Unfortunately it would be hard to build this into regular dice, since they are small and need to be robust. The randomizing can the either be embedded in the board, shown on a display, or some kind of device has to be built if the tactile qualities are regarded important. If the outcome of the die roll should depend on the state of the game, this state must of course be able to measure, and the information has o be transferred to the dice. This would be easily accomplished with the SuperBoard; see the Reflections.

Active Surface

Definition: The surface of the board changes its properties during the game.

Explanation: One example of this would be that the topography of the surface changes physically. It could also be entire sections moving, becoming hot or cold or lit or whatever. If metallic pieces are used, parts of the board could suddenly become magnetic (using electro magnets), effectively capturing or moving the pieces!

Technical suggestions: How this should be done is highly dependent on the way the surface should be active.

Active Tiles

Definition: Tiles recognize which other tiles are next to them and react to this.

Explanation: If tiles can recognize each other they can be made to change appearance or behavior depending on which tile is put next to them. This can be done with the MultiMonsterMania cards as for instance in The Hatchery where monsters with tentacles can steal pellets from other monsters if being placed next to them.

In a resource management game like *Settlers of Catan*[42] where tiles depict fields (delivering grain), meadows (delivering sheep), woods (delivering timber) etc. one could for instance imagine that the sheep would like it better in a larger meadow, i.e. two meadow tiles in connection would result in more sheep than two single ones etc. Or perhaps the inverse; if too many fields are adjacent this might impoverish the soil, also affecting the size of the crop?

Or imagine some other game, where four blue tiles can be put next to each other in order to form a river, which is too wide and rushing to cross (as opposed to a more tame, three-tile river). Now, if the tiles "know" that they form such a river, they can prevent pieces to move over them by beeping for instance and they could change color or appearance (perhaps depicting streaming water instead of calm water).

Technical suggestions: This can be implemented as in the *Triangles* project (Gorbet et al. 1998) where each triangles had a microprocessor inside and magnetic edge connectors which made it possible to physically connect the triangles. Through these digital information was carried. Other ways are to use short-range radio identification or infrared emitters and sensors. Output (i.e. the status of a tile) can be shown by LEDs, or sometimes by sound (e.g. when moving a piece over a forbidden tile).

Espionage

Definition: Using this, players get the possibility to gain more or less accurate information on other player's resources, however risking that the own information leaks over to the player who is spied upon…

Explanation: In many games it can be very important to know how many and/or which resources a player has at hand; it can be money, building blocks, shares or whatever. Some games have built-in spying features (normally an action card that allows a player to take a quick glance at another player's hand or resource pile. In this case it is obvious to everyone who has spied on whom; especially the player who is being spied upon is aware of this.

[42] *Die Siedler von Catan*, a.k.a. *The Settlers of Catan* a.k.a. *Les Colons de Katane*, by Klaus Teuber 1995, published by Mayfair Games, Kosmos and Patnik respectively.

However spying in the real world isn't as obvious and uncomplicated. Sometime spies fail to fins something out, or get somewhat skewed information, or get caught and interrogated, or are even double agents…! This could be implemented in a game. For instance players could send (more or less?) reliable spies out, perhaps with more or less accurate missions (*"Find out the exact amount of money that Blue has"* or "*find out if any of the other players have a Black Lotus in their hand"*).

Players may or may not know how accurate the information is that they get. Players will not know if they are spied upon, but might sometimes get some not-asked for information (simulating a spy being caught or trading information). The player whose spy leaked information may or may not know if or what was leaked.

Technical suggestions: This could be accomplished either if each player had all his information on a PDA, in which case the information could be exchanged wirelessly, or via RFID. The latter requires that each player has his or her own reader that both can register a player's resources and read one's spy request, transmitting it to some central unit that in turn collects information on resources from all readers in order to produce an answer. The latter encourages cheating however, since players can refuse to put their resources on the reader when they earn them.

Ubiquitous Information

Definition: In this case the game itself keeps track of when certain conditions are fulfilled and acts on it. This can be appropriate both as a clou in the game but can also be supportive if there are conditions that are hard to take in.

Explanation: This mechanic is very similar to Keeping Track and can also be applied with any one of the following Goal mechanics: Area Enclosure, Matching, Set Collection, and Pattern Building since all of these are quite easy to detect.

For instance a game can have a winning condition that says that it ends when three out of four players have gathered at lest five of the seven types of jewels (or stones, or buildings or workers, or any other resource) that exist in the game, or if the total amount of earned money in the game exceeds x, or if there are more white tokens gathered than blue ones etc. If these resources are hidden, it is hard or sometimes impossible (tokens can for instance be drawn hidden from a bag) to know if the goal is reached; players can only have a feel for it. However the game will know, and will tell. This impossibility to foresee exactly when the game ends can be a feature. Also, this mechanic would go very well together with Espionage (as described above).

Technical suggestions: This can be made possible if using RFID-technology or some other kinds of sensors, e.g. light sensors or compression sensors that can keep track of the deployment of items on the board or at the player's resource stacks.

Supportive "mechanics"

These aren't mechanics per se, since mechanics normally describe what is done in the game. Instead, these mechanics visualize information and thus enable more complex games and rules without burdening the player with having to implement them or keeping track of them.

Informative Board

Definition: The board keeps track of information that is due to change in a pre-defined way, and always displays the current information.

Explanation: Many complex games are divided into phases consisting of a certain number of rounds. This is used for example in war games where armies' weaponry may change with time. Another example is *History of the World*[43] where players play different empires in different epochs. Between such phases, certain prices may differ, and some actions are only possible in some phases, the number and even existence of certain items change etc.

Now, if the board gets to know which phase it is in, it can automatically upgrade itself by changing the actual information, redraw the map or whatever has changed. In this way it would display information that would otherwise come from more or less complex tables or would have to be kept in mind by the players themselves. There aren't that many games as complex as this today, but if using an informative board, they could be.

Technical suggestions: If many things shall change, it would be easiest to accomplish this with a board that is projected onto the table. If the phases depend on the number of rounds it would be easy to have a switch or something similar to input this information to the game.

Keeping Track

Definition: The game keeps track of a player's resources and displays them to him or her.

[43] *History of the World* by Gary Dicken & Steve Kendall 1991, published by Avalon Hill, Ragnar Brothers, Gibsons Games, Welt der Spiele, Avalon Hill (Hasbro) and Compendium Games respectively.

Explanation: In many games – especially auction/bidding games – players want to keep track on how much money they have. This becomes especially important if you don't get any change back when paying (as in the game *Outpost*[44] for instance); in this case players have to count and recount in order to see what they can actually bid. The same technology can of course be used for keeping track of other things.

Technical suggestions: In this case one might want to keep track by using RFID-technology. Each bill or coin would have an RFID-tag containing its value and the tag reader would read their value. The correct sum would be added and kept by a microcontroller showing it on a small calculator-like screen on the reader. It could also carry a small program that calculates which exact sums can be bid.

Reflections: Computer Augmenting Board Games

Will computer augmenting a game automatically make the game better? *Of course not!* As always, the game must be designed carefully, and if the computer augmentation does not add anything extra to the game experience, or if computer augmenting leads to more trouble than it is worth, then it should be designed out again! And designing a computer augmented game does bring about special problems – as well as possibilities.

Problems

Computer augmenting a board game brings its own difficulties. For instance the game will be more vulnerable than the average board game. It will most likely need electricity to work, or at least batteries. If it breaks it might not be possible to repair it, and similarly if some pieces get lost they might be that easy to replace. It will cost more – this latter problem is diminishing however.

Above all a computer augmented board game will take even more time to design since the interaction design part of the game will need a lot of thought as well; technology should not be in the way, not interfere with the game experience but silently improve it. This is harder than it sounds. There are at least two distinct problems. One, the simpler one, is related to the visual style of a board game. It might look weird to mix beautiful paintings of a fantasy landscape with a small display showing the score, or some light emitting diodes in the midst of it all. Maybe

[44] *Outpost* by James Hlavaty 1991, published by TimJim Games.

a new visual style had to be created, or perhaps these problems diminish if the board is projected onto the table and the numbers/displays thus can be made to fit the rest of the (graphic) design. Perhaps a new aesthetic style for computer augmented board games needs to be developed.

Another solution is to let most of the output come in the form of sound. This is done both in *King Arthur*[45] and *Stop Thief*[46], two of the commercial board games mentioned in the introduction. In the third, *Dark Tower*[47] sound is used too, but along with movement of the tower etc. Sound is of course tempting to use since it is a natural form of output, easy to bring about as well, and does not cramp the visual style. Nevertheless, it also brings with it its own difficulties; the sounds need to be clear, sound well and be easy to distinguish from each other. If the sounds consist of speech (e.g. spoken instructions) the game becomes language-dependant. Also, a sounding game requires that the environment is silent and that players don't speak or make other noises when the game sounds. Thus an opportunity to be able to replay the sound is an absolute necessity.

A larger problem is that interaction with the game might need to be more precise. Note that this mustn't be the case; it depends on the used technologies. But RFID for instance, requires the player to put a tagged item on the reader in order for it to work. This is not a problem if it is natural for the player to do so anyway (e.g. by putting his tagged boat onto a water tile which contains a reader) but it might become awkward if, say, the player first has to "register" a piece before placing it on the board, by placing it on a reader next to the rim. If this is not incorporated in the rules in a natural way (e.g. new tourists enter the island via a bridge which happens to be the reader) it will easily be forgotten, and will also easily be regarded as a nuisance.

In addition it is harder to create prototypes quickly; even adding sensors etc – even if rudimentary – to a game requires both time and skill. Luckily, computer augmentation can be faked by a human of course; this is called *Wizard of Oz-testing* (Salber and Coutaz 1993, Preece et al 1994). A (preferably distant or hidden) human – the Wizard – performs the tasks of the computer, just as the terrifying Wizard of Oz was actually controlled by a small man behind a curtain in the book with the same name. E.g. a human giving away the "computerized clues", calculating the

[45] *King Arthur* by Reiner Knizia 2003, published by Ravensburger.

[46] *Stop Thief*, published by Parker Brothers 1979.

[47] *Dark Tower*, published by Milton Bradley 1981.

anonymous trade etc. but a human will never do this as fast (and sometimes not as accurate) as a computer would. This is a known problem for anyone using low fidelity prototypes in order to simulate functionality; still a low fidelity prototype test can be very valuable, if testing the right things, the ones the prototype can accomplish. It might give a notion of what works and what does not, but when analyzing these results on must keep the differences between computer and human performance in mind. Unfortunately, some things (e.g. the slowly changing color in The Hatchery) are impossible to simulate.

Furthermore, computer augmentation brings with it a lot of extra testing. The interaction per se needs to be tested as does the technical solutions. Among other things the latter implies a need for stress testing the technology by making as many faults as possible, and see if the game can handle the difficulties that appear, which in many cases may require a redesign of the rules. Also extra expertise may be needed to design the game. It's not just the game designer(s) and the illustrator anymore; there might be a need for an additional interaction designer and/or programmer and/or someone knowing a lot about ubiquitous computing.

A possible solution; the Super Board

In order to avoid many of the difficulties mentioned above, one could imagine a possible SuperBoard; a device consisting of (perhaps) a flat screen equipped with compression sensors, capacitive sensors, a net of RFID-readers or whatever, serving as board, enclosed in a frame containing input devices like RFID-readers, buttons etc and output devices like loud speakers and lights (e.g. one for each player), and of course embedded computational power. In this it would be very much the same as a game console for computer games (e.g. Xbox or PlayStation), and in the same manner it would be able to use for many different games, since the appearance and behavior of the board could change from time to time. When buying a game one would get only software, the relevant pieces and perhaps an overlay for the frame or parts of the frame. This would mean that the games would be cheaper and paying for the Super Board would be a larger one-time investment. From the world of computer games it is clear that this model works. It would of course also bring about certain restrictions when designing the game, i.e. if the Super Board does not implement magnetic sensors, there would be no possibility to use magnetic/metallic pieces. Then again, when designing a game for a console there are similar constraints.

Reflections and lessons learnt

When looking back at the game design experiments I have been involved in, it seems that the one being fully implemented – myTHeme, see Facet No. 5 – was the one with the least complicated solution. It is based solely upon RFID-technology, text and Java whereas other concepts have relied on myriad of motors, buttons and diodes. Neither MultiMonsterMania nor The Hatchery require this type of specially constructed hardware, but instead demands code distributed to a number of freestanding but yet communicating units, and complex algorithms as well as touch screens, advanced moving graphics etc. During the workshop when MultiMonsterMania and The Hatchery was created, we actually did have tools and materials enough to implement prototypes and one group actually created a rough version of their game (Björk et al. 2002), but we never even played with the thought of prototyping. The task was simply to complex for that short amount of time.

There are of course several reasons for why some games never make it to the prototyping phase. For instance myself and my colleagues are and were significantly more software oriented than hardware oriented, which is why we chose to create a game like myTHeme; this decision was of course steered both by our intentions as researchers as well as our competences. In a similar manner many of the research projects where games have been created have been about exploring or using known techniques in order to explore something in context of the game, rather than creating a game per se. If we look at the commercial games again, they too are rather simple, in terms of the amount hardware and software they require to work. In the commercial cases this was probably both a matter of cost and limited resources (in terms of expertise) when developing the games.

Still all of these examples give an indication that cannot be ignored: cut your coat according to your cloth. Do not engage in complex technological solutions without sufficient support! Start out with something simple, like RFID; Simple will take you far enough, given the competition. And remember; less is more.

Possibilities

Regardless of all the *don't*s and *if*s and *not*s mentioned above, I truly believe that new and exciting board games can be created along the borderline between board games

and computer games! As described in the new mechanics above the computational power can help the game designer to:

- **Hide information from some or all players** (Anonymous Trading, Computerized Clues, Espionage, Secret Bidding, Secret Partnerships, Ubiquitous information)
- **Clarify and update information** (Active Board, Active Surface, Complex Commodities, Informative Board, Keeping Track)
- **Enable interaction between components, as a reaction to what happens in the game** (Active Board, Active Surface, Active Tiles)
- **Enable complex rules for the behavior of certain occurrences in the game** such as prices or probabilities without burdening the players with calculating or implementing them (Active Dice, Complex Commodities, Informative Board)[48].

All of this enables new game experiences, new possibilities, new adventures and a richer design space for the game designer/interaction designer. It is a whole new area of gaming out there, go get it!

To conclude, it is my firm conviction that computer augmented board games *will* become a part of future gaming. I believe this because computer augmentation *does* bring features humans cannot, and I believe it because hardware becomes cheaper and smaller and easier to manage for every second, and I believe it because computational power as phenomenon is soon as ubiquitous as electricity, and I believe it because this revolution brings with it a new wave of designers skilled in interaction design, ubiquitous computing, programming etc; young people looking for jobs! If the creative minds of these are combined with that of the game designer, there are hardly any limits. Get on with it!

[48] This is especially common in computer games: we happily play the mayor of *SimCity* (Will Wright 1996, published by Maxis) without exactly knowing the complex algorithms for calculating how happy our citizens are, but still we have a general feeling for that they need a sufficient amount of work, shops, school, electricity, garbage removal etc. to be content – and this is enough.

FACET No. 4:

Game Design Patterns – an Approach to Designing Games

This chapter describes the concept of Game Design Patterns, which is both a framework to describe the constituents of a game as well as a set of patterns that can be used when analyzing, discussing, describing or designing games.

"Okay so they want us to create a collectible, interactive card game featuring ***coffee,*** *of all stupid and uninteresting things in the world???"*

"Obviously yes. It seems to be a joint venture with some large food companies. The idea is to have the kids drinking coffee instead of Coke."

"So this is a marketing game???"

"Relax. We can still make it a good game"

"I hope we get a free supply of coffee doing it because we're gonna need it"

"Yeah. Okay, which patterns can we use?"

"If it's supposed to be a collectible we might want to give some kind of ***Reward*** *for* ***Collecting*** *as many types of cards as possible."*

"Okay, if there should be a point in that the different types of cards need to have ***Asymmetric Abilities.*** *"*

"Absolutely. And you shouldn't need the entire collection to play but if you do you'll definitely have better odds"

"Odds – do you want it to be a ***Betting*** *game?"*

"I didn't really think of that but with that poker hype…"

"Okay let's do it! What about the theme… coffee, coffee, what can you do with coffee?"

"Wake up. Study. Invite someone for it. Pour it over someone you dislike… ehhh… get warm, get cold, get abstinence… "

"Better skip the abstinence part!"

"Okay here's an embryo: The game ***spawns*** *a set of people; some are cute and should be hit on – by offering coffee of course – some should be scared off and some should simply be… ah I dunno. And you play different types of coffee cards to do this… it's a kind of* ***Bidding*** *but you* ***bet*** *on the outcome…? Who will get the babe, who will get stuck with the insomniac class mate?"*

"A good set of patterns but the theme is a bit… wobbly, ain't it?"

"We'll have to work on it. Let's go get a cup o' Joe…"

The focus of the Game Design Patterns Project is to create *"a tool for understanding and creating games*" (Björk & Holopainen 2005). It consists of a framework that describes the constituents of a game, as well as a rich collection of Game Design

Patterns that can be used whenever analyzing, describing, discussing or designing games. The project uses the concept of Design Patterns, originally developed for architecture by Christopher Alexander et al. (1977) as basis. This is not uncommon; Design Patterns is a concept of semi-structured formalism that has been used for similar causes in areas such as software engineering, interaction design and many other.

For me, the Game Design Pattern Project was an attempt to meet some of the needs expressed by professional game designers; the need for common game design methods and models as well as a common language (c.f. Costikyan 2002, Spector 1999) [49].

Background: Pattern Languages

The pattern concept is attributed to the architect Christopher Alexander who in the late 1970ies wrote the two seminal books *A Pattern Language* (1977) and *The Timeless Way of Building* (1979). The latter contains the theory as well as instruction for how to use patterns, whereas the former mostly contains a pattern collection. Alexander's notion was that towns and buildings should be made "alive" by the people of that society, and that a common language was needed for this discourse; his proposed pattern language on architecture. In the first chapter of *A Pattern Language* Alexander describes it as follows:

> *You can use it to work with your neighbors, to improve your town and neighborhood. You can use it to design a house for yourself, with your family; or to work with other people to design an office or workshop or public building like a school. And you can use it to guide the actual process of construction.*
>
> *The elements of this language are entities called patterns. Each pattern describes a problem which occurs over and over again in our environment, and the describes the core of the solution to that*

[49] The Game Design Pattern project has been run by Staffan Björk and Jussi Holopainen from 2002 to present. I was a part of the project during the start in 2002 – 2003 and took part in several workshops as well as in setting the pattern structure and writing several patterns. The contents of this chapter are based upon the paper "Game Design Patterns" (Björk et al 2003) which I co-authored, the paper "Describing Games - An Interaction-Centric Structural Framework" (Björk & Holopainen 2003) and the resulting book, "Patterns in Game Design" (Björk & Holopainen, 2005), as well as on my own, current reflections.

> *problem in such a way that you can use this solution a million times over without ever doing it the same way twice.*
>
> *- Christopher Alexander*
> *In: "A Pattern Language" (1977)*

Alexander's patterns cover all aspects of architecture, from what to consider when planning a public transportation network and shopping areas via design of specific buildings like a town hall down to how to design a pleasant office. As an example we can examine the pattern No. 132, *Short passages*. In this pattern, it is stated that the "problem" is long, sterile corridors which are regarded as unpleasant, and the set of possible solutions include: allowing lots of natural light to illuminate the corridor; animating the hallway by providing interior windows to the adjacent rooms (however without disturbing the privacy of the workers); furnishing the hallway with art, chairs and plants etc.; and possibly widen the corridor into smaller rooms every once in a while. The solutions are vague in the sense that no exact drawings are provided – naturally since every site is different, but still rather specific and concrete in what can be done. This is what Alexander means when he claims that the same solution can be used "a million times over without ever doing it the same way twice". Also, patterns are related to each other, for instance the issue on how to keep an open landscape without disturbing workers is discussed in the patterns *Half-private Office* (No. 152) and *Workspace Enclosure* (No.183).

Since, pattern languages have been developed for other disciplines too. Although differing from subject to subject, most of them are concerned with noticing and naming common problems, describing the essentials on how to solve them. The result is access to many different paths through a design process, as well as a way to see how problems are interrelated.

In the book "Design Patterns: Elements of Reusable Object Oriented Software" (Gamma et al 2004)[50] patterns entered the programming sphere. Whereas Alexander and his peers express a clear problem-oriented view (e.g. that people dislike living in skyscrapers and what to build instead), Gamma et al's view is not as strict; even if the terminology is kept here, patterns are presented more like possible design demands rather than problems, i.e. reasoning like *"Some times objects can have different states, and these are the possible ways to achieve this."* Also, the relational structure between patterns isn't as elaborate as in Alexander's collection.

[50] The authors as well as the book itself is often referred to as "The Gang of Four" within programming circles.

Within the fields of Human Computer Interaction and Interaction Design several people have investigated and written patterns creating a language. Thomas Erickson (2000) advocated a "lingua franca" for interaction design and others followed. Jan Borchers (2001) discusses patterns in general and presents a pattern language specific for interactive music exhibits in his book "A pattern approach to interaction design". Also, patterns within specific areas of interaction design have been collected, e.g. patterns on web usability (Graham 2003) and user interface design (Tidwell 2005).

As for game design, Berndt Kreimeier (2002) published an online article called "The Case for Game Design Patterns." Ergo – the dawn of the game design pattern project.

Topic: Game Design Patterns

As described under "Current game Design methods" in the Intermediate, there is clearly a need to analyze games and game design methods from the viewpoint of the game designer, focusing on the most essential part of a game: gameplay. Björk and Holopainen (2005) define gameplay as *"the structures of player interaction with the game system and with other players in the game"*. This is a very interaction design inspired approach as it includes not only the possibilities and results, but also *reasons* for the player to interact with the game and the other players (if any).

All of this led us to the development of Game Design Patterns, as suggested and inspired by Kreimeier (2002). They were partly inspired by and founded in the board game world's mechanics (see Facet No. 3). A game mechanic however, is a very short description defined as "*Part of a game's rule system that covers one general or specific aspect of the game.*" (Lundgren 2002, www.boardgamegeek.com) whereas a game design pattern is a much more elaborate description, including examples, how to use the pattern, the consequences of use and how it is influencing and is being influenced by other patterns; just like in Alexander's original model. However, it must be mentioned again that Alexander's original patterns for architecture are very focused towards problem-solving; one has a problem and sets out for a curing pattern. Furthermore Game Design Patterns have a rather different approach in that they simply describe patterns as parts of a game, rather than "problems". Game Design Patterns cannot be judged in terms of "good", "bad", "problem" and "solution"; instead of one game design pattern being a solution to another, it is

simply so that using one pattern may induce another, or can only exist with another pattern as prerequisite, e.g. the pattern *Dice* automatically instantiates *Randomness*.

There are several reasons for letting Game Design Patterns remain neutral. The most prominent one is that a pattern that is regarded "bad" and unwanted in one context (e.g. *Elimination* in a children's game) may be desired in another (e.g. in a war game). Furthermore, seeing patterns as problem-solution pairs would turn Game Design Patterns into a problem-solving tool instead of a design tool; this view would severely limit creativity and possibilities. Moreover, the relations between Game Design Patterns are very complex; most patterns have relations with a dozen or more others, which firstly makes it impossible to form problem-solution pairs and secondly implies that they do not really work as patterns might do within other disciplines; adding or removing a pattern from a game influences, instantiates or eliminates many other patterns that are part of it, thus changing the balance and gameplay of it. Therefore, using patterns is similar to any other design process since; every pattern choice affects the others just as every design decision affects others, in turn affecting the whole of the design.

Design challenge

The challenge in the game design pattern project was to create a common language for game designers, as well as a common game design tool. This meant collecting and analyzing patterns from all kinds of games, trying to categorizing them as well as ordering them into some sort of framework.

Finding and refining patterns

The patterns were collected by analyzing and improving existing mechanics (e.g. Lundgren 2002), by analyzing games looking for patterns, and by interviewing game developers. The concept was then tested and iterated in a series of game design workshops.

The framework; the context of the game design space

When trying to order or categorize patterns, a framework for them was compiled in the process. This framework is generic and can thus be used to describe any kind of game and it can be used as an overview to the design space of creating games.

Our focus has been that games can be seen as interactive systems with gameplay serving as the living "soul" of the game. Thus, the framework that the Game Design Patterns exist within was constructed under the assertion that playing a game can be seen as making changes in quantitative game states. Still, there are four different levels, or views, or aspects, that can be applied to the components that together enable "playing a game", and they are boundary components, temporal components, holistic components and structural components.

Holistic components

These are the overarching components and concepts that cover the entire activity of playing a game. This includes conceptions like a *game instance* (the whole process of playing the game, from setup to end), *game session* (one player's activities during the game session; if you and I play a game of chess, we share the game instance, but have one session each) and a *play session* (used if players play one game instance in several sessions, i.e. one session of a certain computer game can be played in many play sessions). Other concepts are *extra-game activities* that are not a part of the game instance per se, but still related to it, like putting one's movies created in the tycoon computer game *The Movies*[51] online, collecting and displaying one's Magic cards[52] in special folders or carefully painting one's miniature army before a tabletop battle.

All of these notions are interesting to consider whenever studying games as an activity in relation to other activities, or when studying the social aspects of a game.

Examples of patterns concerned with this include: *Real-Time Games, Single-Player Games, Multiplayer Games, Time-Based games, Turn-Based Games, Early Elimination, Immersion, Character Development, Meta Games, Replayability.*

Boundary components

These set the limits for what can be done in the game; they define the game world in a sense. Typical boundary components are thus the *rules* (i.e. how to play the game) *goals* (how to win the game) and *sub goals* (smaller goals making it possible to achieve the winning goals) and *modes of play* (e.g. going through various phases in a game; from bidding phase to building phase for instance).

Looking at boundary components is useful when trying to define a game, both in terms of activities and goals wanted as well as unwanted.

[51] *The Movies*, published by Lionhead Studios 2005.

[52] *Magic: The Gathering*, a collectible card game by Richard Garfield 1993, published by Wizards of the Coast

Examples of patterns concerned with this include: *Symmetric Goals, Asymmetric Goals, Rescue, Capture, Bluffing, Alliances, Rewards, Penalties, Collection*.

Temporal components

These are used to describe the flow of the game, and therefore deal with volatile and accidental occurrences like *actions* (i.e. what players do), *events* (something suddenly happens, instantiated by the game rules/the game itself or by a special player action), *closures* (e.g. completing something, often reaching a goal or sub goal like capturing one of the opponents pieces in Chess), *end conditions* (who regulate changes of mode or perhaps the end of the game itself) and *evaluation functions* (e.g. scoring).

It is appropriate to investigate these components whenever one is analyzing and improving the flow and interaction forms in the game.

Examples of patterns concerned with this include: *Easter Eggs, Surprises, Player Killing, Betrayal, Cooperation, Movement, Interruptible Actions, Ultra-Powerful Events, Transfer of Control, Time-Limit*.

Structural components

These are the physical and virtual components of a game, as well as abstract phenomena that represent attributes or values. The most important components are *the player(s)* (both humans and artificial opponents), the *interface* used for play (i.e. a graphic user interface for the computer game, or a board game board, or a game master running a tabletop role playing game etc.), the *game elements* (avatars, pieces, dice) etc.

Looking into these is relevant when fine-tuning a game or setting up requirements for which components are needed and how to interact with them.

Examples of patterns concerned with this include: *Game world, Enemies, Obstacles, Avatars, Tools, Score, Lives, Strategic Locations, Inaccessible Areas, Lives, High Score List, Levels*.

Game Design Patterns to the core

At present, the game design pattern collection consists of ca 300 patterns. They are described in a rather formal and structured way, having information about each pattern described within the same structure, with clear headlines. A pattern description consists of

- The name, chosen to be as descriptive or at least idiomatic as possible
- A core definition

- A general description, often with examples from different types of games
- How to use the pattern, i.e. which design choices that have to be made
- The consequences of using the pattern
- The relations to other patterns (in terms of which other patterns they instantiate, are instantiated by, modulate, are modulated by and conflict with)
- References (if any)

Since the relations between patterns are so important, references to other patterns are not only mentioned in the relations-part, but throughout the text.

Below follows the description of a pattern, courtesy of Björk & Holopainen (2005).

Competition

Competition is the struggle between players or against the game system to achieve a certain goal where the performance of the players can be measured at least relatively.

Competition can take many forms, with the primary dichotomy being between having to actively engage against other players to win, direct *Competition*, or being able to win without interacting directly with other players, indirect *Competition*. The first case is the most common and is usually aggressive and destructive (e. g., *Chess*) but is also often perceived as the most emotionally engaging. The second can more easily allow for slow-paced games and constructive gameplay. It also puts more emphasis on competing against oneself.

Example: Many games based on race have indirect *Competition* between the players to reach a certain position in the game as fast as possible. The performance of the players is measured by timing each player's race.

Using the pattern

The easiest form of *Competition* is ***Conflict*** with ***Enemies***, but any situation where players have ***Incompatible Goals***, ***Excluding Goals*** (possibly

through ***Tiebreakers***, or ***Rewards*** (especially ***Individual Rewards***), can cause *Competition*. Two forms of *Competition* that require ***Conflict*** are ***Overcome*** and ***King of the Hill***. Examples of *Competition* without ***Conflict*** are all forms of ***Races*** without ***Interferable Goals*** or ***Last Man Standing*** goals where the players are not the cause of each others' demise. In these types of *Competition* the players are not each others ***Enemies***, but the game may provide other ***Enemies*** through ***Agents***.

Using ***Mutual Goals*** with ***Shared Rewards*** in subgoals of the *Competition* reduces the level of *Competition* between the players, as the players can have ***Cooperation*** with other players in ***Alliances***. By encouraging ***Dynamic Alliances***, the dynamics of *Competition* and ***Cooperation*** usually increase the level of ***Social Interaction*** between the players.

How the *Competition* ends depends on what criteria the different competitors have. ***Symmetric Goals*** allow players to judge more easily their chance of winning and are often used to create ***Player Balance*** between players. ***Asymmetric Goals*** allow different strategies and can create a more varied gameplay and may also promote temporary alliances, although it may be more difficult to balance the game. Further, players may have ***Unknown Goals***, which allows for techniques of masquerading one's intentions.

If one player is certain to win a *Competition*, the motivation for other players to continue competing becomes pointless. In the case of sub goals, this can be a temporary setback, which can be offset by winning other ***Competitions***, but if the outcome of the overall game becomes apparent, the motivation for continuing to play the game may become pointless.

The final outcome of *Competition* varies as well: the winner may gain ***Rewards*** in the form of ***Resources***, information, ***Improved Abilities***, or ***Social Status***; the loser may similarly lose ***Resources***, suffer ***Ability Losses***, or be excluded from the game through ***Player Elimination***. Exclusion from the game near the end of the game is usually not a problem, but early exclusion may be. Specific examples of more complex forms of *Competition* for sub goals within games include ***Bidding*** and ***Trading***.

The use of ***Agents*** allows players to compete without having to risk the social consequences of losing a game to other players. On the other hand, this means loss of opportunities to gain ***Social Status*** by winning the game. The use of ***Ghosts*** has similar effects on *Competition* but may still give ***Social Status***, as the results between different players can be compared.

Consequences

Having *Competition* in a game gives a sense of purpose to playing the game and often creates ***Tension*** during gameplay. If players have chosen to play a competitive game, they have chosen to test their abilities against other players, a computer, or a puzzle. *Competition* can also motivate ***Social Interaction*** in general, but especially in games where there are dynamics of *Competition* and ***Cooperation*** between the players, as is the case in ***Social Dilemmas*** and rivalries in ***Social Organizations***.

As *Competition* makes players want to use their ***Resources*** and abilities as efficiently as possible, the presence of *Competition* discourages ***Experimenting***.

Relations

Instantiates: ***Social Statuses, Conflict, Tension***

Modulates: ***Social Statuses, Social Interaction, Alliances, Dynamic Alliances***

Instantiated by: ***Shared Resources, Bidding, Incompatible Goals, Excluding Goals, Trading, Last Man Standing, Overcome, Race, Ghosts, Enemies, King of the Hill, Red Queen Dilemmas, Rewards***

Modulated by: ***Collaborative Actions, Mutual Goals, Shared Rewards, Symmetric Goals, Individual Rewards, Tiebreakers, Asymmetric Goals, Unknown Goals, Alliances, Social Dilemmas, Social Organizations, Agents, Cooperation, Player Balance***

Potentially conflicting with: ***Experimenting***

Using Game Design Patterns

Apart from providing a common language when discussing games, Game Design Patterns and their framework can be used for analysis of games, both when dissecting someone else's game or playtesting one's own design. The latter can be very important when looking for flaws and ways to fix them. Apart from this Game Design Patterns can also be used to generate new game ideas and to develop and improve these.

In this, it is important to note that even if most games can be described as a rather complex web of patterns, the core of the game, defining it, consists of maybe ten to twenty dominant patterns; normally it is sufficient only to deal with those.

Game Design Patterns as a tool for analysis

When researching a game, trying to get a theoretical understanding of it, one can make a structural analysis of its Game Design Patterns. This can be simplified if considering the holistic, boundary, temporal and structural components respectively, adding them all up. Actually this can be done by only reading the rules or a through description of the game, but playtesting can help finding patterns that appear during gameplay, i.e. aspects of social interaction, and the impact of certain patterns in relation to others, which can be hard to find or detect solely from the rules.

Since most professionals (regardless of field) use playtesting as a tool for improvement anyhow, they may benefit from expressing the game in terms of patterns since this is a way to become aware of how the patterns interact, i.e. if some are obsolete, missing or too weak or strong. This can be very useful, since recommendations on how to interconnect and balance patterns can be found in the collection. However playing the game with the sole aim to analyze it often mitigates certain aspects of it, such as immersion, flow and player-player interaction, and thus it can be suitable not to analyze during the session but instead record the session and/or have non-playing observers and discuss the game afterwards.

Game Design Patterns as a tool for idea generation

Game Design Patterns can be used as an idea generating tool in both unstructured and structured ways. In the first case, some Game Design Patterns are picked more or less randomly and used as starting points in a brainstorming session. If the idea generation is slow, one can dig into the patterns in order to find connections between them. I.e. *Tension* and *Avatars* are not directly connected, can be but if one adds the possibility of *Player Killing,* spawning ideas for a hide n' shoot game. If we instead connect the patterns with *Uncommitted Alliances* we might get a non-violent, *Negotiation* influenced game.

In the second case the wished game must perhaps fulfill some special criteria, i.e. it perhaps has to be partly cooperative, online and it has to be about drinking different kinds of soda (since the customer is a brewery). In this case, the demands set the initial starting patterns; since there are different kinds of sodas, *Collection*

might be a suitable goal or sub goal, or a kind of reward. In the latter case, in order to modulate the *Cooperation*, it could be suitable to use *Shared Resources* in order to give the players an opportunity to *Negotiate* who gets which resource with possible *Conflicts* or *Alliances* as a consequence.

Game Design Patterns as a tool for initial development

After having put together a first version of the rules, a list or a rudimentary graph of patterns can be drawn, showing the main patterns and how they influence each other. As development continues this list or graph may become more fleshed out, since the main patterns are instantiated using sub patterns (e.g. *Cooperation* is achieved using *Shared Rewards* and *Shared Resources*). In early phases of development patterns may be very useful for balancing and fleshing out a game.

Example: Hypothetically fleshing out the MineSweeper

Here, I will use the example of a very familiar game, *MineSweeper* to show how a pattern collection for a game can be put together and fleshed out. Of course it is very unlikely that *MineSweeper* was actually created using Game Design Patterns, or even along the trains of thoughts I describe below. Still, I am using it in this example since the game is familiar to most people and it will thus serve well as an example. Also, it is a very simple game, easy to express and clarify in terms of Game Design Patterns, which will also clarify the explanation. Note that I have chosen to denote the patterns not as a list, but as a graph.

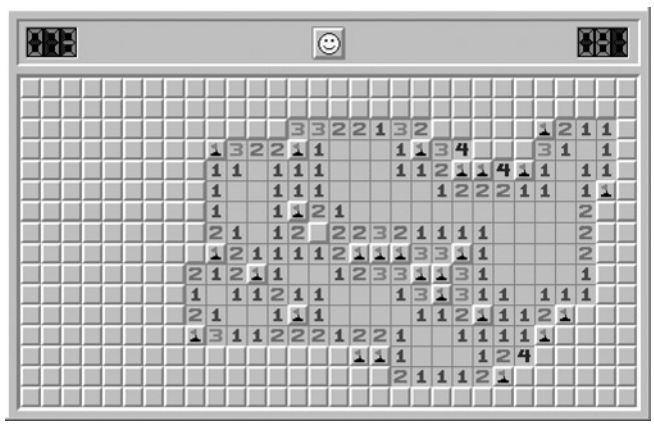


Figure 16: Minesweeper

In *MineSweeper (*See Figure 15), the sole object is to find out where on the board the mines are and mark them. The number in each square indicates how many mined squares it has next to it. The number to the top left corner of the game area (in this case "79") indicates the number of mines left on the field whereas the number to the top right corner ("81") is the time played in seconds. One can place flags on squares where one knows that there's a mine, or just click on a square one knows is mine-free in order to expose its number. The goal is to find the exact locations of all mines in the field by having cleared all other squares (i.e. placing the flags isn't really necessary but a useful help).

The most dominant pattern in *MineSweeper* is *Exploration*, and ways to encourage *Exploration* would be to give *Rewards* for exploring as well as adding a *Goal* as a further incitement for doing this. This would result in a very calm game, perhaps cute but not very interesting in the long run. *Tension* is lacking, and since there are already *Rewards* for exploration, why not add *Penalties* as well? This gives us a very first pattern graph looking like in Figure 17. We can see that the link between Exploration and Rewards is very strong; they support each other. If you explore you get rewards, and getting rewards is good, so you want to explore.

Naturally, this graph isn't very elaborate. For instance, one must decide what kind of penalty there should be, and perhaps moderate the risk of getting it. Here we can imagine a series of playtests elaborating with different rewards (e.g. *Resources* that perhaps need to be organized in a *Collection*, the goal of the game). If the rewards are points instead, the penalty could perhaps be the loss of such points etc. Regardless of these decisions, there is a problem if the rewards and penalties are randomly distributed. The game becomes a game of luck, or of calculated risk at best, something a playtest most likely would show. If this is not wished, this effect can be countered by for instance adding Information to the game. Again, there are design decisions to make; how much information? In MineSweeper, the information about the current game situation is rather satisfactory, and the information one has is always valid – there is never a "3" in a box adjacent to anything else that 3 mines. Still, situations can occur when it is impossible to deduct where a mine is, especially in the beginning of a game, and in corners. Thus, the pattern used is *Imperfect Information*. Having information about the terrain that is to be explored makes explorations safer and reduces the tension, but the latter effect is somewhat countered by the fact that the information is imperfect. Again, the game can run a risk of being boring. This can be countered by increasing the penalty for making a wrong choice… perhaps even Player Elimination? This gives us a more alborate structure, as in Figure 18.

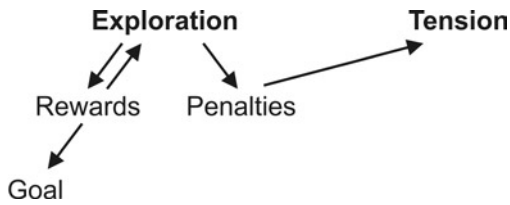


Figure 17: The first version of a pattern map for Minesweeper.

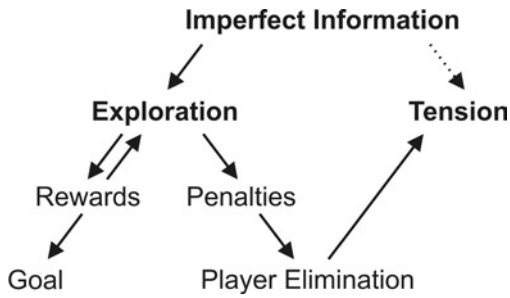


Figure 18: The second version of a pattern map for Minesweeper.

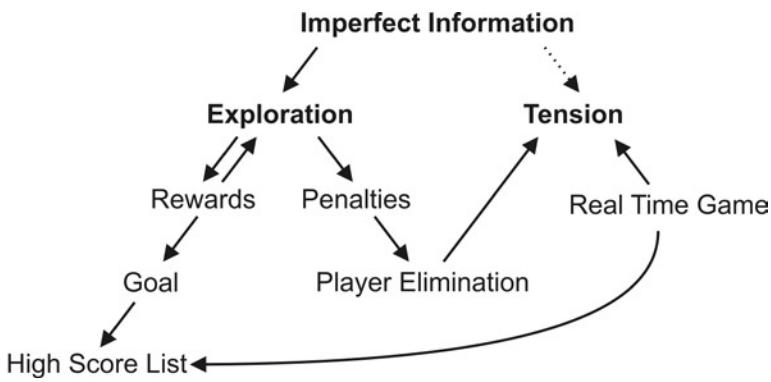


Figure 19: The third version of a pattern map for Minesweeper.

Now we seem to have a working map of patterns, but currently, the game is very much a mind game, running the risk of being very slow and characterized of careful analysis before each player move. If this is not wanted, the game must be speeded up somehow (for instance by adding suddenly appearing *Enemies*). And still, it is unclear what the rewards actually are. If the penalty is as heavy as being eliminated from the game, but quite easy to avoid as long as one has perfect information, then the reward should perhaps just be to find out that the field chosen is safe, thus making it the goal of the game to find and avoid all the mines. In this clearing one random mine field is just as good as clearing any other. Even if this is satisfactory in a sense, it reduces replayability, since the player cannot compare sessions or measure progress. Luckily we can solve both this and the slowness problem in adding only two patterns, making the game a *Real Time Game*, and measuring success in seconds that are saved in a *High Score List*. Then the "final" pattern web for MineSweeper would look somewhat like in Figure 19.

And so on. Of course the graph cannot be elaborated in absurdum, and can be very hard to overlook rather early in the process if the game that is to be developed is very complex, but nevertheless it can be a good tool when trying to keep track of and visualize the design. It also clarifies the relationships between patterns; if we take out the *Real-Time Game* pattern, this affects the *Goal*, which is currently the *High Score List*, and so on. Currently there is no real description; it may be up to the individual development team to choose how to denote patterns and relations of varying importance.

Game Design Patterns as a reparation tool

In the cases when a game has been partly developed with Game Design Patterns used in a more unconscious manner it can be useful to draw up a graph or make a list of them, and try to analyze where something has gone wrong.

Game Design Patterns as a tool for communication

In this, the Game Design Patterns serve the same purpose as do game mechanics: they express a rather elaborate and complex notion in one word. The game developers needn't say "*we need to have a set of actions where each action has an advantage over the next but where the last has an advantage over the first in a circle-kind-of-way*"

because saying "*We need a paper-rock-scissors pattern here*" is enough. This facilitates conversation and also gives a common ground of understanding.

Game Design Patterns as a teaching tool

Game Design Patterns suffer from the same problem as does any language; one has to learn it before one can use it. Thus, I believe that the possible introduction of the Game Design Patterns into the professional game design sphere will come from teaching new game designers how to use them. One reason for this is of course that the framework establishes the design space in a rather concrete way. Another reason is that when learning how to design something – anything – one typically starts out by looking at, and analyzing, existing designs in order to try and understand what makes them work. In this, a structured vocabulary can help very much, and as the Game Design Patterns also point to relations between patterns this can simplify the analysis of a game.

Using Game Design Patterns it would be very easy to set up a series of design exercises on game design, starting out with analysis of existing games, continuing with analyzing and fixing not-so-good or incomplete games, followed by the creation of smaller games, starting with a few set and mandatory patterns – and perhaps analyze and improvement of each other's ideas before designing a larger game. All of this would take place within the framework, so to speak, using patterns and pattern graphs to express connections. The possibilities to use Game Design Patterns as an inspiration for teaching and learning are abundant, and this will most likely strengthen Game Design Patterns as a language, but not necessarily as a tool; as mentioned game designers develop their own sense for the relations between patterns. Time will tell.

Case: A Workshop on Game Design Patterns

Over the years several workshops on Game Design Patterns have been held, both prior to writing the entire collection in order to test the concept and perhaps collect new partners, angles and views, and after, in order to study the use of them. In this case, I describe a one-day workshop where the participants had access to the full collection of patterns as hyperlinked HTML-pages.

Workshop setup

The workshop was a one-day workshop with eight participants in their thirties. Professionally, most of the participants came from some kind of natural sciences background, and several of them were programmers. All of them had a background in gaming; both board gamers, computer gamers, role playing gamers, war gamers and live action roleplayers were represented and many of the participants had experience of several types of games. Several of them had arranged larger gaming activities, such as role playing tournaments, game conventions, live action roleplaying events, board gaming tournaments etc. Two of the participants had published commercial games (computer games and board games respectively) and yet another four had designed games as a part of a course or as hobby projects, however mostly stopping at some level of prototype. About half of the participants had used patterns for software design previously and two of them were aware of Game Design Patterns; one via a lecture by Jussi Holopainen, one by reading Björk's & Holopainen's book (2005).

One hour was spent introducing the participants and describing the pattern concept in general and the game design pattern language in particular, as well as describing some computer augmented games, like myTHeme (see facet No. 5) and the games described in Facet No. 2, in order to widen the participant's views on novel game forms.

Two hours were spent on analyzing a simple game in terms of patterns, trying to tweak it, and after lunch three hours were spent on coming up with new game ideas with some patterns as inspiration/boundary. The day was enclosed with a discussion. The various groups were assigned by myself, trying to get an appropriate blend of professional designers and people of different gaming backgrounds.

Analyzing games

The two games analyzed were *Exxtra*[53] and *Lift Off*[54]. *Exxtra* is a dice game where players roll two special dice[55] in order to "bid" with the outcome of their rolls in order to win points – thus a higher roll is better. Players may re-roll as many times they like, but if they roll one or two X-es the turn is lost and they lose one or two

[53] *Exxtra* by Reiner Knizia 1998, published by Amigo Spiele 1998.

[54] *Lift Off* by Marcel-André Casasola Merkle 2000, published by Queen.

[55] One dice is numbered X, 1, 2, 3, 4, 7, the other X, 1, 2, 3, 5, 6.

points. A double 1, 2 or 3 results in 1, 2 or 3 points respectively. The first player to reach 21 points wins. The points are counted as steps on a track.

The group who analyzed *Exxtra* played a quick round of the game after which they browsed the entire pattern collection in order to find used patterns. They found 24, but then omitted 6, since they thought that some of the other patterns they had found covered them too, e.g. they omitted *Race* after some analysis, since the game is more about reaching a *Score* than actually racing along a track; the track and the goal point of it aren't actually needed to play the game, one might as well just write down the points. In the *Exxtra* case, the track, and the tokens on the track are just ways to represent the score. After about 90 minutes a pattern graph containing the ten most prominent patterns had been drawn (after some discussion on how to draw and point arrows), and the group set out to improve the game by changing some rules, one being that rolling a double and choosing to take these points ended the turn too, the second being that points could be paid to raise a roll, thus strengthening one's bid. In doing this, they made deliberate tries to change the game, not by adding or removing patterns but by trying to change the influence of existing patterns, e.g.. the new rules moderated *Limited Planning Ability, Luck* and *Randomness* since players gained more control over their rolls.

Lift Off[56] is a fast-paced *Real Time Game* where players rush to terraform and/or mine planets. On each planet a player should try to play Freight cards (like Energy Tank, Settlers etc.), a Lift Off card (starting a mission) and executing cards (e.g. Ore Production, Terraforming, Theft). Each player has access to five planets, however competing about each with one other player. The game is played on time; each player has the same deck of cards and turns them up one by one, playing, keeping or discarding them. Whenever a player has run out of cards the game ends and the points are scored. The time-pressure makes the game very fast-paced.

This group too, started browsing the collection, however in a bit more structured way since one of the group members was the one who had read the book already. Also, they divided in pairs doing this. Still, it took them longer time to complete their graph and they only briefly discussed how to improve the game; one was for instance to turn the game into a computer game to limit the importance of being nimble-fingered.

[56] *Lift Off* by Marcel-André Casasola Merkle 2000, published by Queen.

Designing games

The two groups were now turned into three, and they were given the task to design a game which fulfilled the following demands

1. It should use the pattern *Real Time Game*
2. It should use computer augmentation of some sort
3. It should fulfill *at least one* of the following:
   - All players should cooperate towards a mutual goal (possible patterns are *Cooperation*, *Communication Channels* and *Mutual Goals)*.

   or
   - It should be possible to spy on other players without them knowing it (possible patterns are *Gain Information*, *Imperfect Information*, *Uncertainty of Information*, *Asymmetric Information* and *Red Herrings)*.

   or
   - It should be an outdoor game that uses the local architecture and environment as component in the game (possible patterns are *Pick-Ups*, *Inaccessible Areas* and *Strategic Locations*).

   or
   - The game is played in teams, but the team members are at different locations (possible patterns are *Team Play, Communication Channels* and *Team Balance)*.

One group came up with an idea almost immediately and spent a good deal of time discussing it and designing it. Although using terms which are clearly patterns (i.e. *Camping*, *Boss Monster*, *Safe Havens*), they didn't really discuss the game in terms of patterns, and they did not use the pattern collection when designing. Another team got stuck with discussing one game idea for fairly long but being unable to make anything out of it they abandoned it and toyed around with a set of other ideas in a rather unstructured way. They too, didn't use the pattern collection. The third group spent a rather long time on discussing which real time (board) games they knew and liked and a general vision of what they wished to achieve. After this they came up with an idea that different actions in the game should take different amounts of time, and that only one action could be made at a time, i.e. no simultaneous actions were allowed. Thus if the game time was 30 minutes the player could either make 30 one-minute actions or two 15-minute actions or any other combination of available actions that summed up to 30 minutes. After a while they found a suitable theme and started elaborating it. This group used patterns more actively, e.g. specifically

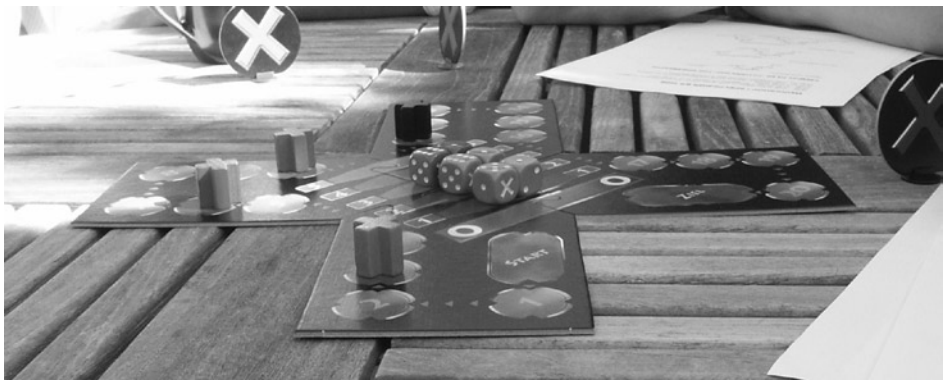


Figure 20: Playing Exxtra.

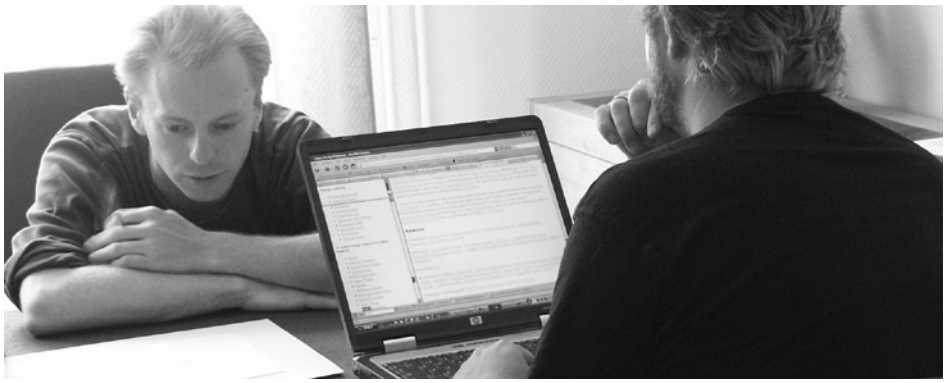


Figure 21: Pondering over the pattern map for LiftOff.

Figure 22: Prototyping MonstroCity.

aiming for *Balancing Effect*s. They also expressed their game in terms of patterns when explaining it. Hardly surprising one of the two members in this group was the one who was already familiar with the collection.

The groups had access to a lot of prototyping material and despite this and the fact that they were actively encouraged to make quick low fidelity prototypes and playtest their concept, no groups did, not even the first group which had plenty of time. There were only scenario-based discussions, i.e. mental playtesting. Perhaps this was due to lack of time, or because none of the participants was used to working extensively with prototypes; perhaps it takes a special kind of personality combined with some training to do this early on in a process.

Concluding the day

In a one-hour discussion the game ideas were presented. In the two cases where the groups had not expressed the games in terms of patterns they were asked if they could name some now on top of their head, and they actually did mention some. To conclude, there was a free discussion about Game Design Patterns as a tool.

Polar Expedition Race

This game idea was created by Ola Janson and Magnus Lundgren. In a Jules Verne-like setting gentleman participate in a polar race in their vessels, which are combined submarines and zeppelins. The race takes place in several legs, and after each leg there is a tea-time session where players can trade items and information. Each leg has a set time, and during it players move in "jumps", flying high in the sky to accumulate enough speed to intrepidly plunge into the water in order to pick up interesting items (leftovers from previous races) at the bottom of the sea. Each movement costs time, which is calculated on how long and how high jumps the vessel makes before diving into the water. Vessels are only visible when in the air, but a long-lasting splash will show where a vessel has dived, information which can be used to block someone from coming up again. The vessels only have windows at the front; only the last player can see all other currently visible vessels. In consequence, the leading player cannot see any other vessels unless the player stops and makes a 180-degree turn, which of course consumes time. The game is played on mobile phones and depending on the activities of the day, players can plan their moves accordingly, i.e. initiating a very time-consuming jump before going into a meeting.

When developing the concept, the players actively used the patterns *Time Limit, Turn based* (tea-time sessions), *Movement Limitations, Inaccessible Areas* (the vessels cannot dive into water everywhere; there are ice floes), *Trading, race, Budgeted Action Points, Imperfect Information* and *Balancing Effects* (only the last player can see the others).

Photonary

This game idea was developed by Johan Eriksson-Lassbo, Kristina Knaving and Ekaterina Rosén. The game is played in teams with mobile phones equipped with cameras. Half of the team is out on the town somewhere, the other is indoors. The outdoor team gets a word sent to their mobile phone(s) and then takes pictures illustrating the word. They send the pictures to a web page which the other team members watch in order to guess the correct word as fast as possible.

When asked the group could come up with the following used patterns: *Team Play, Hidden Information, Social Interaction, Communication Channel,* and *Score*.

MonstroCity

This game idea was developed by Anders Mårtensson, Anders Qvist and Leif Ryd. This too, is a mobile phone game, using a city as a playground. The players' aim is to pick up treasures, but unfortunately the treasures are guarded by monsters and in addition there are other monsters that live within certain enclosed areas (e.g. at places the players might wish to pass more or less often) as well as a terrible, free roaming monster.

Monsters come with different strengths and can be eliminated or scared off if players form more or less temporary teams by walking together. On the screen of their mobile phones, players can see a map of where they currently are, and if there are any other players, monsters or treasures in their vicinity. Also, approaching monsters make the phone beep; the more frequently, the closer the monster. Monsters are especially interested in players who move fast (i.e. opting to get to a treasure first by taking the bus to it). There are also some kind of safe havens in the game. The game is played in sessions set up by players, and the location of the treasures and monster areas differ between sessions.

When asked the group could come up with the following used patterns: *Cooperation*, *Boss Monsters*, *Extra game activities* (e.g. wearing a special cap or making something else to indicate to other players that you are a player too), *Obstacle*, *Safe Havens*.

Concluding the workshop

When discussing the outcome of the workshop it is important to notice three things about the participants; firstly, they were *gamers interested in game design*, rather than experienced game designers, although two of them had published games. Secondly, only one of the participants had read Holopainen's and Björk's book on Game Design Patterns (2005) and was familiar with the collection prior to the workshop, and he – and the groups he was in – thus worked significantly more active with the collection. Thirdly, none of the participants was a trained designer in terms of knowing how to use design methods on idea generation (apart from the ubiquitous brainstorming) or how to quickly prototype or test an idea. Thus their behavior and their opinions come from a beginner's and learner's point of view, rather than from the experienced game designers point of view.

Observations that I made during the day was that even if the names are chosen to be as descriptive as possible, there was still a lot of confusion and the patterns had to be checked out more thoroughly to see what they meant. For example, one group spent a lot of time discussing the difference between *Luck* and *Randomness*, and another mistook *Ghosts* for being, well, ghosts, instead of overlays from a previous session (i.e. driving a second lap in a race and seeing a "ghost car" showing how you drove in the last lap). Also the whole idea with patterns are that they should be as general as possible (in order to decrease the number) but then again this can cause problems; beginners may spend a significant time looking for something that they believe is a pattern in its own right whereas is it actually one of many variants of a large pattern, i.e. one group spend some time looking for *Track* before realizing that the track was only one means to show the *Score*.

Another observation was that even if the participant used patterns and pattern language to the best of their abilities during the first exercise, general brainstorming and other strategies to come up with ideas took over as soon as they were to design their own game; only the participant already familiar with patterns used them, but perhaps more as a tool for discussion than for design. This implies that beginners must be given significant time to browse and get familiar with the pattern collection before using it actively as a design tool.

The latter fits well with the participants own opinion on patterns; they all agreed that Game Design Patterns could be used for analysis and discussion (e.g. one participant said that the game analysis made him aware of a problem he would not have detected otherwise) but they did not agree on whether it could be used as a

design tool. Someone pointed out that it could be useful to pick x random patterns having to use them, just to break boundaries and familiar trains of thought, but although this could work as inspiration it seemed to many (and to me, both as observer and game designer) that the first cycles of brainstorming and discussion work just as well without involving patterns. As someone put it: "A good tool […] but it doesn't save the world".

Another interesting comment that perhaps can be of value is that Game Design Patterns could and should be used in game reviews in order to make them clearer (this requires that an explanation of the patterns is available online somewhere), shorter, more standardized (and thus more comparable) and hopefully more objective.

Overall the participants seemed to be pleased with the day and all of them claimed to have, or have gotten a generally positive impression of the patterns as a tool.

Reflections

Obviously, patterns can be used to develop any kind of game, although a quick browse through the collection suggests that most of them come from, and are used in, board games and computer games. Nevertheless the ability to combine patterns – whether they require computational power or not – will stimulate the development of interactive entertainment artifacts like hybrid games, computer augmented items mimicking magic in fantasy Live Action Role Playing, and computer augmented board games.

Issues and Problems

The main "problem", or perhaps disadvantage is a better word, with the game design pattern language is that it shares the same disadvantage as any other language; it takes time to learn and if you don't want to risk misunderstandings you can only talk it with those who know it too. As a tool it is like any other complex tool; a lot of practice is needed before you can make full use of it. Altogether this means that if one really wants to use Game Design Patterns within a game design group, everyone must spend time and energy in learning the vocabulary (in order to discuss and analyze) and in using the patterns as a web of dependencies. If the group is or gets apt at this they can be a great tool.

Arguably, learning the pattern names and understanding pattern consequences may not be very tedious for experienced game designers since many names are self-

exploratory (e.g. *Team Play*) and describe concepts that are rather familiar for the game designer. In addition he or she normally has a natural, hard-earned feeling for how patterns are related to each other. This is of course an advantage when it comes to using the language, but might mean that they are less prone to use the patterns as design tools; they have already developed other design strategies, or simply rely on their tacit knowledge of how the design process is run and the design space the game resides within. Still, providing common names for the concepts (patterns) they already know, will facilitate conversation.

It seems that only parts of the game design community's has been met; the common language, but perhaps not the game design tool. Then again, it is very hard to develop one common design method since every design project is unique, and the game design sphere covers such a wide range of games – computer games, board games, card games, role playing games, tabletop miniature games, live action role playing, computer augmented board games, pervasive games played on cell phones or PDAs, etc, and all of them can have varying complexity. It is obvious that it is not an easy task to merge all of these possible design procedures into one; a parallel example would be software development which features dozens and dozens of more or less formal methods and tools for development plus, of course, the fact that each larger company have their own modified versions.

Reflections and lessons learnt

The most prominent reflection we have made since starting the development of the Game Design Patterns are that the name is actually misleading. Creating a user interface, or programming rendering algorithms for a simulation game is just as much game design as creating the rules for the game, which is the main focus of the Game Design Patterns. Thus, a more correct name would be gameplay design patterns; perhaps we will manage to slowly shift the name, despite the book title "Patterns in Game Design" (Björk & Holopainen 2005).

Above, it is argued that most games can be expressed in its 10 – 20 most influential patterns. This quick graph or list of patterns will do well for an analysis of the game, or for early development, but using it for finding a critical fault or imbalance may be harder, especially if the game is complex and it is a minor pattern or a forgotten pattern that causes the breakdown. Sometimes also the collection might look alright

and the whole breakdown is simply caused by a balance problem; some rewards somewhere are not balancing the penalties elsewhere etc.

Note that the approach of denoting the patterns as a graph (as opposed to a list) is not explicitly mentioned in the book but trying to arrange the patterns in relation to each other can clarify connections. Also, the whole pattern collection in itself is a graph, with three different types of connections (modulates/is modulated by, instantiates/is instantiated by and conflicts, respectively). However it may be up to the designer and the context whether to try and depict these special connections (and their direction) when making a game-specific pattern graph. This means that a pattern graph needs to be drawn with some care, to avoid misunderstanding and over- or underemphasizing the importance of a certain pattern. Thus drawing a pattern graph is not as easy as it may sound, firstly, it might not be clear which pattern actually affects which and how much, and second the graphs are not two-dimensional; when drawing them one must also try to denote the importance/influence of each pattern, both in relation to the game in general and in relation to the other patterns. In turn, these effects of regulations and modulations may be so vast that they are difficult to foresee theoretically; again playtesting and trial-and-error changes of the system are needed in order to balance the game. In the development of complex games this is a tacit craft, hard to formalize or express – neither in a pattern language nor in any other language.

Interestingly enough connections between two patterns in a graph may not be mentioned in the collection. This depends partly on that only the most important or common connections are mentioned in the collection, partly on the fact that some intermediate patterns may have been excluded in the graph, and partly because due to the myriad of games possible almost any pattern can be closely tied to another if the gameplay design is the right one.

Another possible improvement already used in the software pattern language (Gamma et al 1994) is to have alternative names for each pattern, or to have names of versions of the patterns pointing to the pattern. Arguably, this could be seen as confusing as well.

During the workshop, one group looked desperately and unsuccessfully for trick taking until they decided that from a broader point of view, trick taking can be seen as a form of *Bidding* where the resource used to make the bid is lost. Although it is good to have as wide definitions of a pattern as possible (in order to minimize the collection) it would be appropriate to provide a supplementary reference list

containing other possible names of a pattern, or variants of patterns. It could read for instance "Trick taking – a form of *Bidding*" or "Start point – a form of *Spawn Point*" (a problem another group ran into) and so forth.

Possibilities

When using patterns as a starting point for design (e.g. as a basis for a brainstorming session, I believe that no extensive knowledge on the Game Design Patterns is needed as long as participants either get a very clear explanation of each pattern, or already have a basic experience in games and game design. In the case of experienced designers, reading through the chosen patterns can be inspirational enough, or participants can simply just guess what the names mean to get a starting point for discussion. In order to find more inspiration or getting on with the process the collection can be browsed, jumping from pattern to pattern in search of more inspiration.

Several workshops on patterns have indicated that this is the case. Then again, there is an imminent possibility that the patterns are used solely as triggers for inspiration, rather than as the start of an exploration of the related patterns.

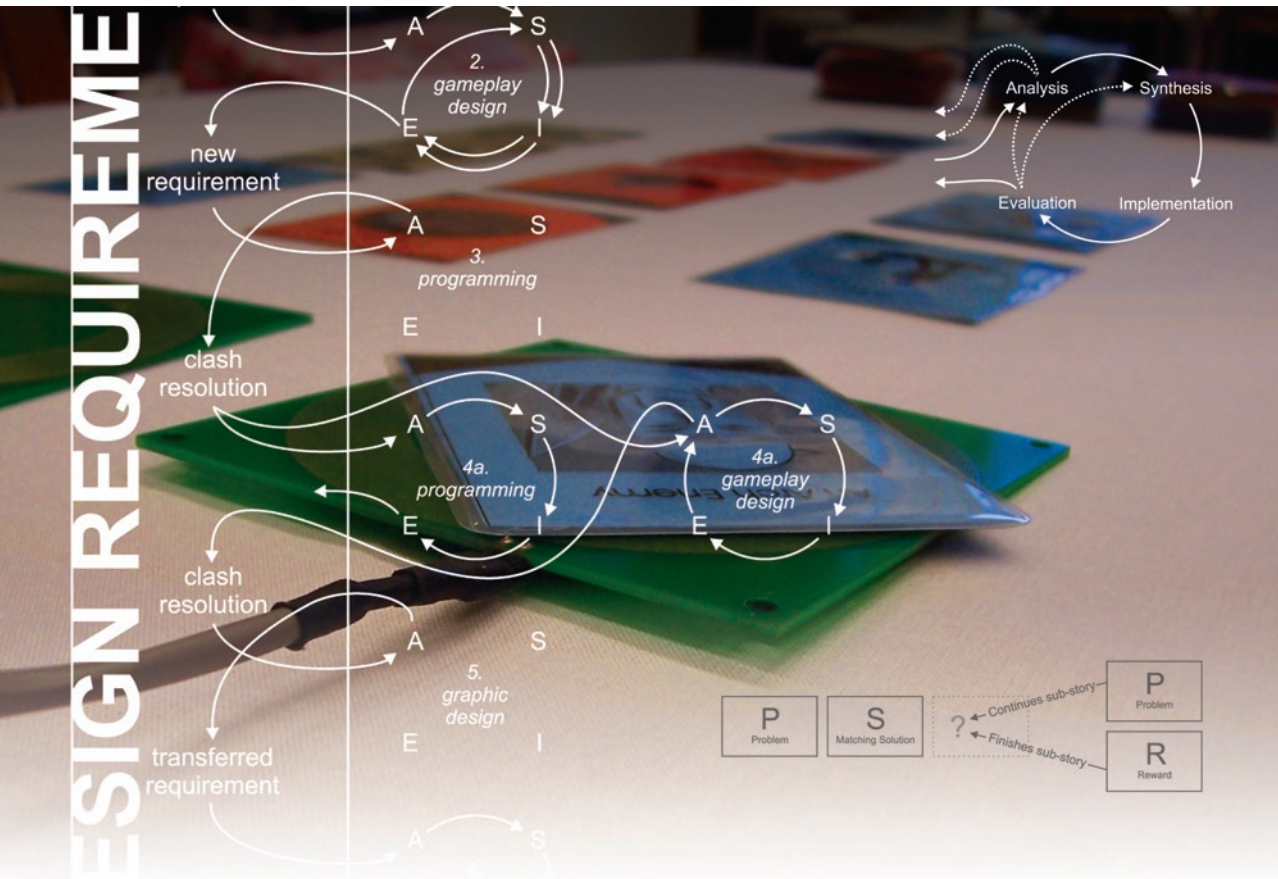
Note however, that this rudimentary use of patterns as a tool for idea generation does not give enough knowledge of the pattern language in order to use it to discuss or analyze games in terms of patterns. In order to do this effortlessly, (e.g. making lists or drawing graphs), one needs to be much more familiar with the pattern collection, at least if one wants to make it rather effortlessly. On the other hand, asking novices to do this forces them to explore the collection, making analysis a tool for getting acquainted with the pattern collection instead of vice versa.

Still, the possibility to quick-start game design with only a few patterns, provides a very pleasant introduction to the pattern collection. This also very strongly supports and suggests the notion that the true future of the Game Design Patterns lie in the use of them as an educational tool. As mentioned they have several strengths in this area; the most important one being that participants must and shall set aside time to actively learn and use the collection and that this learning and exploration process can be carefully planned. Also, the game design pattern language will be shared by the entire class, and members will learn from each others.

Another advantage with the pattern language is that it may be used in game reviews, both as a way to shorten them (rules may not have to be explained extensively in order to prove a point) but also to make them easier to compare and more standardized and perhaps even more objective. However this requires that the public has access to the patterns (or at least limited descriptions of them, perhaps only the descriptive part?), e.g. online.

FACET No. 5:

MyTHeme – in Terms of Forces, Clashes and Remnants



This is the story about the creation of a story. Or, more specifically how we designed a storytelling game for non-storytellers. The design process turned out to be rather complex and this lead to the development of some concepts that can be used in a multidisciplinary design project.

You are playtesting your computer augmented board game. It seems to work rather well, until you suddenly detect an inconsistency between the rules and the actual implementation. The rules state that it is possible to teleport a piece from one square to another non-adjacent. However if you do this, the board looses track of the identity of the piece, since it only has sensors to check the identity once: when the piece is added. Thereafter it just keeps track of its movement. This also means that two pieces cannot enter the same square since the board does not know which piece is which when they leave.

"Oops, we have a clash here" you say.

"What?" says Hannah, who has done a great job with the graphic design.

"Oh, the rules are conflicting with the hardware."

You explain the problem to Hannah and Carl, the hardware guru.

"Oh wait" says Hannah, "the crossing of tracks there on the board is a remnant, don't you remember we put it in to facilitate movement from here to there – and that isn't really needed anymore since we added teleportation too later."

"You are right! But what can we do about the teleportation problem, Carl?"

"Can't you just rewrite the rules?"

"I could, but if we take out both teleportation and the crossing the game becomes unplayable again… Also it is so in line with the initial objectives we had that it can be said to be a force."

"You are right. I'll have to figure something out. Hannah, that might mean that you have to redesign the board somewhat again, ok?"

"Ok. And don't forget to tell Steven that he most likely has some reprogramming coming up."

"Sure. I'll discuss possible solutions with him so that we don't clash."

The main focus of the myTHeme project was to explore the intersection between game design, ubiquitous computing and auto-generated narratives. The outcome was MyTHeme – a computer augmented card game, a storytelling game for non-

storytellers. It was a multidisciplinary project, covering aspects of graphic design, gameplay design, software design, hardware design and design of narrative structures, and as it happened it very clearly came to illustrate the kind of design issues that can arise when several disciplines cooperate, collide or coincide within a project and how to deal with this. Therefore the project has two outcomes; the expected one, i.e. the game, and an unexpected one, being new insights on multidisciplinary game design projects[57].

Design Challenge

Starting out, we wanted to explore computer augmentation of a game as well as how to generate narrative structures. It was conceived as part of a large research endeavor focusing on computer augmented games. Thus, the design challenge was to create a computer-augmented game evolving around storytelling. However we wanted the game to have the same kind of interaction as a traditional game, and – to make sure that the game idea was solid – it should be interesting and well-designed enough to be playable even without the storytelling-theme. Again, these were the known challenges. As it turned out, another challenge was to monitor the design as it moved between disciplines, and how to supervise and record contradicting demands coming from the involved disciplines. The first challenge led to the creation of the game, myTHeme. The second challenge led to some new insights and some new concepts usable in multidisciplinary design projects.

Background: Multidisciplinary Game Design and Iterative Design

Although all game design projects are multi-disciplinary by nature – combining the craft of gameplay design[58] with at least one other skill, e.g. graphic design for

[57] The *MyTHeme* project was run by myself and Staffan Björk when we were both employees at The Interactive Institute. Our colleague Jennica Falk also took a large part in the work, being responsible for most of the text writing. Other contributors were Jussi Holopainen, Nokia Research Center, and Karl-Petter Åkesson, Swedish Institute of Computer Science (SICS). The content of this chapter is based upon the paper Forces, Clashes and Remnants: a Model for Event-Driven Iterative Multidisciplinary Game Design (Lundgren and Björk 2005). However this is a much more elaborate version of the paper.

[58] Note that there is a difference between the terms "game design" – which refers to the entire process of designing a game – and "gameplay design" – which refers to the craft of putting together a game's rule system.

a pure card game – the design of a computer augmented game offers additional challenges.

First, there are often several disciplines involved; game design, hardware design, software design, interaction design, graphic design and/or product design and often an theme-specific discipline (e.g. some insight in history would be suitable for a game on ancient Greece, just as storytelling was an extra subject/discipline in myTHeme).

Second, due to the use of non-standard hardware platforms the designers often have to build the hardware more or less from scratch – a design process that will likely require novel solutions to rapidly-evolving specifications. Third, the interaction in computer augmented games differs from more well-established game formats, and highly affects and is affected of all of the other design disciplines. All of this means that in the design of a computer augmented game the interactions and interrelations between different disciplines are common, important and driving the design process.

Iterative design

An iterative design process is characterized by the process going through a number of cycles. What a cycle consists of can differ between approaches and disciplines, but typically starts out with a goal, followed by analysis, design and synthesis in various steps, ending with an evaluation (Jones, 1992 p 63 - 69). If the goal has not been reached, the same cycle is iterated once again.

Now, most of the design disciplines involved in game design have developed various methods and documented best practices. Despite the various differences required by the design material which is used, many of these find a common general structure in the notion of iterative design. Such a process has been advocated many times in game design (c.f. Cook 2002, Fulton 2002, Fullerton et al 2004 and Salen & Zimmerman 2004) and is regarded as the standard approach in user interface design (c.f. Preece et al 2002), and is indeed one of the central higher level general design methods (Jones, 1992). It is also a very common approach within software design, e.g. being an essential part of the Rational Unified Process, an iterative software development process used by many companies worldwide. RUP[59] is an adaptable process framework but regardless of how it is adapted, any RUP

[59] The RUP was created by the Rational Software Corporation, now a division of IBM.

process will undergo several iterations of the design. Since RUP is rather formal and structured, it is particularly applicable to larger projects with larger teams.

Agile software development

Within software design, the upcoming popular methodologies, like Extreme Programming (Beck 1999) are often agile. Agile software development (Highsmith 2002, Agile Software Development Manifesto[60] is targeted towards adapting to changing requirements, very short, feature-driven iterative cycles, using an incremental design approach, i.e. not specifying the design before the development starts or even before an iterative cycle starts, but in the process of designing.

> *"Projects may have a relatively clear mission, but the specific requirements can be volatile and evolving as customers and development teams alike explore the unknown. These projects, which I call high-exploration factor projects, do not succumb to rigorous, plan-driven methods. "*
>
> *- Jim Highsmith*
> *In: "What Is Agile Software Development?"*
> *(CrossTalk Oct. 2002)*

In short, the practices of agile software development are driven by a belief in continuous adaptation. According to Highsmith (2002), three overarching characteristics define agile development.

Firstly, it has a chaordic[61] perspective, meaning that one acknowledges the world is a mixture of chaos and order; while goals are achievable (and thus ordered in a sense) project details are often unpredictable, i.e. chaotic. Secondly, collaborative values and principles are superior to formalism and rigorously forcing people to adapt to a certain process. This means that there is a focus on individual (specialized) skills and a high degree of interaction both between team members and customers. Understanding typically comes from face-to-face-interaction rather than from documentation. Thirdly, the development should be steered by a barely sufficient methodology and structure. This is of course the most controversial point, but is being justified by supporting the chaordic view and general agility, in balancing flexibility with structure.

[60] The Manifesto can be found at: http://www.agilemanifesto.org/principles.html)

[61] Chaordic, a word created by combing "chaos" and "order", was coined by Dee Hock, founder and former CEO of Visa

Wanted: An iterative method for game design?

From the above we can conclude three things. Firstly, the design of a computer augmented game is by nature multidisciplinary, involving *at least* gameplay design, software design and interaction design *plus* a range of other design disciplines. Secondly, there is a lack of explicit methods, tools and approaches within the game design area[62],. Thirdly, many of the disciplines relevant when designing a computer augmented use an iterative approach to design and development.

Hence, it would be suitable to organize the work in an interdisciplinary game design project around a common, shared, iterative process, method or model. Unfortunately, there is not yet any well-documented and established method or model when it comes to game design that describes how different disciplines affect each other.

Topic: An Iterative Approach to Multidisciplinary Design

Again, one of the major outcomes of the myTHeme project was an iterative game design method (Lundgren & Björk 2005), covering most of the aspects mentioned above. In this event-driven model the interplay between disciplines is taken into consideration. It also provides a classification/terminology useful for keeping track of and discussing the design process.

The model evolves around five concepts, *the Design Requirements*, being the current version of the specs, individual *forces* and *requirements* being parts of the Design Requirements and *clashes* which are contradicting forces or requirements, and *remnants* – leftover designs whose incitement have been removed again

Below these concepts will be described more in detail. Thereafter, in the next part, the development of myTHeme will be described in terms of forces, clashes and remnants, in order to illustrate the concepts.

The Design Requirements

Any design project is initially defined by the requirements or demands involved. They can be very loose, such as "*create a new entertainment product*" or very elaborate, such as "*create a cheaper version of [whatever], as close as possible to the original without breaking any copyright laws*". A very rigid view is that these initial requirements should

[62] See the "Intermediate" section on game design, starting on page 79 for a more thorough explanation.

be written as detailed specifications, which should be set in stone for the rest of the project. Even if this might be a laudable approach, especially for very large projects where different teams are working far apart, it hardly ever works – hence the need for methods such as agile software development – since any design-intense project has a somewhat volatile character; and the more multidisciplinary a project is, the larger the complexity and risk for changes in the initial or later requirements.

Therefore we would like to introduce a concept we call the Design Requirements. Unlike a traditional specification the Design Requirements are subject to change. The term "requirements" – as opposed to "specification" or "demand" – is thus deliberately chosen, since the whole point is that the Design Requirements are flexible; they impersonate the living, ever-changing core of the design.

At the beginning of a project the Design Requirements are a combination of the stated **design objectives** (Jones 1992, p. 194 - 200), e.g. "*it should be a fun game*"; the **fundamental requirements** that come from each involved design discipline (e.g. the chosen technology, material, programming language, hardware), and **external demands**, e.g. commercial or economical concerns, e.g. "*it needs to be released in November and the retail price mustn't exceed 25 Euro.*". This is a very explicit list of demands. However, as soon as the development starts, the Design Requirements are currently being updated with the latest version of the design within each design discipline. These designs might not state new requirements explicitly, but implicitly. Designing the look and functionality of a graphical user interface for instance puts implicit requirements on programming.

One could of course set up rules for formalizing the input to the Design Requirements, depending on the need for steering (e.g. if the team is large or not co-located) or one can rely on frequent communication between designers in the different disciplines. However, since disciplines may differ very much, it can be hard to formalize submissions, since design in each discipline may be best displayed in its own manner, which is not necessarily a written description. In the myTHeme case, which only involved a handful of people, it was sufficient to simply add the latest contributions to the design in any suitable form, e.g. images to show the graphic design and the graphic user interface design, commented code, the rules of the game as text etc. Frequent meetings notified the team members that new updates had been made.

All of this means that the Design Requirements also can serve as an *interface between design disciplines*. In this role it transfers new *requirements* or *forces* from one discipline to another but in some cases these requirements or forces contradict with others; a common phenomena in all development and design. We have chosen to call these common contradictions *clashes*.

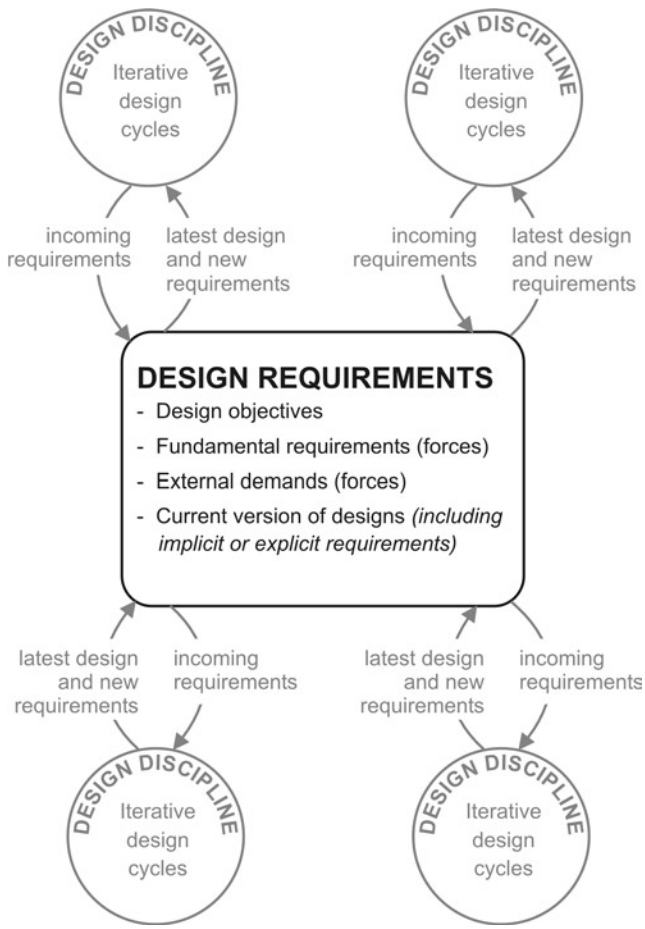


Fig 23: The Design Requirements in relation to four different design disciplines involved in a design project.

Requirements

A requirement is a part of the Design Requirements. It is a neutral request or demand asking for something to be accomplished. New requirements can occur at any point in the development, typically as a result of evaluation, testing, new design or redesign. They in turn typically trigger new design or redesign, meaning that they are the prime events and the prime motor in the development process.

Forces

Forces too, are parts of the Design Requirements. They are very strong requirements based on vital design decisions made within a discipline. They have an inherently large influence on the work in other areas, i.e. actively affecting design and design choices in them. A force will typically overrule or steer design issues in any other discipline, unless another, stronger force contradicts this. Forces typically occur in the beginning of a project – i.e. most if the initial Design Requirements consists of forces – and are normally the result of a fundamental choice of e.g. technology or material. Forces could also occur during a project if many design decisions are tightly aligned towards the same direction.

An example of a force could for instance be that a game should stick to a certain theme or world, i.e. Harry Potter, and that it mustn't contain any violence at all. The former may be a part of the initial Design Requirements whereas the second might occur during user tests and is supported by some other force, e.g. that the game should be suited for children aged 5-7.

Forces can however be ranked; some forces are stronger than others. At some point a force is no longer a force but just a requirement in general, but where and when this occurs depends on the actual project.

Clashes

A clash occurs when there are contradictory requirements or forces (e.g. conflicting designs or ideas). A clash can occur within one design discipline, but more commonly between one or more different disciplines. In any case, a clash always results in a clash resolution which means that re-design has to be made in at least one of the involved disciplines. This may in turn trigger changes within other design disciplines as well.

An example of a clash could for instance be a web designer creating a page with certain data on it. This might clash with the database design if this data is actually not in the database. The clash could be resolved in several ways; either perhaps the data could be calculated by using already collected data (i.e. redesign within database design) or the web designer could design another page earlier in the flow asking for that data (i.e. redesign within both disciplines), or the web designer could omit the data again (i.e. redesign within web design). The latter may trigger changes in how to deal with e.g. orders, perhaps triggering redesign within interaction design, etc.

Remnants

Remnants are left-over design choices whose prime incitement no longer exists, i.e. the demand for them being, working or looking the way they do, is no longer there. Some remnants are harmless and some are valuable since they still may hold important qualities, but some are restricting the overall design. It is often hard to distinguish a remnant from a necessary design choice, but if one records the design in terms of clashes and forces, they are easier to discover since one can backtrack the reasons and validity of each design decision, seeing if they still apply.

A well-known example of a remnant is the typical computer calendar. It still has many of the characteristics enforced by the paper-version, i.e. showing only the days of the current month instead of – for instance – showing the upcoming thirty days regardless if there is a month shift or not. The flexibility of software is standing back for tradition and paradigm.

The event-driven iterative design process for game design

The model is built upon the *Design Requirements* that keep track of and distribute new requirements, forces and clashes between the involved *design disciplines*. During development every design discipline undergoes several iterative cycles, of which some are planned as being normal stages of development and others can occur more or less suddenly in the case of an unexpected new requirement, force or a clash, i.e. a restriction in the capabilities of hardware or code can trigger an instant redesign of the game rules as a preparation for the next prototype for instance.

The planned development can be said to be event driven in the sense that one or two disciplines act on the outcome of what has been accomplished in earlier cycles, e.g. in the case above no programming can be done until the basic game idea is established. Hence, each change of the Design Requirements (i.e. the addition of new

designs and requests) typically triggers a new iteration in one or more disciplines, including the one that updated the Design Requirements. A typical iterative cycle within a discipline has four stages. The flow of one cycle is best described in the following model[63] (see Figure 24 also):

- **Analysis.** In this stage the current development is evaluated against the triggering requirement in particular and the Design Requirements in general. Several things can occur now.
  - If the request is neutral or if there is a clash that can or shall be resolved within the current area, the iteration continues with the next step.
  - If the requirement has already been fulfilled, nothing is done.
  - If the requirement cannot be fulfilled since preparing work first needs to be made within other disciplines, nothing is done and a requirement (or perhaps just a remark) is being transferred to those disciplines via the Design Requirements.
  - If a clash occurs, it has to be resolved before any development continues regarding that special issue. The resolution may result in a new requirement affecting any discipline or the same requirement being returned to the current discipline.
- **Synthesis.** Coming up with and evaluating possible solutions to the demand, keeping the Design Requirements in mind. Briefly documenting this in case there is a later need for redesign. Choosing a solution and how to implement it.
- **Implementation.** This stage is typically characterized by practical work such as programming, drawing or building, but might as well be non-

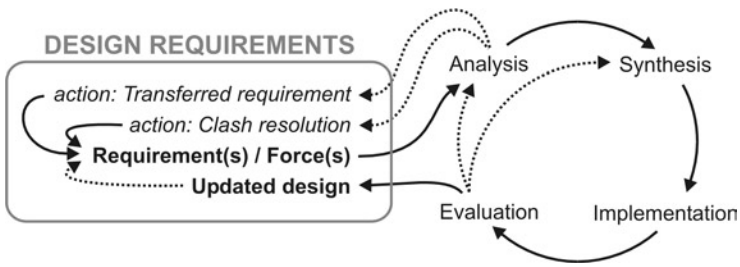


Figure 24: One iterative cycle in one design discipline.

[63] (Note that this is not the exact same flow as presented in the paper by Lundgren & Björk (2005). It is the same in essence but the stages have in some cases been given more descriptive names, and two stages, coming up with solutions and choosing one, have been merged.

practical, e.g. running a test and summarizing the conclusions of it, or writing new specifications on how to create interaction. The latter two activities typically trigger activities within other disciplines since they will add requirements to the Design Requirements.

- **Evaluation.** Finally the solution is evaluated against the triggering requirement in particular and the Design Requirements in general. Several things may occur in the evaluation phase.
    - The requirement was solved and no new requirements arose. The Design Requirements are updated with the latest design.
    - The requirement was solved and one or more new requirements arose. The designer may if it is possible label these requirement as being possible forces or clashes, and also notify the design disciplines that are affected. The Design Requirements are updated with the latest design as well as with the new requirements.
    - The requirement was solved but a conflict with some other part of the (perhaps updated) Design Requirements has been discovered. A new cycle begins instantly, starting with the analysis-phase where it is resolved whether the conflicting requirement is a force or if it is a clash that can possibly be redirected.
    - If the requirement has not been solved, perhaps due to unclear or misunderstood requirements, a new cycle begins, starting with the synthesis phase.

Since the model is event-driven, and events are typically new requirements or re-design requests in form of clashes, starting out with one simple requirement in one area can trigger changes in many other design disciplines as a result of this. Below, such a sequence is described:

1. A requirement starts a design cycle in hardware design. The final result is an updated design and a new requirement, which triggers the next cycle.
2. The next cycle occurs within gameplay design, and during the evaluation phase the designers discover that they had misunderstood the requirements, iterating again, this time starting with the synthesis-phase. The final result is an updated design and a new requirement, which triggers the next cycle.
3. When analyzing this requirement within programming, a clash is detected. The clash resolutions triggers redesign in two design disciplines simultaneously; programming and gameplay design.

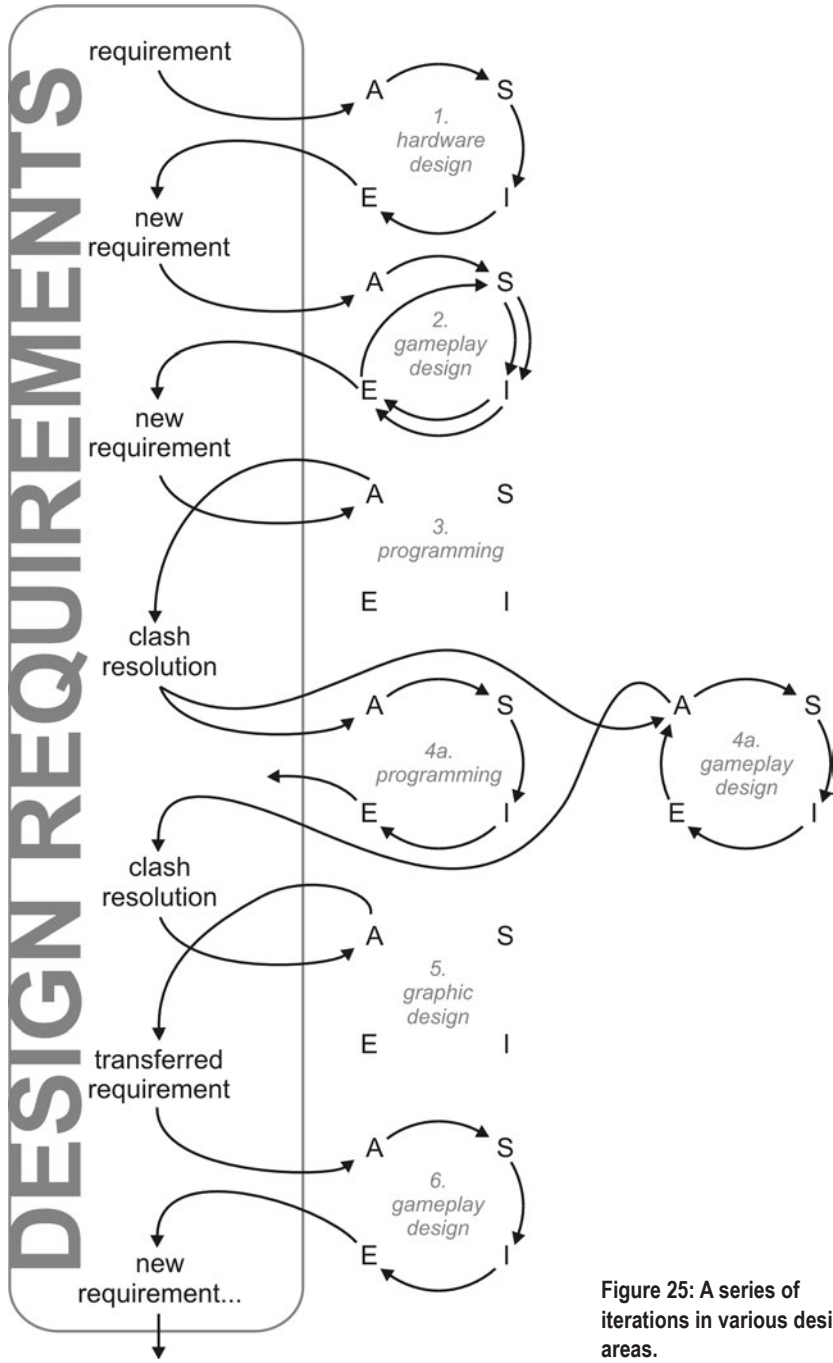


Figure 25: A series of iterations in various design areas.

4. In programming (4a), the iteration goes smoothly, and results in an update of the Design Requirements but in gameplay design (4b) a new clash is detected in the evaluation phase. The clash resolution triggers redesign in graphic design.
5. When analyzing the requirement in graphic design, it is found out that some work needs to be done within gameplay design first; a new requirement is added to the Design Requirements and also communicated to gameplay design.
6. The requirement is solved in gameplay design. The Design Requirements are updated with the new design, and perhaps some new requirement triggering development somewhere…

Even *if* the last cycle did not result in an obvious requirement, development is not over. The next not-yet satisfied requirement in the Design Requirements can trigger a similar cycle, and on it goes until all current requirements have been met, in which case the entire design/development is "done".

The model supports both serial and parallel work (e.g. as in step 4) in different disciplines, providing an opportunity for tight co-development between areas. However, since parallel work in several design disciplines increase the risk for simultaneous changes in the Design Requirements, the risk of clashes is greater than for serial work.

Case: MyTHeme in Terms of Forces, Clashes and Remnants

In this section the development of myTHeme is described in terms of requirements, forces, clashes and remnants. Do note that these terms did not exist during the design; however the concepts did, more or less explicitly since they were developed in parallel with the game; we analyzed the process of designing the game while we were at it. You will notice that the main evaluation process was playtesting, normally leading to changes in the Design Requirements and thus redesign within various disciplines. In other types of projects this may correspond to making user tests, run test protocols etc.

From the multidisciplinary nature of the project it was clear that the development of myTHeme would be both iterative and evolutionary, especially since it was an

experimental design. Further, to ensure that a focus on gameplay would be present the process would start with work on the gameplay design.

Game concept

MyTHeme is a game where players together create a heroic story by playing cards. The game evolves around a number of concepts: cards depict problems, solutions and rewards which together form sub-stories; cards have colors and symbols which are used to create point-generating series; and several sub-stories are created in parallel. Although the game is competitive, the players do not play specific heroes or heroines and the players create the sub-stories together. The story loosely follows the traditional structures found in fairy-tales and myths (Propp 1984) in that the hero has to solve a number of problems, e.g. the young heroine who sets off to save the land by bringing back a magic talisman may encounter and kill various trolls, befriend unicorns, fall into pits, and so forth. In line with this the inspiration to the storytelling content of myTHeme comes from fairy-tales and fantasy stories.

Putting together the Design Requirements

As mentioned earlier, Jones (1992, p. 194 - 200) states that it can be useful to state the objectives of the design or product before starting the design process. For the myTHeme project there were three absolute design objectives:

- The final product should be a computer-augmented game but have the same gameplay as traditional card game
- This game should evolve around storytelling but not require the players to generate the stories
- The game should still be interesting and well-designed enough to be playable even without the storytelling-theme

In addition, the design disciplines that were to be part of the myTHeme project were identified, and some initial decisions regarding technology were made. These were partly depending on "outer" requirements, such as the competence of the researchers, and the fact the hardware from another project was reused. The six areas were:

- Gameplay design
- Programming (Java)
- Hardware (RFID technology)

- Writing of story sections (as distinct from story writing)
- Graphic design
- Interaction design (of a non-traditional user interface)

Together with the first set of game rules these initial choices and objectives made up the first set of Design Requirements, formulated as a set of specific requests; there should be game rules, there should be cards, there should be text snippets, there should be a working story generating mechanism, hardware and software enabling this etc. Some initial forces that affected the Design Requirements were:

- *The story is important*. The most essential part of the game experience was, and is still, the evolving story that is created through gameplay. Thus it was a driving force that the stories created should make sense.
- *RFID-technology should be used*. Coupling RFID readers with Smart-Its devices[64] provided an easy way to track played cards simply by inserting RFID tags in the cards.

First iteration: Creating the rules of the game

When making the first set of rules for myTHeme, only the initial design objectives-part of the Design Requirements mattered, and drove development. Immediately, inspiration came from fairytales, such as the Grimm stories. When analyzing these it was concluded that universal elements were a *hero* or *heroine*, encountering a number of *problems*, to which they had to find *solutions*, earning *rewards* of some sort, finally somehow *concluding* the story. It thus felt suitable to let these elements be represented by cards that somehow had to be played in the right order. However this appeared to be too simple, so the element of *theme* of a story was added as well as a *symbol*, only used for scoring.

Another basic notion was the idea to allow several possible chains of events (later known as sub-stories) in order to enrichen gameplay. A sub-story had to consist of a logical course of events, i.e. one or more Problems being solved using a Solution, with a certain Result, turning this possible course of the events into the truth, i.e. making the sub-story a part of the main story. This introduced a racing mechanism into gameplay, which was desirable to add some tension to the game.

The first set of rules was added to the Design Requirements and put requirements on graphic design in order to create some cards so that we could playtest.

[64] See www.smart-its.org for information about this FP5 EU project.

Second iteration: Creating the cards

A set of cards were created. They have had the same properties throughout the whole process, however the graphic layout has changed from iteration to iteration. Below is a description of the final layout. The cards are simply black-and-white prints on colored paper. Each card in the deck has four properties, a Name, a Theme, a Symbol and a Type.

- The Name is unique – "Dragon", "Food", "A Desert", "Despair", "A rope" etc. This property communicates an idea about what kind of text is associated to the particular card. A picture depicting the subject complements the name to evoke the thematic setting.
- The Theme can be Epic, Comic or Tragic, each represented by a different color. This property is relevant for scoring, and will foster clueless storytellers into sticking to one theme. The Theme is shown by the color of the card.
- The Symbol is a Hexagon, a Circle or a Square, and is shown by the frame enclosing the picture. This property is only relevant for scoring.
- The Type is a Problem, Solution or Result. It is shown by a shape on the edge of the card. This property is relevant for the order in which cards can be played. In addition, the Problem cards and Solution cards have an extra property, a Variant, which is also crucial to gameplay, since the Variant of a Problem and its Solution need to match. There are three Variants: living Opponents (such as monsters, witches, and guards), physical Obstacles (e.g. deserts, fire chasms, and magic zones) and Psychological problems (e.g. fear, homesickness, stupidity). To clarify:
    - —*Dragon* (Type: Problem, Variant: Opponent) can be solved by for instance *Sword* (Type: Solution, Variant: Opponent)
    - —*Desert* (Type: Problem, Variant: Obstacle) can be solved by for instance *Shortcut* (Type: Solution, Variant: Obstacle)
    - —*Despair* (Type: Problem, Variant: Psychological Problem) can be solved by for instance *Love* (Type: Solution, Variant: Psychological Problem)

The Variants are an effect of that it isn't logical to deal with, say, an enemy the same way as a dead obstacle; it would not be a good idea to try to cross a Desert using Love, or to kill a Dragon with a Shortcut. Using a Sword to deal with Despair isn't

something to be recommended either… The Variant was communicated with two corresponding shapes, one on the Problem and one on the Solution.

The Result cards are used to finish off sub-stories, making the part of the final story created throughout the game. In addition, each Result card introduces a new item, e.g. magical sword or rope, to the hero or heroine's inventory. These items activate the possibility to perform extra combos, e.g. using a Heroic Act as a solution to a Dragon when carrying a Magical Sword.

The Theme (color) and the shape are rendered on both sides of the card; this affects gameplay since these properties then become common knowledge to all players.

Evaluation: Playtesting

The first cards had no drawings on them, just words explaining what they were. There were four Themes (colors) and four Symbols. Using these simple cards, playtesting a set of possible rules started. Initially there were Heroes and Conclusions in addition to the other cards, but after having toyed around with them for several playtests without getting them to work within the game they were abandoned. The game was hard to play with only text on the cards; it took a lot of reading to check out the cards and understand which card fit where.

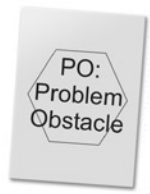


Figure 26: The first version of the myTHeme cards.

- This created a requirement to improve the cards graphically by drawing motifs for each card, but also to graphically distinguish between Variants using concave and convex shapes; a concave shape (determined by the Variant) on the Problem and a corresponding convex shape on the Solution.

During these sessions the general game play was outlined. The general aim of the game is to win by contributing to the story that is being created by the game. This is made by playing cards in places that are allowed (which cards could be played where varied throughout the process due to various forces and clashes). To complicate matters further, there is not one but three parallel stories going on simultaneously.

A finished sub-story consists of one or more pairs of Problem-Solution cards and one concluding Result card. Each of the Problem cards belong to one of three Variants (opponent, obstacle or psychological) and a Solution must match the

Variant to become a pair with the Problem. A sub-story may not be longer than seven cards, i.e. three such Problem-Solution pairs and a Result.

When a Result card is played on a sub-story it is immediately finished. The generated text is moved to the main story and a score board is shown that indicates the number of points each player scored. Thus players want to take part in completing as many sub-stories as possible, meaning that they need to cooperate somewhat with other players. Creating long sequences of cards having the same Theme and/or Symbol increases the score.

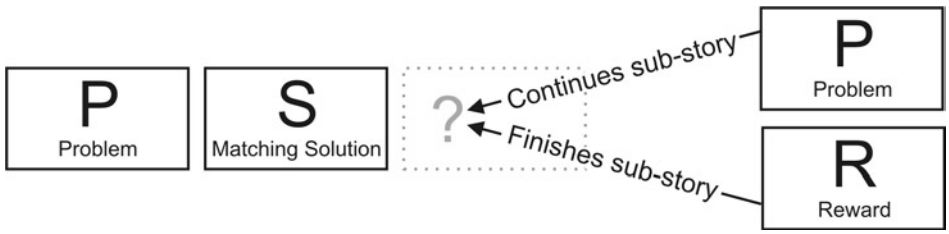


Figure 27: Here, the sub-story can either be continued or finished depending on which card is played next.

The first clashes that arose during playtesting was that the initial four Themes and four Symbols were too many; it was hard to match the Theme or Symbol of played cards in order to score, and this needed to be changed; a redesign of the cards and game rules was needed, however not a significant one.

Due to the first clash, some cards were taken out of the game. The iteration ended with an update of the Design Requirements, in that a list of all cards was put together, since this affected both graphical design and text writing.

Third iteration: Parallel creation of texts, cards and a program

New cards were designed, printed and laminated with an embedded RFID-tag. Meanwhile the first steps within the storytelling discipline were taken: an xml-template was created in order to write text into, and a few test texts were written. Xml was chosen since it is formalized markup language that would be easy to use together with the java code. This of course required some java code to be written in order to interact with the RFID-technology and put together the text snippets in the xml-files.

The RFID-clash

During this development phase two huge clashes arose, due to the choice of RFID-technology. In the early stages of gameplay design the idea was that cards could be played in a different order than they would be used in the final story. For instance, if a sub-story consisted of a Problem-Solution-pair that could be concluded by a Result, a player could block this conclusion, not only by playing a new Problem after the Solution, but also by leaving a "hole" and play a Solution, or leave two "holes" and play a Result etc.

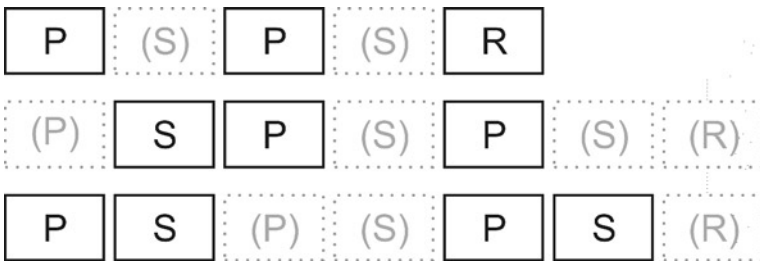


Figure 28. Possible ongoing sub-stories if cards can be played anywhere in random order.

Unfortunately, this would require that the physical game board should be made out of a grid of 3 x 7 RFID-readers. Since the cost of the readers exceeded the budget, a clash occurred where the budget and the choice of hardware limited the number of active play areas to three. This meant that each sub-story was represented by one reader only, not seven as planned. Due to this, there is no way for the game to "know" where on the board a card is being placed; the tag-readers only indicate the sub-story, not where in the sequence.

Although this force clash limited the card readers to three, the clash had still to be resolved; which discipline should deal with the problem, gameplay design or programming? The "holes" *could* be kept anyway; if the program code was written to deal with them. If a Problem was played in a sub-story, and then yet another Problem, wouldn't it then be obvious that there should be a "hole" after the first problem? Or, if a Solution was played, that didn't match the Problem's Variant, two "holes"? This idea was abandoned due to several reasons. Firstly, there was the case when a Problem of a certain Variant was played on the first space and a matching Solution of the same Variant was played on the fourth space (indicating that yet another Solution and a Problem of the same Variant should be played in spaces

two and three). In this case the placement of the cards on the board would differ from the virtual placement of them – very bad from an interaction point-of-view. Secondly, the program would not be able to distinguish a wrongly played Solution, not matching the Problem Variant it was being placed next to; the player would place the card next to the Problem on the board, and the program would conclude that it had been placed elsewhere. Again a discrepancy between the physical board and the virtual board would occur. This, the idea of keeping the holes was abandoned, and the RFID force and the clash resolution resulted in the following:

- The rules needed to be changed so that cards had to be played in sequential order only.
- It impaired the game from an interaction design point of view – players now first had to play their card on the reader and then place it on the table instead of just placing it.
- The original game board was never built as had been planned and designed.

Evaluation: Playtesting and tweaking the working prototype

When playtesting again three flaws were quickly discovered. The first was a clash between interaction design and graphic design. The Variants of cards had been communicated during convex and corresponding concave shapes in order to show which cards could be played where (different shapes for each variant, and convex shapes for problems), but this formed arrows pointing "upwards" the story which made interaction feel a bit strange.

- This triggered redesign of the cards

Figure 29: The second version of the myTHeme cards.

Another flaw occurred rather early in the testing phase when only a few cards had been made. The cards in the deck circulated very fast, so when the Dragon was played for the third time, its text felt worn out. The solution to this was to program some variable properties for each card, in the Dragon's case its color would vary.

- This triggered redesign in story-writing – the variable properties had to be added

- This also required extra programming; adding a variable property to the text had to be taken into account when parsing the text snippet.

The last flaw was significant and consisted of story-writing clashing with itself – according to the initial requirements that the story was important – and as a result also affecting graphic design. The stories created weren't very good! A huge part of the feel of the game was concealed within the text snippets that would become the stories, and the problem turned out to be how to make all these texts combine into a story with a good textual flow. Partly the problem was solved in advance by the rule that Problems and Solutions played in pairs had to have the same Variant (e.g. a Problem card of the Variant Physical Obstacle had to be matched with a Solution for a Physical Obstacle). This rule reduced the possible combinations of cards being played together, but still eighteen possible Problems of a Variant should be matched by eighteen Solutions of the same Variant. This provide a significant challenge to having a logically consistent story: you may very well use a *Rope* (an early Solution to Obstacles) to climb out of a *Canyon* (a typical Obstacle), but not to cross a *Haunted Forest* (another typical Obstacle).

This problem was finally corrected by making the Solutions and their corresponding texts more general; for example, by having Solutions such as *Endurance* ("Our hero summoned all his strength and endurance to conquer [the obstacle]"), *Luck* and *Avoid*. This solution works well, but triggered a lot of rewriting and drawing of new motifs for the cards. This last clash forced a large update of the Design Requirements: a new set of cards was defined, adding quite many new cards and omitting others

- This lead to significant design and redesign of cards
- This also required a lot of writing and rewriting of texts

Fourth iteration: Creating the projected game board, rewriting texts and programming varying properties

Until this stage, the stories created had been output as plain text on the computer's screen. Now it was time to take immersion a step further by instead projecting the ongoing story on the wall. The Design Requirements concerning this were that the projected board should be projected on a wall close to the players, being updated for each card played. Roughly the area was divided into five parts; one for the main

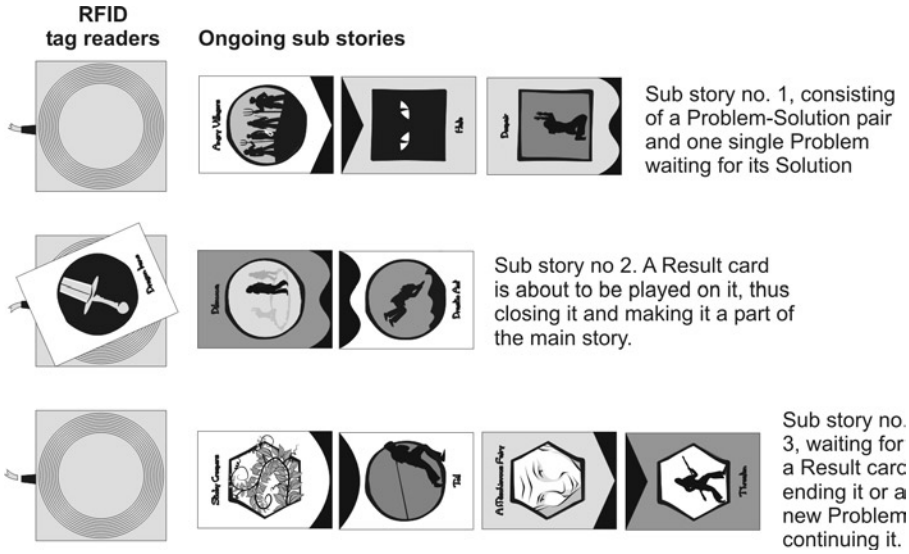


Figure 30: A theoretical example of an ongoing game. Notice how the played card first has to be played upon the tag reader (see third iteration). Note also how the different Variants are visualized using different convex and concave shapes. The pointy shape signals an Opponent, the rounded an Obstacle, and the wavy a Psychological problem. Since this is expressed graphically on the cards there is no need to confuse players with labels on the card.

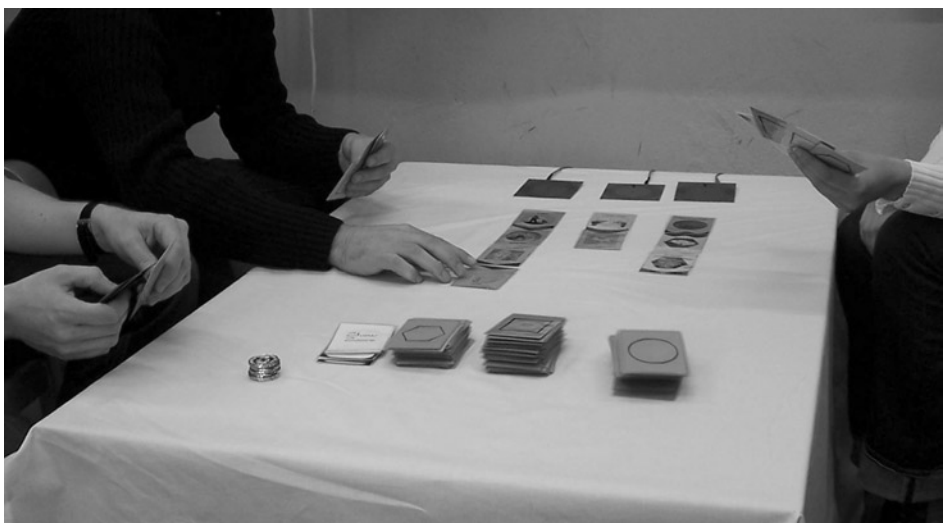


Figure 31: …and myTHeme played in reality.

story, one for scoring, one for inventory and three for each of the ongoing sub-stories.

Simultaneously the varying properties (e.g. the dragon's changed color) were incorporated in texts and in programming. The new cards and texts were created.

Evaluation: Playtesting the working prototype and the projected game board

Once again, the game was playtested, with new cards and new texts, as shown in Figures 30 and 31. This time, there was a clash between the graphic design of the projected game board and the texts. The area of the projected game board was to small, or the texts to long; even if projected upon a wall the effective area to design for was no larger than 1024 x 768 pixels with standard screen resolution, and in addition the font size had to be large since the texts were to be seen at a distance. A first solution to this was to replace not-yet-solved Problems texts with a shorter text. E.g. the Troll card would not display the entire text (*"In this region a huge troll terrorizes a small village. The villagers feel so threatened that they are afraid to tend to their crops. The closer our hero gets to the village, the heavier the stench of the troll gets. Suddenly cracking noises are heard from the shrubbery that lines the path and the beast emerges in front of our hero."*), but a so-called waiting text ("*…the air is thick with the pungent stench of trolls...*") taking up much less space. The interaction design logic of this was the hero or heroine would not be fully aware of the nature of the Problem until encountering it.

Figure 32: The final version of the myTHeme cards.

- The first clash led to additional writing, creating the waiting texts, and additional programming, dealing with them.
- Also a new requirement arose; a function making it possible to flip between pages of the finished story had to be created too; there was no way it would ever fit on one screen.

The first remnant

In this iteration a larger deck of cards was used, since additional cards had been fabricated. This made the entire problem with cards showing up often disappear! This made the varying properties a remnant, since the initial incitement for enforcing them was gone, but they still added flavor to the game, and thus they were kept.

Fifth iteration: Minor redesigns

In this iteration texts were expanded with waiting texts, the program was changed accordingly. A few cards were redrawn.

Evaluation: Playtesting yet again

Would the game finally come to a completion? No. When playtesting again, interaction design clashed with itself. Up until this point all the meta procedures of the game (initializing the game, setting up the correct number of players, quitting the game etc.) had been done via the keyboard of the computer. But when all other obstacles had been removed, this violation against immersion was detected. The solution was to create an extra set of cards, Control Cards, which provided the possibility to do such things. This also led to the waiting texts being omitted again since control cards for navigating through sub-stories were created.

- This clash led to design of a few new cards and the programming of their function. The use of the "waiting texts" was omitted again.

Grand finale: Showing it to the public

In this version myTHeme was finally fit to be demonstrated to the public. Since, it has been shown at an internal mini conference of the Interactive Institute Sweden, at the Digital Games Research Conference in the Netherlands 2003 and at several workshops. No further clashes have been detected, but playtesters have contributed with a lot of creative ideas that have been used to improve gameplay.

The Final Game Rules

After these five iterations we stopped the development of myTHeme, even if there were still things that could be improved – for instance it was suggested thet sound could be added to enrichen the game experience even more. Thus, the "final" game rules are described in the following section.

Game components

The game consists of the following physical components: *cards*, an area to play them (the *physical board*), *control cards* (e.g. "New player", "Quit" etc), three *card readers* connected to a *computer* which is in turn connected to a *projector* that projects

a *digital game board* on the wall. Non-physical components consist of a set of *text snippets*, one for each card, used when putting together the story the players create by playing the game. This is done by a *Java program*.

Radio Frequency ID (RFID) technology is used to keep track of which cards was played, where, and in what order. The abbreviation stands for Radio Frequency ID, and as implied, they use very weak radio signals to communicate. The RFID-technology consists of two components; small flat tags that are easy to embed into objects (in the myTHeme case the cards), and readers, that are flat and can be built into for instance game boards. The latter need power from some source and cost fairly much, as opposed to the very cheap tags.

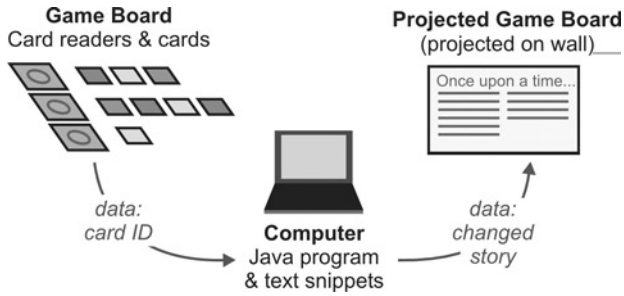


Figure 33: An overview of the components and the dataflow in myTHeme

Setup

A new game is started by playing the New Game control card on any card reader. The program responds by presenting a randomly generated hero, including name and epithet. This presentation indicates the gender of the hero and the main theme of the story; e.g. Shiana the Courageous is female and probably belongs in an epic story, whereas Stefano the Stubborn seems to be a male comic character. The program also presents the main three goals of the hero or heroine's quest; these are simply three random problem cards from the deck. This information also tells the players when the game will end, which is when all these problems have been solved. Each player registers to the game by playing a New Player control card. Each player is dealt five cards with the remaining cards divided into three roughly even drawing piles. The game proper starts with the first registered player placing a problem card on one of the tag readers.

Rules concerning sub-stories

The game allows three sub-stories to be created in parallel. A finished sub-story consists of one or more pairs of Problem-Solution cards and a concluding Result card. The cards must be played in order, i.e. a Problem card cannot be played when another Problem card is lacking a Solution and the Result card can only be played when all Problems and Solutions match up. A sub-story may not be longer than seven cards, i.e. three such Problem-Solution pairs and a Result. When a Result card is played on a sub-story it is immediately finished. The generated text is moved to the main story and a score board is shown that indicates the number of points each player scored. The players are assumed to remove the cards from the column of the imaginary board and place them in a discard pile. The other ongoing sub-stories are left untouched.

Besides finishing a story, each Result card introduces a new item, e.g. magical sword or rope, to the hero or heroine's inventory. These items activate the possibility to perform extra combos, e.g. using a Heroic Act as a solution to a Dragon when carrying a Magical Sword.

A player's turn

A player's turn simply consists of playing a card and drawing a replacement. However, a number of rules must be observed. Firstly, a player may play a card on any one of the ongoing sub-stories but must follow the logical order: a Problem must be played before its corresponding Solution and one may play either a Result (to close the sub-story) or a new Problem (to prolong it) after a Solution. The variant of a Solution (opponent, obstacle or psychological problem) must match the variant of the Problem. A player must play a card if he or she can; if not he or she will discard a card and draw a new one until he or she can play the newly drawn card. Note that the player has to play the card on the RFID-reader representing a certain sub-story before he or she actually places in on the table. Having done this is indicated by that the text associated with the card shows up on the projected game board.

Thereafter the player draws a new card from one of three drawing piles. Since Theme and Shape are visible on the back side of the cards there's a point in having three piles to choose from and players may also benefit from looking at the public side of their opponents' cards.

Scoring

The basic mechanism for gaining points in the game is to create series of shapes and/or colors: a series of three or more gives points when the sub-story is finished and each card in the series is worth as many points to the person who played it as the length of the series. Playing the ending result card gives 1 point and activating a combo gives 2 points. In addition, receiving the gold coin described below is worth 1 point.

Game end

The game ends as soon as all the three Problem cards of the hero's quest – the ones defined by the program in the beginning – have been played and their sub-stories have been closed. The player who has the most points wins.

Reflections

MyTHeme

The current version of myTHeme is satisfactory in that it fulfilled the three initial objectives; it is a computer augmented card game with natural gameplay, it is concerned with storytelling and there is am abstract variant of the game which is playable and thus supports the rules as sufficient without the "make-up" of the story-telling element. Since these goals were fulfilled, the project was shut down in this state.

Problems

In retrospect it would have been nice to evaluate myTHeme more, but this was not the scope of the project. One typical problem is that players that have played the game several times are not interested in the lengthy texts anymore; they "only" play the game. For them, the entire storytelling-immersion-factor is lost. Once the "slow" exploration (see Facet No. 1 if you want to know more about Slow Technology) of the texts is over, the game loses a lot of its charm.

This suggests that the game is best suited for children, or for being played only once or twice or perhaps rather seldom, in order to prolong this exploration, but then again; why design a game that can only be played three times? Well, if being

placed at an amusement park, or at the rehabilitation ward in a hospital, or as an installation in a library, the game could still serve very well.

Lessons Learnt and Reflections

Looking back at the design process of myTHeme, it is clear that the different iterations should have been planned more carefully, i.e. what was to be done in which order. For instance elaborate motifs of the cards were drawn rather early, and when the storytelling clash occurred after testing the third iteration (the one resulting in Problems being solved by rather vague Solutions, i.e. a *Desert* being crossed using *Endurance* instead of a *Rope*) a lot of the cards – and carefully made drawings – were omitted and new motifs fitting the new cards had to be drawn. In retrospect it is easy to say that the first motifs only ought to be rough sketches, especially since there was even an initial force saying that the story was important and thus, changes in the storytelling was bound to lead to clash resolutions affecting other areas.

Possibilities

MyTHeme can still be improved in many ways. As mentioned, we who have played it a lot, have come to find the long texts rather tiring – we don't read them anymore but focus on the board and on the game, whereas first-hand players tend to like exploring the cards and the texts. A way to satisfy both new and experienced players would be to increase the variable properties in each text snippet. Even larger difference could be made if each card had two or more different variable texts associated with it. Also, more elaborate code and text could keep and display environmental variables, like that the adventure is set in a certain country, ruled in a certain way. Intermediate snippets about what goes on in the country or what happens elsewhere could thus be woven into the story.

Another improvement that would make the game richer would be to associate sounds with the cards too. Having the card's text read to the players would probably be too much – it might very well be perceived as annoying (unless the players are smaller children that cannot read), but sounds could be used to enhance emotions and associations, e.g. a strange shriek or hoot could be played when the card *Haunted Valley* is played. As discussed elsewhere it is risky to have sounds as sole carriers of information, since this means that players need to listen up, that the environment needs to be quiet and that players can't talk freely.

One could argue that all of these improvements serve only to create a richer story and a richer experience; they do not improve gameplay per se. This is not entirely

true; players are or can be tempted to rather play a card that scores less but fits the story as the player wants it to evolve; i.e. basing play on emotional rather than rational grounds.

Using The Forces-Clashes-Remnants Model

As many models the Forces-Clashes-Remnant is not a static model; if you read the original paper (Lundgren & Björk 2005) and compare it with this text you will already find several refinements and clarifications.

Problems

Even if the model does support simultaneous work in different design disciplines as well as subsequent work, one must keep in mind that the more disciplines involved at the same time, the larger the risk is that clashes are designed into the system since developers are currently unaware of what is going on in other disciplines; the Design Requirements have not yet been updated.

In addition, the Design Requirements consist of a variety of different materials; code, drawings, written specs etc (see motivations for this in the next section), and this can sometimes make it hard to overlook. For instance it is not easy for a graphic designer to read and understand code and from it conclude that there is a clash somewhere. Thus, new probable clashes and new requirements need to be clearly communicated from one design discipline to another.

Both these problems imply that the method probably works best for smaller multidisciplinary projects where it is possible for each team member to have at least a chance to overlook what is being done within other design disciplines.

Lessons Learnt and Reflections

If using the model, there are a few things that are important to point out. Firstly, it is not defined how the overall Design Requirements should be described. We do not wish to formalize this in the method since each project and each discipline is unique in terms of existing demands and the best suitable ways to express them, i.e. the objectives of many disciplines cannot be described in a way that is very formal (e.g. interfaces in programming languages) and/or easy to measure (e.g. "The game must be entertaining"). Depending on the discipline, the way to communicating design choices can take many different forms, e.g. prototypes, scenarios, storyboards, and design patterns. This can of course be seen as a weakness, since some forms are harder to intuitively understand than others, e.g. can it be hard to quickly judge

how a part of the code will affect interaction or expression. This problem will grow if many disciplines are involved (since no one can be an expert of all) and/or if the project is so large that it is run by a project leader lacking profound expertise in too many areas.

Secondly, if using the method it must be kept in mind that the Design Requirements are subject to change; they *shall* be regarded as being flexible, ever-growing, ever-changing, ever-improving. Especially gameplay design is a very volatile process that is bound to affect the Design Requirements one time after another, as opposed to for instance hardware requirements that are set from start.

All of this necessarily means that the overall Design Requirements will consist of a heterogeneous collection of information which may not be reducible to one type of description. Further, even the most essential parts of the initial design objectives may be changed due to external, unforeseen events such as the release of a similar product, or an overall decision that all or some of the initial objectives were partly wrong and do not work and thus need to be revisited.

It should also be kept in mind that, at least in the myTHeme case many of the clashes only became visible during playtesting. Then again we did not make any other attempts to find them – playtesting was simply our method for doing this. This might suggest that very frequent testing is needed to support the model. Note, however, that requirements could, and were, detected and communicated during cycles.

Also, in myTHeme we were only three main developers. This meant that we were responsible for different areas; Staffan took care of programming, Jennica of the story-writing and I of the graphic design and the interaction design. Although I wrote the first version of the rules they were worked out by all of us and by several playtesters in the various evaluations. Anyhow, this responsibility for a certain discipline was both good and bad. It was obvious whom to turn to when a certain requirement had arisen, and it was also obvious who should carry it out. However this strong division in various disciplines of course led to a tendency to defend one's territory – "*Why must **I** redesign this by redrawing, can't **you** just as well change your code?*". In some agile software development methods, e.g. Extreme Programming (Beck 1999) programmers never work alone with a certain code; there are always two developers. The pairs change often, so that each programmer has taken part in an individual collection of code. This means both that no one has sole responsibility for anything, but also that a pair of programmers together overlook a greater amount

of the total code, which will reduce clashes. Also, in case of a clash, this sense of defending one's territory would not be so strong within the pair.

Forces – good bad or evil?

As mentioned, forces are design decisions that steer design decisions in other design disciplines. Sometimes a force, typically an initial one, can be very obstructive (e.g. when the choice of RFID-technology limited the possibilities in gameplay design) but on other occasions it can lead to very positive effects (e.g. establishing Variants for the Problem and Solution cards as a gameplay design decision strongly reduced the number of texts that had to match). Thus it is impossible to say in general if a force from one design discipline is "bad" in the sense that it is preventing good design solutions in another discipline or "good" in that it provides unexpected design opportunities.

Possibilities

Of course the model is of course idealized and needs further testing. After myTHeme, the model has been used in a master Thesis (Eriksson & Ernevi 2004) and in another computer augmented game project, Wizard's apprentice (Peitz et al 2006). This is a game aimed at children playing together with an adult; the children play actively all the time whereas the adult is only active from time to time, enabling him or her to play in a somewhat more preoccupied manner without ruining the experience for the children. It seems as if the model, and in particular the concepts forces, clashes and remnants served to help development. Interestingly the model seems to have worked for two different approaches of gameplay design. In myTHeme, the gameplay evolved from a simpler version to more elaborate one, i.e. at the same pace as the rest of the components. In Wizard's Apprentice, the iterative design process with its forces and clashes resulted in removal (rather than evolution) of some concepts in order to make the current design more consistent and clear, in this way focusing on the central gameplay design goals.

Unlike other game design methods, this model provides a way to deal with interdisciplinary matters and it is iterative to support the general habits of working within many of the disciplines associated with game development. Since the model is event-driven the design process can be started in any design area, and it is easy to follow how the design jumps from one discipline to another, making it possible to backtrack a process if necessary. Of course the model is idealized, and the design

process as described in retrospect may appear more structured than it actually was. It can be difficult to define exactly in what phase the design process the project currently is in. Still, we claim that the Forces-Clashes-Remnants model can support design in four ways.

- It highlights and clarifies interdependencies between design disciplines involved in a project.
- It provides a way to express and document every major design decision and its influences.
- It aids in finding reasons for a broken-down work process.
- It aids in backtracking the reasons for making each design decision, thus making it possible to see if incitements for certain obstructions are obsolete, i.e. if the obstructions (remnants) are obsolete themselves.

Agile gameplay design?

As it turns out, the Forces-Clashes-Remnant model has very much in common with the agile software development models (see page 163). Like agile software development, the focus is not on formalism (i.e. there is no formal format for how to update new designs to the Design Requirements) which in turn means it relies heavily on communication (preferably face-to-face). In addition, the iterative cycles are very short. And just like in agile software development, there is an integrated view that the initial conditions are bound to change during development. Change is embraced, not shunned (Highsmith 2002).

However unlike agile software development the iterative cycles are not necessarily finished increments/functionalities, but might as well just be about creating another non-autonomous part of the process, or supporting something else. Moreover, the Forces-Clashes-Remnants-model has to deal with several very different disciplines, unlike agile software development which is mainly concerned with software only.

Also, many agile software development methods advocate a close contact with the customer; e.g. in Extreme Programming (Beck 1999) a representative customer with the right to make and approve decisions shall, in principle, work side-by-side with the programmers, ready to answer and take customer responsibility for every major design decision. This is not built into the Forces-Clashes-Remnants model, but of course this manner of working can be used side-by-side with the model.

PART III:
DESIGNING FOR FUN?!

DISCUSSING THE DESIGN SPACE

In the facets I have designed what I call interactive entertainment artifacts, i.e. artifacts that I've defined as artifacts who *use computer augmentation to entertain and engage an active user by providing a tempting challenge and/or evoking a strong emotional response*.

According to this definition, the key factors of the design space are (*tempting) challenge*, *emotional response,* a degree of *interaction* and qualities of *computer augmentation* e.g. as hidden complex functionality, a "memory" of what has been done, etc. How do these properties benefit entertainment and can any conclusions regarding the need for interaction and complex computer augmentation be drawn?

Examining some Interactive Entertainment Artifacts

Below, the degree of challenge and emotional response of a set of interactive entertainment artifacts is being discussed. Of course the interactive entertainment artifacts described in the facets are discussed, but as a contrast and perhaps more valid subjects, some commercial artifacts are discussed as well.

Emotional response

Of the artifacts mentioned in the cases, the Iron Horse was probably the one evoking the most emotional response. Just like the commercial products mentioned in the Iron Horse facet, (e.g. Sony's robot dog the AIBO, and the Furbies), it seems that the main characteristic of the Iron Horse is the emotional response it triggers partly by just alluding on a horse, partly by in turn reacting to what the user does. Its animal-like character triggers care, empathy and immersion. In addition, it is autonomous in some aspects, i.e. only greeting its user "at will" which also might strengthen the perception of it having its own personality. None of the products engage by providning challenge to any significant degree (even if the horse jumping application to the Iron Horse clearly is a challenge; it acquires a lot of training to be good at it), although both the Furby and the AIBO can be trained. This of

course implies that if the user does not connect emotionally to the toy, the whole entertainment value is lost.

Interestingly these three artifacts differ a lot in the degrees of interaction possibilities and the complexity of the computer augmentation; whereas the Iron Horse is a rather "simple" device from a computer augmentation point of view, a Furby is more complex and an AIBO hides a lot of functionality, not only in registering and remembering its users behavior, but also in very advanced responses to it, such as fetching a ball, i.e. a delicate mix of software and mechanics/robotics which the other two lack, even if the Furby can do simpler things like wag its ears and close its eyes. In alignment with this, you can actually interact with the Iron Horse in very few ways; mainly by riding it, rearing it and approaching it (hoping to trigger a neigh), whereas you can play simpler games like peek-a-boo with the Furby and more complex games with the AIBO. However in all cases interaction is more or less continuous; if the user stops interacting with the toy, it "stops".

Regardless of these differences, all three artifacts can be used as *tools* for play and imagination, rather than *being* the entertainment. This feature is especially significant for the Iron Horse, which more or less requires some fantasy to come alive. Just as a hammer is a suitable tool for hitting a nail, the Iron Horse can be a useful tool for imagining that you are riding through the countryside. In this, it's only a part of the entertaining experience, not the focus of it, only a means for the entertaining experience, not the experience per se.

As mentioned the above toys rely on their emotion evoking qualities alone. However, other artifacts that are more or less about challenging the users can be strengthened by adding an emotional dimension. Of the facets, the MultiMonsterMania game system is the most obvious example. If the user wants to, he or she can engage in the monsters as beings – rather than as cards – by breeding them and training them, treating them as virtual pets in addition to playing games with them.

Another, more commercial example can be found in for instance Tycoon games. Tycoon games are computer games, in general economic simulation games. The goal is to create and run whatever is simulated in the game (e.g. an amusement park, a zoo, railroads) as successfully as possible, thus becoming a tycoon. Typically, the player has a limited amount of money and/or time and is given a task to solve, e.g. getting more than 1000 visitors per day, or connecting at least ten cities. The player controls some parameters (e.g., which animals or roller coasters to buy, how to landscape, marketing, research etc.) but not all (e.g. what the animals, population or visitors

think, or their mood, the weather, prices etc). Thus, these games provide a significant amount of agency i.e. "the player's delight in having an effect on the electronic world" (Platt 1995, quoting Janet Murray) which is brought by an extensive creative freedom which also in turn results in the user's pride of, and emotional response to what he or she has created. Acknowledging this delight, many of these games offer a lot of not-so-game-dependent creative activities, like naming objects like people and sites in the game, as well as the opportunity to custom-make places and facilities. In *Roller Coaster Tycoon*[65] it is possible to design one's own roller coaster from scratch, if one dislikes the standard ones that are also available. Not only does this provide an tempting challenge, but also emotional sensations like for instance pride when the visitors line up to ride it. Similarly, in *Zoo Tycoon*[66], the player can feel a sense of pride, reward and emotional response when an almost-extinct animal succeeds in breeding, and a litter of very adorable offspring suddenly appears!

Again, in both the MultiMonsterMania case and the Tycoon case, the emotional response is not crucial for the entertainment factor, but the user may choose to engage, perhaps strengthening the experience. Then again, too much emotional response in such games can also be a burden; what if the home-made roller coaster isn't popular, and what if the animals die? Still this built-in risk is a part of the game.

(Tempting) Challenge

Firstly, it must be noted that whenever the aim is to create a tempting challenge, one needs to determine when the challenge is sufficient as well as tempting enough for the chosen target group. Evaluating this is not a part of the tools per se.

Of the artifacts presented in the cases, the two most challenging are the Interactive Quilt and The Hatchery. Interestingly, they contrast in almost all other aspects; The Interactive Quilt is a very simple construction from a computer augmentation point of view, whereas The Hatchery is very complex; in the Interactive Quilt there is only one way to interact, and interaction is sparse. In The Hatchery, interaction is varied and continuous, since the game is time-based. The Interactive Quilt is slow, inviting reflection, whereas The Hatchery is fast-paced and demands constant concentration.

[65] *RollerCoasterTycoon* by Chris Sawyer 1999, published by Hasbro Interactive.

[66] *Zoo Tycoon* by Blue Fang 2001, published by Microsoft Game Studios.

Similar contrasts can be found in the realm of computer games. *Tetris* is one of the most well-known computer games, designed by Alexey Pazhitnov in 1985. A Google search for the words "Tetris" and "online" generates 11 500 000 hits. Thus it is safe to say that *Tetris*, being free (or cheap) and everywhere for two decades, has resulted in so much entertainment, or at least mental distraction, that Hollywood cries its heart out. The general game idea is very simple, it's just about packing falling pieces as tight as possible; full rows disappear. Of course there are numerous variants (cooperative play, competitive play where the rows one player loses appear on the other player's field, *Tetris* with bombs etc…), but the falling pieces remain in all versions. Just like in The Hatchery, players need to think, and think fast, but the number of decisions is limited and some players fall into a kind of trance after having played a while; after much practice playing Tetris the mental challenge is lost to more or less automatic reactions, and the challenge is instead physical; how fast can you be? Interaction is continuous but very limited – all one can do is to rotate pieces, move them sideways, and choose to drop them immediately, and the underlying complexity is thus very small.

A complete opposite to this small and fast paced game is another seminal computer game with several sequels, namely *Myst*[67]. In addition to being a success, it has had many followers, spawning an entire genre of clue-solving games. It was first released in 1993 and has sold over 6 million copies, being a best-seller throughout the 1990ies. It's graphics were state of the art for the time being and it was one of the first first-person adventure-puzzle games. The whole object of the game was to roam five worlds trying to find out what had happened to the missing creator/inhabitant(s). Thus, *Myst* is a set of beautiful and very different places full of mysterious items and mechanical devices that need to be figured out. Each world can be explored in any order but some problems may not be possible to overcome until others have been solved. Also, the worlds lack inhabitants (although there are a few animals), and there is no risk of dying. There is a strong sense of abandonment which is very much supported by the background sounds. It was later followed by a set of sequels which are basically the same but which elaborate the underlying story about a man and his sons, and of a people that create worlds. Thus, the underlying logic of the game is rather complex, although the interaction types are rather limited; players can point out in which direction to move and whether they wish to move/touch movable objects. The game is interaction-driven, but just like the Interactive Quilt simple interactions are sometimes intervened with rather long

[67] *Myst* by Robyn and Rand Miller et al, 1993, published by Cyan Inc.

Figure 34: Screendump from Myst.

times of reflection. The entire challenge in *Myst* is about problem-solving; since it is only about exploration and finding out it can actually be seen as a slow technology artifact, a very successful one. This is very significant in that once it is solved, it is *solved*. It has hardly no replayability-value at all. Unlike Tetris, or The Hatchery it is slow, intriguing but in a calm way.

Returning to the Tycoon games (as being described in the previous section) they are a mixture of these two extremes. They have a lot of hidden complex functionality, but unlike *Myst*, they do provide a lot of possible interactions. Also unlike *Myst*, and like *Tetris*, these games are time-based but – as a compromise – time can be stopped to allow for reflection, planning and some interaction. Whereas Myst typically is played only once, and one sequence of play isn't necessarily "better" than any other, optimizing is important in Tycoon games, sometimes triggering an entire restart of a challenge in order to solve it faster or more effective. Tycooning can be practiced, and so can *Tetris* – but there is no point in practicing how to play *Myst*.

Intermediates

Above we have discussed interactive entertainment artifacts that are strong in either intellectual challenge or that evoke emotional response. What about those interactive entertainment artifacts that only reach a moderate degree of challenge and/or emotional response. Are they less successful?

MyTHeme is such a game; the computer augmentation is moderate and the interaction possibilities are restricted by the cards at hand, but this restriction is part of the challenge of the game. Also, the creative part of the game, the creation of the story, invites to an emotional response to the hero, or the creatures in the story. However, in my opinion, neither is significant enough to create a real "hit". Pretty much the same can be said about the commercial computer augmented board game King Arthur[68]. In this game the players wander in a landscape – the board – and run into various encounters like robbers, dragons, friendly villagers, the Lady of the Lake etc. What happens in a certain spot is randomized by the game, or semi-randomized. It also keeps track of the score and certain player-specific information. The main output is sound. Computer augmentation is rather simple and the interaction possibilities are few, since the moves and actions are limited by the placement on the board. The challenge – to collect money via quests in order to buy a horse, armour and a sword – isn't sufficient, and there is hardly any emotional response (and no one is aimed for indeed). As for its commercial success I haven't found any sales numbers but it has not been nominated for any large awards.

Then again my argumentation against these games is based on my adult gamer opinions of what a good game is, and perhaps the challenges featured in the above games are sufficient for someone else.

Conclusions?

Can any conclusions be drawn by this; is there a formula for fun? Unfortunately not. There is only an indication; Less is *not* more – more is better! A well thought through challenge or strong means to create emotional response are better than less elaborate attempts – and combining them can increase the fun factor even more, as in creating the Horse Jumping application to the Iron Horse.

As perhaps expected neither a high degree of computer augmentation or complex interaction is necessary to create an interactive entertainment artifact; hardly surprising since there are several non-computer augmented yet still entertaining things in the world. Thus, computer augmentation and the interaction means it enables shall *support* the design of the interactive entertainment artifact, not *steer* it. Obviously it is the creation of the tempting challenge or the emotional response that *is* the challenge.

[68] *King Arthur* by Reiner Knizia 2003, published by Ravensburger.

The Tools and the Space

Now, making the claim that emotional response or tempting challenge are the key properties of an interactive entertainment artifact, we can return to the main question: *How can we go about to create interactive entertainment artifacts in general, and how do we design for emotional response and/or tempting challenge in particular?*

How the five tools are applied, as well as their pros and cons, are discussed in depth in each facet. However there are still some more over-arching questions left to discuss – which tool shall be used for what when and can they be combined to create an even more successful design?

Since the Forces-Clashes-Remnants method is probably – albeit not yet tested – applicable on any multidisciplinary design project, and the design of an interactive entertainment artifact per definition is multidisciplinary (interaction design and programming being two inherent disciplines, plus the project-specific ones), it can be used in any project where an interactive entertainment artifact is developed. In this, it is a low-level method, since it only aids in creating the artifact, not the important properties (emotional response and tempting challenge respectively). In this, it is similar to the game mechanics for computer augmentation listed in Facet No. 3. These too, are a rather low-level since using them does not automatically bring strong qualities of emotional response or tempting challenge with them. Nevertheless most of them can be used to increase or decrease the mental challenge to a "suitable level", i.e. make the challenge tempting by either clarifying or keeping information or hiding or obscuring it; as you might remember most of the novel mechanics are information-oriented in some form. Some mechanics are more targeted towards this than others, i.e. mechanics like Computerized Clues and Secret Partnerships can contribute very much to a tempting challenge whereas Active Dice or Active Tiles have a more supporting role, enabling other rules or mechanics that in turn make the game more challenging.

In comparison with the game mechanics the Game Design Patterns are a more elaborate tool, partly because the collection is larger but mostly because here, we find lots of patterns with the explicit goal to increase challenge or emotional response. For instance, emotional response can be increased by using patterns like *Immersion, Creative Control, Characters, Identification, Character Development* etc. Some kind of challenge is of course present in any game, but can be moderated with patterns considering the quality and distribution of information, rewards and penalties,

action control, resource control etc. And, last but not least, Game Design Patterns and the game design pattern framework is of course also a low-level tool for creating interactive entertainment artifacts in the form of games.

On the high-level end of the spectra we find Animal Expression Transfer and Slow Technology which are both highly targeted towards creating emotional response and tempting challenge respectively. Of these, Animal Expression Transfer is probably more suited for artifacts than games, whereas Slow Technology aspects can be used in many kinds of projects to increase the challenge. However, the use of these tools are not elaborate as the Game Design Patterns or the Forces-Clashes-Remnants model, perhaps because they are approaches depending highly on context and goal, rather than hands-on tools.

Also, the tools differ in how easy they are to adapt and learn. Both Expression Transfer and Slow Technology are relatively easy to understand and rather easy to have a go at. However using some practice, user testing and careful analysis in various stages of development will definitively improve the outcome. In contrast, using game mechanics is very concrete, but limited since there are rather few. When using those, the issue is more about implementing them than anything else. As for the Forces-Clashes-Remnants model, it is rather easy to understand and use, but it requires that all designers stick to it, constantly updating the Design Requirements and discussing new requirements and potential clashes. Thus they require a commitment from the whole design team. The same can be said about the Game Design Patterns; if they are to be used effectively, the whole team must understand and use them. Also, as mentioned, the Game Design Patterns is the tool that requires most effort in getting to know it, but as a compensation it is very versatile and can aid many aspects of design.

Another comment is that some of the tools can be used for other types of projects as well, i.e. Expression Transfer or Animal Expression Transfer can be used as design methods to create novel artifacts in general, focusing on something else than emotional response, and the Forces-Clashes-Remnants model can be used for any multidisciplinary project. Nevertheless they are useful tools in the context of designing interactive entertainment artifacts.

Conclusions: when and where!

If the project is the right one, all tools can be combined, supporting different aspects of the design. If, for instance the project is to create a real-life pervasive game which is played, say, on the streets using interactive animals, then game mechanics, Game Design Patterns and Slow Technology may aid in creating the game rules and a tempting challenge, whereas Animal Expression Transfer could be used to create an emotional response to the game pieces, i.e. the animals, in order to make the game even more interesting. The development in all of the integral design disciplines would then be steered by the Forces-Clashes-Remnants model.

To summarize:

- In the general design of interactive entertainment artifacts, the Forces-Clashes-Remnants method and Slow Technology can be used.
- When aiming for emotional response, Animal Expression Transfer and Game Design Patterns can be used.
- When aiming for a tempting intellectual challenge, Slow Technology and Game Design Patterns can be used.
- When designing games in particular, Game Design Patterns and game mechanics can be used.

Of these the Forces-Clashes-Remnants model and the game mechanics are low level tools (i.e. tools targeted towards creating the artifact itself rather than the dimensions of emotional response and/or tempting challenge). Slow Technology and Animal Expression transfer are in turn high level tools and Game Design Patterns can be found somewhere in-between depending on use.

BABBLING INCOHERENTLY? PERSONAL REFLECTIONS

Of course it is as obvious to me as to anyone else that none of the design tools that are proposed in this book have been evaluated in a formal, or extensive, way and neither have the cases that exemplify them. There are several reasons for this.

As for the cases, the most prominent reason is that there was simply no time or resources set aside for extensive evaluation of in any of the projects. The second is that the goals that were set could sometimes be evaluated "by checklist", e.g. the Interactive Quilt goal to build a working prototype which was easily answered with a yes. Since none of the designs were commercial projects the goals set were explorative, rather than explicit. "*Can we do this?*" e.g. map fabrics to music or create a narrative game or put together a framework for game design, or "*What happens if we…*" e.g. merge the qualities of a horse and a bike/ computer augment a card game. Above all, these projects are about lessons learned throughout the design procedure, the Interactive Quilt-project and the Forces-Clashes-Remnant model being the most prominent examples. The third is that of all qualities emotional ones are the hardest to measure; how do you evaluate fun? Also, the outcome of these results are not as interesting as when evaluating for instance the ergonomic aspects of an office chair, because nothing can be considered being fun by everyone, and therefore it is not interesting to even try and please everyone, as might be the case if running a huge software program project creating a word processor.

As for the tools, they are even harder to evaluate since the effect of using them is sometimes invisible and thus hard to measure. Also a method might work well in a specific design situation and/or in a specific group whereas it is totally useless under other circumstances. May things affect a design process; it can be the task itself, it can be the methods, it can be unforeseen demands or possibilities, it can be a sudden synergy between team members, or perhaps conflicts. Above all the skills, luck and creative sparks of the designers matter.

I'm not saying that I wouldn't like to evaluate the proposed design tools further, I'm just saying that this too is a huge task. I.e. it would be very interesting indeed to use the Forces-Clashes-Remnants method formally in a series of projects to see under which circumstances it seems to work best, but this includes running some five-six multidisciplinary game design projects, which isn't exactly done over night.

The Reflective Practitioner

Still, both my own integrity and the scientific practice requires me to try and analyze the pros and cons of the methods I propose, as well as the outcome of the projects, which has been done in each facet respectively. This has been an introvert process. In this, I and other designers can be classified with the pertinent words of Donald Schön (1983); we are "*reflective practitioners*". Schön discusses the current positivistic view of technical rationality, saying that it is focused on *problem solving* whereas it ignores the aspect of *problem setting,* which he describes as "the process by which we define the decision to be made, the ends to be achieved, the means which may be chosen" (p 40), claiming that in the real world, problems are not clear textbook problems with all matters defined; they are puzzling, multi-faceted and sometimes hard to distinguish and delimit. The process of doing this is what Jones (1992) refers to when he talks about the designer as a self-organizing system. In doing this, the designer is reflecting-in-action; he or she is "learning by doing" or is "feeling the way about" or is engaging in "trial-in-error". All of these activities demand that the designer is alert, analyzing, open-minded and ready to act on whatever the analysis results in.

Since I see myself as a reflective practitioner, the cases in this book can be seen as descriptions of my (and my colleagues) conscious reflections made in the action of designing whatever was to be designed. By making these reflections the chosen design tool is analyzed and evaluated too, in the same way.

Being a Designer – a Matter of Art, Skill or just Having the Right Neurosis?

It is my true belief that design is not science. To me, design is a skill, closer to art than to engineering[69]. Jones (1992, p. 45 - 57) speaks of three ways to regard the designer. The designer can be seen as a magician or a black box; demands are input and via various magic processes an output in the form of a design is created; this is the creativity-oriented view. A rational viewpoint is that the designer is instead a glass box, through which the way the input is processed to output is clearly viewable, being an explained, logical, rational process. And last but not least, the designer can be seen as "a self organizing system" that is capable of finding ways through the unknown territory of each new design task. Personally, I believe that no design

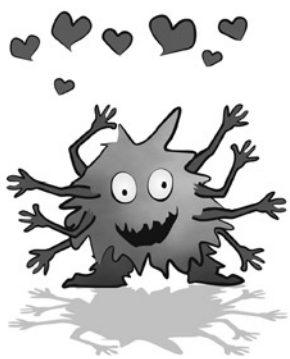
[69] Jones (1992) claims it to be an even mixture of art, science and mathematics

process ever can be carried through without at least one magic, unexplainable black box spark. Jones describes this creative spark in wonderful way, saying:

> *"Perhaps it is not so much a question of being creative or uncreative but of being blessed or cursed with the right blend of experience and neurosis to be able and willing to resolve the particular kind of conflict that exists within a given design situation."*
>
> *John Chris Jones "Design Methods" (1992), p 47*

My point here is that even though the tools described in the upcoming facets can be seen as ways to support glass-box design and self-organizing designer systems, all of them contain elements of black box design. Thus the projects and tools described below are not absolute recipes and certainly not the absolute truth; they are meant to be *inspiration*. Use whatever you ideas you like, tweak them, twist them, and turn them upside down if you believe that will aid your design process. And let me know how it worked!

Sus Lundgren, October 2006



REFERENCES

Abowd, G. D. & Mynatt, E. D. (2000) *Charting Past, Present, and Future Research in Ubiquitous Computing*. In: ACM Transactions on Computer-Human Interaction, Vol. 7, No. 1, March 2000, Pages 29 – 58.

Alexander, C., et al (1977) *A Pattern Language,* Oxford University Press, New York, ISBN 0-19-501919-9.

Bersak, D., et al. (2001) *Intelligent Biofeedback using an Immersive Competitive Environment.* Paper at the Designing Ubiquitous Computing Games Workshop at UbiComp 2001, Atlanta, GA, USA.

Björk, S. et al. (2001) *Pirates! - Using the Physical World as a Game Board.* In: The Proceedings of Interact 2001 IFIP TC.13 Conference on Human-Computer Interaction.

Björk, S., Holopainen, J. (2005) *Patterns in Game Design.* Charles River Media, ISBN 1-58450-354-8.

Björk, S., & Holopainen, J. (2003) *Describing Games - An Interaction-Centric Structural Framework.* In: Copier, M. & Raessens, J. (Eds.) (2003) Level Up - CD-ROM Proceedings of Digital Games Research Conference 2003, Utrecht, The Netherlands, 4-6 November 2003.

Björk, S., Holopainen, J., Ljungstrand, P., and Åkesson, K.P (2002) *Designing Ubiquitous Computing Games - A Report from a Workshop Exploring Ubiquitous Computing Entertainment.* In: Journal of Personal and Ubiquitous Computing, Volume 6 (2002), 6th issue: Special issue on Ubiquitous Gaming.

Björk, S., Lundgren, S. and Holopainen, J. (2003) *Game Design Patterns* In: Proceedings of Level Up - 1st International Digital Games Research Conference 2003, Utrecht, the Netherlands.

Borchers, J. (2001) *A pattern approach to interaction design*, John Wiley & Sons, ISBN: 0 471 49828 9.

Buchanan, R. (2001) *Design Research and the New Learning*. In: Design Issues, 17:4, pp. 3-23.

Caillois, R. (2001). *Man, Play and Games*. University of Illinois Press. ISBN 025207033X.

Cook, D. (2002) *Evolutionary Design. A practical process for creating great game designs*, [online]. Available from: http://www.gamedev.net/reference/design/features/evolution/ (retrieved 2006-05-03).

Cooper, A. and Reimann, R. (2003) *About Face 2.0: The Essentials of Interaction Design*, Wiley; 1st edition, 2003, ISBN: 0764526413.

Costikyan, G. (1994) *I Have No Words & I Must Desig*n. In: Interactive Fantasy magazine, 1994, can be downloaded at: http://www.costik.com/nowords.html (Retrieved 2006-06-06).

Costikyan, G. (2002) *I Have No Words & I Must Design*. In: Conference Proceedings of Computer Games and Digital Cultures, pp. 9-33, Tampere University Press. Note that this is a rewritten version of the 1994-article.

Crawford, C. (1984) *The Art Of Computer Game Design: Reflections Of A Master Game Designer,* Osborne/McGraw-Hill 1984, ISBN: 0881341177.

Crawford, C. (2002) *The Art of Interactive Design: A Euphonious and Illuminating Guide to Building Successful Software*, No Starch Press Inc., San Francisco 2002, ISBN: 1886411840.

Dunne, A. (1999) *Hertzian Tales; Electronic products, aesthetic experience and critical design*. RCA CRD Research publications; London, UK, 1999.

Erickson, T. (2000) *Lingua Francas for Design: Sacred Places and Pattern Languages*. In: The Proceedings of DIS 2000 (Brooklyn, NY, August 17-19, 2000). New York: ACM Press, 2000, pp 357-368.

Eriksson, D. & Ernevi, A. (2004) D*evelopment of Electronics for Interactive Toys on the Concept of Free Play.* Masters Thesis in Electrical Engineering, Department of Electrical Engineering, Chalmers University of Technology, Göteborg, Sweden.

Fjellman, E. & Sjögren, J. (2000) *Interaktiv underhållning inför framtiden*. In: Telematik 2004. KFB-rapport 2000:10 & TELDOK Rapport 133. TELDOK och KFB – Kommunikationsforskningsberedningen, Stockholm.

Fullerton, T., Swain, C., & Hoffman, S. (2004) *Game Design Workshop: Designing, Prototyping, and Playtesting Games*. CMP Books, 2004. ISBN 1578202221.

Fulton, B. (2002) *Beyond Psychological Theory: Getting Data that Improve Games.* In: Proceedings of Game Developer's Conference 2002, San Jose, CA, USA, March, 2002. Available from http://www.gamasutra. com/gdc2002/features/ fulton/fulton_01.htm (retrieved 2005-02-14).

Gaver, W., Beaver, J. and Benford, S. (2003) *Designing design: Ambiguity as a resource for design*. In: Proceedings of the conference on Human factors in computing systems, April 2003. ACM press.

Gamma, E., Helm, R., Johnson, R., Vlissides, J. (1994) *Design Patterns: Elements of Reusable Object-Oriented Software,* Addison-Wesley, ISBN 0201633612.

Gorbet, M., Orth, M., and Ishii, H. (1998). *Triangles: Tangible Interface for Manipulation and Exploration of Digital Information Technology.* In: Proceedings of CHI 1998, ACM Press.

Graham, I. (2003) *A Pattern Language for Web Usability*, Pearson Education; 1st edition (January 31, 2003), ISBN 0201788888.

Haahr, M., Lundgren, S. and Reijers, N. (2002) *Multi Monster Mania.* A part of *Designing Ubiquitous Computing Games - A Report from a Workshop Exploring Ubiquitous Computing Entertainment* by Björk, S., Holopainen, J., Ljungstrand, P. & Åkesson K-P. In: Personal and Ubiquitous Computing January, Volume 6, Issue 5-6, pp. 443 - 458.

Hallnäs, L., Jaksetic, P., Ljungstrand, P., Redström, J., & Skog, T. (2001) *Expressions; Towards a Design Practice of Slow Technology*. In: Proceedings of Interact 2001, IFIP TC.13, 2001, Tokyo, Japan.

Hallnäs, L., and Redström, J. (2001) *Slow Technology – Designing for Reflection.* In: Personal and Ubiquitous Computing January 2001, Volume 5 Issue 3.

Hallnäs, L. & Redström, J. (2002) *Abstract Information Appliances; Methodological Exercises in Conceptual Design of Computational Things*. In: ACM SIGCHI DIS 2002, pp. 105-116. ACM Press.

Huizinga, J. (1986). *Homo Ludens: A Study of the Play-element in Culture.* Beacon Press.

Ilstedt Hjelm, S. & Brovall, C. (2000) *Brainball – using brain activity for cool competition*. Demonstration at NordiCHI 2000 — Design versus Design.

Ishii, H. and Ullmer, B. (1997) *Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms.* In: Proceedings of CHI 1997, ACM Press.

Jones, J. C (1992). *Design Methods*, 2nd Edition, John Wiley & Sons, 1992. ISBN: 0471284963.

Kahn, P. H., Jr., Friedman, B., & Hagman, J. (2002). *"I Care About Him as a Pal": Conceptions of Robotic Pets in Online AIBO Discussion Forums.* In: CHI 2002 Extended Abstracts of the Conference on Human Factors in Computing Systems (pp. 632-633). New York: ACM Press.

Kahn, P., Friedman, B., Perez-Granados, P. R., Freier, N. G (2004) *Robotic Pets in the Lives of Preschool Children*. In: The proceedings of CHI 2004.

Kramer, W. (2001) *What Makes a Game Good?* [online]. Available from: http://www.thegamesjournal.com/articles/WhatMakesaGame.shtml (retrieved 2006-05-25).

Kreimeier, B. (2002). *The Case For Game Design Patterns*. [online]. Available from: http://www.gamasutra.com/features/20020313/kreimeier_03.htm.

Landin H., Lundgren S., Prison J (2002) *Aesthetic artefacts: The Iron Horse: a sound ride*. In: Proceedings of the second Nordic conference on Human-computer interaction October 2002, Aarhus, Denmark, October 19-23, ACM Press.

Laurel, B. (1993). *Computers as Theatre.* Addison-Wesley, Reading, Massachusetts, ISBN 0201550601.

Lundgren, S. (2002) *Joining Bits and Pieces - How to make Entirely New Board Games using Embedded Computer Technology.* M.Sc. Thesis in Interaction Design at the Department of Computing Science, IT University of Göteborg, Göteborg University and Chalmers University of Technology; supervised by Björk, S.

Lundgren, S., Johansson S., Nilsson F., Stenberg P., and Thorin, P. (2003) *Mapping Fabrics to Music: Lessons Learned.* In: Proceedings of The ninth IFIP TC13 international conference on Human-Computer Interaction (Interact 2003), Zürich, Switzerland.

Lundgren, S. and Björk, S. (2003). *Game Mechanics: Describing Computer-Augmented Games in Terms of Interaction*. In: Proceedings of the 1st International Conference on. Technologies for Interactive Digital Storytelling and Entertainment (TIDSE 2003), Darmstadt, Germany.

Lundgren, S. and Björk, S. (2005) *Forces, Clashes and Remnants: a Model for Event-Driven Iterative Multidisciplinary Game Design*. In: Proceedings of ACM SIGCHI International Conference on Advances in Computer Entertainment Technology (ACE 2005).

Löwgren, J. and Stolterman, E. (1997) *Design av Informationsteknik – Materialet utan egenskaper.* Studentlitteratur 1997, ISBN 91-44-00681-0.

Löwgren, J. and Stolterman, E. (2004) *Thoughtful Interaction Design: A Design Perspective on Information Technology.* MIT Press 2004, ISBN 0262122715.

Mandryk, R. L., Maranan, D. S., Inkpen, K. M. (2002) *False Prophets: Exploring Hybrid Board/Video Games*. In: Proceedings of CHI 2002. ACM Press.

Manovich, L. (2001) *The Language of New Media*. MIT Press, 2001, ISBN 0262632551.

McGee, K, and Harup, A. (2003) *Contact Expressions for Touching Technologies.* In: Proceedings of COSIGN 2003, the 3rd International Conference on Computational Semiotics for Games and New Media, Middlesbrough, UK, September 9-12, 2003.

Melson, G. F., Kahn, P., Beck, A. M., Friedman, B., Roberts, T., and Garret, E. (2005) *Robots as Dogs? – Children's Interactions with the Robotic Dog AIBO and a Live Australian Shepherd*. In: Proceedings of CHI 2005.

Monö, R. (1997) *Design for Product Understanding*, Liber AB, Stokholm 1997, ISBN: 914701105X.

Murray, J. (1998). *Hamlet on the Holodeck: The Future of Narrative in Cyberspace*. MIT Press; ISBN: 0262631873.

Parlett, D. (1999). *The Oxford History of Board Games*, Oxford University Press, ISBN 0192129988.

Peitz, J., Björk, S. & Jäppinen, A. (2006) *Wizard's Apprentice - gameplay-oriented design of a computer-augmented board game*. In: Proceedings of ACM SIGCHI International Conference on Advances in Computer Entertainment Technology (ACE) 2006.

Platt, C (1995) *Interactive Entertainment: Who writes it? Who reads it? Who needs it?* In: WIRED Magazine, Issue 3.09, September 1995. Can be downloaded at http://www.wired.com/wired/archive/3.09/interactive.html (retrieved 2006-07-18).

Pinker, S. (1997) *How The Mind Works*, Penguin Books, London 1998, ISBN 0-713-9911305.

Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S. and Carey, T. (1994) *Human-Computer Interaction*, Addison-Wesley Publishing Company, ISBN 0201627698.

Preece, J., Rogers, Y., Sharp, H (2002) *Interaction Design*, John Wiley & Sons, New York, USA. ISBN: 0-471-49278-7.

Propp, V. (1984) *Theory and History of Folklore*. University of Minnesota Press, 1984, ISBN 0816611823.

Salber, D. and Coutaz, J.(1993) *Applying the Wizard of Oz Technique to the Study of Multimodal Systems.* In: Proceedings of East-West HCI conference '93. ACM Press.

Salen, K. & Zimmerman, E. (2004) *Rules of Play: Game Design Fundamentals*. The MIT Press, 2004, ISBN: 0262240459.

Schneider, J. & Kortuem, G. (2001). *How to Host a Pervasive Game - Supporting Face-to-Face Interactions in Live-Action Roleplaying.* Position paper at the Designing Ubiquitous Computing Games Workshop at UbiComp 2001, Atlanta, GA, USA, September 30, 2001.

Schön, Donald (1983). *The Reflective Practitioner: How Professionals Think in Action*. Basic Books Inc, Reprinted by Ashgate Publishing Ltd, Aldershot, England. ISBN 1857423194.

Spector, W. (1999). *Remodeling RPGs for the New Millennium*. [online] Available from: http://www.gamasutra.com/features/game_design/19990115/remodeling_01.htm.

Starner, T. et al. (2000) *MIND-WARPING: Towards creating a compelling collaborative augmented reality game.* In: Proceedings of Intelligent User Interfaces (IUI) 2000 pp 256–259.

Streitz, N.A., Geißler, J., Holmer, T., Konomi, S., Müller-Tomfelde, C., Reischl, W., Rexroth, P., Seitz, P., and Steinmetz, R. (1999) *i-LAND: An interactive Landscape for Creativity and Innovation*. In: Proceedings of CHI '99 , ACM Press, New York.

Strommen, E. (1998) *When the Interface is a Talking Dinosaur.* ACM Press/ Addison-Wesley Publishing Co. New York, NY, USA, pages: 288 - 295 Series-Proceeding- Article.

Tamura, H. (2000) *What Happens at the Border Between Real and Virtual Worlds The MR Project and Other Research Activities in Japan*. Invited talk at ISAR 2000.

Tidwell, J. (2005) *Designing Interfaces. Patterns for Effective Interaction Design*, O'Reilly Media 2005, ISBN 0596008031.

Wada, K., Shibata, T., Musha, T., and Kimura, S. (2005) *Effects of Robot Therapy for Demented Patients Evaluated by EEG*. In: Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems.

Weiser, M (1993) *Some Computer Science Issues in Ubiquitous Computing*. In: Communications of the ACM, Back to the Real World, Special issue on Computer Augmented Environments, Vol 36, No 7, pp 75-84, ACM Press.

Weiser, M. and Brown, J. S. (1996) *The Coming Age of Calm Technology* [online]. Available from: http://www.ubiq.com/hypertext/weiser/acmfuture2endnote.htm (Retrieved 2006-07-17).

Winograd, T. (1997) *From Computing Machinery to Interaction Design*. In: Denning. P. and Metcalfe, R. (eds.), Beyond Calculation: The Next Fifty Years of Computing, Springer-Verlag, pp. 149-162. Also available at: http://hci.stanford.edu/~winograd/acm97.html (Retrieved 2006-07-17).

Wright, W. (2003). *Dynamics for Designers*. Presentation at Game Developers Conference 2003. Presentation available at http://www.gdconf.com/archives/2003/Wright_Will.ppt.