University of Ljubljana
Faculty of computer and information science

Jakob Marovt

Gamification of software applications

diploma thesis

undergraduate university study programme
computer and information science

Advisor: assist. prof. Peter Peer

Ljubljana, 2012

Univerza v Ljubjani
Fakulteta za računalništvo in informatiko

Jakob Marovt

Igrifikacija programske opreme

diplomsko delo
na univerzitetnem študiju

Mentor: doc. dr. Peter Peer

Ljubljana, 2012

Univerza *v Ljubljani*
Fakulteta *za računalništvo in informatiko*

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 01870/2012
Datum: 14.09.2012

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **JAKOB MAROVT**

Naslov: **IGRIFIKACIJA PROGRAMSKE OPREME**
GAMIFICATION OF SOFTWARE APPLICATIONS

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

Igrifikacija je ena izmed trenutno najbolj popularnih metod oziroma tehnik za zagotavljanje dolgoročne uporabniške izkušnje. V uvodu razložite razliko med igro, igrifikacijo in pomembnost slednje v današnjem času. Opišite pomen ogrodja MDA za igrifikacijo. Nato uporabite ogrodje na konkretnem primeru igrifikacije programske opreme. V nadaljevanju naredite pregled obstoječih igrifikacijskih platform ter izpostavite znane primere uporabe igrifikacije.

Mentor:
doc. dr. Peter Peer

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:
prof. dr. Nikolaj Zimic

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Jakob Marovt, z vpisno številko **63050171**, sem avtor diplomskega dela z naslovom:

Igrifikacija programske opreme

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Petra Peera,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 18. septembra 2012 Podpis avtorja:

Za mentorstvo, vodenje in pomoč se zahvaljujem doc. dr. Petru Peeru.

Za podporo pri izdelavi diplomskega dela in v času celotnega študija se zahvaljujem vsej družini, ki mi je vselej stala ob strani in nudila več kot potrebno motivacijo.

Contents

Razširjeni povzetek

Načrtovalci programske opreme vedno stremijo k čim boljši uporabnosti svojih produktov. Dandanes velika večina programske opreme deluje preko interneta; bodisi v obliki spletnih strani oz. aplikacij bodisi v obliki nameščenih aplikacij, ki pa za povezovanje in sinhroniziranje podatkov uporabljajo internet. Slednja oblika se vse večkrat uporablja pri dandanes zelo popularnih mobilnih aplikacijah. Konstantna sinhronizacija podatkov omogoča načrtovalcem programske opreme zelo natančno spremljanje in diagnosticiranje uporabe aplikacij. In v javnosti se vse večkrat pojavljajo globalne metrike uporabe in frekvence vračanja uporabnikov, ki so zelo zaskrbljujoče za načrtovalce in razvijalce programske opreme. Ena izmed raziskav je pred kratkim pokazala, da v povprečju več kot 26 odstotkov uporabnikov preizkusi in uporabi aplikacijo zgolj enkrat. In trend kaže, da se ta številka povečuje iz meseca v mesec, predvsem na področju aplikacij namenjenih končnim uporabnikom za neprofesionalne namene. Razlog za to je predvsem zasičenost uporabnikov z vedno novimi produkti in posledično pomanjkanje časa za učenje uporabe novih aplikacij.

To pomeni, da moramo biti načrtovalci aplikacij vedno bolj pozorni na to, kakšne vzvode in tehnike uporabljamo za pridobitev zaupanja uporabnikov in spreminjanja njihovih navad. S temi problemi se ukvarja dandanes zelo veliko število raziskovalcev, predvsem s področja uporabniške izkušnje in obnašanja uporabnikov. Naša teza je, da bi moral vsak načrtovalec programske opreme poznati vsaj osnove igrifikacije, saj bi to rezultiralo v boljši uporabniški izkušnji. Posledica le-tega bi bila večja uporaba programske

opreme in manj potrošenega in zavrženega dela razvijalcev.

Igrifikacija je ena izmed trenutno najbolj popularnih metod oz. tehnik za zagotavljanje dolgoročne uporabniške izkušnje. Kot lahko sklepamo že iz imena samega, izhajajo tehnike igrifikacije s področja iger oz video iger. Razlog se skriva v večni privlačnosti iger in času, ki ga ob igranju preživijo uporabniki le-teh. Metrike uporabe in frekvence vračanja uporabnikov iger so nekajkrat višje od tistih, ki jih lahko opazujemo pri drugih vrstah programske opreme. Izraz igrifikacija se prvič uporabi v strokovni literaturi leta 2008. Tedaj so številni raziskovalci začeli razmišljati globlje o tem, kaj dela igre tako zanimive in privlačne za uporabnike ter katere elemente iger bi bilo mogoče uspešno prenesti v načrtovanje aplikacij zunaj konteksta iger.

Čeprav je lahko to zelo "mehko"področje, so raziskovalci našli določene skupne točke in mi smo se za reševanje našega problema odločili uporabiti ogrodje MDA za načrtovanje igrifikacijske rešitve. Kratica MDA označuje tri glavne komponente tega ogrodja: igralne mehanike, igralne dinamike, ki se vršijo tekom uporabe in pa estetiko. Igralne mehanike so tista komponenta, na katero ima načrtovalec programske opreme direkten vpliv: razvijalec namreč uporabi in vstavi posamezne mehanike v uporabniško izkušnjo, odvisno seveda od namena in načina uporabe aplikacije. Najbolj znane in najpogosteje uporabljene mehanike so: različne oblike točk, privajanje uporabnika na aplikacijo, stopnje, lestvice, dosežki, prilagajanje izkušnje po meri uporabnika, izzivi ter zanka družbene angažiranosti. Igralne dinamike tvorijo okolje odzivov mehanik na uporabnikovo interakcijo z aplikacijo. In kot zadnja komponenta, estetika, tvori vse tiste vzvode, katere so raziskovalci identificirali kot ključne za uporabnikovo zadovoljstvo in angažiranost med uporabo aplikacije.

Mi smo se v našem delu spoprijeli z izzivom igrifikacije dveh povsem ločenih aplikacij s pomočjo ogrodja MDA. Prva je bila mobilna družbena aplikacija za izmenjavo fotografij Eeve. Za ta problem smo naredili dve iteraciji rešitev. Prva iteracija vključuje nekaj osnovnih mehanik in zelo spominja na implementacijo igrifikacije v znani mobilni aplikaciji Foursquare. Druga

iteracija je globlja in temelji bolj na intrizinčni motivaciji uporabnikov in vključuje le nekaj bolj subtilnih igralnih mehanik. Mehanike so bile zastavljene tako, da bi spodbujale uporabnike k fotografiranju, jih nagrajevale za originalne in družbeno popularne zbirke fotografij in vedno znova nudile možnost izziva. Na žalost te implementacije ni bilo mogoče preizkusiti v javni različici aplikacije, temveč le v testni fazi. Podjetje, ki je glavni razvijalec aplikacije, se je namreč odločilo za globalno spremembo strategije.

Naslednji projekt pri katerem smo uporabili ogrodje MDA je bila risalna aplikacija Psykopaint. S to aplikacijo smo se ukvarjali več časa in uporabili praktično kompleten sistem igralnih mehanik. Skupaj z ekipo Psykosofta smo ta sistem tudi implementirali in je trenutno v uporabi v aplikaciji, ki jo mesečno uporablja več kot 1.000.000 uporabnikov. Igrifikacija je nedvomno pomagala k nadaljni rasti uporabe in predvsem vidno rezultirala v večji frekvenci vračanja uporabnikov. Hkrati imamo še nekaj predlogov za nadgradnjo sistema, ki bodo verjetno zaživeli v bližnji prihodnosti.

S popularizacijo igrifikacijskih metod se je seveda razvil tudi celoten ekosistem platform, ki podjetjem nudijo hitre rešitve za uvedbo nekaterih osnovnih igralnih mehanik v njihove aplikacije. Od leta 2010, ko je ta trg popolnoma oživel in se je nenadoma pojavilo kar nekaj tovrstnih platform, se je trg že precej konsolidiral. Mi smo tako identificirali 3 ključne platforme na trgu; eno od njih, Badgeville, smo tudi sami uporabili na projektu in jo podrobneje opisali. Drugi dve BigDoor in Bunchball pa smo prav tako vzeli pod drobnogled, opisali njihov produkt in izpostavili razlike oz. podobnosti med njimi.

Za vzpostavitev kredibilnosti igrifikacije smo podrobneje analizirali tudi aplikacije, ki v tem trenutku že globalno uspešno izrabljajo pozitivne učinke igralnih mehanik. Podrobneje smo pregledali več kot 30 tovrstnih aplikacij in izpostavili nekaj najbolj pomembnih primerov s področij produktivnosti (tako osebne kot na ravni celotnih podjetij), zdravstva, izobraževanja in drugih. Našli smo tudi nišo, ki bo mogoče v prihodnosti imela največji vpliv na naša življenja – aplikacije, ki izkoriščajo moč množice uporabnikov. Ti

uporabniki opravljajo majhna dela v velikem globalnem projektu. Zelo znan primer takšne aplikacije je Fold.it, ki je uporabila enostavne igrifikacijske tehnike za rešitev znanstvenega problema iskanja strukture enega izmed encimov virusa HIV. Znanstveniki so se s problemom ukvarjali več kot 10 let, Fold.it-u in njegovim uporabnikom pa je uspelo primer rešiti v obdobju 3 tednov.

Upamo, da bo naše diplomsko delo opogumilo in populariziralo nekatere metode igrifikacije med študenti računalništva, saj je nenazadnje uporabnost njihov aplikacij ključna za uspešnost našega dela. Tako si želimo, da te metode ne bi ostale zgolj v domeni psihologov in tistih, ki se profesionalno ukvarjajo z uporabniško izkušnjo. Načrtovalci aplikacij imajo dandanes verjetno najboljše možnosti za prodor na globalni trg, vendar je z nasičenostjo le-tega potrebno razviti ne le s tehničnega vidika odličen produkt, temveč tudi, oz. prvenstveno, uporabniku prijazen in zasvoljiv produkt. Igrifikacija se tako po našem mnenju uvršča med ene izmed temeljnih znanj, ki bi jih moral osvojiti prav vsak načrtovalec programske opreme.

Ključne besede

Igrifikacija, igre, uporabnost aplikacij, angažiranost uporabnikov, metrike uporabe in frekvenca vračanja uporabnikov, mehanika, dinamika, estetika

Abstract

More and more software applications operate online, which means that we, as software architects, are able to much better track how our users are using and engaging with the applications. Some of the results lately, have been staggering – more than 26% applications are not utilized and have poor retention and engagement rates. Behavior design and user experience researchers have been trying to solve these problems in many ways. Lately, one of the most popular methods has been the use of gamification techniques. These techniques try to imitate some of the mechanics that have made video games immensely popular and implement them in non-gaming software applications. In our thesis we first try to disassemble some of the core reasons behind engagement and fun of video games in order to understand the principles behind them. Later, we utilize a formal approach for gamification by using the MDA framework and gamify two software applications. Further on, we make an overview of all gamification platforms on the market. We sum it up by finding, featuring and describing gamification techniques used by some of the most popular software application.

Keywords

Gamification, games, engagement, engagement and retention metrics, mechanics, dynamics, aesthetics

Chapter 1

Introduction

Software applications have been born from a human need to help us make our lives more efficient, build things that would be otherwise impossible, and make the work processes easier to handle. They are doing a superb job at it; if people that were supposed to use them (let us call them users from this point on) actually use them and if they use them in the appropriate manner.

This is a well known problem in the software industry. For example, it has been widely reported recently that more than 26% of all mobile applications are used just once [**?**]. One of the most common reasons is the so-called "lack of user engagement". The problem is becoming even bigger, as users are now daily bombarded with new applications and do not have time and willingness to learn every new application. Most of the applications are not designed and architected with the end user in mind. Subsequently a lot of those applications do not meet user's needs, which leads to very low usage and eventual failure. This results in disgruntled application developers and it is the main reason why we started working on this problem.

There is one field in the software industry that has continuously proved to have "out-of-the-park" user engagement rates. Video games. Since their beginning in the 1970s they have absorbed users in a previously unfamiliar manner and have since thoroughly mastered the arts of how to engage user, get them absorbed or even addicted in the game.

So what are games doing right that we can learn from them and subsequently design better, more engaging experiences even beyond game contexts. It has been widely researched that people who play games express unfamiliar dedication to the activity. We, as software architects, could use this knowledge on motivation and engagement to design, for example, better collaboration tools, productivity tools, even email applications. Applications that we all use on a regular basis and would be much more enjoyable, fun and engaging if they utilized a few of the tricks from the treasuries of the gaming industry.

There is a whole movement emerging in the IT sector (and moving even beyond it) and it is called *gamification*. Gamification tries to distinguish what elements of games make them so powerful and engaging, distills and examines those elements and then apply them to non-gaming concepts. Have you ever seen that always present meter in the right sidebar of Linkedin (`http://linkedin.com/`) showing you that you have only completed 80% of your profile? An element from games. How many times have you tried a new software application that had three subsequent onboarding steps for you to learn how to use it? This is a game element too.

We we see this kind of elements appearing more and more. They have been especially popularized lately in the web applications, where the iteration cycles are much faster and developers and software architects can test and try different techniques on how to pull more and more people to use and engage with their applications. Usually we are not even aware of those elements; greatly implemented elements morph naturally into the overall experience of the application and just make it easier to use.

For illustration of how the trend of gamification has taken off lately, let us look at Figure 1.1, which is a screenshot from Google's Trends product (`http://google.com/trends/`). It shows the number of queries for the gamification term.

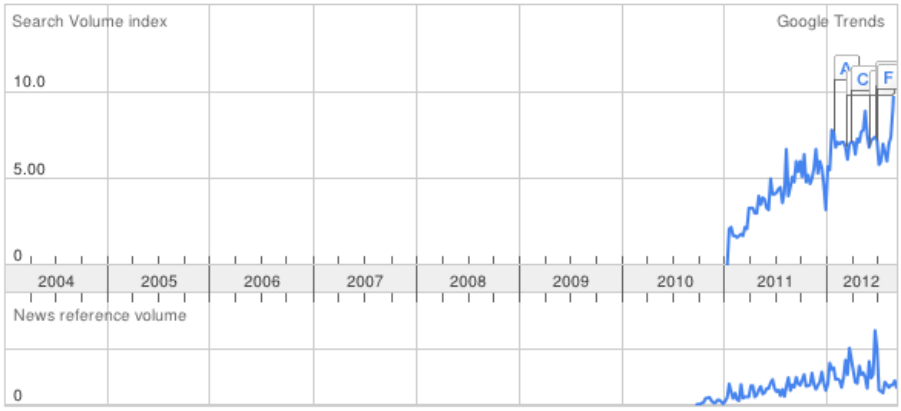

Figure 1.1: Growth of gamification related queries in Google's search engine

That is why we have decided to dig into it a bit more. We aim to uncover the reasons for success of gamification techniques, where gamification comes from, dissect it into logical pieces and offer and find an appropriate tool for every software architect.

In Chapter 2 we discuss what are games, what's definition of a game and gamification and why gamification is so crucial for driving user behavior. In Chapter 3 we discuss and present the *MDA framework*, which helps software architects design gamified solutions. We then use the framework in the main part of our thesis, Chapter 4. There we propose two solutions for two different projects – first one is a photo sharing application Eeve and the second one is a painting applications Psykopaint. In Chapter 5 we look at the *gamification platform* market, analyze the three most promising products and describe one of them, Badgeville, in more detail. We genuinely wanted to clarify the importance of gamification and the best way to do it is to present some of the successful applications out there using it. This is what we focus on in Chapter 6 and then conclude our thesis in Chapter 7.

Chapter 2

Background on games and gamification

Before we start to discuss and research the topic of gamification it is extremely beneficial for us to know where it comes from. As the word itself signifies, the term comes from games, so we first look into various definitions of games in Section 2.1. After we grab some of the commonalities of several definitions of games, we start looking at the term gamification more closely in Section 2.2. The remainder of this chapter in Section 2.3 then tries to see why gamification has been so popular, what we can learn from it and what makes the principles behind it so powerful. We take a deeper look at types of motivation in Subsection 2.3.3. We conclude the chapter by examining the different types of players/users that we have to consider when architecting gamified software applications in Subsection 2.3.4.

2.1 What is a game?

There are plenty of definitions of a game, but they vary a lot depending on the standpoint of the one who defines it.

Most of the definitions of a game define it as a structured playing, with strict rules and something that is usually done as of pure enjoyment or fun

instead of work, which most often involves a compensation for the activity.

On the other hand, academics have not been able to standardize a definition, so they vary a lot: here is an example from Roger Caillois' as featured in Jesse Shell's book Art of Game Design [**?**]: "activity which is ... voluntary ... uncertain, unproductive, governed by rules, make-believe."

Jesse Shell [**?**] also cites Jesper Juul, a Danish game researcher's definition: "A game is a rule-based formal system with a variable and quantifiable outcome, the player exerts effort in order to influence the outcome, the player feels attached to the outcome, and the consequences of the activity are optional and negotiable."

Although all these definitions are correct from their perspective, they are not related to how we want to define a game in the context of gamification. We get closer to a definition we seek by looking at how some of the most notable game designers define games.

Chris Crawford, a well-known US computer game designer defines it as [**?**]: "Games are a subset of entertainment limited to conflicts in which players work to fill each other's goals, just one of many leaves of a tree that includes playthings, toys, challenges, stories, competitions, and a lot more."

Sid Meier, the father of the legendary strategy game series Civilization defines a game as a [**?**] "series of interesting choices." This quote has been somewhat controversial in the past and did not have much context around it. But not long time ago, he explained it a bit further: interesting choices are choices that are presented to the player and they always have to involve some trade-offs – there is no absolute best option.

Ernest Adams and Andrew Rollings, authors of the book On Game Design, narrowed a game to "one or more causally linked series of challenges in a simulated environment" [**?**].

For Greg Costikyan, game designer and science fiction writer, the definition goes like [**?**]: "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."

Katie Salen and Eric Zimmerman define a game in their book Rules of Play [**?**] like: "Game is a system in which players engage in an artificial conflict, defined by rules, that results in a quantifiable outcome."

Raph Koster, author of A Theory of Fun for Game Design is quite keen on the next definition, which considers the game in the context of play and also explained it beautifully in one of his latest articles [**?**]: "Playing a game is the act of solving statistically varied challenge situations presented by an opponent who may or may not be algorithmic within a framework that is a defined systemic model."

To sum it up, we can say that there is no easy and well-rounded definition of what is a game and that it varies depending on context of strict video games or games in general. That said, we can see some of the components of a game that seem to resonate with most of the definitions above:

- Game is a system that has *a strict set of rules*
- Games present *structured challenges* to the players
- Game is played in *a simulated environment*

We see later on in our research that these constraints of a game can be (and are) utilized in every significant gamified software application.

2.2 What is gamification?

Gamification is a widely used and lately more and more popular term that can be spotted in several different contexts, so it is worth examining it in detail. The term started to appear in various articles around 2008 and was first coined by social gaming experts James Currier and Bret Terrill [**?**].

It got more traction and became popularized in the second half of 2010. It is worthwhile to note that it mainly spread out through the blogosphere and different conferences on behavior design and social games; one telling example is Barry Kirk's talk Gamification of Loyalty from Loyalty Expo

in 2009 [**?**]. There Kirk explained some of the basic game mechanics and encouraged a broader set of marketers and product oriented professionals to study and consider where "the game tactics fit" in their own business case and take full use of it.

Even though the term is relatively fresh, the experts in the field nowadays appear to be quite aligned on the definition, so let us look at it a bit closely.

The following definition is our aggregated and condensed version of several definitions that appear in the research field: gamification is the use of *game design techniques*, *game thinking* and *game mechanics* outside of gaming context.

Gamification aims to encourage people to engage with applications more than they would without applying these techniques. It engages users to start using applications more easily, leads them throughout the learning curve to mastery and autonomy. It also helps solving otherwise tough or uninteresting problems and makes the technology in general more appealing and easier to grasp.

We will explain all the elements of the definitions more in details later, but it is crucial to note how gamification relates to traditional video gaming, games and play. Let us take a look at a research workshop called Gamification: Toward a Definition [**?**].

2.2.1 Game versus Play

The previously mentioned work [**?**] focuses on coining the definition of the game and trying to put a context around everything encompassing it (Game, Elements, Play). Firstly it tries to distinguish between game and play. It says "games are characterized by rules, and competition or strife towards specified, discrete outcomes or goals by human participants."

On the other hand, it defines play as "a free activity standing quite consciously outside ordinary life as being not serious but at the same time absorbing the player intensely and utterly."

That means we have to distinguish games from just basic design for play-

Figure 2.1: Diagram depicting relationship between play, game and gamification

fulness or playful interactions.

2.2.2 Game elements versus whole game

The distinction between gamified experiences or applications and real games is not always very clear. So how do we set the borderline between games and artifacts with game elements? Games are, as we have seen previously, defined by a strict set of rules and boundaries. Artifacts with game elements (in our case, gamified software applications) are more or less just composed of a subset of technical elements of games, and are missing the social elements to become a real game. We can see a visual representation of these relationships in Figure 2.1.

The research [**?**] gives an example of the gamified application Foursquare

(`http://foursquare.com/`): it is usually interpreted as a powerful gamified application. It is not a game, though – it could be considered a game, for example, if a smaller group of users organized and imposed an informal set of additional rules.

2.2.3 Non-game context

It is necessary to explain what we, and the rest of the scientific community on this subject, mean by the term "non-game context". It is the use of game elements in non-typical use-cases, which are mainly aimed at entertaining users.

The research [**?**] further proposes that we do not limit the aim of gamification to, for example, joy of use, engagement, improvement of user experience as it is still in the very early stages of development and the current popularised use cases might be further subcategorized or too limiting.

2.2.4 Game design elements

So what are then the elements that help us design gamified experiences? The research paper [**?**] proposes a classification of the GDEs (*Game design elements*) into five different abstract levels:

1. *Interface design patterns*, such as badges, levels, or leaderboards.

2. *Game design patterns*: reoccurring interaction relevant to game play. An example is a well-known Paper rock scissors pattern, sometimes also referred to as triangularity.

3. *Design principles or heuristics*: guidelines for approaching a design problem.

4. *Conceptual models of game design units*, such as the MDA framework, Malone's challenge, fantasy, and curiosity, or the game design atoms described in Braithwaite and Schreiber [**?**].

5. *Game design methods*, including game design-specific practices such as playtesting and design processes like playcentric design or value conscious game design.

2.3 Importance of gamification

2.3.1 What can we learn from games?

We can probably all agree that games, at least certain games, are fun. We have all probably witnessed a moment when we got totally immersed in a game of Tetris or solving a very hard Sudoku. But what makes the games so fun and can we, as system architects, learn anything from them?

"Fun is all about our brains feeling good - the release of endorphins into our system." That is how Raph Koster put it in his book A Theory of Fun [**?**]. But what makes the brain feel good?

Research that Raph Koster uses tells that the brain constantly needs motivation to do something. It is on a constant search for achievement, for a moment in time, when we learn something or master a certain task.

This is where the fun comes from in the games. Our brain is constantly trying to find the right pattern to solve the mystery or the task and when it finally finds it has a small moment of satisfaction. But soon afterwards the brain needs a new challenge and that is also the reason why some games become boring so quickly – we master the pattern and if the game is not challenging us into finding a new one, it becomes tiresome and repetitive.

A good example here might be a game of tic-tac-toe. It is very fun and challenging – for the first five minutes. But once players figure out the optimal way of playing it, it becomes very tiring and as a player might say "too easy".

On the other end of the spectrum is frustration – games can be frustrating if they do not offer a "fair" challenge; the user is unable to locate a logical pattern to form a solution to the problem.

So if games are teachers of patterns and if we like to play or learn them

when we are having fun in a fair and not boring way, then subsequently this means that the more fun and engaging the experience is, the more we learn. If we can apply techniques from great games to software applications, they can become fun to use: in other words, users will stay longer and have more fun using them.

2.3.2 Engagement

We often use the term *engagement* when describing the connection between user and the given thing. It is very often mentioned within the scope of marketing, especially now in the social media world, when advertisers seek engagement between users and the brand. But what is engagement really, and why is it so significant and important in our case?

Engagement signifies a special connection between the user and product, service, or in our case, software application. Like Tom Chatfield, British author and game and technology theorist, states in his TED talk [**?**]: "... individual engagement can be transformed by the psychological and the neurological lessons, we can learn from watching people that are playing games. But it is also about collective engagement and about the unprecedented laboratory for observing what makes people tick and work and play and engage on a grand scale in games ..."

So how are games able to get all this amazing amount of engagement? People are, for example, playing a famous game World of Warcraft even for more than 8 hours daily. We have learnt previously that human brains need constant stimulus for it to stay engaged and have fun. Fun is when we are trying hard and staying immersed in the experience. And to achieve this constant fun, the user needs constant gratification and for that games for example use a technique called reward schedule. Reward schedule, as the name refers, aims to deliver rewards and new challenges just at the optimal time. We talk more about it later in the game mechanics and game dynamics sections (Section 3.1 and Section 3.2, respectively).

2.3.3 Types of motivation

Why are people so motivated to play games? Why do they keep coming back to them? We distinguish two core motivations: those that arise from within users or *intrinsic motivation* and those that are triggered by some external mechanics and are called *external motivators*.

Intrinsic motivation

Intrinsic motivation is driven by interest or enjoyment that derives from doing a task itself. It comes from within an individual rather than relying on some external triggers. It heavily relies on taking pleasure in the activity itself.

It is vital to note what researchers perceive as the most common reason for people to work on a task being simply motivated intrinsically: people need to have a certain level of autonomy. Let us look at a common example from video games: the first time users sit down and drive a car down the virtual city of New York in the game Grand Theft Auto (`http://rockstargames.com/grandtheftauto/`) they do not need any external motivation. It is just fun.

There has been a lot of research on the topic of intrinsic motivation: one of the best is the one done by Daniel Pink that manifested in his bestselling book Drive [**?**]. One of his basic thesis is that money is a really lousy motivator for complex tasks. He tells something counterintuitive – involvement of such straightforward motivators actually reduces people's performance on creative tasks.

Another well known researcher on this topic is a Hungarian psychologist Mihaly Csikszentmihalyi. He coined the term "state of flow" [**?**], which describes people in their most productive state: state where the time flies by and they feel just properly challenged but in the same time capable of completing the task. This is every gamification architect's dream state, and the state we would like to create in the user's interaction with the application we design.

Extrinsic motivation

Extrinsic motivation refers to the type of individual motivation that comes from favoring the outcome of a successfully completed task, as opposed to just completing a task for the pleasure of doing it. So if we look back to the previous example of driving a virtual car: the sole task of driving is fun in the very beginning, but then it eventually becomes extremely tiresome.

What do the game designers do in this kind of cases to make the experience lasts longer? They motivate players by giving them time limited tasks and throw more and more obstacles on the road (e.g. chasing police cars).

There have also been some very radical approaches to this topic – one of those is the work by B. F. Skinner, a psychologist from the early 20th century, who set a totally new paradigm called *Radical Behaviorism*. His work [**?**] on the topic completely disregards intrinsic motivators and relies solely on extrinsic motivation. It has thus been the basis for many of the game mechanics like point systems and game dynamics like reward schedules (more about it in his book Schedules of Reinforcement [**?**]).

One of the latest and more accomplished researchers on the topic is also B. J. Fogg. He is a Stanford professor and researcher, who lays out an intriguing model [**?**] for behaviour change. He states that for an action to occur there have to be three key elements: *trigger, ability and motivation*. His FBM (*Fogg behavior model* – in Figure 2.2) can help design products and functions, and better understand why certain mechanics/dynamics work or do not work.

What is the best kind of motivation?

There is a popular belief that intrinsic motivation is somewhat better than extrinsic, and it creates more value. Game designers would love it if all players would be intrinsically motivated to play in their "sandboxed" worlds. It would be easier to create this kind of environments. But they, as game

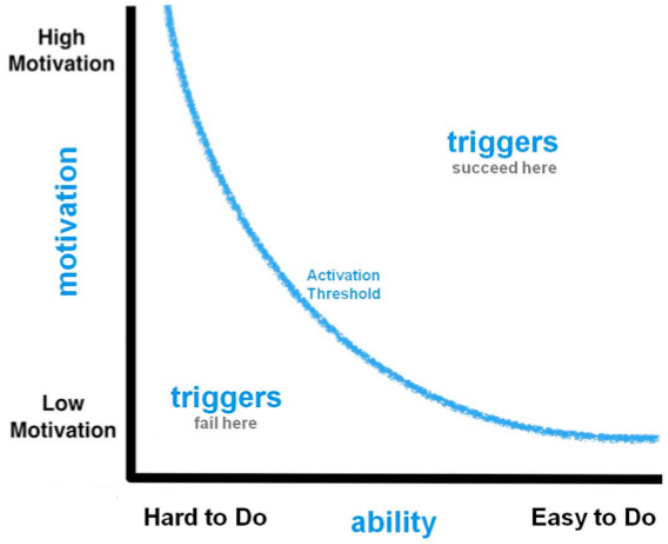

Figure 2.2: Visualization of B. J. Fogg's behavior model

designers, and us, as information architects, cannot rely solely on the user's inner needs to engage with our product.

With extrinsic motivators, we can more surely predict users actions and drive their activity in a processed manner. The greatest thing about it is: if we use game mechanics in a systematic and smart way, they dissolve in the experience and users might even perceive them as intrinsic.

2.3.4 Types of players/users

Why do we care what types of users/players engage with our application? It is much easier to start designing something that you know how it will be used and what it will be used for. So it is crucial for every game designer or gamification specialist to know their users or more specifically different types of users.

Richard Bartle, a British game researcher, spent lots of time researching and figuring out how to put player types in a few buckets in order to make designing gamification experiences a bit easier in the future. He studied

MMOG (*Massively multiplayer online game*) players, particularly players who played MUDs (*Multi-user dungeon games*), and identified four specific types [**?**] (later on he divided players in many more categories):

- explorers,
- achievers,
- socializers,
- killers.

Bartle also set the foundation for a questionnaire now known as Bartle's test [**?**]. The questionnaire itself was developed by Erwin Andreasen and Brandon Downey, and is a set of 30 questions with the goal of categorizing players into four before-mentioned categories. Let us look at some of the characteristics of specific player types, so we can identify and adapt to them when designing our applications. Figure 2.3 also shows the relation of specific player/user types to characteristics.

1. Explorers

Explorers are always seeking new things, adventures, secrets. Imagine people that always wanted to discover all secrets treasures and rooms in games, and people who "play" Foursquare mainly to get/discover all different badges. They feel distinct type of joy when they find something special or well hidden in an application: an easter egg or, surprisingly, even a bug.

2. Achievers

People that are categorized as achievers tend to be goal oriented. They like to achieve goals and feel particularly frustrated if they fail to do so. Moreover, they seem to lose their interest in playing the game that they do not win. Let us look at the Foursquare example again: achievers would play it mainly to win the badges. They would not play it for the

purpose of winning the badges themselves, but also to achieve and successfully complete all the "goals" behind them. They also enjoy collecting points.

3. Socializers

Socializers play or use applications in order to make new friendships or engage with existing social networks. An application for them is sometimes merely seen as a tool to meet new people both inside and outside of the scope of game/application. Actually, most of the people are first and foremost socializers.

4. Killers

Killers especially thrive in the MMOG type of games where they are able to interact and compete with a great number of other people. It is a bit difficult to incorporate elements specifically targeted at killers in "normal" applications: one of the examples could be mayorships in Foursquare. Users there compete with other players to be mayors of real-life venues. People that focus on "ousting" other players and becoming mayors take more pleasure in that than merely owning the venue.

Beyond Bartle's model of player types

There have been some critiques and additions to the Bartle's model recently. These have been somewhat covered in a research paper by Dan Dixon [**?**], a lecturer and researcher of creative technologies. The problem is not so much exactly in Bartle's research and the four player types he had identified; it is more the application of his model. Lately, the model has been used for real game design challenges and gamification purposes, but the problem is that people/players/users do not strictly belong in one and only one group of players. It is more common that we are all a combination of all of them, just with different distributions.

Therefore, Nick Yee, a researcher of self-representation and social interaction in virtual environments, has conducted a thorough research [**?**] on

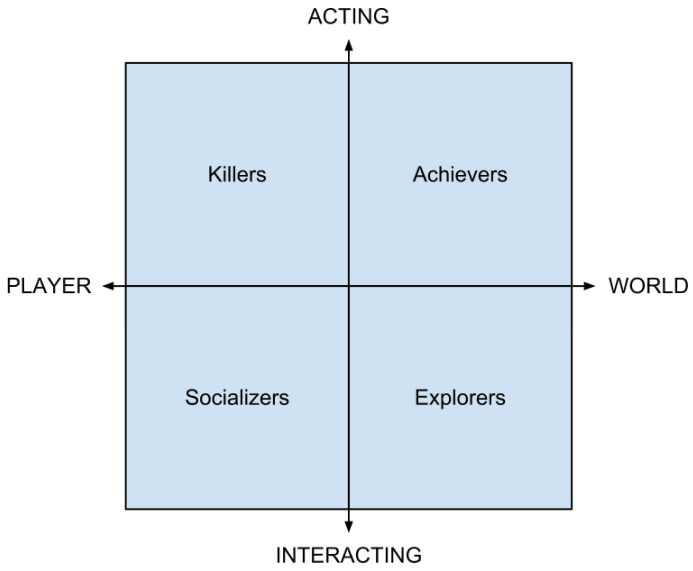

Figure 2.3: Four types of players/users matrix from Bartle's model of player types and their relation to four core characteristics of player behavior

MMORPG (*Massively multiplayer online role-playing game*) players. It resulted in an updated model of Bartle's model with three main components and ten subcomponents:

- achievement: advancement, mechanics, competition
- social: socialising, relationship, teamwork
- immersion: discovery, role-playing, customization, escapism.

He does not clearly define "new" player types, but gives a new perspective on user's motivation. To give an example of it: characteristics of Bartle's explorer type are split between the mechanics and discovery subcomponents. On the other hand, characteristics of Bartle's killer type relate strongly with the competition subcomponent.

This topic is lately being pretty heavily researched. There have been many empirical studies, but they most often deal with games instead of gamified software applications. This means that we can look at the upper

cases and use them as a good basis, but should not rely entirely on them and design all aspects of a gamified application around them. There is obviously still plenty of room for researchers to dig into millions of data points created right now by gamified system and create real user types for this whole new generation of experiences.

Chapter 3

MDA framework

We stumbled over a common problem while architecting our solutions – people and researchers tend to use terms game mechanics and game dynamics interchangeably. Therefore, we decided to follow a proposal from Robin Hunicke and colleagues who propose usage of the so-called MDA (*Mechanics, dynamics, aesthetics*) framework [**?**]. This framework was introduced in their research paper called MDA: A formal approach to game design and game research. The goal of this framework is to make it easier to understand and design translation of all aspects of the game design theory.

Let us start in the beginning: what are the game mechanics and dynamics? Is there something more to it? It turns out the MDA framework can be very well utilized when designing a gamified software application. It helps to bridge the gap between the outcomes (which is an engaged user of an application) and the inputs, which as in every software application are the architecture and common patterns that software architects can use and work with.

Let us look at the MDA framework from the perspective of software development. One hand we have the set of requirements that specify the final behaviour, components and constraints of the given system. This is how we, as software architects, envision the final product. On the other hand, there is the working product, which is essentially code that brings the

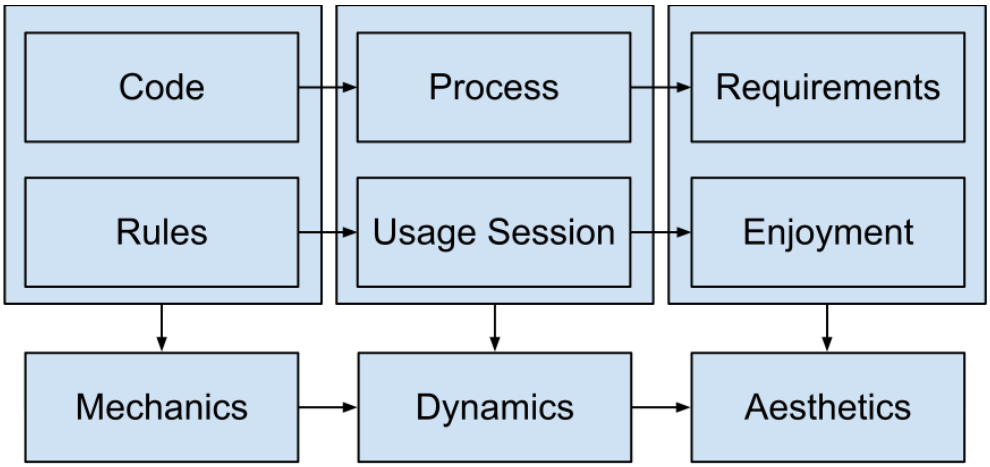

Figure 3.1: Visual translation of different components of the MDA framework

whole system into reality. In the middle lies the process of executing that code and making the system working.

We can look at games or software applications on a higher conceptual level through a similar prism. Requirements become enjoyment, joy and engagement of the user. On the other hand, there are rules that set the whole application in motion. And in the middle we have the user session, which is similar to the process from the previous paragraph.

This is how we come to the MDA framework. Aesthetics is the top layer, layer that the user interacts with and the way it makes him feel about the application. On the other side are mechanics, which are simply functioning components of the game. And in the middle are dynamics, which could be translated into a run-time behavior of the gamified application as a system.

Let us look at the Figure 3.1 from the perspective of both the software architect and the user. It is far more crucial for us to think and deeply understand the software architect's point of view; where specific mechanics lead and engage certain dynamics that in the end invoke required aesthetics (or not required, if the system has not been designed properly).

Only thinking from this perspective can sometimes be dangerous for the software architect and can result in excessive features or the so-called *"feature*

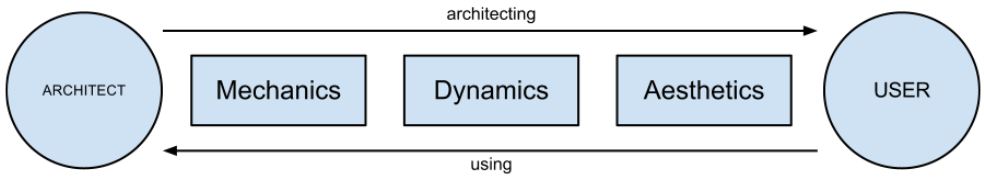

Figure 3.2: MDA framework from perspective of an architect and a user

creep". As LeBlanc et al state in their research paper [**?**], it has become more and more popular to think and design a gamified system (or preferably any other system) from the user's standpoint. This way aesthetics set the tone of the experience, which is a result of observable dynamics and finally operable mechanics. This approach to designing software applications is called the *experience-driven design*.

In the Figure 3.2 we can see both perspectives: software architect usually sees the application architecture from the mechanics standpoint, contrary, user sees the application from the prism of engagement and joy of using it, thus, from the application aesthetics.

This chapter is dedicated to understanding the MDA framework for architecting gamified software applications into details. The remainder of this chapter is organized as follows. We describe main components of the framework. The first and the most important one is Section 3.1 that goes deep into different game mechanics. Section 3.2 and Section 3.3 then describe the game dynamics and game aesthetics components, respectively.

3.1 Mechanics

As we said before, mechanics in the gamified system, are functioning components. As Gabe Zichermann and Christopher Cunningham state in their book Gamification by Design [**?**], components, when used correctly, "promise to yield a meaningful response (aesthetics) from the players". Mechanics should always be chosen for the specific experience if they are to result in

the appropriate aesthetics.

So let us look at some of the most suitable and often used mechanics when designing gamified software applications.

3.1.1 Point systems

We have all experienced point systems before as they form the basis of a lot of video games. They have proven to be an extremely powerful motivator. Just look at the sports for example. Most of them are based on some kind of point system.

Points, however, may sound terribly boring and not compelling at all for a gamified system today. We see them every day in a variety of forms: the bank account balance, followers/friends scores on various social networks, credit scores etc. This is just the tip of the iceberg; just remember how many new (especially) mobile applications have followed the lead of Foursquare and implemented a basic point system. Consequently, we as users, may not react to them as we did just a couple of years ago.

Nonetheless, they are still crucial when designing a gamified system; even if we do not expose them directly to users, it is extremely important to have a simple experience point system in place (even if it is just in the back-end of our application). It logs down every interaction with the system and tells valuable information about the usage – that way we can then go back and tweak the system and make it better or more engaging.

The other vital part of every point system is always assigning the right points value for specific tasks. Point values for a given task are not really important, but become really important when the system has a set of different tasks with different associated point values. This forms a hierarchy between them and signals the importance of a specific task in relation to others. It is super hard to get this relations right in the first try and that is where observing the system and subsequent tweaking come into play.

We have identified three most crucial types of points that should be considered when building a gamified application:

- *experience point system,*
- *karma point system,*
- *redeemable point system.*

Experience (XP) point system

Most basic point system that we have briefly described above use the experience point system. It can not be maxed out (it is limitless), and it can not be used to "purchase" or redeem anything: its sole purpose is to measure a specific metric or a set of actions. They mainly serve to show-off and attract the achiever type of users. However, it can be reset on a regular basis to attract users to engage more often and constantly chase new points. A very basic example could be the number beside the username on the online forums indicating the user's number of posts.

Karma point system

This is a pretty unique system, that does not find its way too often into video games. On the contrary, it can be (and it has already proven to be) powerful system in non-gaming gamified context. How does it work? Users are rewarded by the system for accomplishing specific tasks, which is no different from any other point system. But instead of "spending" points for some other virtual objects they can award other users with the points they have earned. This forms a compelling feeling of community and altruism.

We have seen it remarkably well utilized in a form of customer service forum for an alternative mobile carrier Giffgaff (`http://community.giffgaff.com/`) in the UK, where users help each other solving technical problems in the context of the mobile operator. In return, other users may reward them with karma points. This behavior helps running an almost self sustainable support system for tens of thousands of users. We talk more about this specific case in Section 6.5.

Redeemable point system

This is a hugely popular point system that we interact with almost on a daily basis. Real life examples are *frequent flyer points* or *loyalty shopping points*. All those points are earned for finishing specific tasks within the system and can at certain thresholds be exchanged for another item. Therefore they form a virtual economy, which means that the application architect may be constrained in the implementation, because of certain legal and regulatory issues.

Other notable point systems

There are two other point systems that we have encountered: *reputation point system* and *skill point system*. An example of the former is a rating system for restaurants. A telling example is Google Places (`http://google.com/places/`). Skill point system is, for example, a system where users do not earn special skill points for accomplishing vital tasks, but for tasks within a field. For example, if we were to gamify the training of tennis, trainees might be getting special skill points for practicing only smash return shots.

3.1.2 Onboarding

Onboarding is the process of user's first experience with the system and the way the gamified application leads users to accomplish specific tasks to get to know the system. It can sometimes also be referred to as the application *sign-up process* or *sign-up flow*. This process is becoming immensely crucial for all new web and mobile applications as users are bombarded with hundreds of new services and do not have much time to study every new application. Sometimes a new service has under a minute to lure the user over and get to the core of it.

We have experienced this ourselves when designing the first user experiences within Eeve and Psykopaint. It has proven that it is way more valuable to hide the complexity of the service and then slowly reveal all the possibilities when user goes down the onboarding funnel and becomes an active user. Moreover, it is particularly crucial not to make user fail and give him very

quick positive feedback on every action. Onboarding, thus, represents one of the biggest hurdles in new applications; it should be extremely well thought through, observed on real test users and constantly modified based on the real usage data.

There is a recent example of the social networking application – Twitter (`http://twitter.com/`). Even though Twitter has been widely successful and attracted millions of users, they still had immense problems keeping those users engaged and explaining the system well enough so they would come back to the site and convert to active users. That is when the back-then Twitter's data analyst Josh Elman took a deep look at their existing terabytes of data and discovered that people become more engaged as soon as they follow more than 30 other users. This led Twitter's team to dramatically change the onboarding process where they lead new users slowly through steps and recommend them more existing (and also more contextually relevant) twitter users to follow, which in the end results in more active Twitter users. This kind of hiding and slowly revealing information is sometimes also referred to as the *Cascading Information Theory*.

3.1.3 Levels

Traditional levels that most of us know from video games are not quite possible and viable solution for gamified experiences. There are a few elements that we can take from the knowledge base of levels: progress bars, for example. We have learnt how valuable positive feedback is to users and how tempting it is to fill up half accomplished tasks. That is why progress bars that we see in essentially every software application should be utilized to drive user behaviour.

Take, for example, one of the most famous cases of Linkedin: we can still remember how awful it felt going to the site and always seeing the 80% filled up profile bar on the right side. So eventually we gave up and accomplished a few more steps to reach the 100% level. Another example is level-uping in case of internet communities, where a certain amount of points leads to a

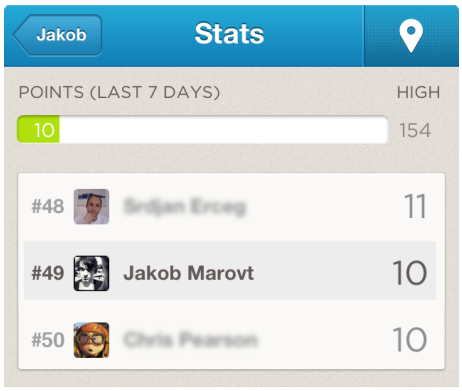

Figure 3.3: Example of a social leaderboard

higher level status.

3.1.4 Leaderboards

Leaderboards are designed to present the user's score (for example his overall experience points) against other users and to make clear visual comparisons from it. In a gamified application, they mostly serve the purpose of incentivizing the user; showing his position always in the middle no matter where on the social leaderboard he stands (if he is number 49 we would then show his between number 48 and number 50). We also mentioned social – that is because usually this kind of leaderboards show the player against his peers (especially now when social context is becoming ubiquitous). Thus, the player always knows where he stands, what he has to accomplish to climb up (the steps should be obvious), and that there are also people that are performing worse. Figure 3.3 depicts an example of such social leaderboard from one of the latest versions of Foursquare.

3.1.5 Achievements

Achievements stand for visually representing a task or a group of tasks. The power of clear visual achievements has been well known for a while: just

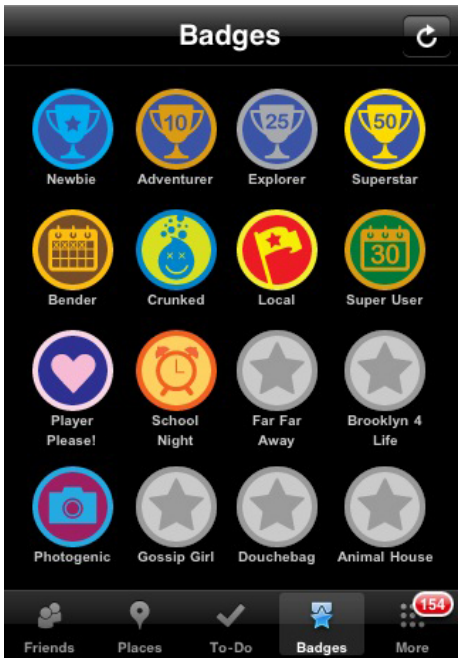

Figure 3.4: Example of the Foursquare Badges view

think about military and boy scouts badges. They are easily identifiable on uniforms and achievers take pride in wearing and showing them off.

In the last years, the achievement badges have become widely popular in the gamification scene, especially after the global success of Foursquare, which was one of the first mobile applications to use them in an appealing and highly effective way. Figure 3.4 depicts its badges view.

Since Foursquare hundreds of other applications have tried to mimic it and usually simply plastered some non-innovative badges on their existing product. We would like to warn software architects to think a bit deeper when implementing this kind of system as users have become really bored of it lately. If it is implemented well it can be really effective, but it has to augment the core of the product and not just stand on its own. Later on, we show how we thought about implementing this kind of system in our application Psykopaint in Section 4.2.

3.1.6 Customization

Customization of the application experience is a pretty old and widely used mechanic that we have all encountered and probably even see and use every day. It is basically allowing and enabling the user to change and modify some parts of the experience, usually visual representation of his online presence within the context of the given application.

Examples include customizable avatars, skinning of the experience (WinAmp audio player, used-to-be widely popular social network MySpace, Twitter, even Facebook with its new timeline cover) etc.

These kind of techniques have proven to be really successful in keeping users excited, more engaged and giving them a chance to show off their creativity, personality and commitment to the application. But we, as software architects, should be very wary while designing this kind of experience; too much choice (especially presented at once) can lead to confused and dissatisfied users. Moreover, giving users a lot of free room for creativity and leverage over the whole visual experience can be fatal for the application as we have seen in the example of MySpace. In MySpace (`http://myspace.com/`) users took openness of customization options to a whole new level and transformed their personal profiles into crazy blinking sites. This has proven to be a big turnoff for the majority of audience, which soon after switched to a more controlled and cleaner version that we all use nowadays – Facebook (`http://facebook.com/`).

The bottom line is: customization can be very successful if it is highly controlled and does not present users with too many options, which has been deeply explored in Barry Schwartz' article The tyranny of choice [**?**]. His conclusion is that happiness of users rises proportionally with added possibilities of customization, but at a certain point starts decreasing. Eventually then added choice only decreases happiness of users.

3.1.7 Challenges

Challenges or quests from games are powerful drivers of behaviour. Users or players crave to be led and to know what to accomplish or do next, and this kind of mechanics are widely present in games. When thinking about software applications, it is sometimes hard to think about challenges – they are not as obvious as in games. Recently challenges, or just user behaviour changing tasks, have become widely used as a part of the onboarding process where software architects strive to lead users through a few most crucial steps. Software architects previously identify those few crucial steps the user has to do to get to know the core experience and be able to use the application on his own.

Let us take, for example, the before mentioned Twitter example; Figure 3.5 shows us how Twitter's onboarding process presents the user with extremely small, easily achievable challenges. This tweaks alone have had a dramatic impact on *activation rates* of new Twitter users.

Twitter's onboarding experience slowly takes the user through the steps and give him easily achievable tasks, e.g. follow a few well-known contextually relevant people (Figure 3.5.1. + Figure 3.5.2.). The next step is to follow a few celebrities from specific categories, like music and sports (Figure 3.5.3.) and then a few of your real-life friends through any of user's e-mail accounts (Figure 3.5.4.). The final steps are adding some personal information (Figure 3.5.5.) and introducing the user to the stream of tweets (Figure 3.5.6.). Overall truly elegant solution, which slowly presents small tasks, gives back the user positive feedback (also through a clever use of progress bars – check Levels mechanic), and encourages him to continue and slowly reveal more options. This technique alone has risen their *retention rates* significantly (retention rates up from cca. 25% to more than 40% and sign-up completion process up by 29%). Previous Twitter's onboarding process consisted only of 2 steps: the first step recommended a few known users to the new user and the second step led the user directly to the twitter stream (quite similar to the step depicted in Figure 3.5.6.).

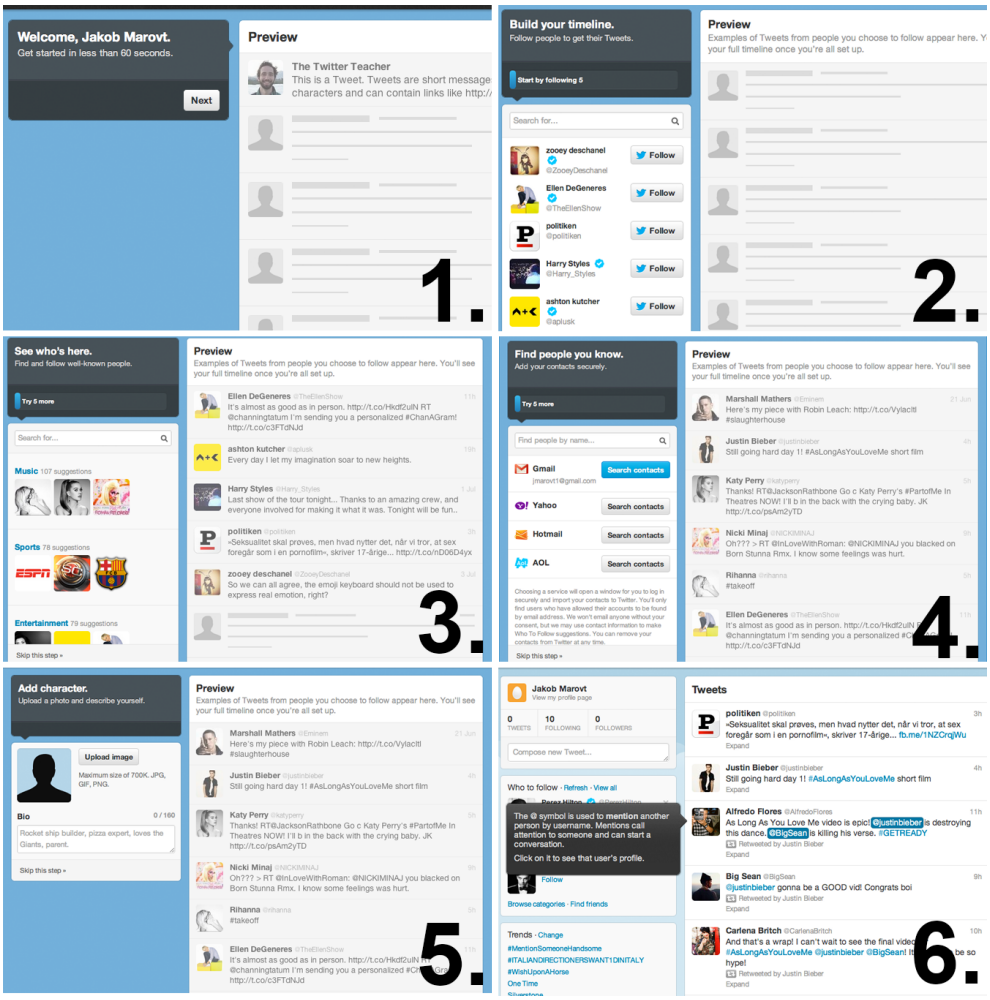

Figure 3.5: Twitter's new six steps onboarding process

A well known technique here is also a so-called collaborative challenge, which heavily taps into the social aspect of experience and tries to leverage the intrinsic motivation for social connection. For example, there might be some challenges or tasks in an application that only a group of people working together can accomplish. This is really powerful mechanic as it leverages the existing users to organize among themselves, which then also results in social pressure to perform the given. This eventually results in more committed users. It is sometimes also referred to as *Communal Discovery*.

3.1.8 Social engagement loop

We have all heard the term *viral loop* and it is one of the most studied and fundamental concepts in modern software applications. Its quality and whether it is a part of the core experience in many cases determines the success of modern social applications. It is crucial nowadays for the application designer not only to think about how user uses the core functionalities of the system, but also what brings him back, how he interacts with other users and what motivates him to engage with the application in the long-term.

Thus, researchers in the field have identified four essential steps [**?**] that have to be present (and of course effective) to complete the social engagement loop:

- *motivating emotion,*
- *social call to action,*
- *player re-engagement,*
- *visible progress/reward.*

We can look at the visualisation of the concept of social engagement loop in Figure 3.6.

Social engagement loops are sophisticated mechanics and can not be effectively easily added to an existing product. Rather, they have to be present in

Figure 3.6: Social engagement loop visualization diagram

the core of the product itself. Social engagement loop also embodies some of the other more simple mechanics, such as onboarding, leaderboards, leveling, customization etc.

Let us look at one of the most famous examples of the last year (2011) – social music listening and curating service Turntable.fm (`http://www.turntable.fm/`) launched and experienced almost unprecedented viral "hockey-stick" growth. We help ourselves here with a really informative article [**?**] about this specific case from the social Q&A site Quora.

1. Visible progress/reward

- DJ points: users receive points for each "love" they get from the public.
- Fans counter: users have their fans – people who follow them and get notification when they play in a specific "room".
- Avatars: user can choose avatars for their online representation; the more popular/experienced the user is the more extravagant his avatar can be.
- Stickers: users can stick more and more stickers to their virtual DJ laptops through the whole experience.

2. Motivating emotion

- Points: users are motivated by getting more "awesomes" on their song plays and subsequently satisfying more people.
- Head shaking avatars: really great subtle tactic – each "awesome" on the user's song also means that the user's avatar starts to shake the head a little bit; taps into basic intrinsic motivators such as satisfaction and mastery.

3. Social call to action

- Becoming a DJ and playing a songs: playing a song allows people to express their music taste and offers it to other to judge it.
- Voting on a song: users can "awesome" (or "lame") songs, and thus publicly express their satisfaction level and connect with the DJing player on some social level.
- Chat: chatting allows people to express their emotions, knowledge and connect with other players.

4. Player re-engagement

- Smart e-mail notifications: notifying users when their favorite DJs are playing or when their real-life friends joined the application, which motivates people to come back.
- Desire to progress & get more "love": as mentioned before, users collect DJ points (which are standard XP points) and in order to keep up with the rest of the players they have to be present regularly.

3.1.9 Other notable mechanics

There are many more game mechanics beside the ones we described above. It is necessary to state, that the mechanics listed before are the ones more widely used and more easily applicable. We will list down some others here and try to describe them briefly in order to have a full toolbox of mechanics covered:

- *Behavioral Contrast*: behavior of users can shift dramatically when expectations change – after the player gets a certain achievement he is not so excited about getting it again or getting a lower value achievement.
- *Countdown*: there is a certain amount of time for users to do something – this mechanic is particularly useful for time limited discounts, which have proven to be super effective.

- *Disincentives*: this mechanic penalizes the player for doing something in the wrong way – one well known example are real life speeding traps.
- *Epic Meaning*: it has been proven that users, as referred in Jane McGonigal's book The Reality is Broken [**?**], are way more willing to cooperate and perform better in some task, if they believe they are working on something huge, something beyond them.
- *Free Lunch*: it is a mechanic that persuades users that something is more available, because someone else has performed the work instead of them – a widely known example are various daily deal sites where users get huge discounts because so many of them buy the deal.
- *Ownership*: this is not a particularly huge mechanic, but it has been widely used and successful – users are motivated by being able to "own" something, even if it is just virtual; a good example are mayorships of places in Foursquare.

3.2 Dynamics

Dynamics describe the run-time behavior of the mechanics acting on user's inputs and each other's outputs over time. We could, therefore, say that they form a feedback loop between game mechanics and the user and form the actual art of playing or using in our case.

Here are some of the dynamics that may take place during a user session:

- *Pacing*: pacing of user progression.
- *Appointments*: rewards for using the system in a specified time frame.
- *Progressive unlocks*: serendipitous, unexpected unlocks of achievement.
- *Rewards schedules*: specific schedules of rewarding the user.
- *Dynamic systems*: dynamic adjustment of the events based on player's usage characteristics.

- *Peer pressure*: individual user within a group of other users striving toward a common goal is more motivated as he knows that other users depend on him achieving the goals.

3.3 Aesthetics

Aesthetics of the system represent how the game or a gamified experience makes the user feel during the interaction with the given software application. Game aesthetics can be viewed as the composite outcome of the mechanics and dynamics as they interact with and create emotions. To boil it down to the simplest form – it is everything that makes an experience using the software application "fun" or engaging.

Software architects, implementing gamification techniques, try to invoke several emotional states by Hunicke et al [**?**]. Although these emotional states derive directly from game design, they can be useful when planning gamified systems:

- *Sensation*: game as sense pleasure.
- *Fantasy*: game as make-believe.
- *Narrative*: game as drama.
- *Challenge*: game as obstacle course.
- *Fellowship*: game as social framework.
- *Discovery*: game as uncharted territory.
- *Expression*: game as self-discovery.
- *Submission*: game as pastime.

The above list of aesthetics is sometimes also referred to as eight kinds of fun. Although it is worth noting that these models do not capture all of

the positive human emotions, which researchers have discovered lately by observing players. Jesse Shell notes in his book Art of Game Design [**?**]: "these models ... have gaps, and when misused can gloss over subtle pleasures that might easily be missed." He then goes on and lists a few of the emotions discovered and not otherwise captured in the above models:

- Anticipation
- Delight in another's misfortune
- Gift giving
- Humor
- Possibility
- Pride in an accomplishment
- Purification
- Surprise
- Thrill
- Triumph over adversity
- Wonder.

This is an extremely interesting topic for every software architect as more and more design approaches now start from the user's perspective. This is why we should know as much as possible about the motivations behind why people engage with applications. One of the most intriguing researches in this field was done by Nicole Lazzaro, a researcher in the field of player experience design. She did an extensive observation and consecutive analysis [**?**] on what motivates players to play and engage in games. She found out there are four main keys to why people enjoy playing games that might be helpful when designing gamified software applications:

- Hard fun: players were particularly engaged in overcoming a structured problem/challenge.
- Easy fun: players enjoyed intrigue and curiosity while discovering and exploring the game.
- Altered states: players enjoyed games that made them feel better or less bored.
- The people factor: players expressed different emotions while playing within a group or competing against each other.

We can see many similarities between these four keys from Lazzaro's research [**?**] and the above mentioned eight aesthetics by Hunicke et al [**?**]. Let us try and see how we can connect each one of the keys with the related aesthetics and see if eventually researches can merge these relations into knowledge that we can use further:

- Hard fun : challenge
- Easy fun : sensation, narrative, discovery
- Altered states : expression, submission
- The people factor : fellowship.

Obviously, we were able to identify seven out of eight aesthetics and link them with appropriate keys. This means that the researchers' findings do not differ much and that the eight aesthetics or kinds of fun can be used for our architecture in the Chapter 4.

Chapter 4

Applications of the MDA framework

Now that we know how MDA framework functions and how we can use it, we start with designing the gamified solutions. During our work on the thesis, we have been involved in two projects that required a gamification overhaul.

First one is Eeve, a mobile (iPhone) event based photo sharing application, which lacked engaged audience. We architected two solutions that are presented in Section 4.1. Next, slightly bigger, project was a gamification of a web based painting application Psykopaint, which we have been working on for a longer period. We describe our solution and architecture in Section 4.2 and conclude this chapter with the effects or results of our work after the custom solution implementation.

4.1 Gamifying photo sharing application – Eeve

Eeve is a simple event based photo-sharing iPhone application. It allows people to start an eeve (an eeve is essentially an event) that is tied to a specific location via the GPS (*Global positioning system*) signal. When an eeve is going on at a specific location, users are able to take more photos and

co-create an event photo set. It is a super easy application, that is trying to get into the crowded photo-sharing market, by allowing users to create visual collaborative experiences. The main Eeve's feed view is portrayed in Figure 4.1.

4.1.1 Previous state

In the current state, it does not feature many game elements, so we have decided to take a deeper look at it and try to make it a bit more engaging and fun for the end users. Current game elements:

- One-way following system with exposed number of followers (twitter model follower system)
- Ability to co-create event streams with other people
- Like system (people are able to like photos of other people)

Application rewards creativity and motivates people for curiosity and excitement about things surrounding them. Moreover, application is encouraging people to co-create their eeves (referring to specific photo sets). This makes the experience more rewarding with each new user participating and co-creating an eeve.

These mechanics are perfectly fine and encourage people to connect and create meaningful eeves in order to attract more followers and likes. Let us take a deeper look at how we could engage users even more and thus create more engaging experience, that would, in the long run, retain more users.

4.1.2 Mechanics

Iteration 1

When we first started to design a compelling gamification of the system we started off with some of the more usual game mechanics and basically implement Foursquare-like model of gamification that relies heavily on external motivation of users:

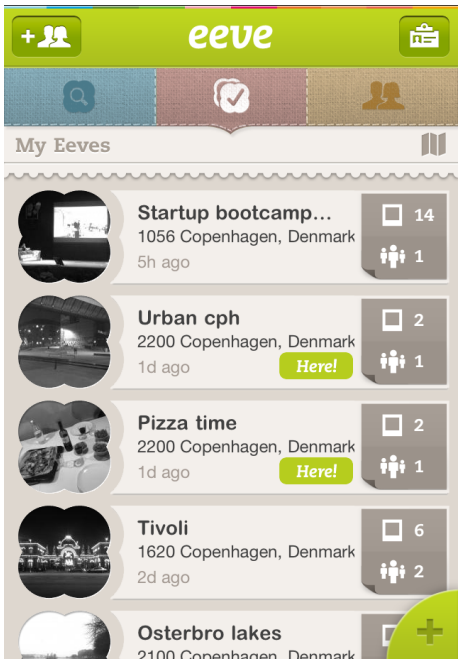

Figure 4.1: Eeve's main screen with the personal feed of eeves

- points,
- achievements (in the form of badges),
- leaderboards.

Points

We proposed to use a "normal" experience (XP) point system that would only store the point value for the last week and then reset it weekly to level the playing field between users. Points would be allocated for:

- creating an eeve,
- participating in a trending eeve,
- points for every like on the user's photo,
- points for every share user's photo receives.

Achievements

We would then allocate different kind of badges to users; an example would be for starting the first eeve, starting the fifth eeve, starting a trending eeve etc. By giving out badges we would motivate users to create more eeves, do a different kind of eeves, invite more users etc.

Leaderboards

People love to see how they stand in relation to their peers. Thus, we would introduce a social leaderboard which would let users see how they stand in relation to their own social circle. Because Eeve uses a one-way following model (similar to the one of Twitter), the social leaderboard of each user would only feature mutually followed peers. This would positively encourage users to compete against each other and drive more engagement throughout the application.

Iteration 2

Then we took a deeper look at our Iteration 1 plan and realized that it was superficial and looked a lot like something we (and probably also users) have seen before. Even more, this proposal looked like a separate layer on top of the existing core product and in the end would not feel like a natural part of the application. We went back to the beginning and found out that we wanted to make eeve-ing (the process of using the application) in its core more fun and tap into users' intrinsic motivation.

Points

First of all we wanted to get rid of the normal points. They feel used and have kind of grown old with users. Instead, we came up with the idea of eeveies, special points that would be allocated to users in only 2 cases: when they take a photo or when someone, in one of the eeves they started, takes a photo. This metric (eeveies) would then be shown on user's profile.

Because there is a high possibility of gaming this kind of system, we would add some constraints:

- User cannot get more than two points if he is the only participant in an eeve.
- User cannot start another eeve if there is already an eeve started by him at the given location.

From the testing we had done, we concluded that it was great fun seeing people participating, taking photos and commenting in an eeve you started. So why not encouraging this behaviour even more? Moreover, we thought of how to incentivize people to be more active and joining existing eeves at a given location (e.g. conference, night club, stadium etc.).

We came up with a system that would promote both an eeve starter and the user with most eeveies in the current eeve. The latter user would be called "Rocker of the eeve" and the former one would be named "Starter of the eeve". This mechanic would encourage users to participate in more eeves and collect more eeveies. It would also greatly expose and reward active users by featuring them on top of the eeve in the application and on eeve feed dedicated URL on the web.

Let us not forget the other important point metric, which is similar to the one on Twitter: because of the one-way following system we, of course, have a parallel metric of followers, which we emphasize and show in the specific user's profile.

Levels

Because the system would reward active players, it would, as we know from the history of gamification, encourage players to try and game the system (users would, for example, create a few different artificial users and then engage with themselves within an eeve to earn more points). Thus, it would be particularly relevant to implement an even tighter control over the game and possibly introduce levels. There could be only a handful of levels in the beginning, but the important one would be the step from a novice to

a beginner, which would require the user to first create a handful of eeves and have at least 15 followers in order to be able to contend for the "Rocker of the eeve" spot. This way we would be able to mitigate the problem of gaming the system by creating artificial users and self serving meaningless eeves.

Social engagement loop

Let us refer also to the Social engagement loop mechanic that somehow combines all of the mechanics listed above.

1. Motivating emotion:

- Showing of the creativity and experience of the life.
- Exploring beautiful photos.
- Seeing plenty of other people (that the user maybe even do not know personally) join your eeve and co-create the experience.

2. Social call to action:

- Sharing, like-ing, creating a new eeve.

3. User re-engagement:

- Smart e-mail and iOS notifications when users' Facebook friends join the application and for all the interactions that happen inside eeves.
- Because of the tight integration with Facebook seeing your friends starting new eeves would lead users to participate again.

4. Visible progress/reward

- Eeveies from creating meaningful and crowded eeves.
- Getting new followers.
- Achieving the visual "Rocker of the eeve" and "Starter of the eeve" statuses.

4.1.3 Dynamics

We have designed the system in a way that we encourage or even enable some important dynamics. During the journey of using the application, the user experiences:

- Pacing: the user is not as powerful in the beginning as he is later on in the experience; he can not achieve, for example, the "Rocker of the eeve" status.
- Progressive unlocks: the user unlocks specific abilities while using the application and attracting a certain number of followers.
- Peer pressure: the more people at the same location use the application the more popular the particular eeve becomes within that city, making users that are close to each other bond together and compete with other eevers in the city.

4.1.4 Aesthetics

If we look at LeBlanc's eight aesthetic components, it is quite easy to see those of them that are evoked by current game mechanics:

- Sensation: the application is about taking and exploring beautiful photos, which make this one self-explanatory.
- Fellowship: the application encourages people to meet new people around them in the real-life and follow their friends and other interesting people all around the globe forming a new vertical social network.

- Discovery: by motivating people to go wild and start eeves in crazy, unique places it also taps into discovery mechanic as it allows people to look at the world through a different prism and directly see things/events that are happening around the globe.

- Expression: the application provides almost zero value if the user only explores and does not create content; by creating photos and eeves it allows user to both express their creativity and get insight into what kind of photos get more feedback, which in the end makes his learn about good practices and improve.

- Submission: the application allows players to step back in time and recreate past moments by looking and exploring his personal feed of eeves.

4.1.5 Conclusion

In the end, the application creators that we have worked with decided to spin the whole application experience in the direction of non-event based photo sharing application with heavy emphasis on sets of photos created by a single user. This particular version of the application, that we have been involved with, never got into the App Store. It would be great to see this implemented as it was our first gamification project, but the life of a young startup sometimes involve changing direction quickly.

The application is now known as Popset, (`http://popset.com/`) and the team has moved to Silicon Valley since our involvement in 2012 and was a part of the most successful startup accelerator Y Combinator (`http://ycombinator.com/`).

4.2 Gamifying painting application – Psykopaint

Psykopaint is a different kind of painting application. It allows users to create stunning paintings from their photos. Users come to the site (`http://psykopaint.com`) and have a choice of going into one of the two biggest part of the site – the public gallery, where painters around the world constantly post new paintings or the actual painting tool.

What makes Psykopaint so truly special is its unique painting style – users not just simply apply a filter to the painting, but choose one of the many brush styles (like Monet, Van Gogh, Picasso) and then paint over the photo. When the user make a brush stroke, the application automatically samples the colors of the underlying pixels of the photo, so that the user does not have to care too much about the colors and can rather focuses on his style and expresses the creativity.

We have been involved with it for the past half year and seen it grow from around 600,000 unique visitors/month to more than 1,000,000 unique visitors/month. During the time, we have implemented or improved several gamification elements, all with the core mission to augment the experience and make it even more fun, but still staying true to the core spirit of the application.

4.2.1 Previous state

Psykopaint was already a thriving community of painters before we joined the team, but we quickly noticed several gaps in the experience from the usage statistics. Further on, we conducted a few user surveys and interviews and found out some other problems in the user experience that needed addressing.

Most notable problems were connected with lack of real onboarding process; users were simply thrown into the experience of the painting tool without much explanation of what it is and what they can expect. Moreover, they started painting and were met with a ton of different options, which

made it really difficult for novice users to be comfortable using the application. In the gallery (`http://psykopaint.com/gallery.html`) part of the application, we have noticed some activity, but realized that there could be much more users participating there, commenting on each other's paintings etc. We slowly iterated during the past few months and implemented almost all of the desired game mechanics and are finally able to see the results of it. Let us look at what we did.

4.2.2 Mechanics

Points

Psykopaint used to have a simple point system in the application even before we started working on it. However, we just recently implemented redeemable points (they are called Psykopoints) in connection with more options to spend those points on. It makes little sense having a point system without lots of meaningful things to spend those points on. We also restricted some parts of the application that used to be free. In the beginning of *user journey* now there is only one brush option unlocked and users have to buy the rest of them. This whole new experience was started for a few reasons: people found it difficult to navigate through all different options in the beginning and we wanted a way to be able to incentivize positive behaviour with different kinds of challenges and reward users with meaningful points afterwards.

One of the behaviors we wanted to improve was the *social referrals* – the social action that users take to recommend the application to their peers. After the recent introduction of Psykopoints, the number of social referral have risen dramatically (by way over 500%) and has been seen as one of the main factors for the strong overall growth. Two screens from Psykopaints current point system are represented in Figure 4.2.

Onboarding

As mentioned previously, we have noticed the same problem with lots of users: they would come into the application and leave it shortly after.

Figure 4.2: Two screens depicting the Psykopaint point system

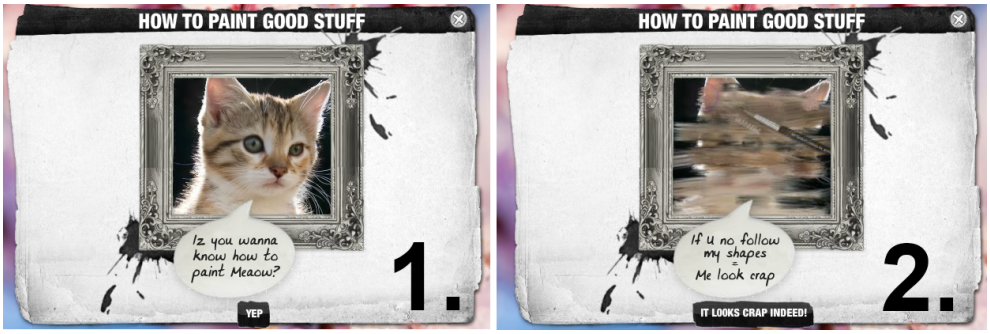

Figure 4.3: Simple interactive onboarding process

We figured out that one of the most apparent reasons for this behavior was the lack of tutorials. Thus, we recently implemented a remarkably simple onboarding tutorial that teaches users how they should paint to make their first decent creation. We can see a screenshot of this interactive video in Figure 4.3.

Moreover, we changed the first view users encounter when they come into Psykopaint – previously it used to be just the dialog box with two options about how to start their painting. Now, we have changed it now into a beautiful view of some of the best works from the public gallery and an option to explore the gallery before even starting to paint. The reason for this change is quite straightforward – we found out that many people needed an additional push to start, they needed to be inspired. And what better way

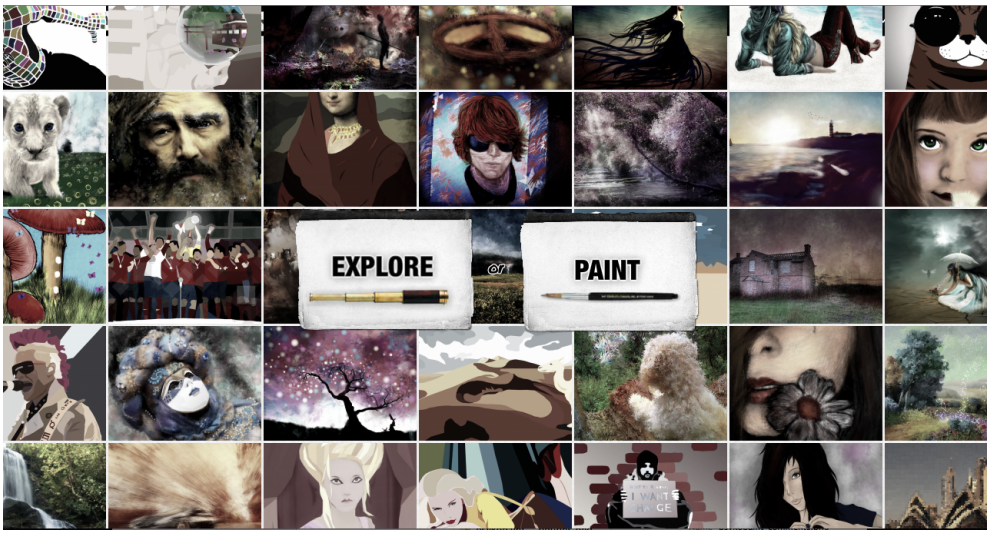

Figure 4.4: New first screen of the Psykopaint application

to inspire them than to show them a few featured works from our gallery. Screenshot of the first screen is seen in the Figure 4.4.

We are fully aware that the current onboarding steps are not good enough and should be improved even further. We have learnt from our experience now that the limiting functionality is not bad at all, and we plan to take this even a step further by delivering a truly directed first experience. One of ideas is that users will not be able to choose their photo to work on for the first few times. They will rather be presented with a predefined one, but will be given a step-by-step tutorial that would almost certainly produce great result. Users crave for self-accomplishment and to be able to create something good looking from (almost) nothing, delivers a great deal of satisfaction and should improve Psykopaint's early retention rates a lot more.

Achievements

We have started thinking about achievements a long time ago and had a few of them implemented even before we started to work on the gamification overhaul. We needed the achievements to be able to reward users for their

successfully accomplished challenges (alongside points). We struggled quite a lot on how to present achievements in the most meaningful way and not just with some sort of badges. We were pondering several ideas: one was to reward users with virtual replicas of the most famous real-life paintings, the other was to have a different kind of carnival masks from all around the world.

In the end, we decided for a bit different solution: 3D objects that would fit well on the user's' virtual trophy shelf, which can be seen in Figure 4.6. These objects were inspired by various pop art works (Warhol's portrait of Marilyn Monroe), old fashion art works (Dali's clock), some are even just random super stereotypical objects depicting our modern culture (a hamburger, funky sunglasses). This is an example of well thought-through achievement system that feels remarkably natural to the whole experience.

Challenges

Challenges are not directly visible in the current system, but are implied through the list of achievements, where we specify what the user has to do to achieve a certain achievement. Current achievements include, for example, an achievement for having a painting that has been viewed 50 times. Thus, we incentivize sharing to other social networks to encourage their other friends to come in to Psykopaint and see the painting. Other challenges require users, for example, to use at least 3 different brush styles (here we try to incentivize exploring different painting options to develop a broader set of painting skills) etc.

This is a part of the experience that could be further developed so that the user would always know what are the three to five things he can do at any given point to progress in the experience.

Another particularly appealing challenge type that we might implement in the future would be contests. We have actually done some testing on user behavior by running simple painting contests with a predefined original picture on our Psykopaint Facebook Page (`http://facebook.com/psykopaint`).

Figure 4.5: Picture of the virtual trophy shelf with some of the achievements

This allowed us to test the behavior and effectiveness of this game mechanic, without implementing it in the real application. Results of the first test of users' interpretations of the famous painting Mona Lisa were extremely encouraging; more than 40 different paintings and hundreds of likes within Psykopaint. We noticed some really interesting social behavior like users encouraging each other to paint some more, users posting links to their paintings on their Facebook profiles and asking their real life friends to join Psykopaint and help them win the contest etc. Afterwards we decided to implement this kind of contests into the Psykopaint itself, so we will always have a contest running there. It will serve the purpose of engaging users in new kinds of social interaction and inspire them to enhance their skills to beat the other painters.

Social engagement loop

1. Motivating emotion:

- Unlocking new achievements.
- Painting something from scratch and receiving positive feedback on it.

2. Social call to action:

- Sharing, liking, commenting, rating the paintings (Figure 4.6 and Figure 4.7).
- Posting a new painting to the gallery.
- Repainting a certain painting (users start a new painting from another painter's photo; we plan to make a reference back to the original, which will make it really rewarding to see that your painting got many repaints – kind of Psykopaint's version of a retweet from Twitter).

3. User re-engagement:

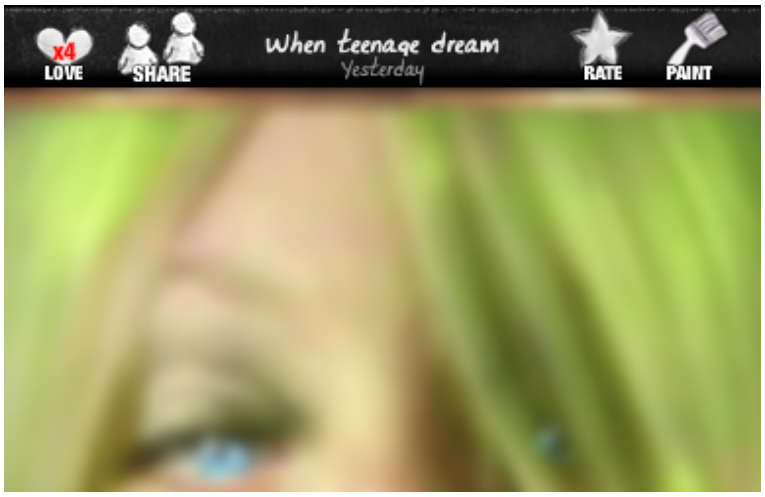

Figure 4.6: Example view of the social calls to action

- E-mail notifications for new likes, comments, regular newsletters.
- Users feel special drive to paint something new to delight their followers.

4. Visible progress/reward

- Achievements: the more engaged users are the more different achievements they are able to collect.
- Points: users are collecting points and with them unlocking more and more of the functionality of the system.
- Followers: users are gaining a following of fellow painters who admire their work and love to see their paintings appear directly in their feed.

4.2.3 Dynamics

The system that we have designed enables users to experience several important game dynamics:

Figure 4.7: An example of comments on the painting in the gallery

- Pacing: people slowly progress to the mastery within Psykopaint; first time they visit the application they experience a nice video tutorial, later on they are able to watch Youtube video tutorials, we drop them small hints on how to become better painters when they reach certain breakpoint.

- Appointments: we have a few achievements that are given out only once a year – for example on a Picasso's birthday we have a special secret Picasso achievement.

- Progressive unlocks: we tried to architect an achievement system that rewards users for unexpected behavior; painting with a certain brush, painting on a certain day, combining different styles etc.

- Reward schedules: certain achievements work on a so-called *fixed ratio reward schedules*; for example, log in ten times within five days.

As we now look back at designing the system, we can clearly see that we almost encompass all the dynamics from Section 3.2 (except peer pressure and dynamic systems). We did not design the system with all of them in mind, but it turned out that they can be immensely powerful motivators for re-engagement and retention of users. In the future, we might try adding some mechanics that would encourage even the peer pressure dynamic; for example, by having a challenge of breaking a world record of different interpretations of a famous painting done in a single day. This way people would have a clear, very big goal, that they could only reach and be a part of if they helped gather and encourage an even bigger group of painters.

4.2.4 Aesthetics

Let us look at LeBlanc's eight aesthetic components to see what kind of fun we might anticipate from users:

- Sensation: the application helps and encourages people to enjoy in creating art without any previous academic knowledge needed. It also

encourages them to explore other painters' works, learn from them and appreciate their work.

- Challenge: the applications slowly tries to challenge users in creating better and more unique paintings – recently also to compete against other users in creating the best work of art based on the same original photo.
- Fellowship: the application has recently made steps into becoming an implicit creative social network with more ways to appreciate the works in the gallery and with the ability to follow other users' works.
- Expression: this component is at the core of Psykopaint as the application enables users that previously could not express themselves artistically to be able to finally produce something they can be proud of – we see it even being used in the art therapy for these exact reasons.

4.2.5 Conclusion

We have spent much more time working on the gamification layer of Psykopaint than on the one of Eeve. Gamification now feels as an integral part of Psykopaint, which we are really proud of. It has also proved to be effective in some of the metrics Psykopaint previously struggled with (for apparent reasons we can not disclose the absolute numbers, but let us try to present some of them in terms of percentages):

- The retention rates from first to second week have gone up more than 120%.
- Average time on site have gone up by more than 60%.
- Average number of comments per painting in the gallery have gone up by 130%.
- We have increased the number of paintings posted in the gallery per week by more than 200%.

It is crucial to note here, that not only gamification techniques have helped drive this numbers and that we have also done some performance and navigation improvements, but are responsible for the most of growth the application has seen recently.

Chapter 5

Existing gamification platforms

Gamification has truly taken off lately and progressed into a real industry with some of the biggest players raising significant multi-million dollar rounds of venture capital. What does this mean? Big gamification platform vendors like Badgeville, BigDoor, Bunchball and iActionable have proven that online gamification actually works. It has proven to raise engagement, retention and, in the end, bring in more money for web publishers and application developers.

Especially Badgeville, BigDoor and Bunchball are interesting for developers looking to enrich their applications with the use of gamification. They are all building a different kind of SDKs (*Software development kit*) to help developers do that with the smallest possible effort and applying the industry best practices. So let us look at what some of them can offer to software developers and architects.

We have tested implementation of Bunchball in the process of gamifying application Psykopaint (which we talk about in Section 4.2). We have decided not to use it in the end as we wanted a bit more custom design and experience and the price point that they offered did not make financial sense for us (at the traffic we had at the time it would cost us slightly more than $1,500/month).

Nevertheless, we have gained some knowledge into how gamification plat-

forms work and for what kind of purposes they can be utilized and will try to present it in this chapter. We have learnt that it is crucial for software architects to set goals and methods of how to quantify and measure those goals along the way before proceeding with applying a gamification solution.

The remainder of this chapter is organized as follows. First, in Section 5.1, we discuss the necessary steps every gamification implementation team should consider before implementing any of the above mentioned platforms. In Section 5.2 we present high level implementation example of gamification platforms Badgeville. In Section 5.3 we then look at the other two platforms: BigDoor and Bunchball. Section 5.4 sums up the commonalities between all three platforms. We look at the most promising open source platforms in Section 5.5. Conclusions to this chapter are given in Section 5.6.

5.1 Gamification platform project plan

Here is a list of things that should be considered before starting the implementation of a gamification platform project. It involves knowledge and work of both, an architect and a developer. The architect should understand the underlying business goals and gamification benefits and caveats, the developer, on the other hand, should get to know the platform specifics (API structure, widget configuration etc.).

This can sometimes even be done by one person, and this is the reason we are producing this thesis in order to introduce and educate software developers and architects in some basic knowledge of gamification.

Things to consider when starting a gamification project are:

- Clear definition of the business objectives and metrics you want to improve (architect responsibility).
- Look at your current product, distill the core of it and try to locate the right kind of behavior that you would like to see in your product (architect responsibility).

- Look at the aesthetics and the kinds of fun you would like users to get themselves immersed with (architect responsibility), selection of appropriate game mechanics that would drive the desired behavior from the previous points: levels, badges, points (architect responsibility).

- Most of these gamification platforms have the so-called publisher modes, where architects can identify and prepare the desired taxonomies and models (architect, developer responsibility).

- Preparation of the methods that would call the platform's developer API (*Application programming interface*) – they are all RESTful (*Representational state transfer*) nowadays (developer responsibility).

- Preparation of the methods that would accept the platform data in JSON (*JavaScript object notation*) format and then further parse it in the appropriate application format (developer responsibility).

- Preparation of the server or client side logic to track the behaviour of users (developer responsibility).

- Configuration and installation of the platform specific widgets (developer responsibility).

- Study, exploration and automation of the possibilities of the analytics functionality on the platform's side (developer responsibility).

5.2 Badgeville example

We have a bit more experience with Badgeville (`http://badgeville.com/`) as we have helped Psykopaint (as mentioned in the introduction to this Chapter) implement a part of it for testing reasons. It is a comprehensive, but robust solution that allows developers and architects to build something very quickly from scratch and then further customize it depending on their needs.

Short implementation example

Steps that are needed to set-up a small Badgeville project (this is a more specific version of the steps in the general description). In parenthesis, we specify who is generally responsible for a particular step in the process.

1. Define the game mechanics that you would like to utilize (architect)

This is a planning phase of the project where the architect identifies the desired behavior within the application. Best practice is to make a list of all the different mechanics he plans to use and then specify hierarchy and custom taxonomy for each one of them.

2. Enter the mechanics in the Publisher Module (architect)

This is the architect's first contact with the Badgeville platform (for the given project). They have an easy to start and use creator for defining game mechanics and specific gamification business rules. For example, when a user does event X for the 10th time, reward him with a badge Y.

3. Get familiar with the Badgeville API (developer)

This is the step when a developer starts working actively on the project. It is important in this step to prepare the methods that would help connecting with the API and parsing the data. We will not list specific examples for such methods here as they are fairly trivial and depend on the programming language in use for the application.

For example, there can be a helper method that makes a request and takes any Badgeville API endpoint as a parameter, makes a request and returns the parsed data.

Let us also look at an example API call that registers an activity within the Badgeville platform:

```
http://badgeville_host/api/publisher_id/users/user_sess
```

```
ion/ user_badges?behavior=behavior
```

4. Register users and enable tracking of behavior with the API (developer, architect)

There are two ways to register users of the application and track their behavior within the Badgeville platform: one is by utilizing Badgeville's Javascript tracker, and the other one is by calling the API endpoints. Since we started this example project with implementing everything with the API, we will provide this solution (which is also more customizable in the long-term).

To register a user for the first time within the Badgeville platform, simply call its users/update_info endpoint:

```
http://badgeville_host/api/publisher_id/users/update_in
fo?email=user_email&display_name=user_name
```

This endpoint then returns a unique user identifier for the given user that you can then use in subsequent calls when triggering specific behavior events of this user.

5. Display various user relevant information in your application (developer)

It is important to have flexible options for showing the data to the user. You may want to show his user profile, badges, achievements, a leaderboard, activities etc. Most of this are possible to present by utilizing Badgeville's widgets or by calling the API and rendering the data manually.

In order to display information manually you just call the relevant API endpoint, store the result in a variable and present it to users. If you rather use the predefined widget here is an example javascript code snippet for it:

```
<script id="badgeville_widget_activities"
type="text/javascript" src="http://api4.badgeville.com/
api/444/widgets/activities?tags=hotels&session=current
usersessionid" >
</script>
```

Badgeville also offers visual customization of those widgets through its Widget Studio, where developers are able to set their own CSS styles.

6. Set up analytics (developer, architect)
We have learnt that analytics are a vital part of every gamification project as they enable architects to see their failed or validated assumptions from the real user behavior data.

There are 2 ways to set up analytics on Badgeville: one and simplest way is just to log in to the platform and check the analytics there. Most of the applications already have their own analytics dashboard so integrating these analytics into that existing platform is a reasonable option. Badgeville offers it through its analytics API.

This is a simple example of how easy it is to set up a custom gamification layer with the Badgeville platform.

5.3 BigDoor and BunchBall

5.3.1 BigDoor

BigDoor (`http://bigdoor.com/`) seems to be on the cheaper side of these solutions; they specialize in offering super fast deployment and are the cheapest solution out of the three we are presenting here. Their primary offerings are widgets (which can be easily deployed as a javascript snippet). It is a

For the advanced usage they of course offer also a REST API, which then allows their customers to build a more custom-tailored solution. Apart from the other two competitors on the market, they also seem to be more focused and niche and are specifically providing solutions that are consumer facing. This is different to BunchBall and Badgeville, which also help big organization gamify their internal systems such as CRMs and productivity tools.

The interesting bit in BigDoor is that they seem to be the only solution

currently out of the big three that also offers a free, so-called Lite version of their product (for developers and architects that have less than 25,000 monthly active users). This version of the product allows software architects to easily deploy a solution (after they have been approved by BigDoor); loging in to their dashboard, defining a few challenges and copying the code snippet of Javascript code on to the website are enough for a simple gamification solution.

5.3.2 BunchBall

Bunchball (`http://bunchball.com`) is the oldest (it was founded back in 2005 – years before the term gamification became popularized) of these platforms and seems to be very well positioned in the enterprise sector of the market. Their customers are established corporations in different sectors like Toyota, Adobe, Cisco, Hasbro, T-Mobile, ESPN etc.

It may not be the most suitable solution from our perspective as independent software architects and developers. The solution is a bit too complex and expensive. Their three larger product offerings are:

- Nitro: they call it a gamification engine and it is basically a full blown gamification platform that features widgets, analytics, easy to use builders for badges, challenges and a comprehensive API, which makes it possible to integrate it with the existing products.

- Spark: more easy to set-up solution with more or less predefined challenges, newsfeeds, leaderboards.

- Fuse: mainly an internal solution to engage and help motivate the workforce; integrates with SAP, Jive, Salesforce and tens of other SaaS tools and on-premise systems.

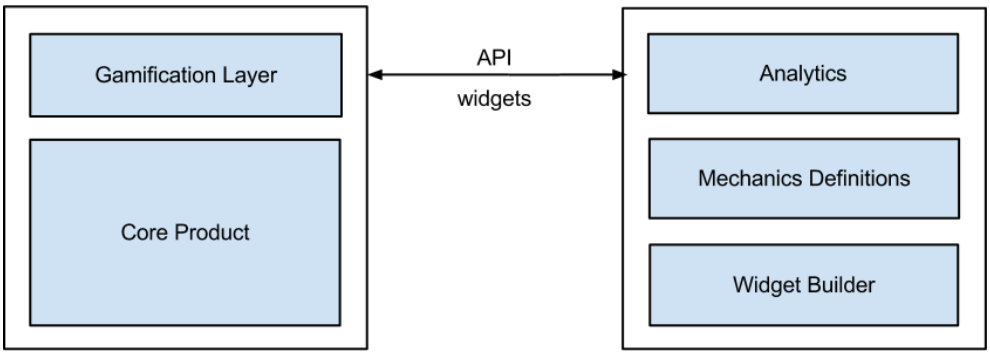

Figure 5.1: Diagram depicting gamification platform architecture

5.4 Commonalities between the platforms

It looks like that at their core all three solutions, that we describe in Section 5.3, have quite similar gamification engine. They all cover some of the basic game mechanics, have accomplished analytics systems, offer the API for developers and easy to use and configure widgets. That is why we did a diagram that depicts a common gamification platform architecture in Figure 5.1.

Bunchball and Badgeville have obviously also located a very substantial need and a lucrative market in the enterprise as they both offer solutions that gamify enterprise cloud applications, including Salesforce (`http://salesforce.com/`), Jive (`http://jivesoftware.com/`), Yammer (`http://yammer.com/`) etc.

5.5 Open source gamification platforms

5.5.1 Mozilla Open Badges

Mozilla Open Badges (`http://openbadges.org/`) is a really fascinating project that has been widely promoted and started by Mozilla Foundation. It is not a comprehensive gamification solution, but it is a great project for many reasons:

- It has a clear incentive for the issuer (software application) as it motivates its users.

- It is mainly aimed at educational applications.

- It has a clear incentive for the earner of the badges as he can grow and display his own portfolio of badges around the web and will, if it evolves, serve as a great hiring tool.

The API for the "issuer" is really simple and is available at: `https://github.com/mozilla/openbadges/wiki/Issuer-API`.

5.5.2 Userinfuser

Userinfuser (`http://code.google.com/p/userinfuser/`) is a gamification platform that has been developed by a company named CloudAptive (`http://cloudaptive.com`), which specializes in gamification and provides their own gamification platform. They have made the platform Userinfuser public and open sourced and everyone can decide to take and modify the code and host its own gamification platform. The game mechanics currently supported are the basic ones: points and achievements.

They provide the whole code suite: client side API libraries for PHP, Ruby, Python, Java, and a server side code to deploy on App Engine or Appscale. For displaying the data on the site, the client side API provides calls for iframe widgets that can then easily be implemented in the application.

The last code changes were made in June 2011, and there have been less than 400 total downloads of the codebase, but we hope that the CloudAptive team will continue with the development of the platform. It is the most comprehensive free and open-source gamification platform available at the moment.

5.6 Conclusion

As many times before in the history of technology innovation cycles the gamification land grab in the past years has attracted many companies. Many have tried and only a few are now left in the market – all of them with tens of millions of venture capital funding in the pursuit to make gamification a significant and viable market.

Architecture and implementation-wise all of the three competitors are quite similar with slightly different approaches depending on their current market focus (big media companies, enterprises etc.). It is obvious that there is a enormous need for maximizing the user engagement on the ever more cluttered internet and these companies are unquestionably well positioned to help solve this problem.

Besides the external facing engagement efforts, some of these gamification platforms have also identified a big enterprise employee engagement market. This is especially apparent with Bunchball and Badgeville; both of these companies offer various integrations with cloud and on-premise softwares, such as SAP, Salesforce, Jira, Jive and others. They bring all these services together with a coherent and unified gamification layer on top of them. We can imagine this kind of service working exceptionally well in the ideal scenario where this kind of solution is completely internalized and well communicated throughout the organization.

The final observation is that although these companies are building large businesses, it is hard to see that these one-stop-shop solutions would be appealing for user and developers. That is, if the application architect does not use time and resources on locating the core of his product and gently augment it with specific case-based solutions. In this regard, we also see the above gamification vendors as not only providers of the products but also consulting and implementation service provider in helping their customers really understand the impact. Implemented incorrectly, they can do more harm than good. The problem from our standpoint as system architects is then not only in the generic approach to gamification, but also in the high

price point as we have experienced when inquiring at Badgeville regarding our Psykopaint project. Consequently, as we can see in Chapter 6, none of the prominent gamification use-cases has used the before mentioned platforms.

Chapter 6

Promising use-cases

So far we have looked at what is gamification, how we can structure it with a framework and worked on it on a few examples. Ggamification would not be so popularized without a few success stories.

There are now hundreds of software applications outside of gaming context, that are powered by the learning of games. We can find those applications in various fields: personal productivity, enterprise productivity, healthcare, education, customer support etc.

Obviously, there are some core gamification principles that are effective. In fact, Bunchball gamification platform states that its Spark product, for example, increases page views by a factor of two and increases the average time-on-site by 85% [**?**]. Also, Richard Taylor, SVP (*Senior vice president*) for communication and industry affairs at the ESA's (*European Space Agency*) research [**?**], found out that 70% of major employers utilize interactive software and games to train employees (in 2008). The number is probably even higher today.

The four biggest gamification platforms (Badgeville, BigDoor, Bunchball, Gigya) have altogether $110 million in venture capital investments. This is according to Crunchbase (`http://crunchbase.com/`) data on date July 12th 2012 and it makes gamification one of the biggest venture investment markets in the recent years.

Moreover, highest profile spokesperson have mentioned gamification recently. Former vice president of the USA Al Gore said in his speech at the Games for Change conference in 2011 that "games are the new norm" [**?**]. Not only that, he went on and implied at gamification even more directly: "... game interfaces and scoring rules have become standard ... the gamification trend is really, extremely powerful." He believes that games and gamified experiences can make a massive difference in the world; bringing social good and helping solve the toughest problems of the society.

The other famous speaker talking about gamification recently was J. P. Rangaswami, Chief Scientist of Salesforce. He had a keynote on gamification at the ReadWriteWeb's 2Way Summit in 2011 and talked more deeply on why gamification is the future of work [**?**]. Accidentally or not, just a few months later Salesforce acquired one of the biggest players in the "gamification of workplace" space – Rypple [**?**].

The remainder of this chapter is organized as follows. We want to identify some of the most promising and biggest sectors in the industry, where gamified solutions are already proving to be very effective. We start it off with personal productivity in Section 6.1, then continue exploring the solutions in the enterprise productivity (Section 6.2), healthcare (Section 6.3), education (Section 6.4), customer support (Section 6.5) and Q&A (Section 6.6). We continue this chapter by looking into a really interesting space, that could maybe have the biggest impact of all – crowdsourced initiatives in Section 6.7. We conclude the chapter by describing a few other uncategorized solutions in Section 6.8.

6.1 Personal productivity

We always wanted to be more productive or find new ways to keep ourselves focused and aligned towards a single goal. We help organize our day-to-day operations with tons of applications, so it makes perfect sense that somebody has tried gamifying a to-do application.

EpicWin

One of the best examples is an iPhone app called EpicWin (`http://rexbox.co.uk/epicwin/`), which turns the tedious task of our daily life into a quazi RPG style gaming environment. It naturally relies heavily on challenges and leveling.

SuperBetter

Another interesting example that we have found is an iPhone application called SuperBetter (`https://www.superbetter.com/`). It helps people going through life and helping them improve the physical, mental, emotional and social aspects of their lives. It is designed around a series of challenges in each one of the before mentioned categories and it strives towards the crucial goals called epic wins. Epic win is, according to Jane Mcgonigal, who coined the term [**?**], "an outcome that is so extraordinarily positive you had no idea it was even possible until you achieved it ... it was almost beyond the threshold of imagination and when you get there you are shocked to discover what you are truly capable of." We have been trying it out and it is a really immersive experience designed around constant feedback and slow, but sustainable progression curve.

6.2 Enterprise productivity

Two of the biggest gamification platform vendors that we mentioned before have also identified and built products that are aimed at increasing productivity in the enterprise.

Bunchball Fuse

Bunchball is offering a stand-alone product called Bunchball Fuse (`http://bunchball.com/products/fuse`), which integrates and pulls in information from more than 80 different cloud and on-premise applications.

Thus, it creates a stream and overview over different activities in an enterprise company and offers a way to increase productivity around common goals, real-time feedback and social collaboration.

Badgeville products

Badgeville, on the other hand, offers a line of products (`http://badgeville.com/solutions/employee-facing`) that integrates into existing applications that are already in use within an enterprise (Yammer, Jive, Lithium, Salesforce, Zendesk etc.). The products' core engagement systems are similar to the one of Bunchball Fuse.

Rypple and Worksimple products

There are two other companies with attractive products in this space. First one is Salesforce, with the before mentioned product Rypple (`http://rypple.com/`). The other one is a fairly new product by company Worksimple (`http://getworksimple.com/`). They both build separate standalone products that try to engage the team in a way of setting clear goals, providing better performance reviews and analytics that can then be used to assess the performance of the teams and individuals.

6.3 Healthcare

There are a few different categories of products (and companies associated with them) that are trying to solve the healthcare issues. Currently, they are all approaching it from a point of trying to get people to be in a better shape and consequently improve their health. The solutions differ from the markets they are addressing, so let us look at a few of them more carefully.

6.3.1 Enterprise wellness program platforms

Keas

Keas (`http://keas.com/`) is the most known company in this sector. They provide social wellness incentives to help employers essentially pay less in insurance costs and having a fitter workforce. Employees form teams and set goals based on their personal fitness level. Then those compete against each other for cash and other prizes. Thus, it creates a positive peer-pressure environment and provides team members with constant challenges. Keas has been recently cited to increase user engagement by 100% [**?**] after they implemented different gamification mechanics such as challenges, points, levels and achievements.

Shapeup

Another company in this space is called Shapeup (`http://shapeup.com/`) and offers quite a similar approach to enterprise wellness program by enabling companies to establish a private wellness social network.

6.3.2 Individual fitness reward applications

Healthrally

Most of the individuals want to get in shape on their own, so the other approach is to address them directly and offer them different kind of incentives. That is the approach of several companies: Healthrally (`http://healthrally.com/`), for example, enables people to tap into their real-life social network and leverage the social pressure and external rewards to accomplish fitness goals or to encourage others to do it.

GymPact

Another company called GymPact (`http://gym-pact.com`), which we have personally collaborated with at Psykosoft in Boston at the Techstars programme, offers people to put their own money on the line if they do not

accomplish their desired goals about going to the gym. They claim that their gamified system has been so effective that people accomplish 90% of the fitness goals.

6.3.3 Individual tracking applications

This is about to become a billion dollar industry on its own. These products usually comprise a tracking device and an application to sync the data from a tracking device, get intelligence on top of it and compare the results with peers. The devices usually consists of a pedometer or an accelerometer (the newer models) that measures the motion of an individual throughout the day.

Fitbit

One of the first and most known applications is Fitbit (`http://fitbit.com/`). Its gamification side is not so emphasized, but it does leverage the peer pressure dynamics by exposing some of the user statistics publicly (depending on user's privacy profile). Potentially exposed statistics include calories burnt, miles travelled, steps taken in a day etc.

Nike+ line of products

Nike has a strong IT engineering department and has long been leveraging gamified systems in order to help runners achieve their goals. They have started with the Nike+ devices (`http://nikeplus.nike.com/plus/`) and are now further driving this change with their FuelBand product (`http://nikeplus.nike.com/plus/products/fuelband`).

The most intriguing part is the application that helps runners understand and drive the behavior change after they sync their running or walking data. It connects users with other people that are facing similar challenges, they organize large, real-life challenges (we have experienced a real city challenge for residents of London in early 2011), gives them real-time feedback even while running etc. We can say from our own experience that the first Nike+ product had been vital when we started to run after a few years of not so

regular exercise.

Zamzee

Another fascinating example is a product called Zamzee (`https://www.zamzee.com/`). We call it "a Fitbit for kids". They essentially tackle one of the most obvious problems of the US and beyond – obesity. They are tackling it from the ground up – from kids. Their research has proven that kids wearing Zamzee did move 30% more on average than those who did not. In its core, it is heavily based on the levels mechanic, with quick leveling and badges that are genuinely appealing and effective with kids.

Wii Fit

This list would not be perfect without possibly the most widespread phenomenon of all the above – Wii Fit (`http://wiifit.com/`). Wii Fit is a remarkable accessory for the gaming console Wii that helps users get in shape by providing the most interactive fitness experience ever. Coming from a gaming company Nintendo (`http://nintendo.com/`), it is of course equipped by some of the most advanced game mechanics (customization, points, levels, achievements etc.). Wii Fit and its successor Wii Fit Plus have combined sold more than 43 million copies worldwide which puts it among the best selling franchises of all time and shows the scope of success and opportunity in this market.

6.4 Education

Education is one of the pivotal and most obvious use-cases for gamified applications. People have been dealing with this problem for quite a few years now, actually the first so-called edutainment game Carmen Sandiego originates from 1983 [**?**]. The key question here has always been how we stimulate people to spend as much time on learning (and be at the same time so engaged in it) as they spend playing games. Right now

this market is blooming, so let us look at some of the best executed examples.

Khan Academy

Khan Academy is a brilliant source of educational videos but also features a new sub-site (`http://khanacademy.org/exercisedashboard`), which enables users to graphically represent their knowledge of math and progress through it. It steadily challenges users from basic problems of addition and subtraction and all the way to differential equations by giving out points and skilful use of leveling. Besides this, they have recently started pushing out achievements throughout the site – for example, when a user first sees an educational video, he gets a small Meteorite badge.

Typewriting applications

There are a whole lot of applications that have turned a terribly tedious task of learning how to type properly with ten fingers into a game. We have ourselves learnt how to type with an application called Typingweb (`http://typingweb.com/tutor/games/`). It features different styles of games where they first teach users how to position their fingers properly and then slowly builds up their WPM (*Words per minute*) metric by playing silly games and pressing appropriate letter on request. As a trivia, well-known game publisher Sega published a full-fledged PC game title called The Typing of the dead in 2001, which was a type-writing tutorial dressed in a proper PC game.

Programming learning applications

There have been a massive influx of interactive gamified applications for learning how to program. They are all based on more or less the same principles – interactive courses, online editor, small and easy challenges, achievements and constant feedback. Some of the most popular platforms currently are Codecademy (`http://codecademy.com/`) and Treehouse (`http://teamtreehouse.com/`). Codecademy alone, for example, has passed

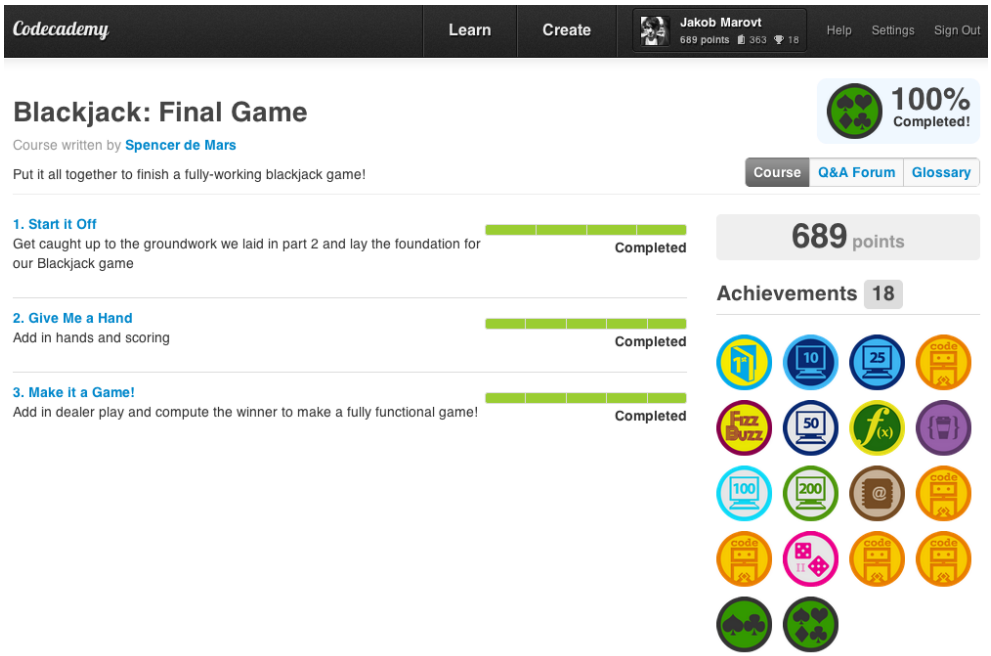

Figure 6.1: Our Codecademy Javascript course dashboard with achievements

1,000,000 users in its first five months of existence [**?**]. We have done the whole JavaScript course in it in a matter of a few days and can say that the experience is impressive, but lacks a bit of depth. So, for now, it can not replace real programming experience. However, we would highly recommend it for any newcomer to programming as it steadily presents users with the basics of programming in a fun and entertaining way. We can see the main dashboard in Figure 6.1.

6.5 Customer support

Recently we have seen some remarkable initiatives that have found out that utilizing the community can be tremendously effective in helping other users solve their problem. This is especially useful when dealing with a huge community of people, and when the cost of having real people responsible for answering user queries can be substantial.

Giffgaff

Giffgaff (`http://giffgaff.com/`) is an "alternative mobile operator" in the UK. They are one of the few mobile operators in the world without a call center and instead rely heavily on their community of "gaffers" to help and support each other. They have built a special community forum which enables people to answer other's questions, get rewarded for it in karma points and then redeem those points for real life minutes or SMSes (*Short message service*). This is an appealing positive reward system and it seems like it is working for them – the average response time is outstanding 90 seconds, any day or hour of the week [**?**].

Uservoice

Another great example in this category is a customer service platform Uservoice (`http://uservoice.com/`). They have build into their products various ways of engaging the community and particularly the support team. In comparison to Giffgaff their product does not rule out the in-house customer support team, but rather help them to be more productive and engaged by giving users the option to give out karma points to the support staff answering their issues and requests. Additionally user are also actively engaged in building the future roadmap of the products that utilize Uservoice by being able to vote on various feature requests.

6.6 Q&A sites

StackExchange

StackExchange (`http://stackexchange.com/`) is a Q&A site that started off with and is still especially well-known for its core product aimed at programmers called StackOverflow (`http://stackoverflow.com/`). It became immensely popular since inception in 2008 and they have deliberately used various game mechanics to make it so. They have beautifully implemented

the user voting mechanisms, karma points and leveling mechanics, which have made the network the "go to" place for programmers and beyond. In the recent years, they have started to add more domain specific Q&A networks, for example, for users of Mathematica, developers on the Facebook platform, system administrators etc.

Quora

Another very well known example of Q&A site is Quora, which is especially well-known for its super high quality of answers. This fact has a little bit to do with their ability to attract most of the high profile people in the software industry. The remarkable fact is that the quality has stayed pretty much the same even now, when the site has grown beyond the startup oriented community. It has implemented various game mechanics from up and down voting to the point system, the following system (similar to the one of Twitter), which is itself a point metric on its own and real-time feedback via a very clever notification system. They also have a karma point system that rewards users for answering certain questions that others want answered, which then in turn helps those users to get the questions they want answered quicker (by promoting questions to a bigger audience of people).

6.7 Crowdsourced initiatives

Internet has connected the humankind in an unprecedented way and made it possible to connect millions of people interested in a certain, sometimes even obscure, topic from all around the world. As we have discussed in Section 3.1, people seem to be sometimes really motivated in performing a task that unites them with others and seems to be far beyond their individual capabilities (almost incomprehensive for a single human being). If the system is designed well enough, this kind of mechanics can yield tremendous success, so let us look at some of the examples.

Twitter translation effort

There have been a few translation initiatives of internet services, but one that stands out recently is the one done by Twitter (`https://twitter.com/about/translation`). They have managed to gather a group of 350,000 people working together on a single problem of translating Twitter into as many languages as possible. They have set up a special system, where users can contribute small bits of translations. In exchange they are a part of something really big (translating one of the most important social network into their native language) and earn a distinctive and scarce visible achievement on Twitter – the Translator badge. Moreover, for every snippet of translation, users are awarded experience points and are progressing through different translator levels the more translations they provide.

Foldit

In 2008 researchers from University of Washington released an application, which looks more like a game. It is called Foldit (`http://Fold.it/`) and enables users to fold the structures of selected proteins as best as they can. Users then upon submission of a folded protein get a score feedback. The best scored structures are then further observed and studied by researchers that determine whether there is a "native structural configuration". In just a few years since its inception, hundreds of thousands of its users managed to solve a long withstanding scientific problem of the structure of enzyme found in an HIV-related monkey virus. This has also been the first occurrence of a large group of gamers finding a solution to a hard scientific problem. The game mechanics used in Foldit are points, leaderboards and leveling.

6.8 Other

Of course, there are many other examples of applications that have leveraged the behavior triggers that gamification offers. We can find them in

many other verticals, like entertainment, social influence, recommendation services, personal finance, services for social good etc.

Foursquare

Foursquare (`https://foursquare.com/`) is a social recommendation service that have heavily based its success on the game mechanics like points, achievements, leaderboards. We have mentioned it in some of the previous sections, so we will not go deep in details here.

Getglue

Getglue (`http://getglue.com/`) is an entertainment service that allows people to check-in to their favorite television shows. This alone can be pretty a tiring activity, but they have managed to popularize it with challenges, achievements, points and building a community around specific shows. On the other hand, content producers have, for the first time, almost real-time feedback on their newest content, which will in the end lead to hopefully more high quality TV content.

Klout

Klout (`http://klout.com/`) is a pretty controversial service that aims to measure the social capital on the web with a single score metric. They have implemented an even more gamified version of the service in the recent months where people earn achievements and karma points for their influence in certain topics.

News aggregation sites

News aggregation sites have been revitalized in the recent years by heavy use of game mechanics to drive behaviour change. Thus, game mechanics like up and down voting, karma points and leveling have become vital parts of sites like Reddit (`http://reddit.com/`), HackerNews (`http://news.ycombinator.com/`) and Digg (`http://digg.com/`). These

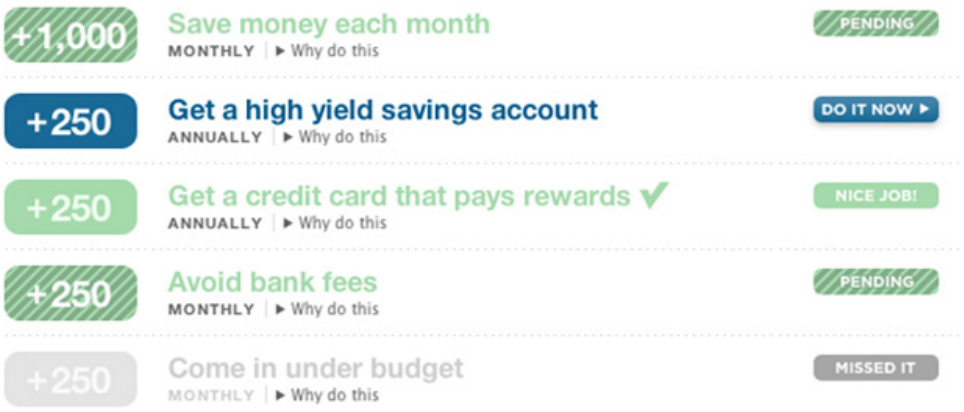

Figure 6.2: Simple example of challenges within finance tracking application Mint

sites are now vital parts of the so-called new media revolution driving incredible amounts of web traffic.

Mint

Personal finances are a vital part of each individual's life, and recently a few services have emerged, especially in the US, that are tackling the problem also with the help of gamification. The most successful one of those is Mint (`http://mint.com`), which provides users with a quick and easy overview of their finances and helps them stabilize those through a series of challenges that are either individual or social. The before mentioned challenges can be seen in Figure 6.2.

Kiva

Kiva (`http://kiva.org`) is a social good application that connects people that need small amounts of money for a concrete task (e.g. $600 to buy crops for example) with the lenders from all around the world who would like to help. They have been around since 2005, but did a fantastic shift to gamification in 2010 [?] and utilized some of the game mechanics

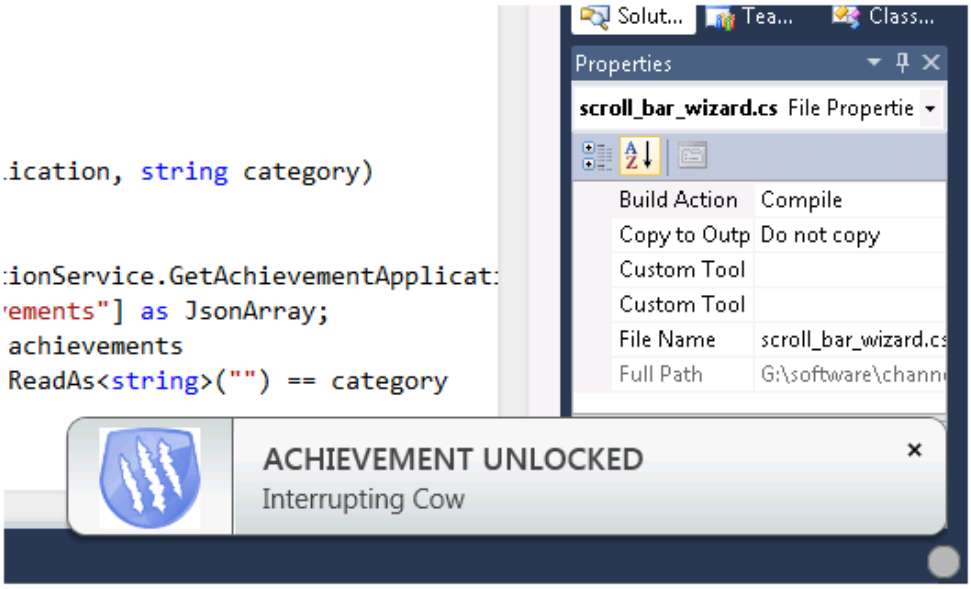

Figure 6.3: Example of an achievement within the Visual Studio

like leaderboards, points, group lending etc.

Microsoft Visual Studio Achievements

Even Microsoft could not avoid tapping into gamification. They have released a plugin for their programming development tool Visual Studio (`http://channel9.msdn.com/achievements/visualstudio`), which rewards programmers for performing various tasks within the tool with virtual badges. For example, users can get rewarded for opening several new projects, using several assemblies, having ten breakpoints when debugging etc. Further on, users are encouraged to share their achievements and compare themselves with other programmers on Microsoft's platform via various leaderboards. We can see an exemplary Visual Studio achievement notification in Figure 6.3.

Even though the idea of gamifying programmer's work feels ok, this project turns out to be one of the worst examples of a gamified software application. The whole achievement system feels defined in a haste and with-

out much thought put into it. It does not encourage collaboration between programmers and the badges are really badly designed. Some achievements even encourage strange or somewhat bad code design, like: “Write 20 single letter class level variables in one file. Kudos to you for being cryptic!”. This surely has not come from a programmer, and it just implies how important it is sometimes for system architects that deal with gamification to also be developers at the same time. All in all, it really comes out as a toy, and we do not think we will see many real programmers installing and using this plugin.

Chapter 7

Conclusion

During the work on our thesis, we have become highly familiar with the subject and its current state. Because of the rapid and quick expansion of this field, many naysayers have been critical to this whole field, as for example some of the analysts in the article by Janna Anderson and Lee Rainie [**?**]. As we have seen in our own research and work, gamification can be sometimes applied to an existing application without much thinking of augmenting the core of the product. As we have seen both from our own experience working on gamification of Psykopaint and also from the solutions that were presented in the last chapter, gamification can be an extremely useful, sometimes even critical, tool to increase engagement and usage.

In the example of applying gamification principles with the usage of MDA framework at Psykopaint, we have seen some immediate effects on participation and community. Right after releasing the version with some of the game mechanics people have started spending way more time in the application, participated and engaged more in the gallery part of the site and actually started to ask questions and seek advice on how to become better painters. They all wanted to progress through the application, gain followers and share their art with the rest of the community.

We have also looked deeply and even worked with some of the gamification platforms out there that promise companies fairly high engagement

results. They are an obvious result of the gamification trend, but should be applied and used with caution; simply taking a few of the most popular game mechanics and applying them on to the product does not make it better by itself. For example, basic achievement system within an ERP (*Enterprise resource planning*) system like SAP would probably not be the most suitable use of gamification. Instead, companies should approach gamification platforms already equipped with a fair bit of knowledge on the subject and only if they do not have the ability to build a custom solution in-house.

The best examples of gamified application were presented in Chapter 6. Most of those applications and system take gamification tool for granted, apply them with caution and see positive results from it. Foursquare, Wii Fit, WorkSimple, Fold.it and many others are applications that would simply not exist without game mechanics as they were, especially, in the early days of the applications, the main motivating force for users.

Consequently, we think it is important for system architects to know and apply game mechanics whenever it makes sense. For example onboarding mechanics and challenges should be applied to every software application. Onboarding helps users understand and more effectively use the application. Challenges in a broader sense are a way of letting people know what they should do next or what tasks can be accomplished in a reasonable amount of time. Software architects that are building consumer facing applications should be even more familiar with the whole subject and use it as a tool that serves and augments the core use of the product. There are more and more applications hitting users every single day (especially with the rise of mobile platforms) and users do not spend much time on trying to get used to them. Sometimes, even the first minute experience can separate applications that become widely successful and those that do not break through.

List of Figures