INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Gamification Design with a Domain-Driven Engineering Approach

PEDRO GUILHERME PINTO FERRAZ DE AGUIAR
novembro de 2019

POLITÉCNICO
DO PORTO

Gamification Design with a Domain-Driven Engineering Approach

Pedro Guilherme Pinto Ferraz de Aguiar

A dissertation submitted in partial fulfillment of the requirements for the degree of Master of Science, Specialisation Area of Software Engineering

Supervisor: Isabel de Fátima Silva Azevedo

Porto, October 27, 2019

Dedicatory

I dedicate this project to all of the thousands of hours spent learning in order to come this far.

Abstract

Companies use a variety of methods and processes to improve the quality of their services, which, in turn, would increase the satisfaction of their users and hence their popularity. One of the enhancements that have been used in recent years is known as gamification. With the benefit of being virtually independent of business types, one of the goals of gamification is to solve user engagement issues. Even so, companies fail to achieve their goals after instantiating gamification into their services, and one cause is related to poor gamification design. The main objective of the developed project was to acquire and develop a possible solution to this problem through the use of a specific set of methods, technologies and the Model-Driven Engineering (MDE) approach. In this sense, in-depth research was done into previous gamification applications and other previous attempts to solve the problem at hand. Several gamification concepts were analyzed, gathering as much data as possible about the subject before the conceptualization of the solution's domain through an MDE approach.

Keywords: Gamification; Design; Model-Driven Engineering; Domain-Specific Language.

Resumo

As empresas usam uma variedade de métodos e processos para melhorar a qualidade dos seus serviços, o que por sua vez aumentaria a satisfação de seus utilizadores e, consequentemente, sua popularidade. Um exemplo do mesmo que tem sido utilizado nos últimos anos é conhecido como gamification. Com o benefício de ser praticamente independente dos tipos de negócios, um dos objetivos de gamification é resolver problemas relacionados com a interação entre o serviço e o utilizador. Mesmo assim, as empresas não conseguem atingir os seus objetivos após a adição de gamification nos seus serviços, e uma das causas está relacionada a mau design de gamification. O principal objetivo do projeto é desenvolver uma possível solução para o problema através do uso de um conjunto específico de métodos, tecnologias e da abordagem Model-Driven Engineering (MDE). Nesse sentido, uma pesquisa rigorosa foi realizada sobre aplicações existentes de gamification, como também sobre outras tentativas de resolver o problema em questão. Vários conceitos de gamification foram analisados, de forma a reunir o máximo de informação possível sobre o assunto antes da conceituação do domínio para a solução através da abordagem MDE.

Acknowledgement

I'd like to thank everyone who has helped me come this far, my family and friends in particular. I'd also like to thank my supervisor for being incredibly helpful during the whole development of this project.

Contents

List of Figures **xiii**

List of Tables **xv**

List of Source Code **xvii**

List of Acronyms **xix**

1 Introduction **1**
1.1 Context and Problem . . . 2
1.2 Objectives . . . 2
1.3 Methodology . . . 2
1.4 Structure . . . 3

2 Value Analysis **5**
2.1 Innovation Process . . . 6
2.1.1 New Concept of Development . . . 6
2.2 Value . . . 8
2.2.1 CANVAS Model . . . 10

3 State-of-the-Art **13**
3.1 Gamification . . . 14
3.1.1 Value Generated . . . 16
3.1.2 Examples . . . 17
3.2 Frameworks . . . 17
3.2.1 The Octalysis . . . 18
3.2.2 Six steps to Gamification . . . 19
3.2.3 Mechanics, Dynamics, and Aesthetics . . . 21
3.2.4 Problems with current frameworks . . . 22
3.3 Model-Driven Engineering . . . 23
3.4 Project Technologies . . . 24
3.5 Similar Solutions . . . 26
3.5.1 GaML . . . 26
3.5.2 MEdit4CEP-Gam . . . 27

4 Design **31**
4.1 Metamodel Design . . . 32
4.1.1 Design Process . . . 32
4.1.2 Current Design . . . 33

5 Development **41**
5.1 Implementation process . . . 42

5.1.1 Project Setup 42
5.1.2 DSL Development 42
5.1.3 Code Generation 49

6 Evaluation and Experimentation **51**
6.1 Solution Testing 52
6.1.1 Problem 1 - DSL 52
6.1.2 Problem 2 - Gameful Experiences 55

7 Conclusion **57**
7.1 Current Objective Completion and Limitations 58
7.2 Future Work and Other Remarks 58
7.2.1 Notes on Gamification 59
7.2.2 Notes on Model-Driven Engineering 59

Bibliography **61**

A Illustration of the Analytic Hierarchy Process **63**
A.1 Step 1 – Hierarchical Decision Tree 63
A.2 Step 2 – Comparison between Hierarchal Elements 64
A.3 Step 3 – Element's Relative Priority 64
A.4 Step 4 – Evaluate the consistency of the relative priorities 65
A.5 Step 5 – Evaluate each alternative by each element 65
A.6 Step 6 – Obtain composed priority for the alternatives 66
A.7 Step 7 – Choose the best alternative 66

B Illustration of a Value chain **67**

C Evaluation Method: Questionnaire **69**
C.1 Questionnaire Details 69
C.2 Question 2: Results 72

D Evaluation Method: Running Simulator **77**

List of Figures

2.1 The Process of Innovation . . . 6
2.2 New Concept Development . . . 7
2.3 Benefits and Sacrifices . . . 9

3.1 Abstraction in Game Elements . . . 15
3.2 Octalysis Framework . . . 18
3.3 Activity Cycle . . . 21
3.4 Process of Consumption . . . 22
3.5 Mechanics, Dynamics, Aesthetics Sequence . . . 22
3.6 Usual and MDE approaches . . . 25
3.7 MEdit4CEP-Gam High-level Gamifying Process . . . 27

4.1 First Gamify Metamodel . . . 32
4.2 Second Gamify Metamodel . . . 33
4.3 Second Iteration in Gamification Concepts . . . 33
4.4 Gamify Metamodel . . . 38
4.5 Other Gamification concepts . . . 39

5.1 Early DSL Specification . . . 43
5.2 Early DSL Example . . . 43
5.3 Simple Cross-reference Example . . . 45
5.4 Portion of a Gamification Strategy . . . 45
5.5 Example of the DSL assistance . . . 48
5.6 Guidelines from the Item Entity . . . 48
5.7 Gamify's Compile Function . . . 50

6.1 Results Graph for Question 1 . . . 53
6.2 Results Graph for Question 3 . . . 54
6.3 Results Graph for Question 4 . . . 55

A.1 Hierarchy Tree . . . 63

B.1 Value Chain . . . 67

C.1 First Questionnaire Section . . . 70
C.2 Part 1 - Second Questionnaire Section . . . 70
C.3 Part 2 - Second Questionnaire Section . . . 71
C.4 Third Questionnaire Section . . . 72

D.1 User Interface of Running Simulator . . . 79
D.2 Notification for JourneymanRunner Achievement . . . 81

List of Tables

3.1 Game conditions (Huotari and Juho Hamari 2012) 15

4.1 Basic visual game mechanics 35
4.2 Aggregated visual game mechanics 36
4.3 Gamification reward strategies 37

7.1 Table of set objectives 58

A.1 Table of comparison between hierarchal elements 64
A.2 Table of comparison between hierarchal elements with sum per column . . 64
A.3 Table of element's relative priority 64
A.4 Table of comparison between alternatives about the *Metamodel* Design Correctness element 65
A.5 Table of comparison between alternatives about the DSL Completeness element 65
A.6 Table of comparison between alternatives about the DSL User-friendliness element 65
A.7 Table of comparison between alternatives about the Apply complex gamification strategies element 66

List of Source Code

5.1 Terminal rules 44
5.2 Developed rules 44
5.3 Excerpt from Function getFirstLine in DSLEObjectHoverProvider 46
5.4 Excerpt from Function getDocumentation in DSLEObjectDocumentation-Provider 46
5.5 Excerpt from DSLUiModule class 47
5.6 Excerpt from DSLGenerator class 49
5.7 Functions from the Condition class 50
C.1 List of answers for question 2 72
D.1 Gamification Strategy for RunningSimulator 77
D.2 Integration with Running Simulator 79

List of Acronyms

AHP	Analytic Hierarchy Process.
DSL	Domain-Specific Language.
EMF	Eclipse Modeling Framework.
FEI	Front End of Innovation.
MDE	Model-Driven Engineering.
VC	Value for the Customer.

Chapter 1

Introduction

1.1 Context and Problem 2
1.2 Objectives 2
1.3 Methodology 2
1.4 Structure 3

1.1 Context and Problem

Gamification is known as a process that enhances existing services by adding game-like elements to systems/applications in non-game contexts (Deterding et al. 2011; J. Hamari, J. Koivisto, and Sarsa 2014).

The implementation of gamification has been becoming more common in modern systems, independently of the context that these systems were built for. Despite this, according to Gartner (Burke 2014), a high percentage of gamified applications would fail to meet their business objectives due to poor design caused by the fact that gamification is based on games, which are already of a complex nature (Morschheuser et al. 2018; Swacha 2018).Furthermore, a game's primary purpose is to entertain, while gamification's primary purpose is to affect human behavior, this change of end-goal also proves to be an issue when designing gamification.

Gamification is a topic of relevance with room for improvement as many companies have failed to integrate gamification in their applications due to poor design (Robson et al. 2016). As such, the solution developed pretends to assist with the conceptualization and implementation process of gamification strategies, with the intention of improving the success rate of gamified applications.

1.2 Objectives

The following objectives were set for this project:

1. Examine reported failures in gamification adoption related to design issues, but also successful examples;
2. Analyze different ways to incorporate gamification in applications;
3. Delineate MDE approaches for the baseline design of gamification;
4. The solution should allow the implementation of high-level complex gamification strategies and components through an MDE approach to be then instantiated to specific systems, as well as providing guidelines through the conceptualization process of gamification strategies.

1.3 Methodology

The work methodology adopted consists of two main phases, related to the due dates set for the development of the project.

1. **Research, Analysis, and Initial Design:** At first, thorough research is conducted on various articles about gamification and gamification design, with the intention to understand where and why many companies failed to integrate it on their applications. Afterwards, using the information gathered, an initial design for the problem at hand is developed;
2. **Development, Finished Design, and Project Review:** After the conclusion of the first phase, the implementation of the solution begins, along with the necessary tweaks

to the design previously developed. Once the implementation and the design enter their final stages, a review of the work done takes place, listing what goals were achieved, how optimal is the solution that was developed, and what parts of the solution can be improved.

1.4 Structure

The document is built by several different chapters, the following list succinctly describing each:

- **Chapter 1 - Introduction:** Provides context about the theme of the project, as well as the problem to be solved. The objectives and the methodology adopted for the development of this project are also defined within this chapter;
- **Chapter 2 - Value Analysis:** Describes the methods and techniques used on defining the Front End of Innovation. Ascertains the project's value for its customers and a possible CANVAS Model;
- **Chapter 3 - State-of-the-Art:** Contains an analysis of different gamification concepts and related frameworks. Introduces the MDE approach, the concept of a DSL, the technologies used for the development of the project. Lastly, descriptions of similar solutions are provided and discussed;
- **Chapter 4 - Design:** Presents and describes the design for the solution developed, explaining the design decisions taken in the process;
- **Chapter 5 - Development:** Gives a rundown of the different implementation phases, detailing the important decisions made in each phase;
- **Chapter 6 - Evaluation and Experimentation:** Illustrates how the project is tested, providing the methodology used, as well as results obtained;
- **Chapter 7 - Conclusion:** Lists which objectives were completed and limitations found during the process of development, lastly, informs of possible work to be developed in the future to improve the current state of the project;
- **Appendix A - Illustration of the Analytic Hierarchy Process:** Presents a possible use of the Analytic Hierarchy Process for the project developed;
- **Appendix B - Illustration of a Value Chain:** Contains a possible Value Chain for the project in question;
- **Appendix C - Evaluation Method: Questionnaire:** Describes the questionnaire developed for the evaluation phase;
- **Appendix D - Evaluation Method: Running Simulator:** Provides details about the application developed for the evaluation phase.

Chapter 2

Value Analysis

2.1 Innovation Process 6
2.1.1 New Concept of Development 6
2.2 Value 8
2.2.1 CANVAS Model 10

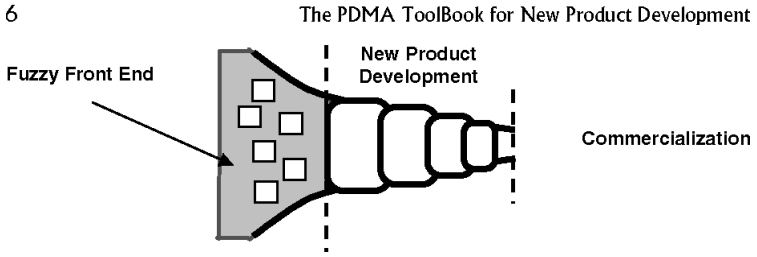


Figure 2.1: The Process of Innovation (Koen, Bertels, and Kleinschmidt 2014)

2.1 Innovation Process

On (Koen, Bertels, and Kleinschmidt 2014), the innovation process consists of three different parts: the Front End of Innovation (FEI), the New Product Development (NPD) process, and commercialization, as represented by the following figure.

Due to the nature of this project, the analysis will be performed on the portion named Front End of Innovation (FEI) of the process of innovation, which is known to improve the overall innovation process, generally characterized by having an experimental nature with unpredictable commercialization dates, undefined funding, depending on the project, uncertain revenue expectations, risk minimization and optimization of potential, and progress is measured by strengthening the project's concepts.

With these characteristics in mind, this approach was deemed more appropriate than the value analysis approach, due to the lack of a real organization context.

As such, the "Front End of Innovation" uses the "New Concept of Development" (NCD) model as its framework and the rest of this section will explain its contents in further detail, while also providing information about the process of defining the project's concept.

2.1.1 New Concept of Development

The model consists of the division of three front end areas:

- **Engine:** Positioned at the center of the model, "consists of two separate segments—organizational attributes and teams and collaboration" (Koen, Bertels, and Kleinschmidt 2014);
- **Wheel:** Positioned at "the inner part of the model, comprises the five activity elements of the front end: opportunity identification, opportunity analysis, idea generation, idea selection, and concept definition" (Koen, Bertels, and Kleinschmidt 2014);
- **Rim:** Positioned at the outer part of the model, it contains influential environmental factors, such as "company's organizational capabilities, competitor threats, customer and worldwide trends, regulatory changes, and the depth and strength of enabling sciences and technology" (Koen, Bertels, and Kleinschmidt 2014).

On (Koen, Bertels, and Kleinschmidt 2014), it is noted that projects should begin by either opportunity identification or idea generation and enrichment, as represented by the arrows

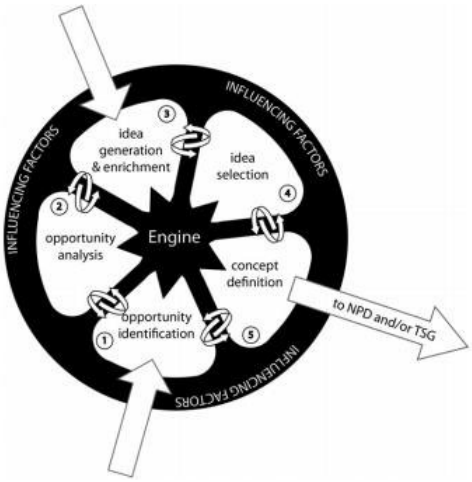


Figure 2.2: New Concept Development model (Koen, Bertels, and Kleinschmidt 2014)

pointing inward in the picture above. For this project, the following order of the front end elements will be assessed:

1. Opportunity Identification;
2. Opportunity Analysis;
3. Idea Generation and Enrichment;
4. Idea Selection;
5. Concept definition.

Opportunity Identification

Usually, the first element to be assessed by companies, it's through this element that an organization "identifies the opportunities that the company might want to pursue" (Koen, Bertels, and Kleinschmidt 2014). Since this project is not connected, nor does it have the support of any company, starting the front end process with this element is not as opportune as it would otherwise be if this project was developed within the context of a company, due to the fact that this element "is typically driven by the goals of the business" (Koen, Bertels, and Kleinschmidt 2014), which do not apply in this context.

For this first element of the process, a meeting between this document's author and his Supervisor took place, to discuss possible themes for the project. Upon further discussion about what type of project the author was looking for, the theme of gamification was ultimately chosen.

Opportunity Analysis

In this element, the previously identified opportunity becomes an object for analysis. This analysis consists in the identification of focus groups, performing "market studies and/or scientific experiments" (Koen, Bertels, and Kleinschmidt 2014). When performing this analysis, the following are the factors to be considered: "attractiveness of the opportunity, the size of the future development effort, the fit with the business strategy and culture, and the risk tolerance of the decision makers".

Having chosen the theme of gamification, research on articles was performed about its current state, and its success ratio, while also researching for examples of its deployment in different areas of the market. On (Burke 2014) it is mentioned how 80% of gamified applications will fail to meet their business' objectives due to poor design, meaning there is a need in the market that is not being properly met that could be explored further.

Idea Generation and Enrichment

Also known as "Idea Genesis", this element is defined as "the birth, development and maturation of the opportunity into a concrete idea" (Koen, Bertels, and Kleinschmidt 2014). This definition implies that in this element, many ideas will be generated, some of them will be worked upon, and others will be disregarded in a later stage.

For this part of the process, following the previously mentioned analysis, there was a meeting between the author of this document and his Supervisor of the project with the purpose of discussing a concrete theme for the project to be developed.

The following are two of the main ideas discussed:

- The implementation of gamification in the Moodle platform;
- The usage of a Model-Driven Engineering approach to assist gamification design.

Idea Selection

This element consists of the selection of an idea to pursue from the list of ideas previously discussed upon. According to (Koen, Bertels, and Kleinschmidt 2014), "Selection may be as simple as an individual's choice among many self-generated options or as formalized as a prescribed portfolio method".

On this phase, the selection process consisted of further discussion between this document's author and the author's Supervisor. It was then decided to select the idea "The usage of a Model-Driven Engineering approach to assist gamification design", due to it being both more technically challenging and, on (Burke 2014), it does not have a clear solution to its primary problem: Poor gamification design.

Concept Definition

This final element consists of "the development of a business case based on estimates of market potential, customer needs, investment requirements, competitor assessments, technology unknowns, and overall project risk" (Koen, Bertels, and Kleinschmidt 2014).

At this stage, the project's concept had been defined as "Gamification design with a domain-driven engineering approach", since this approach hasn't been properly tackled within the context of gameful design. At a later time, the concept went through a formalization process, which was presented the context, problem to be solved, and the objectives of the project.

2.2 Value

On (Walters and Lancaster 2000), creating value is a "concept that is difficult to achieve, understand, model and/or conceptualize".

BENEFITS		SACRIFICES
Attributes	Outcomes	
Perceived quality	Functional benefits	Price
Product quality	Utility	Market price
Quality	Use function	Monetary costs
Service quality	Aesthetic function	Financial
Technical quality	Operational benefits	Costs
Functional quality	Economy	Costs of use
Performance quality	Logistical benefits	Perceived costs
Service performance	Product benefits	Search costs
Service	Strategic benefits	Acquisition costs
Service support	Financial benefits	Opportunity costs
Special service aspects	Results for the customer	Delivery and installation costs
Additional services	Social benefits	Costs of repair
Core solution	Security	Training and maintenance costs
Customisation	Convenience	Non-monetary costs
Reliability	Enjoyment	Non-financial costs
Product characteristics	Appreciation from users	Relationship costs
Product attributes	Knowledge, humour	Psychological costs
Features	Self-expression	Time
Performance	Personal benefits	Human energy
	Association with social groups	Effort
	Affective arousal	

Figure 2.3: Benefits and Sacrifices (Woodall 2003)

The following are some ways to measure value, and having those concepts in mind, the value of the project, for each stakeholder, will be roughly measured.

Value for the Customer

Value for the Customer (VC) consists of a customer's personal perception of advantage, or disadvantage, when they associate themselves with an organization's product/service. This sense of whether or not a product/service is advantageous for a particular customer is the result "of any weighed combination of sacrifice and benefit" (Woodall 2003).

Perceived Value

The term "customer value", within the context of marketing literature, is used to "portray both what is derived by the customer from the supplier, and also what is derived by the supplier from the customer" (Woodall 2003).

Benefits and Sacrifices

Value for the Customer has associated elements, each of these elements having two main categorizations as either a benefit or a sacrifice. Within the category of benefits, there are two sub-categories, the attributes, and the outcomes. Therefore, should one add a certain attribute to your product, it is expected to obtain a specific outcome, while sacrificing another element.

Project's Value

Taking the previous concepts (Value for the Customer, Perceived Value, Benefits and Sacrifices) into account, the overall value of the project, for each stakeholder involved with the service to be developed was roughly measured.

To the author of the document:

- Strengthens the author's knowledge about gamification and usage of model-driven engineering software tools, which may be useful in the future, while sacrificing time, human energy and effort.

To the service's customers:

- The customers are provided with the necessary tools to better design gamification for their services while being able to deploy the previously designed gamification strategy directly into their services. In turn, the gamification elements should increase users' interactivity with the customer's service;
- Customers can sacrifice the price of the service, should they desire additional options.

To the customer's services' users:

- They are provided a gameful experience on their used services, which will improve their personal experience with said services;
- Possible gamification-related costs.

2.2.1 CANVAS Model

A business model defines how a company will generate and capture value (Osterwalder and Pigneur 2010). Both systematic and practical way to create a business model is to follow the CANVAS model. The model consists of various elements, each of them being capable of answering questions related to the business model.

Considering this project is not being developed with the support of a company, most of the answers to the following elements will be attempts to replicate the point-of-view of an organization.

Value Proposition

Value Proposition describes the created value of a product or service for a specific Customer Segment.

The Value Proposition for this project is the following: A service to assist with the gamification process, through providing guidelines over the course of the development of a gamification strategy for a company's own services, to later be deployed in said services.

Customer Segments

In this element, it is defined who are the most important customers, or the customers who would find the most value in a certain service or product, for the company in question.

The customers of our product would be any company with the need to implement gamification on their services.

Customer Relationships

Now that the customer segments have been defined, the relationship between the company and those customers needs to be defined as well. Therefore, the company needs to assess what type of relationship their customers expect to have with the company.

The company would provide personal assistance online where users could communicate through an e-mail, providing information about what issues the user may be having with the service in question.

Channels

With the customer segments and relationships defined, it is necessary to assess how the company will make their product/service reach its customers of choice. As such, some questions need to be addressed:

- How will the company raise awareness about their product/service?
- How does the company help the customers evaluate their value proposition?
- How will the customers buy the company's product/service?
- How does the company provide post-purchase support?

The company will attempt to raise awareness about the service to be developed by contacting companies which specialize in software reviews, in order to use positive reviews as a catalyst to raise awareness. As for the rest of the questions to be addressed, the company would provide a website with information about its value proposition, how to purchase the service and customer support.

Revenue Streams

At this point in the model, the company has assessed who are their customers and how they'll reach them, but the following questions still need to be answered:

- How much do they currently pay for a similar product/service?
- How much would they be willing to pay for for the product/service the company's working on?
- How are they currently paying and is there any preference on the way the company should do it?

The service's customers would pay a subscription fee, depending on which game element choices they'd desire to have access when conceptualizing their own gamification solutions.

Key Activities

With the Value Proposition defined, the company should assess what are the key activities required in order to fulfill the needs of the previously defined proposition.

In order to fulfill all of the requirements defined in the value proposition, the following activities have been proposed:

- **Production:** To work on new features of the service;
- **Service maintenance:** To maintain the quality of the features already present in the service;
- **Website maintenance:** To maintain the quality of the website;

- **Customer Service:** To assist customers of the service.

Key Resources

The Key Activities are required to fulfill some if not all, the needs of the defined Value Proposition, but the Key Activities, the Distribution Channels, the Customer relationships, and even the Value Proposition require resources.

Resources such as a database of our customers, a team to work on each of the previously mentioned Key Activities, servers, and an online platform may be imperative for the success of the business.

Key Partners

Having the Key Activities and the Key Resources defined, the following questions emerge:

- What are the company's Key Partners?
- How does the company obtain its Key Resources?
- How are the company's Key Activities performed?

Key Partners may be companies who can raise awareness of the service to be developed, and those who may assist with the process of setting up an online platform.

Cost Structure

Lastly, the company will measure the costs of building the defined business model. As such, important costs for the business should be defined, as well as which resources or activities are the most expensive.

Most of the costs would be directed to maintaining the value of the service, as such, employee's salary, rent, and utilities (software and hardware) would be the important costs for the company.

Chapter 3

State-of-the-Art

3.1 Gamification 14
3.1.1 Value Generated 16
3.1.2 Examples 17
3.2 Frameworks 17
3.2.1 The Octalysis 18
3.2.2 Six steps to Gamification 19
3.2.3 Mechanics, Dynamics, and Aesthetics 21
3.2.4 Problems with current frameworks 22
3.3 Model-Driven Engineering 23
3.4 Project Technologies 24
3.5 Similar Solutions 26
3.5.1 GaML 26
3.5.2 MEdit4CEP-Gam 27

3.1 Gamification

Gamification has many different definitions, one of them being "the use of game design elements in non-game contexts" proposed by (Deterding et al. 2011).

To justify the word choices for the definition of gamification, (Deterding et al. 2011) begins by comparing the core concepts of "play" and "game" which are essential to distinguish the concept of "playfulness" from "gamefulness", the latter being a term adopted by McGonigal (McGonigal 2011) which is considered as a systematic complement to playfulness, where "'playfulness' broadly denotes the experiential and behavioral qualities of playing (paidia), 'gamefulness' denotes the qualities of gaming (ludus)" (Deterding et al. 2011). In order to proceed, new terminology is provided so that the term gamification could be distinguished from playfulness and playful design. The following represent the previously mentioned terminology:

- **Gamefulness:** "The experiential and behavioral quality" (Deterding et al. 2011);
- **Gameful Interaction:** "Artifacts affording that quality" (Deterding et al. 2011);
- **Gameful Design:** "Designing for gamefulness, typically by using game design elements" (Deterding et al. 2011).

For the definition of "game elements" in this context, the solution is to consider them as a "set of building blocks" or common game features. As for how this definition should be interpreted, the article presents two approaches, one being strict, in which "game elements" are only the specific/unique elements of said game, and the other being very liberal, in which "game elements" would be any element implemented within the game in question. Neither of these options is deemed "correct" since the strict interpretation would barely give any result, and the liberal interpretation would give too many results. Therefore, (Deterding et al. 2011) suggests that the "game elements" should be restricted to only those which are "characteristic" to a game, these elements being common but significant in a said game type.

On the topic of design within "Gamification", it is clarified where the use of game-related tools in other contexts stand within the topic at hand with the following sentence: "For the purposes of terminological and conceptual clarity, it is more helpful to reserve the term 'gamification' for the use of game design, not game-based technologies or practices of the wider game ecology". Also within the same topic, it was found that various levels of abstraction for game elements were identified by other sources and proposes that these levels should be included in the definition of game design elements. The following table represents the levels previously mentioned.

Lastly, "non-game contexts" can be summed to the following sentence "the only thing that 'nongaming contexts' explicitly intend to exclude is the use of game design elements as part of designing a game, since that would simply be game design, not 'gamification'" (Deterding et al. 2011). However, it's up to discussion whether or not a "meta-game" can be considered a form of gamification within a game context.

This definition of gamification has been used as a base for some projects and studies (Calderón, Boubeta-Puig, and Ruiz 2018; Philipp Herzig et al. 2013; Huotari and Juho Hamari 2012) about gamification.

Level	Description	Example
Game interface design patterns	Common, successful interaction design components and design solutions for a known problem in a context, including prototypical implementations	Badge, leaderboard, level
Game design patterns and mechanics	Commonly reoccurring parts of the design of a game that concern gameplay	Time constraint, limited resources, turns
Game design principles and heuristics	Evaluative guidelines to approach a design problem or analyze a given design solution	Enduring play, clear goals, variety of game styles
Game models	Conceptual models of the components of games or game experience	MDA; challenge, fantasy, curiosity; game design atoms; CEGE
Game design methods	Game design-specific practices and processes	Playtesting, playcentric design, value conscious game design

Figure 3.1: Levels of abstraction for game elements (Deterding et al. 2011)

The next studied paper consists in a new proposal for the definition of gamification, by focusing on the "experiential nature of games and gamification" instead of the previous approach (Huotari and Juho Hamari 2012). In this context, the concept in question is tied to service marketing, because the origin of gamification is related to the necessity of fulfilling marketing goals.

After evaluating the contents of some studies about games and gamification, it is concluded that games are built with both a systemic component, which defines how the game is constructed and an experiential component, describing the human involvement within said game. Using this information, a table of game conditions is formed, with three different levels of abstraction.

Table 3.1: Game conditions (Huotari and Juho Hamari 2012)

Level of abstraction	**Systemic conditions**	**Experiential conditions**
1st level (common to all games)	Games are systems	Games require voluntary involvement of player-s/users
2nd level (characteristic to games, although not necessarily to all games)	Rules; Conflicting goals; Variable and uncertain outcomes	Generates hedonic pleasure; Generates suspense; Generates gamefulness
3rd level (unique to games)	?	?

To propose the new definition for gamification, games are considered as service systems, where a game is developed with a partnership between a developer and a player, where the developer sets up the story and rules of the game, and the player(s) part of the production and value-creation is to evaluate the story and rules created, by interacting with them or by playing the game. The authors also take note that the notion of what a game is subjective

to each individual customer, therefore, the value of a service is measured by a customer's subjective experience.

Considering the information provided, the second analyzed definition of gamification is as follows: "A process of enhancing a service with affordances for gameful experiences in order to support user's overall value creation" (Huotari and Juho Hamari 2012).

This definition highlights the goal of gamification rather than the methods behind its implementation, unlike the previous paper's definition which is based on the use of game elements.

Another possible definition for gamification was proposed by Yu-kai Chou, the author of a gamification framework known as Octalysis (Chou 2015), has studied the subject of gamification for 10 years, and he has defined gamification as a "(...) design that places the most emphasis on human motivation in the process. In essence, it is Human-Focused Design (as opposed to 'function-focused design')". For the definition of Human-Focused Design, it's described as "a design process that optimizes for human motivation in a system, as opposed to pure efficiency", which unlike Function-Focused Design (systems that were created to be efficient, disregarding any variable of human behavior), it takes human behavior into account, so it may better motivate any person who may interact with the system to take a certain action. More details about the Octalysis framework will be provided within the following section of the document.

These three different proposals for defining gamification seem to focus on distinguished components involved within the process of gamifying a service. On (Deterding et al. 2011), it was primarily focused on enhancing a service with game-like elements, by following designs which were already set in the game industry. However, according to (Huotari and Juho Hamari 2012), a component related to human involvement with a gamified system has to be considered when developing gamification strategies. The last proposed definition of gamification presented by the author of the Octalysis Framework (Chou 2015), seemed to focus on the component of generating motivation to perform tasks that would not be performed otherwise. Since each of the previously mentioned definitions is related to how gamification is developed, and how it can, and should, affect human behavior, they can support each other in order to achieve an in-depth definition of what gamification is, and what problems it aims to solve.

3.1.1 Value Generated

(J. Hamari, J. Koivisto, and Sarsa 2014) describe a study about gamification effects on systems, assessing the overall value a gamified system can offer, depending on the given context. To conduct this research, the authors chose the concept of gamification proposed in (Huotari and Juho Hamari 2012), consisting in a three-step process starting with the implemented motivational affordances causing psychological outcomes, which will, in turn, produce the desired behavioral outcomes. The results about gamification usefulness varied depending on the motivational affordances used and the expected psychological outcomes which would develop behavior changes on the target audience.

The study revealed that points, leaderboards, and badges were the most prominent types of motivational affordances used in gamified systems, as well as the prominence of behavioral analysis over psychological analysis about the effects of gamified applications on its respective user base. The overall result of the research conducted is that gamification does provide benefits, should it be implemented under the correct circumstances, two of the main factors

to consider before adding a gamified application would be the role of the context to be gamified and the types of users who would engage with the application.

3.1.2 Examples

The following are two examples of companies who have successfully deployed gamified services:

- **Fitocracy:** Aims to motivate users to uphold a healthier lifestyle, using game mechanics present in roleplaying games (RPG), rewarding users with in-game experience and levels should they work out or eat healthy (Juho Hamari and Jonna Koivisto 2015). Fitocracy is based on a traditional form of gamification providing rewards, as well as social features, granting the user means to find online fitness groups who will encourage each other to stay healthy;
- **Duolingo:** Helps users learn new languages through gamified lessons. More than 30 are available with free lessons in this language-learning platform. It allows their users to assess their progress, customize their profile, socialize through language, play through learning lessons using a retainable but limited life pool, which restrains the user from taking a lesson should they run out of lives. Due to badges having a decreasing effect over time, winning streaks are used to keep users motivated (Huynh, Zuo, and Iida 2018).

Not well-defined objectives are among the usual explanations for failure in gamification design (Mora et al. 2017). An inadequate understanding of what motivates target users can also cause some problems. Some companies failed to meet their business objectives, for instance:

- **My Marriott Hotels:** To attract new employees, this gamified application would have its users take on specific positions as an employee of a hotel, fulfilling its various hotel-related activities, to provide a realistic experience as an employee (Robson et al. 2016). Depending on how successful the users were in keeping their customers happy, they would either gain or lose points. After 1 year, this application was removed from its host due to the failure to attract employees. The main culprit of this failure was the application's overall design, as the points players would collect had no real purpose, as well as the lack of social elements to keep people motivated;
- **Google News:** The goal was to encourage users to read news through the Google platform by rewarding users with badges related to the news topic that would appear on the user's profile page. Users were not attracted to the concept of earning unusable badges, neither to sharing information about what news they have read through the badges they earned. The gamification strategy in question did not consider the target users and their interests, thus the platform failed to motivate users as was intended.

3.2 Frameworks

In the context of this document, frameworks should be seen as tools to assist designers, researchers, and scholars with the overall process of gamifying a service or developing a game. As such, a framework will serve as a guide which, either by providing steps to follow or by breaking down complex concepts into simple components, will successfully support the

development of an abstract design. A recent study examined 40 frameworks (Mora et al. 2017). It is beyond the objectives stated for this project to perform such extensive analysis. This section introduces some frameworks widely used in gamification design. They are all broadly recognized, but they also have detailed descriptions and available support. Moreover, they do not only provide guidelines but align them to users and their possible motivations

The Mechanics, Dynamics, and Aesthetics framework is not directly related to gamification, but the information it provides regarding game elements and strategies should not be neglected.

3.2.1 The Octalysis

According to Yu-Kai Chou, who proposed Octalysis (Chou 2015), a game purpose is only to please the individual playing it by appealing to several specific "Core Drives", which, in turn, motivate them to continue playing (see Table 1). Yu-kai Chou defines "White Hat Gamification" and "Black Hat Gamification", the former encompasses positive motivators, while the latter includes the negative motivators. It is reassured that "Black Hat Gamification" is not necessarily bad since it can motivate people to take either beneficial or harmful actions.

The Octalysis framework is visually divided vertically, having the drives on the left associated with logic, calculations, and ownership, and on the right associated with creativity, self-expression and social aspects.

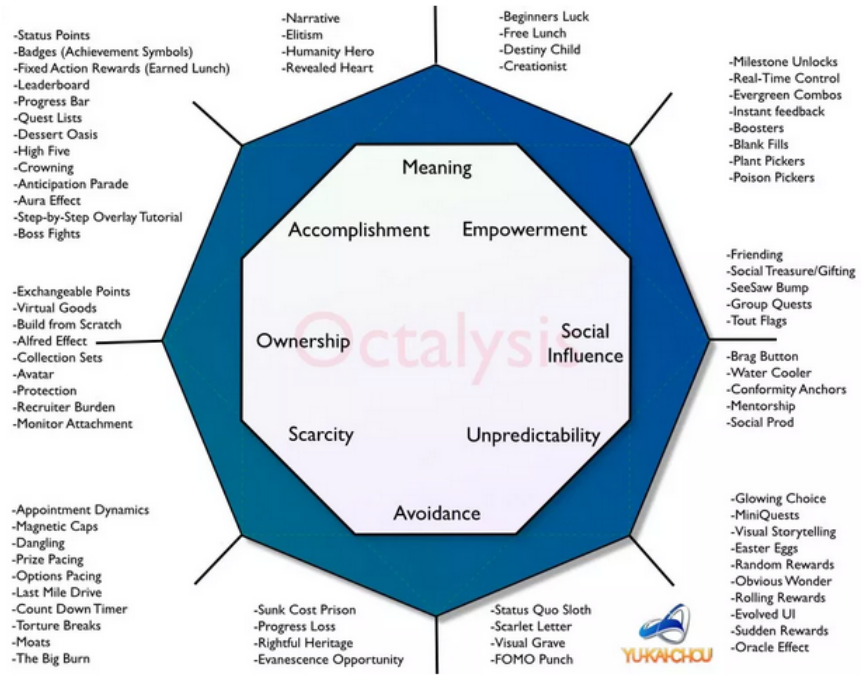


Figure 3.2: Octalysis Framework (Chou 2015)

Figure 3.2 represents the 8 Core Drives in which Octalysis is based on, the following list provides detailed information about each of the represented Core Drives:

- **Epic Meaning and Calling:** This first Core Drive affects the individuals who believe that their actions are contributing to something greater than themselves. These players tend to work on projects which benefit the entire community in question. This Core Drive also includes players who have been lucky in the earlier stages of the game, whether they have completed a difficult task earlier than expected, or they were gifted with a particularly rare item;
- **Development and Accomplishment:** This Core Drive is focused on personal progress, challenges, and developing skills. The use of badges or leaderboards is important to represent which challenges the user overcame. The more challenging it is to attain a particular achievement, the more meaningful it feels;
- **Empowerment of Creativity and Feedback:** Whenever users find themselves repeatedly trying different combination, in an attempt of discovering the best possible combination, in a system that provides instant feedback to the decisions made and, ultimately, rewards users for being successful in their endeavor, the "Empowerment of Creativity and Feedback" is the Core Drive that appeals these types of users. These kinds of game dynamics also allow the game developers to have longer development periods before adding new content due to the large amounts of existing combinations;
- **Ownership and Possession:** This is the drive containing the type of users who have their possessions as one of their main types of motivation. As such, these users search for ways to improve the items already in their possession or ways to own more valuable items;
- **Social Influence and Relatedness:** The Core Drive "Social Influence and Relatedness" contains the users that are motivated by social elements such as companionship, mentorship, and competition. These users tend to become encouraged whenever one of their friends reaches a higher level, driving them to reach the same level as they have;
- **Scarcity and Impatience:** This drive takes advantage of people desiring the unattainable, having people thinking about obtaining a significant reward during an entire day because they can't have it at that specific moment;
- **Unpredictability and Curiosity:** This Core Drive enlists the users that find the suspense, or randomness, to be its own reward. These users wonder about all the possible positive outcomes whenever they perform a specific action, which maintains their motivation should they not obtain the outcome they desire;
- **Loss and Avoidance:** This final drive contains the users who avoid negative repercussions, which motivates them to act accordingly so that they do not lose anything significant. Temporary opportunities to obtain unique rewards fit into this drive since it compels these types of users to act quickly before they lose their opportunity to be rewarded.

3.2.2 Six steps to Gamification

The "Six steps to gamification" (Werbach and Hunter 2012), or 6D Framework, is based on six different steps:

- **Define Business Objectives:** The first step consists in the development of a list with concrete objectives related with the performance objectives of the system to be gamified (e.g. Increasing customer retention, improving employee productivity), and not the organization's mission objectives. After creating the list of objectives, it is important to remove every objective that isn't an important achievement, meaning that the list should not have any objective that is simply "means to an end". Once the list is finalized, each objective should have a description as to how it will help the company grow;
- **Delineate target behaviors:** In this second step, the tasks to be performed by the users should be specified. Performing these tasks should reward users, but it's important to avoid "all-or-nothing" situations since these do not encourage progression. Then success metrics should also be defined for each key performance indicator of the gamified system in question. The ratio of monthly active users, or the number of rewards that users have collected, are two examples of possible key performance indicators which should be considered when analyzing the system's overall performance;
- **Describe your players:** This step requires an in-depth understanding of the targeted player. At this point, it should be specified whether the player-base will consist of employees or customers, since employees and customers may not have the same kind of motivators;
- **Devise your activity cycles:** Unlike various games, gamified systems cannot rely on a linear progress system where there is a beginning and there is an end since it is required of the users to keep performing their activities for extended amounts of time. As such, the usage of cycles is crucial to keep users motivated. There are two types of cycles:
  - Engagement loops contain three different components, one that describes what players can do, another describes why they do it, and the third component describes the system's feedback. Figure 3.3 shows how these components are connected;
  - Progression stairs are used to change the users' experience as they progress, providing bigger challenges, and increasingly more difficult scenarios as they become more experienced while providing rewards which are fit for the current stage of progression.
- **Don't forget the fun:** After all the design choices made until this step, it is important to assure that the system in development will grant its users a fun experience. If the users are performing actions that are, subjectively, fun, the more likely it is that they will keep performing said actions;
- **Deploy the appropriate tools:** Lastly, utilizing all the work developed in each previous step, the implementation step begins. Through the usage of the most appropriate tools, all the gamified system's mechanics and dynamics should be implemented, providing better user experience.

Related to the step "Describe your players", it is important to know who the system's users are, since what may motivate one user, may not motivate another, and if the developed motivators are not fit for the current player-base, the gamified system will fail. As such, creating several different groups of people, and developing different kinds of motivators, may be effective in dealing with the issue. To assist with what may motivate a specific

Figure 3.3: Activity Cycle (Werbach and Hunter 2012)

player-base, the authors of (Werbach and Hunter 2012) refers to (Bartle 1996), concluding that there are 4 types of players: achievers, explorers, socializers, and killers.

Achievers are interested in rewards such as badges, explorers look for new content to enjoy, socializers tend to engage with friends, and lastly, killers "want to impose their will on others, typically by vanquishing them". Each specific individual has elements of the previously mentioned archetypes hence it is important not only to identify these archetypes within the player-base but also to have a system prepared for changes since the players may have a shift in their motivations over time.

3.2.3 Mechanics, Dynamics, and Aesthetics

The Mechanics, Dynamics, and Aesthetics (MDA) framework is an approach to related to understanding games, instead of gamification, which attempts to connect game design with development, game criticism, and technical game research (Hunicke, Leblanc, and Zubek 2004).

This framework is not directly related with gamification, but it is important in the current context due to the information it provides regarding game design strategies, which in turn enhances the overall understanding of gamification strategies by breaking down the consumption of games and game design into concrete components.

A game is consumed like any other entertainment product but its consumption is comparatively unpredictable (Hunicke, Leblanc, and Zubek 2004). To better assist designers with design decisions regarding a specific game, it is important to note that this framework considers games as artifacts, meaning "that the content of a game is its behavior - not the media that streams out of it towards the player".

To clarify the consumption process of games, the MDA framework formalizes it through a sequence of distinct components as represented in Figure 3.4, as well as their respective design counterparts in Figure 3.5.

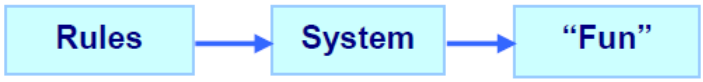


Figure 3.4: Process of Consumption (Hunicke, Leblanc, and Zubek 2004)

Figure 3.5: Mechanics, Dynamics, Aesthetics Sequence (Hunicke, Leblanc, and Zubek 2004)

The design components can be described by the following:

- **Mechanics:** Contains information about game components, such as data representation and algorithms;
- **Dynamics:** The behavior of the mechanics behind a player's input and its output is described within this component;
- **Aesthetics:** This last component represents the "desirable emotional responses" whenever a player interacts with the system to develop.

3.2.4 Problems with current frameworks

Deterding (2015) reviewed the current gamification frameworks discussed on articles (Burke 2014; Kapp 2012; Kumar 2013; Paharia 2013; Werbach and Hunter 2012; Zichermann and Cunningham 2011), analyzing their specific characteristics. The following list consists of the common issues found in the study:

- **Little formative research:** General lack of specification on formative research, commonly disregarding data collection methods (with some exceptions);
- **Reliance on player typologies:** Overuse and misuse of the typology presented on (Bartle 1996);
- **Appeals to motivational psychology:** Use of untested motivation models based on the Self-Determination Theory (SDT) (Deci and Ryan 2012);
- **Inherent-additive, pattern-based approach:** Misunderstanding of both how game elements should be used, and of MDA taxonomy. Recommend a pattern-based approach when developing gamification concepts;
- **Lacking guidance in game design pattern choice:** Small amount of guidance regarding to which design pattern to use, and how to customize it, within a specific context. It is suggested to apply mechanics that are appropriate to a type of user, but there is no indication of what mechanics are suitable for each of the player types;
- **No iterative prototyping:** Lack of methods to evaluate alternative design decisions;

- **Data-driven design:** It is recommended to monitor and track user engagement after deploying the gamification instance, neglecting data of user behaviors on prototype stages of development.

Mora et al. (2017) also performed an analysis on a wide array of gamification frameworks, categorizing some frameworks (and their respective issues) regarding their main areas of application: business, generic, health, and learning. The study revealed the most common context for gamification frameworks was the business environment, as well as the predominance of user-centered designs, along with the overall disregard of business-related issues, such as risk, feasibility, and investment. As is mentioned on (Deterding 2015), psychological factor is heavily considered in most frameworks, but the respective preferences of each user type are not broadly taken into account. Further details regarding the frameworks reviewed in both studies should be consulted on (Deterding 2015; Mora et al. 2017).

3.3 Model-Driven Engineering

Model-Driven Engineering (MDE) is an approach which uses models as the main artefacts for the software development process (Brambilla, Cabot, and Wimmer 2017). It avoids the implicit complexity of application development (Schmidt 2006). A Domain-Specific Language (DSL) facilitates the use of the considered concepts.

In this context, models implement, at least, two roles through abstraction:

- **Mapping feature:** Models are based on the original system;
- **Reduction feature:** Models only contain a relevant selection of the original system's properties.

Models also attend to different purposes when developing software following an MDE approach, as they can be used for both descriptive purposes, such as describing a system or a context, prescriptive purposes, permitting the development of a method to study a problem, and lastly, to define how the system should be implemented.

To follow the MDE approach, appropriate tools are necessary to define both models and transformations during the implementation phase, as well as suitable compilers or interpreters to execute and produce the software artifacts desired. Since MDE is based around models, the definition of the modeling language is also realized through a model, this procedure is called metamodeling. This procedure can be recursive into increasing levels of abstraction, thus, the result of modeling a metamodel is a meta-metamodel. A Domain-Specific Language (DSL) is a possible approach when it is required for a language to easily define a specific set of tasks (Gronback 2009).

A DSL defines the base structure, behaviour, and requirements related to a specific domain. Metamodels can be used to set relationships between concepts in a domain, but also to specify the key semantics/constraints related to each of these concepts. Domain-Specific Languages are typically used to simplify development processes but also to validate what is been specified within the domain context. Once the design of the DSL is complete, a generator can be used to produce source code, or other different types of artefacts, such as model representations.

Figure 3.6 contains a representation of important steps when developing a gamification instance for a system, comparing the usual and the MDE approach. It is important to

understand that the step “Gamification instance” (see Figure 3.6) is not final, and that it is always necessary the assistance of an IT expert to link the generated gamification application with the system to be gamified, as well as providing necessary increments. For instance, authentication services may be considered. However, by having the gamification expert formalizing the gamification design through an MDE approach, the application's code is directly connected to the model, thus there is no loss of information between the gamification expert and the IT professional in the implementation step.

MDE approaches have emerged to ease the inclusion of game elements in non-game applications. Two solutions are described in detail within section 3.5: GaML, MEdit4CEP. The solution Gamify is also highlighted in this document, having chapter 4 and chapter 5 dedicated to its design and implementation. Though these solutions are meant to assist with developing successful gamification strategies, they are not methods to bypass the need of defining concrete business objectives, nor means to disregard valuable information about the user base of the system to be gamified. Each of the solutions to be presented follow a similar process to the MDE approach shown in Figure 3.6.

3.4 Project Technologies

Over the development course of the project, a specific set of tools were necessary to achieve the desired result, as stated in section 1.2. As the purpose of this project is not to compare software, the decisions made regarding the tools used are due to existing familiarity attained through past experiences. Possible alternatives to the software used are presented in this section, though they are not compared with any of the listed tools.

The following list contains the technologies used, as well as a description of their purposes:

- **Eclipse Modeling Framework (EMF):** Modeling framework based on Eclipse, which is also a framework, as well as a code generation facility for building applications through a structured data model (Gronback 2009). With EMF, by modeling the desired application, describing what it is supposed to perform, the code that would otherwise be manually developed can instead be generated. As such, it supports the MDE approach as intended;
- **Xtext:** Supports the development of grammar for DSLs, which can later be used to generate Ecore-based metamodels (Gronback 2009). Although Xtext allows the specification of grammar through importing an existing metamodel, it is expected to manually specify the grammar desired in order to generate an Ecore model, which is not ideal when following an MDE approach. To assist developers with defining grammar, Xtext provides syntax highlighting, code assist and outline view.

Software that allows the development of DSLs, code generation, or the development of complex models, are eligible alternatives for the development of a similar project. The following technologies are other possible software alternatives for the development of this project:

- **EMFText:** Similar to Xtext, can be used to develop text syntax for "languages described by an Ecore metamodel" (*EMFText* 2018). EMFText provides a simple way to define textual Domain-Specific Languages, by removing the need to learn new technologies and concepts;

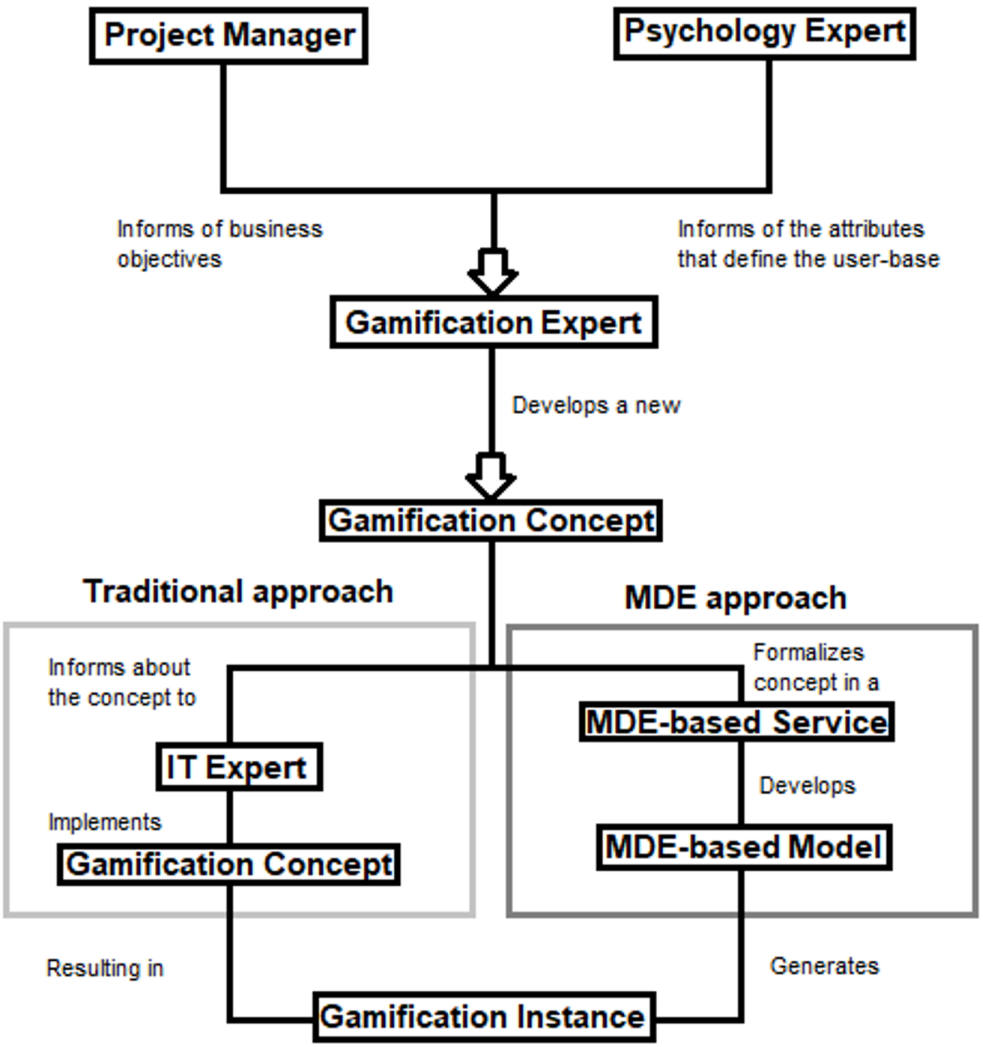


Figure 3.6: Usual and MDE approaches

- **JetBrains MPS:** Is yet another alternative to develop a DSL and to generate code, which also grants the possibility of developing non-textual notations, such as math notations, diagrams, and forms (*MPS* 2019).

3.5 Similar Solutions

This section presents solutions of similar caliber to the solution proposed regarding the problem stated. As such, both of the solutions follow an MDE approach, granting different solution-specific benefits to the domain-experts that would use them when developing gamification strategies.

3.5.1 GaML

GaML is a "language for modeling gamification concepts" (Philipp Herzig et al. 2013) with the primary objective of developing a readable language to non-technical gamification experts.

To develop the intended language, the developers of GaML structured gamification concepts using the taxonomy of game design elements provided by (Deterding et al. 2011), categorizing the concepts as game design elements with five different levels of abstraction, as previously described. The language itself focuses only on the first two levels (what "visual concepts exist and how these elements relate to each other" (Philipp Herzig et al. 2013)), while the other levels are related to the creation of a compelling gamification design, which is associated with the conceptualization of a specific design and not with the language.

For the first level of abstraction (game design patterns), basic visual gamification elements are pointed out, as well as possible synonyms in the specific context instance and its subtypes.

The second level of abstraction in the taxonomy of game design elements, defined as "Game Design patterns and mechanics", determines the gameplay factors of gamification, for instance, rules and conditions, since these elements insert logic into the gamification context. Further information about the approach adopted, and the game design elements chosen for each level of abstraction can be consulted on (Philipp Herzig et al. 2013).

The next step in developing GaML was the language specification, which was separated into the three following phases:

- **Design Objectives:** Four primary design objectives are defined for the language:
  - Domain experts should be capable of formalizing previously developed gamification concepts in GaML;
  - It should be possible to deploy an automatically compilable, valid instance of GaML into gamification platforms;
  - Trivial IT knowledge should be necessary to make GaML understood by domain experts;
  - IT experts should be fully capable of developing strategies in GaML, while domain experts should only be partially capable of doing so.

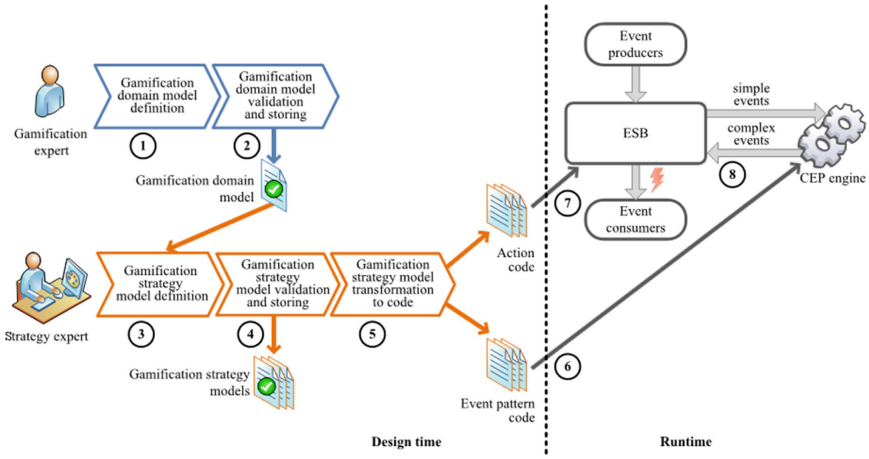


Figure 3.7: MEdit4CEP-Gam High-level Gamifying Process (Calderón, Boubeta-Puig, and Ruiz 2018)

- **Model and Syntax:** Describes the higher-level aspects of GaML's grammar, such as how to define the model, the game mechanics, for instance;
- **Semantics:** Explains how static semantics were defined in the language, and in which aspects they can be found, to assist the user with designing their gamification instance.

In short, GaML assists in the conceptualization and implementation of gamification. It was used to implement an achievement system within a Unity serious game called "Stop Smoking", to motivate the game's users by rewarding them with badges (Matallaoui, P. Herzig, and Zarnekow 2015). Even though one of the primary objectives was to create a language that could be partially writable by domain experts, it was stated that domain experts could not develop a model with complicated gamification strategies (Philipp Herzig et al. 2013).

3.5.2 MEdit4CEP-Gam

MEdit4CEP-Gam (Calderón, Boubeta-Puig, and Ruiz 2018) is a model-driven solution that can also be used by non-technical gamification experts but, unlike GaML (Philipp Herzig et al. 2013), graphical DSLs are used to allow gamification design graphically. This approach can successfully hide the implementation details when defining the desired model, which will later be transformed into the code to be executed by their system, designed with an Event-Driven Service-Oriented Architecture.

The procedure of MEdit4CEP-Gam approach present in figure 3.7 can be described by the following:

1. The gamification expert develops a graphical gamification domain by defining its event types and event properties;
2. Should the finished gamification domain model be invalid, the gamification expert is warned to fix the detected problems. Once the model is valid, it will be saved, so it is ready for import/export;

3. Using the previously defined model, "the strategy expert will create the gamification strategy models describing the activities the participants can perform, the awards they can receive and the analytics that need be monitored";
4. Like the domain model, the previously developed strategy model will also be automatically validated, and the user should correct any error so that this model can also be saved and ready to be imported/exported;
5. The strategy models will then be "automatically transformed into code, which consists of both the code implementing the conditions that must be met so that the Complex Event Processing (CEP) engine can detect situations of interest, and code of actions to be performed in the Enterprise Service Bus (ESB) when detecting such situations";
6. The first part of the generated code will be added to the Complex Event Processing (CEP) engine at runtime;
7. The second part of the generated code is added to the ESB at runtime;
8. Lastly, the ESB sends both simple events and previously defined event patterns, so that the CEP engine may create new complex events, which will be sent back to the ESB, in order to broadcast this new event information to each user of the platform in question, and the designers in question.

To define the domain-specific elements of the gamification context, the developers of (Calderón, Boubeta-Puig, and Ruiz 2018) used the first level of abstraction in the taxonomy of game design elements (Deterding et al. 2011) and proposed the definition of its domain be separated by the components category, and the mechanics category. The following is a small description of each category and its elements:

- **Components:** Contains the concepts which identify the context, as well as both the elements related to gamified systems and the game elements involved with gamification strategies.
  - **Application:** Identifies the system to be gamified;
  - **Course:** Represents relevant information about the course (educational context) to be gamified, within a said application;
  - **ActivityType:** Identifies possible user actions within the Application element (e.g. pressing a button/link);
  - **Event:** Defines a feature of an ActivityType. The feature can be monitored, evaluated, and measured;
  - **RewardType:** Establishes the types of rewards that can be obtained by the users. The proposed rewards are as follows: Points, Level, Badge, Leaderboard, Status, Prize, Certificate, Good (virtual goods).
- **Mechanics:** Embraces concepts "involved in the design of a gamification strategy". Its elements are used to define interactions between game components and the system.
  - **Strategy:** It's the main element in defining a gamification strategy. It consists of three different elements: Activity, Criterion, Reward;
  - **Activity:** Identifies an ActivityType element in a gamified Application, involved in the gamification strategy;

- **Reward:** Identifies a RewardType element, which has a weight attribute attached to it, that allows a designer to set the value of each reward of the strategy in question. The rewards can assist with measuring each user's performance;
- **Criterion:** Sets the conditions to accomplish for the Event element, depending on each user's performance. It also sets the Reward elements a user should receive, would the criterion be satisfied.

With these categories and elements in mind, the authors of (Calderón, Boubeta-Puig, and Ruiz 2018) proceeded to develop the metamodel through the usage of software such as the Eclipse Modeling Framework (EMF). The finished result of the metamodel in question is an extended version of the Model4CEP metamodel with both the gamification elements previously described and the components related with the CEP engine. (Calderón, Boubeta-Puig, and Ruiz 2018) provides more information about how the gamification components interact with the CEP setup.

This solution provides a solid tool for conceptualizing and implementing gamification while being user-friendly for domain-experts. Moreover, due to the usage of an MDE approach, it can transform the models developed by domain-experts into code, to later be monitored and controlled by their event-driven service-oriented architecture system. But, due to it having such a high-level graphical design, it fails to assist with the development of complex gamification strategies.

Chapter 4

Design

4.1 Metamodel Design 32
4.1.1 Design Process 32
4.1.2 Current Design 33

4.1 Metamodel Design

The core idea of this project is to devise a new method for gamification experts to enhance their services through gamified applications, which assists them in both processes of conceptualization and implementation. To do so, it was required to build a domain model attending to the studied best practices and design for gamification, as well as the development of a language that is clear enough to have any gamification expert being capable of writing their strategies.

Due to the project being based on the MDE approach, the design of the metamodel is of great importance to the overall development of the service since it affects the following development phases, as well as it is through this design that the users will be capable of developing complex gamification strategies. On this stage of the process, the EMF technologies were used to design and build the model represented in Figure 4.4, the model being its current version after various iterations, some of which are discussed within the next subsection.

4.1.1 Design Process

When designing the first iteration of the metamodel, the following game design elements were considered: game mechanics, social mechanics, achievements, conditions, rewards, actions, and triggers. The attributes of the system to be gamified were also considered when developing the metamodel due to one of the main objectives set, to assist domain experts in designing gamification strategies for their services, as previously stated. Figure 4.1 represents the first iteration of the metamodel designed.

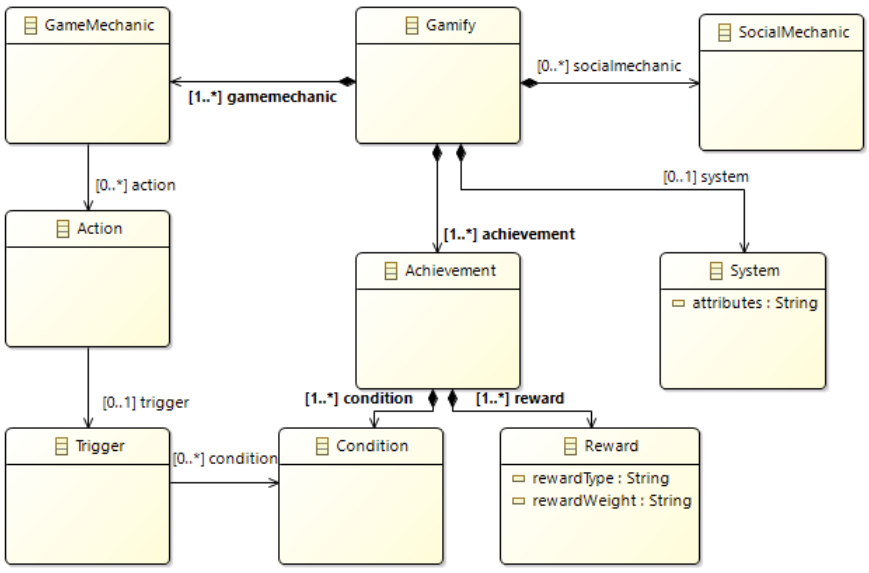


Figure 4.1: First Design Iteration

On the second iteration of the design process, new elements were added to the metamodel, enhancing some of the previously added elements, based on the information studied on (Deterding et al. 2011; Hunicke, Leblanc, and Zubek 2004; Werbach and Hunter 2012). In

this new iteration of the metamodel, it was concluded that social mechanics were within the category of game mechanics, thus there was no need to have both elements. Figures 4.2 and 4.3 represent the second iteration of the metamodel, along with new gamification concepts, while the following list contains general information about the new elements added, as well as their respective justification regarding the design decision made:

- **Types:** Elements such as DynamicTypes and ActionTypes were added to provide gamification experts clear options when defining their design strategy;
- **Reward Subclasses:** Allows gamification experts to choose and customize the type of reward users would attain should they complete a specific achievement;
- **Dynamics and Mechanics:** Presents various different dynamic presets for the gamified application to the domain expert, while allowing the customization of the main actions a user of the gamified application can take.

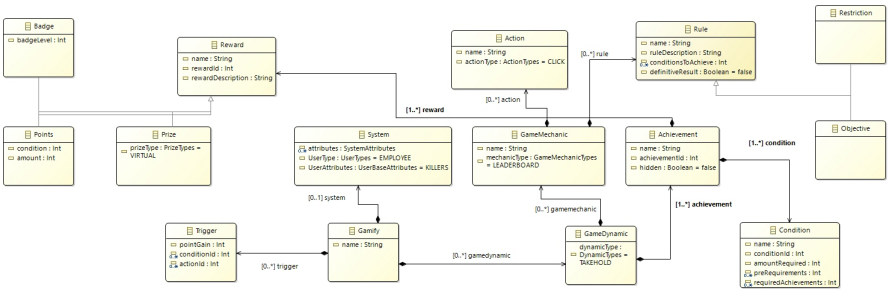


Figure 4.2: Second Design Iteration

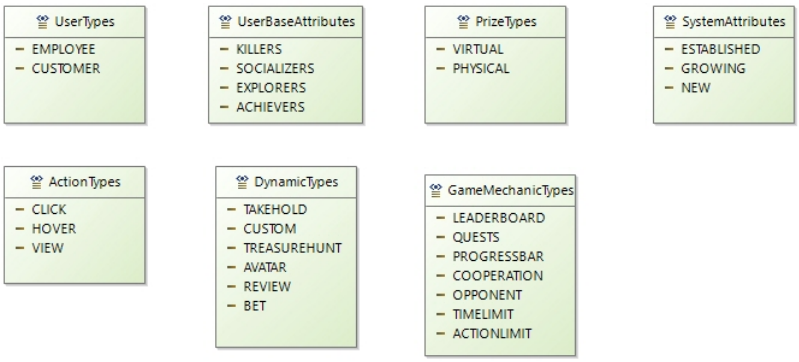


Figure 4.3: Second Iteration Gamification Concepts

The third, and current, metamodel design can be consulted on figure 4.4.

4.1.2 Current Design

With the information gathered, and documented, in chapter 3, the metamodel represented in figure 4.4 was developed, through an iterative process described in the previous subsection.

The various entities represented in the model listed below contain information about their purpose when designing a gamification strategy:

- **Gamify:** Serves as the main element for the development of gamification strategies;
- **System:** This entity allows the domain-experts to specify the state of their system to be gamified and to describe its user base. Once defined, the information saved in this entity will be used to assist the domain-expert with the gamification design to be developed (based on (Werbach and Hunter 2012));
- **GameDynamic:** Referring to the information acquired in (Hunicke, Leblanc, and Zubek 2004), this entity's main purpose is to guide users when developing their gamification design. It is used to define the overall dynamics of the gamification design desired, which can later be used to generate DSL presets, automatically creating a set of game mechanics and achievements depending on the dynamics defined by the user. A certain dynamic may be recommended depending on the user base attributes set in the entity "System";
- **GameMechanic:** Allows the specification of mechanics, events and rules, following guidelines provided by the MDA framework (Hunicke, Leblanc, and Zubek 2004);
- **Event:** Can be used to define user actions, or events, existent within a previously defined game mechanic, specifying its name and type of event;
- **Restriction:** Applies additional restrictions to a previously defined event, such as time limits or action limits;
- **Achievement:** Consists of a set of conditions and rewards, which can be obtained should the user perform the necessary actions which would satisfy the previously set conditions. An achievement can be hidden from the view of the users until its conditions are completed so that its rewards can be bestowed upon the user (Octalysis' sudden rewards strategy (Chou 2015));
- **Condition:** Contains the threshold required to satiate a game mechanic's condition, as well as requirements necessary for the condition to be active;
- **Reward:** Sets the rewards acquired within the list of available items whenever a user completes a set of conditions, such as badges, prizes, or points which can be used to complete another set of conditions;
- **Item:** Contains information about possible user rewards, such as prizes, badges, or points.

General details about the various types and presets available to the user when designing a gamification strategy are the following (see Figure 4.5):

- **SystemAttributes:** Contains different possible states of the system to be gamified;
- **UserTypes:** Used to specify who will be the users of the gamified system (based on section 3.2.2);
- **UserBaseAttributes:** Contains different user types based on the basic player types from (Bartle 1996);

Table 4.1: Basic visual game mechanics

Game Design Element	Synonyms	Subtypes	References
System	Service; Application.	-	(Chou 2015; Werbach and Hunter 2012)
Event	Dynamic Event; Mechanic Event; User Action.	-	(Hunicke, Leblanc, and Zubek 2004)
Condition	Requirement.	-	(Chou 2015; Werbach and Hunter 2012)
Restriction	Regulation; Limit.	-	(Deterding et al. 2011; Hunicke, Leblanc, and Zubek 2004)
Item	Goods; Collectible; Currency.	Badge; Points; Prize.	(Chou 2015), (Deterding et al. 2011)
Reward	Earned Goods/-Collectibles/Currency.	Fixed Reward; Random Reward.	(Chou 2015; Deterding et al. 2011)

- **DynamicTypes:** List of available preset dynamics (based on section 3.2.3), with the option to be generated into DSL text. The custom option does not provide any text generation;
- **GameMechanicsTypes:** List of game mechanics to be chosen by the user, which can generate text related to the option selected;
- **EventTypes:** Contains both possible events and user actions that can affect conditions;
- **PrizeTypes:** Type of goods that can be obtained by users;
- **RestrictionTypes:** Type of limits to be added on a specific event;
- **ConditionTypes:** Defines how will a condition be satiated.

Tables 4.1 and 4.2 contain information about each of the elements that reside in the metamodel. However, Table 4.3 provides information about the various reward strategies that the metamodel is prepared to replicate. The strategies in question are based on the Octalysis Framework (Chou 2015).

The following list contains information related to the column "Major Findings" of Table 4.3:

- **(1):** The usage of this strategy is particularly relevant in competitive scenarios between companies/shops. Users are more likely to buy items in a certain shop that ultimately rewards them for doing so;
- **(2):** Random rewards are often more effective in later stages of play, once all the non-repeatable fixed rewards had been collected. An exception to the previous affirmation

Table 4.2: Aggregated visual game mechanics

Design elements	Synonyms	Aggregates	References
Gamify	Context.	Item; System; GameDynamic.	-
GameDynamic	-	GameMechanic; Achievement.	(Hunicke, Leblanc, and Zubek 2004)
GameMechanic	-	Event; Restriction.	(Hunicke, Leblanc, and Zubek 2004)
Achievement	Quest; Mission;	Condition; Reward.	(Chou 2015)

is companies that specialize in providing random items to customers who find the surprise to be its own reward;

- **(3):** Depending on how the sudden reward is implemented, it can either cause users to share their experience, allowing other users the chance to replicate said experience or, should the sudden reward seem random, it can cause speculation within the community, having the users creating theories on how to obtain the reward in question;
- **(4):** This strategy allows users to have a chance of gaining very valuable goods, with relatively small effort. By maintaining the possible rewards visible, and by making this mechanic available in the early stages of play, it can attract new users to try their luck;
- **(5):** Social treasures often require a special type of points or currency which can only be obtained through interaction with other users. As such, this mechanic usually stimulates users to invite their friends to join them, or to socialize with other users, which helps both with solidifying a sense of community, and with increasing the general user base;
- **(6):** This last strategy rewards dedicated users by providing them with collectibles that have no value until all the pieces regarding a specific category are collected. Once a user gathers a full set, they receive a fitting reward, depending on the category of the pieces.

When designing the reward strategies for a gamification design, the variables "requiredAchievements" (Achievement) and "preRequirements" (Condition) can be used to either create a sequence of achievements (e.g. To obtain a level 2 badge, it is necessary to obtain the level 1 badge), each having their specific reward, or to create a complex condition for a particular achievement, effectively developing a challenge (e.g. a user needs to succeed on a set dynamic 10 times while using less than 10 total actions).

This current solution aims to succeed in being user-friendly for domain-experts, even when developing complex gamification strategies, due to the use of guidelines which adapt to the domain expert's choices. However, the solution currently lacks dynamics, game mechanics, and event types available, which may limit some gamification designs.

Table 4.3: Gamification reward strategies

Strategies	**Core Drives**	**Experiential Effects**	**Examples**	**Situations that make rewards visible**	**Major Findings**
Fixed Action Rewards	2;4;6.	Increases engagement; Builds loyalty;	Virtual or physical goods; Points; Collectibles; Currency.	System notifies the user about the actions required to get a specific reward.	(1)
Random Rewards	2;4;6;7.	Builds loyalty; Can enhance engagement of veteran users.	Randomized virtual or physical goods, collectibles, or amount of currency.	After successfully overcoming a previously set challenge or spending currency on a box of random goods.	(2)
Sudden Rewards	1;3;4;5;7.	Augments socialization within user base; Increases engagement.	Virtual or physical goods; Points; Collectibles; Currency.	Completing a hidden set of actions, or by finding an Easter Egg.	(3)
Rolling Rewards	1;2;4;5;6;7.	Can enhance user engagement substantially.	All types of rewards are eligible, from minimum to significant value.	Interacting with a lottery-like mechanic.	(4)
Social Treasure	3;4;5;6.	Augments socialization.	Points, currency, collectibles which cannot be obtained by other means.	Sending in-game gifts to friends; Inviting friends to join the platform.	(5)
Prize Pacing	2;4;5;6;7;8.	Augments engagement.	Collection of categorized shards, which turn into specific rewards once all shards are collected.	System informs the user about the obtainable reward after collecting a full set of shards.	(6)

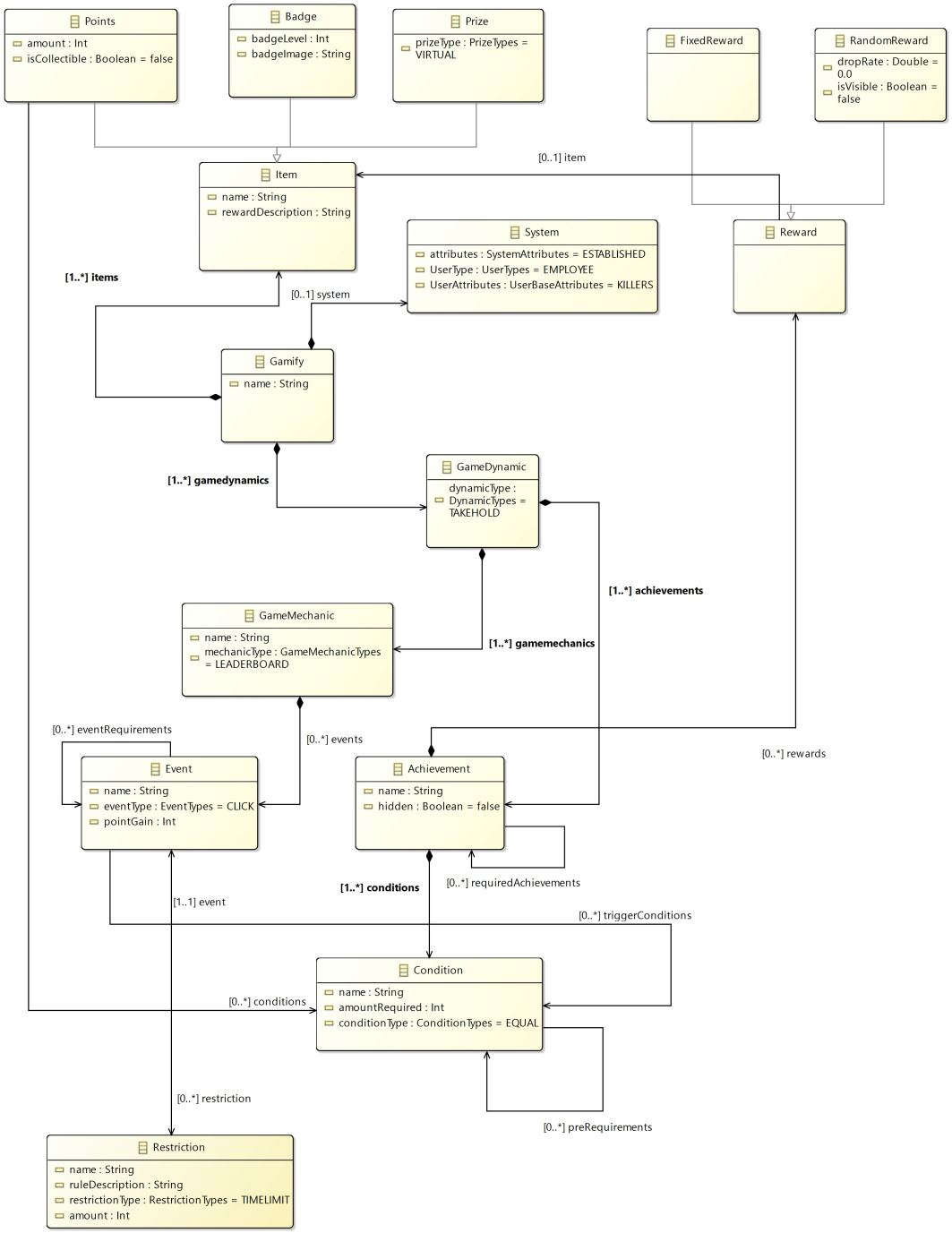


Figure 4.4: Metamodel Design

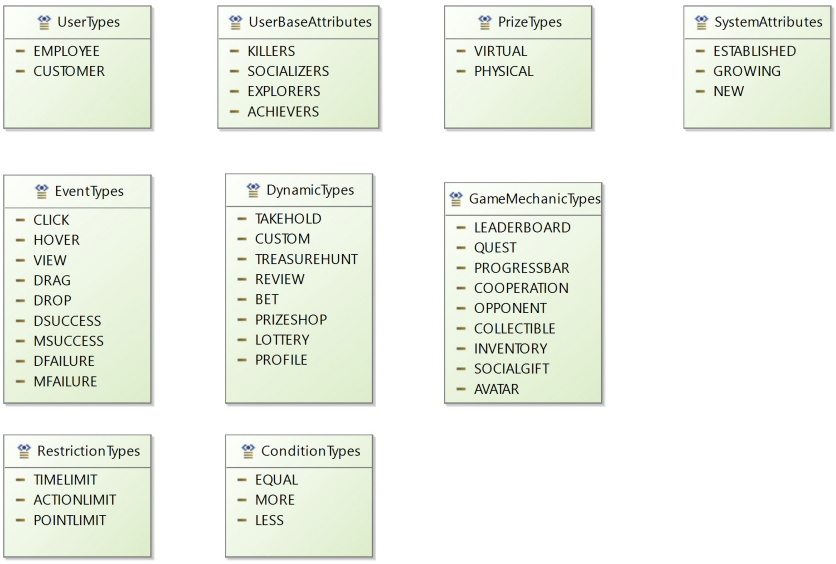


Figure 4.5: Other Gamification Concepts

Chapter 5

Development

5.1 Implementation process 42
5.1.1 Project Setup 42
5.1.2 DSL Development 42
5.1.3 Code Generation 49

5.1 Implementation process

In order to describe the implementation phase of the project, the entire process was split into different main stages. Each stage contains information about which technologies were used, decisions made, and its level of completion.

The following are general descriptions of the stages to be presented:

- **Project Setup:** Presents the initial steps taken to create and prepare the project for the following implementation phases;
- **DSL Development:** Contains information about the state of the textual DSL, through short examples of the textual DSL;
- **Code Generation:** Provides insight on how the code is generated, as well as some examples of how the textual DSL affects the code generation.

Although the work developed in each stage of implementation began in the order previously defined, they did not find their final state of completion without the development of following stages, provoking the necessity of updating previous stages (excluding the setup stage).

5.1.1 Project Setup

In this first stage of implementation, the previously designed model is used to generate the necessary components for a new Xtext project. In doing so, Xtext prepares the new project with the information provided by the model, allowing it to create a new Xtext (.xtext) file with all of the classes, attributes and relationships set in the designed model. Although the model's initial file type is Ecore (.ecore), a Xtext project can only be created using an EMF Generator Model (.genmodel). The setup process of the project can be summed up to the following steps:

1. Create a new EMF Generator Model (.genmodel) file using the previously developed Ecore (.ecore) model (consult Chapter 4 for further information about the model);
2. Using the model file created in the previous step, begin the process of creating a new Xtext project;
3. From the various classes set within the model, "Gamify" is chosen as the entry rule for the new project;
4. The name for the project is established, as well as the extension for the DSL.

5.1.2 DSL Development

For the development of the DSL, the previously set Xtext project was utilized to design the desired language, including the development of guidelines to assist users when writing their gamification strategies. Figure 5.1 consists of an excerpt of the textual DSL in an early stage of development, while figure 5.2 is an example of how a user could write a gamification strategy with the said DSL.

Although the textual DSL was already in a decent shape, some changes were necessary to make it fully functional. In Xtext, to describe a new textual DSL it is first required to

```
System returns System:
    {System}
    'System'
    '{'
        ('attributes' attributes=SystemAttributes)?
        ('UserType' UserType=UserTypes)?
        ('UserAttributes' UserAttributes=UserBaseAttributes)?
    '}';


GameDynamic returns GameDynamic:
    'GameDynamic'
    '{'
        ('dynamicType' dynamicType=DynamicTypes)?
        'gamemechanics' '{' gamemechanics+=GameMechanic ( "," gamemechanics+=GameMechanic)* '}'
        'achievements' '{' achievements+=Achievement ( "," achievements+=Achievement)* '}'
    '}';

Item_Impl returns Item:
    {Item}
    'Item'
    '{'
        ('name' name=EString)?
        ('rewardDescription' rewardDescription=EString)?
    '}';

enum SystemAttributes returns SystemAttributes:
                ESTABLISHED = 'ESTABLISHED' | GROWING = 'GROWING' | NEW = 'NEW';

enum UserTypes returns UserTypes:
                EMPLOYEE = 'EMPLOYEE' | CUSTOMER = 'CUSTOMER';

enum UserBaseAttributes returns UserBaseAttributes:
                KILLERS = 'KILLERS' | SOCIALIZERS = 'SOCIALIZERS' | EXPLORERS = 'EXPLORERS' | ACHIEVERS = 'ACHIEVERS';
```

Figure 5.1: Early DSL specification

```
Gamify{
    name Gamification
    gamedynamics{
        GameDynamic{
            dynamicType BET
            gamemechanics{
                GameMechanic{
                    events{
                        Event{
                            name Betting
                            eventType CLICK
                        }
                    }

                }
            }
```

Figure 5.2: Early DSL Example

declare the EPackages to import, which includes the developed Ecore model represented in Chapter 4, as well as the Ecore library, so it is possible to access and utilize the specific types present Ecore models. Other tweaks were made to ensure the full functionality, involving the generated grammar rules when the project was first created. To appropriately process any input provided by users of the DSL, Xtext has a base set of terminal rules prepared to receive various types of input. The following list describes each of the terminal rules used in the development of the grammar:

- **ID:** Used as a unique identifier for entities, the `ID` rule is more commonly used to process inputs for the "name" attribute contained in each entity of the metamodel. This rule does not allow the use of spaces as well as most symbols, making it an appropriate option in scenarios in which spaces or symbols could cause issues. Listing 5.1 represents the terminal rule in question;
- **INT:** Used to handle integer inputs (e.g. amount of points to be earned);
- **STRING:** Allows users to type freely. This rule is mostly used as a means of communication between the gamification expert and the team of developers. The written text must be within quotation marks.

```
terminal ID:
('^')?('a'..'z'|'A'..'Z'|'_') ('a'..'z'|'A'..'Z'|'_'|'0'..'9')*;

terminal INT returns ecore::EInt:
('0'..'9')+;

terminal STRING:
'"' ( '\\' . /* 'b'|'t'|'n'|'f'|'r'|'u'|'"'|"'"|'\\' */ | !('\\'|'"') )*
    '"' |
"'" ( '\\' . /* 'b'|'t'|'n'|'f'|'r'|'u'|'"'|"'"|'\\' */ | !('\\'|"'") )*
    "'";
```

Listing 5.1: Terminal rules

Through the previously described terminal rules, it is possible to develop more complex rules fit for different purposes. Listing 5.2 contains rules developed to process its respective input. The rule `DOUBLE` is prepared to handle with decimal numbers, while `FQNString` can handle paired up `ID` rules in conjunction with a dot.

```
FQNString returns ecore::EString:
    (ID)'.'(ID)

terminal DOUBLE returns ecore::EDouble:
    INT ('.' INT)?;
```

Listing 5.2: Developed rules

Xtext has a unique feature that allows the declaration of cross-links, allowing users to perform cross-references which consists in the act of referencing a previously defined entity by inserting its assigned `ID`. Figure 5.3 is a simple example of this feature in action, though the rule `FQNString` is prepared for complex cross-references.

By applying the tweaks previously described on the grammar presented in Figures 5.1 and 5.2, it is possible to design a gamification strategy and proceed to the next phase of the

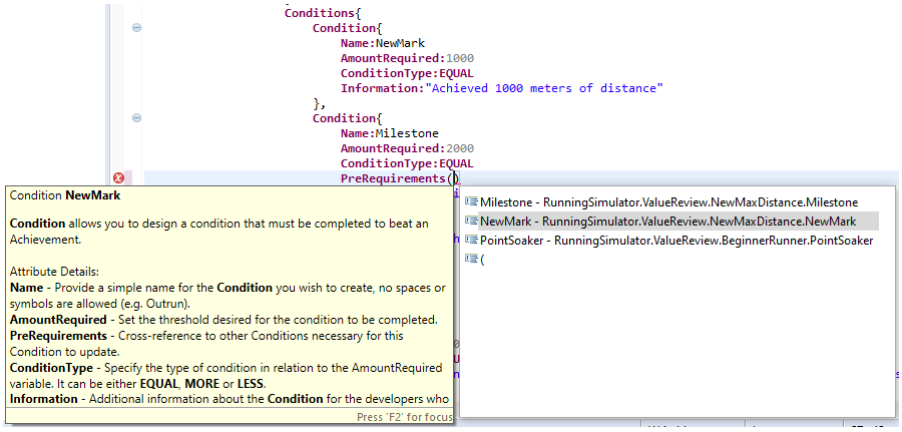


Figure 5.3: Simple Cross-reference Example

process: to generate the desired code. However, one of the main objectives of this project is to provide a user-friendly textual DSL, as such, the language suffered modifications to become easier to read and write for gamification experts with little knowledge in programming languages. Having the possibility of these modifications not being enough to assist every user in understanding the language, guidelines were developed which provide information about the various elements of the language, as well as providing viable gamification strategies. Figure 5.4 represents a portion of the language developed in its current state.

```
GameMechanics{
    GameMechanic{
        Name:NewThresholds
        MechanicType:QUEST
        Events{
            Event{
                Name:Running
                EventType:MSUCCESS
                PointGain:10
                TriggerConditions(BeginnerRunner.PointSoaker)
                Restrictions(
                    Restriction{
                        Name:TimeRestriction
                        RuleDescription:"Limits the amount of time left until the quest can no longer be attainable"
                        RestrictionType:TIMELIMIT
                        Amount:10
                        Information:"Link this restriction with the service code"
                    }
                )
                Information:"Insert a quest event to further reward users"
                }
            }
        Information:"Link game mechanic with service code"
        }
    }
```

Figure 5.4: Portion of a Gamification Strategy

As previously explained, to increase the language's readability and to provide further insight about gamification, guidelines were developed which regard each entity present in the model. With the purpose of making the guidelines feel both opportune and helpful, it was decided to have the information visible whenever a user is hovering their cursor over a specific entity. This was achieved through editing the UI portion of the Xtext project, by performing the following modifications:

- **DSLEObjectHoverProvider:** Extension of the Java class DefaultEObjectHoverProvider, allows users to have access to examples of how to write a specific entity by hovering over it (see function in Listing 5.3);
- **DSLEObjectDocumentationProvider:** Extension of the Java class IEObjectDocumentationProvider, it is used to provide the full documentation about each of the

model's entities, which includes a small description of the entity in question, attribute details, examples, and strategy guidelines (see Listing 5.4);

- **DSLUiModule:** Java class used to register the new previously created UI components (see Listing 5.5).

```
@Override
protected String getFirstLine(EObject o) {
    ...

    if (o instanceof Item) {
        return "Example:<br>"
            + "<b>Prize{</b><br>"
            + "    <b>Name:</b> PrizeExample<br>" +
            "    <b>RewardDescription:</b><b>\"</b>
    Example of a description of a reward<b>\"</b><br>" +
              "    <b>PrizeType:</b> VIRTUAL<br>" +
              "    <b>Information:</b> <b>\"</b>
    Example of important information.<b>\"</b><br>" +
              " <b>}</b>";
    }

    ...
        return super.getFirstLine(o);
    }
```

Listing 5.3: Excerpt from Function getFirstLine in DSLEObjectHoverProvider

```
@Override
public String getDocumentation(EObject o) {
    ...

    if (o instanceof Item) {
        return "An <b>Item</b> is where you may design the rewards for
    your users.<br>"
            + "An <b>Item</b> can be defined as <b>Points</b>, <b>Prize
    </b>, or <b>Badge</b>.<br>"
            + "Depending on the initial choice you've made, the system
    will provide you with different choices in attributes for the Item
    you desire to create by pressing <b>CTRL+SPACE</b>.<br>"
            + "<br>"
            + "Attribute Details:"
            + "<br>"
            + "<b>Name</b> - Provides a simple name for the <b>Item</b>
    you wish to create, no spaces or symbols are allowed (e.g. for an
    Item of type Prize, an example could be a Car).<br>"
            + "<b>RewardDescription</b> - This description is read by
    users who wish to know more about what the reward in question entails
    . Use quotation marks.<br>"
            + "<b>Information</b> - Additional information about the <b>
    Item</b> for the developers who will be working with the generated
    code. Use quotation marks.<br>"
            + "<b>PrizeType</b> - Applies only for Items of type <b>
    Prize</b>. Clarifies the type of prize to be rewarded, it can be
    either <b>VIRTUAL</b> or <b>PHYSICAL</b>.<br>"
```

```
16              + "<b>BadgeLevel</b> - Applies only for Items of type <b>
    Badge</b>. Insert a number to function to establish how meaningful
    the badge is, the higher, the more meaningful it is. <br>"
17              + "<b>BadgeImage</b> - Applies only for Items of type <b>
    Badge</b>. Add a link pointing to where the developers may find the
    appropriate image for the badge. Use quotation marks.<br>"
18              + "<b>Amount</b> - Applies only for Items of type <b>Points
    </b>. Insert the amount of points a user may receive.<br>"
19              + "<b>Conditions</b> - Applies only for Items of type <b>
    Points</b>. Insert the conditions that will be affected by the
    rewarded amount of points previously set. It's recommended to already
     have conditions prepared in order to list them.<br>"
20              + "<b>isCollectible</b> - Applies only for Items of type <b>
    Points</b>. Establishes whether or not the user is being rewarded
    with pieces of a greater reward or simply with points. Can either be
    <b>true</b> or <b>false</b>.<br>"
21              + "<br>"
22              + "Strategy Guidelines:<br>"
23              + "Rewards are fundamental in any gamification strategy, the
     <b>Item</b> element is used to create these rewards so you can later
     link them to various other elements of type <b>Achievement</b>.<br>"
24              + "The types of rewards recommended differ depending on the
    user-base in question. If the users in question give importance to
    their status in comparison with others, rewards linked with specific
    leaderboards are advised. These rewards should be meaningful to
    motivate users in maintaining/improving their positions."
25              + "<br>Should the users be considered as achievers, then a
    set of achievements providing small rewards leading to a heftier
    reward may be more appropriate.";
26      }
27
28      ...
29      return null;
30  }
```

Listing 5.4: Excerpt from Function getDocumentation in DSLEObjectDocumentationProvider

```
1  @FinalFieldsConstructor
2  @SuppressWarnings("all")
3  public class DSLUiModule extends AbstractDSLUiModule {
4    public DSLUiModule(final AbstractUIPlugin plugin) {
5      super(plugin);
6    }
7    public Class<? extends IEObjectHoverProvider>
       bindIEObjectHoverProvider() {
8      return DSLEObjectHoverProvider.class;
9    }
10
11   public Class<? extends IEObjectDocumentationProvider>
       bindIEObjectDocumentationProviderr() {
12     return DSLEObjectDocumentationProvider.class;
13   }
14 }
```

Listing 5.5: Excerpt from DSLUiModule class

By adding the class extensions and registering them in the DSLUiModule class, users now have access to all the documentation regarding each entity, as well as the assistance provided by the DSL when writing gamification strategies, as presented in Figures 5.5 and 5.6.

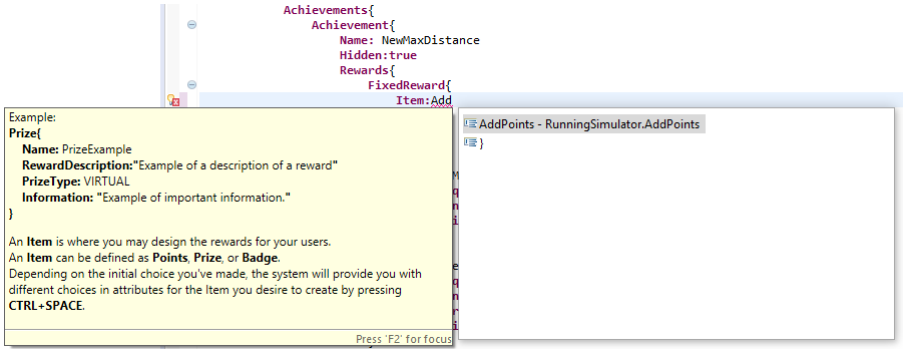


Figure 5.5: Example of the DSL assistance

Example:
Prize{
Name: PrizeExample
RewardDescription:"Example of a description of a reward"
PrizeType: VIRTUAL
Information: "Example of important information."
}

An **Item** is where you may design the rewards for your users.
An **Item** can be defined as **Points**, **Prize**, or **Badge**.
Depending on the initial choice you've made, the system will provide you with different choices in attributes for the Item you desire to create by pressing **CTRL+SPACE**.

Attribute Details:
Name - Provides a simple name for the **Item** you wish to create, no spaces or symbols are allowed (e.g. for an Item of type Prize, an example could be a Car).
RewardDescription - This description is read by users who wish to know more about what the reward in question entails. Use quotation marks.
Information - Additional information about the **Item** for the developers who will be working with the generated code. Use quotation marks.
PrizeType - Applies only for Items of type **Prize**. Clarifies the type of prize to be rewarded, it can be either **VIRTUAL** or **PHYSICAL**.
BadgeLevel - Applies only for Items of type **Badge**. Insert a number to function to establish how meaningful the badge is, the higher, the more meaningful it is.
BadgeImage - Applies only for Items of type **Badge**. Add a link pointing to where the developers may find the appropriate image for the badge. Use quotation marks.
Amount - Applies only for Items of type **Points**. Insert the amount of points a user may receive.
Conditions - Applies only for Items of type **Points**. Insert the conditions that will be affected by the rewarded amount of points previously set. It's recommended to already have conditions prepared in order to list them.
isCollectible - Applies only for Items of type **Points**. Establishes whether or not the user is being rewarded with pieces of a greater reward or simply with points. Can either be **true** or **false**.

Strategy Guidelines:
Rewards are fundamental in any gamification strategy, the **Item** element is used to create these rewards so you can later link them to various other elements of type **Achievement**.
The types of rewards recommended differ depending on the user-base in question. If the users in question give importance to their status in comparison with others, rewards linked with specific leaderboards are advised. These rewards should be meaningful to motivate users in maintaining/improving their positions.
Should the users be considered as achievers, then a set of achievements providing small rewards leading to a heftier reward may be more appropriate.

Figure 5.6: Guidelines from the Item Entity

5.1.3 Code Generation

The third stage of development is based on defining the code to be generated once the gamification expert designs their strategy, depending on the user's design choices. It is required that the generated code faithfully follows the strategy defined in the textual DSL, although it will not be fully prepared for deployment and thus it is necessary the assistance of IT experts to integrate the generated code on the service in question.

In this Xtext project, the code generation process is based on a single Xtend (.xtend) class named DSLGenerator that allows the specification of the files to be generated, as well as the definition of the code to be added in each of the previously specified files. The DSLGenerator is a class extension to Xtext's AbstractGenerator, and through it, the various files to be generated were declared by using the function presented in Listing 5.6.

```
1  class DSLGenerator extends AbstractGenerator {
2
3
4      override void doGenerate(Resource resource, IFileSystemAccess2 fsa,
       IGeneratorContext context) {
5
6          for (e : resource.allContents.tolterable.filter(Gamify)) {
7              fsa.generateFile(e.name + ".cs", e.compile)
8          }
9          fsa.generateFile("Achievement.cs", compileAchievement)
10         for (e : resource.allContents.tolterable.filter(Achievement)) {
11             fsa.generateFile(e.name
12             + ".cs", e.compileCustomAchievement)
13         }
14
15         ...
16     }
```

Listing 5.6: Excerpt from DSLGenerator class

Due to the code generation process being similar for every entity in the model, the following is an example describing the said process for the Gamify entity:

1. The function begins with a search for each entity of type Gamify written by the gamification expert.
2. For each entity of type Gamify, a new file is created using the name established for the entity in the DSL.
3. The code present in the compile function is added to the file previously created (see Figure 5.7).

Although most of the generated code falls within the lines of what is presented in Figure 5.7, the functions from the Condition class in Listing 5.7 play an important role in assuring that an Event entity can, ultimately, lead to a Reward.

Lastly, the Main class is entrusted with handling the declaration of each entity generated, thus simplifying the process of integration with existing services since it is only necessary to create a new Main object for it to instantiate the objects related to the gamification strategy established.

```
def compile(Gamify f) '''
    using UnityEngine;
    using System.Collections;

    public class «IF f.name !== null»«f.name.toFirstUpper»«ELSE»Gamify«ENDIF» {

    private List<Item> items;
    private List<GameDynamic> gameDynamics;

    public «IF f.name !== null»«f.name.toFirstUpper»«ELSE»Gamify«ENDIF»(){
        this.items = new List<>();
        this.gameDynamics = new List<>();

    }

    public «IF f.name !== null»«f.name.toFirstUpper»«ELSE»Gamify«ENDIF»(List<Item> items, List<GameDynamic> gameDynamics){
        this.items= items;
        this.gameDynamics= gameDynamics;

    }

    public String getGameDynamics() {
        return this.gameDynamics;
    }

    public void setGameDynamics(List<GameDynamic> gameDynamics) {
    this.gameDynamics = gameDynamics;
    }
    public String getItems() {
    return this.items;
    }

    public void setItems(List<Item> items) {
    this.items = items;
    }
    /*
    Additional information:
    «f.information»
    */
    }
'''
```

Figure 5.7: Gamify's Compile Function

```
1  public Boolean checkPreRequirements ()
2  {
3    if (this.preRequirements != null) {
4      foreach (Condition condition in preRequirements) {
5        if (!condition.checkCompleted ()) {
6          return false;
7        }
8      }
9    }
10   return true;
11 }
12
13 public void updateAmount (int amount)
14 {
15   if (checkPreRequirements ()) {
16     this.amountCurrent += amount;
17     checkCompleted ();
18   }
19 }
20
21 public Boolean checkCompleted ()
22 {
23   if (amountCurrent == amountRequired) {
24     Debug.Log ("Milestone completed");
25     this.completed = true;
26     return true;
27   }
28   return false;
29 }
```

Listing 5.7: Functions from the Condition class

Chapter 6

Evaluation and Experimentation

6.1 Solution Testing 52
6.1.1 Problem 1 - DSL 52
6.1.2 Problem 2 - Gameful Experiences 55

6.1 Solution Testing

To test the project, the problem presented in chapter 1 is analyzed further in tandem with what was implemented to split the main problem into different parts, allowing the possibility of testing specific aspects of the project in question.

As defined in chapter 1, the developed solution intends to mitigate the problem of gamified applications failing to meet their business objectives due to poor design (Burke 2014). To achieve this, the solution provides a baseline design through the usage of a metamodel, as well as a textual domain-specific language, allowing users to write their gamification strategies while being implicitly guided by the metamodel's structure. Once the strategy is defined, the user can generate code that mirrors what was previously specified in the DSL. The generated code is later sent to the team of developers who were charged with handling the deployment of gamification on their service to successfully link the deployed code with the service in question, as well as making any necessary changes.

With the process behind the solution explained, there are two main possibilities of failure that can nullify the solution's usefulness:

- The domain expert does not understand how to develop gamification strategies on the textual DSL;
- The code generated does not support the process of providing gameful experiences.

Although the solution presented aims to increase the chances of a gamified application to meet a company's set business objectives, it only applies to the situations where poor gamification design was involved, as such, tests related to business risks behind integrating gamification in services are not performed, since it is up to the company's stakeholders to conclude if the said integration would be successful. Thus, the following subsections contain detailed information about how each of these possibilities of failure is tested.

6.1.1 Problem 1 - DSL

Following the possibilities of failure previously assessed, the first problem to analyze envelops both the quality of the DSL structure and its guidelines. As such, the problem has been defined as follows: "The textual DSL is understandable and writable by gamification experts with little IT knowledge".

Metrics

To evaluate the issue, the metrics used are based on the results obtained through a questionnaire that verifies the level of IT knowledge the participant has before presenting a picture with a sample of a guideline developed in the implementation phase. It is then requested of the participant to write a part of the strategy, following the suggestions provided by the guideline previously presented. Thus, both the effectiveness of the guidelines and the simplicity of the DSL are tested.

Test Methodology

As previously mentioned, the data used to evaluate the language and the guidelines developed consists of the results obtained through a questionnaire that can be consulted in Appendix C. The preferred minimum number of samples gathered would be 30, to not only ensure

accurate results, but also to allow hypothesis testing. A split population between participants with IT knowledge and participants without it would be desirable for the test in question.

Test Results

At the time of this delivery, not enough samples were gathered to formally prove an hypothesis. As such, the following is a showcase of the results obtained from the questionnaire, accompanied by an analysis regarding these results.

Since the goal of this test is to prove that participants with little programming experience can understand and write gamification strategies through the language developed, a comparison between participants with programming experience and without is made. By splitting the population and evaluating the results obtained from each side, it is possible to conclude whether or not the language is just as readable for both types of participants. To do so, a question was set to gather the level of experience the participant has with programming languages, providing them with 4 different options:

- None;
- Little;
- A good amount;
- Expert.

The population was divided between the participants with little to no knowledge about programming languages, and the participants who are experienced with programming languages. Figure 6.1 is a graph containing the results from 17 participants obtained regarding the question presented. With the information gathered, the population can be split, having 9 participants that are not experienced with programming languages, and 8 participants that have experience in the area.

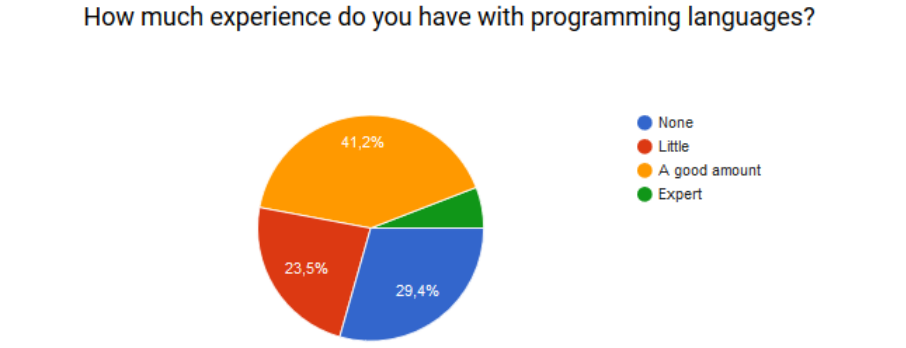


Figure 6.1: Results Graph for Question 1

After reading information about what is a DSL, and how this particular DSL can be used, the participants are met with another question, allowing them to develop an Item entity of type Prize (consult Appendix C for further information). The results obtained from this question are used to prove the hypothesis. An answer is considered correct if it follows the structure developed to create a Prize, but the success rate varies depending on how strict the evaluation method is. Since any participant is prone to make mistakes, independently of how knowledgeable they are about this subject, a system of points was developed to handle the correctness of each answer. The evaluation of an answer is split into two parts: the

structure and the variables used. Correct answers are expected to have 5 different structure elements and 4 variable elements. Each of these elements is worth 1 point, signifying that a fully correct answer can go up to 9 points. Should an element be partially incorrect, only half its worth is considered for the sum calculations.

The following number list are the results (ranked in order) obtained when evaluating the answers with the method previously mentioned:(4.5, 7, 7, 7.5, 8, 8, 8.5, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9).

By analyzing the list of results, the following observations were made:

- Participant's average (approximately): 8.27;
- Participant's median: 9;
- Non-experienced participant's average (approximately): 8.17;
- Non-experienced participant's median: 9;
- Experienced participant's average (approximately): 8.37;
- Experienced participant's median: 9.

From the data gathered, it is concluded that the participants can understand and define their entities using the structure provided. This conclusion is due to the 8.1% score difference between the average answer and the fully correct answer, which can, theoretically, be mitigated by the additional assistance the DSL supplies (see Chapter 5). There is also a 2.4% point difference between the participants with experience in programming and the participants without, making it a positive result towards proving that inexperienced participants can understand the language in question.

In theory, if the average points amount were to reach a value below 6, it can be considered that the DSL, as well as the guidelines, failed to assist the users in designing the Prize requested. Considering the number of samples gathered is 17, for the average of 30 samples total to become below 6, the missing 13 samples' average answer had to score below 3.73 points. Thus it is concluded that the DSL and its guidelines are likely to be sufficient in providing the necessary information for the users to design their strategies.

In the final section of the questionnaire, participants can evaluate the complexity of the language developed, as well as the usefulness of the guidelines provided. Figures 6.2 and 6.3 represent the results obtained regarding each of these subjects.

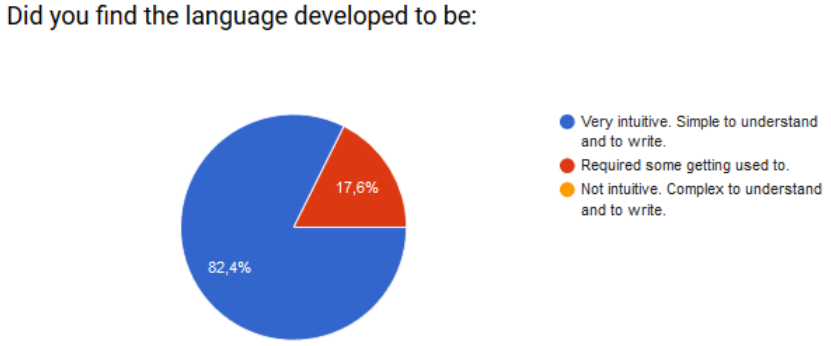


Figure 6.2: Results Graph for Question 3

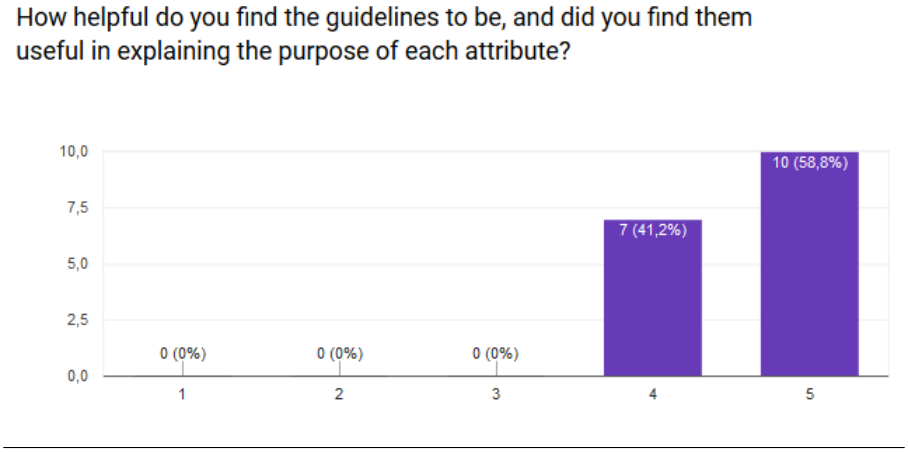


Figure 6.3: Results Graph for Question 4

Using the information gathered in this final section in conjunction with the results obtained in the previous one, the participants had a positive response towards the language and the guidelines presented to them.

6.1.2 Problem 2 - Gameful Experiences

The second issue detected regards the generated code obtained through the usage of the DSL previously discussed (consult Chapter 5 for details). The problem in question gauges whether or not the generated code is capable of providing the effects desired when designing gamification strategies: to affect human behavior. As such, the issue is defined as follows: "The generated code is capable of providing a structure for a gamified application based on the design decisions written on a DSL instance."

Metrics

To evaluate the problem in question, a design strategy was devised through the textual DSL developed to generate the necessary code for the test. Once the code is generated, it is deployed and integrated into a simple Unity application previously set. This application is later used as the environment for the evaluation (further information about the strategy and the application developed can be found in Appendix D).

Through the developed application, it's possible to measure how much users interact with it, as such, two versions of the application are used:

- Unity application without any gamification elements;
- Unity application with gamification elements.

To gather the needed user-data, both applications are shared in Unity's forums, providing it some visibility. As such, data about the overall interactions of the users with both versions of the application can be gathered to proceed with the evaluation.

Hypothesis

Having μ_b representing the data behind the users' interactions with the base application, and μ_a as the data regarding the users' interactions with the gamified version of the application:

$$H_0 : \mu_a - \mu_b = 0 \tag{6.1}$$

$$H_1 : \mu_a - \mu_b > 0 \tag{6.2}$$

The goal of this hypothesis is to prove H_1, signifying that there is a tendency for an increased number of interactions when the application has elements of gamification.

Test Methodology

As before, a minimum number of 30 participants is necessary to attain reliable results. To obtain the required data for the hypothesis previously set, the A/B test experiment is used, which consists of granting half of the population access to the base Unity application without any gamification elements, while the other half interacts with the gamified version of the said application. Once enough data is gathered, a paired Student's t-test is performed to prove the hypothesis in question.

Test Results

At the time of this delivery, the application has yet to receive results.

Chapter 7

Conclusion

7.1 Current Objective Completion and Limitations 58
7.2 Future Work and Other Remarks 58
7.2.1 Notes on Gamification 59
7.2.2 Notes on Model-Driven Engineering 59

7.1 Current Objective Completion and Limitations

In Chapter 1 of this document, a set of objectives involving research about the state of gamification, different examples of its successful and failed incorporation in services, and about other possible solutions to the problem of poor gamification design, were established. Table 7.1 lists each of the previously defined objectives and their respective current state of completion.

Table 7.1: Table of set objectives

Objectives	**State**
Examine successful examples of gamification	Completed
Examine reported failures of gamification	Completed
Analyze different ways to incorporate gamification in applications	Completed
Delineate MDE approaches for the baseline design of gamification	Completed
Solution uses MDE approach and allows the implementation of complex gamification strategies	Completed

Difficulties and limitations found during the initial phase of the project development were mostly related to the value analysis process, due to how the theme of this project was set, many of the methods related to the Front End of Innovation did not seem to fit within the context at hand, the same could also be applied to the CANVAS model.

During the design and implementation phase of the project, various issues related to the high number of design and implementation possibilities made it difficult to devise the best solution for the objectives set, as well as overall difficulties with the setup and code generation phases of the Xtext project.

7.2 Future Work and Other Remarks

As for work to be developed, the solution currently grants a limited number of types (e.g condition types, types of game mechanics, between others), as such, an expansion of the available options would be a welcome addition to the solution. The code generation portion of the solution is also only prepared to generate code in CSharp, thus, adding a new attribute in the metamodel that provides different options for the programming languages that the code generator is prepared for would also contribute for a more versatile solution. Addressing other framework problems presented in section 3.2.4, such as the overuse of the Bartle model when establishing user types, is also a possibility for future work. Regarding the evaluation phase, it was not possible to obtain the necessary samples to perform hypothesis testing for the problems established, thus it is left as future work.

The solution is publicly accessible, and any further developments on it can be consulted using the link below:

- https://bitbucket.org/1140459/gamify/

The information gathered in this project, as well as the solution developed, was also used for the realization of a chapter named "Gamification: Model-Driven Engineering Approaches" for the book "Software Engineering for Agile Application Development", which has now passed its first review stage. For further details about the book, consult the link below:

- https://www.igi-global.com/publish/call-for-papers/call-details/3875

The following subsections contain personal notes about the two main topics researched for the development of the solution presented: Gamification and Model-Driven Engineering.

7.2.1 Notes on Gamification

Gamification is an intriguing concept, but especially in the context in which the user-base consists of employees of a specific company. In this scenario, an incredibly thorough analysis of the possible benefits of integrating gamification in the system is required, to assess if the actions that are being promoted in the gamified system will greatly benefit the company. This is mainly due to the array of rewards necessary to make it function since employees will not feel interested, nor motivated, to perform an action they wouldn't otherwise do without knowing they will be handsomely rewarded for doing so (e.g. an increase of monthly income). For the company to maintain growth, the actions promoted in the gamified system have to outweigh the possible rewards to the employees, while maintaining their interest.

On another note, although there are various articles about its concept, gamification remains without a specific, unarguable, definition. Using the information learned from the research made about the subject, this may be because of how drastically different two successful gamified solutions can be due to the wide array of variables to consider when developing such solutions.

7.2.2 Notes on Model-Driven Engineering

With the work developed involving the solution presented in this document, Model-Driven Engineering seems to be a viable approach to solve problems related to design, but it also seems it could be a possible solution to reduce workload should the need be adding a new, moderately complex, functionality to an already existing system, which can be an interesting option to explore in the future.

Bibliography

Bartle, Richard (1996). "Hearts, clubs, diamonds, spades: Players who suit MUDs". In: *Journal of MUD research* 1.1, p. 19.

Brambilla, Marco, Jordi Cabot, and Manuel Wimmer (2017). *Model-Driven Software Engineering in Practice: Second Edition*. English. 2 edition. San Rafael, Calif.: Morgan & Claypool Publishers. isbn: 978-1-62705-708-0.

Burke, Brian (2014). *Gamify: How Gamification Motivates People to Do Extraordinary Things*. isbn: 978-1-937134-85-3.

Calderón, Alejandro, Juan Boubeta-Puig, and Mercedes Ruiz (2018). "MEdit4CEP-Gam: A model-driven approach for user-friendly gamification design, monitoring and code generation in CEP-based systems". In: *Information and Software Technology* 95, pp. 238–264.

Chou, Yu-kai (2015). *Actionable gamification: Beyond points, badges, and leaderboards*. Octalysis Group.

Deci, Edward L and Richard M Ryan (2012). "Motivation, personality, and development within embedded social contexts: An overview of self-determination theory". In: *The Oxford handbook of human motivation*, pp. 85–107.

Deterding, Sebastian (2015). "The Lens of Intrinsic Skill Atoms: A Method for Gameful Design". In: *Human-Computer Interaction* 30, pp. 294–335.

Deterding, Sebastian, Dan Dixon, Rilla Khaled, and Lennart Nacke (2011). "From game design elements to gamefulness: defining gamification". In: *Proceedings of the 15th international academic MindTrek conference: Envisioning future media environments*. ACM, pp. 9–15.

EMFText (2018). original-date: 2012-08-07T06:36:34Z. url: `https://github.com/DevBoost/EMFText` (visited on 02/08/2019).

Gronback, Richard C. (2009). *Eclipse Modeling Project: A Domain-Specific Language (DSL) Toolkit*. en. Pearson Education. isbn: 978-0-321-63519-8.

Hamari, J., J. Koivisto, and H. Sarsa (2014). "Does Gamification Work? – A Literature Review of Empirical Studies on Gamification". In: *2014 47th Hawaii International Conference on System Sciences*, pp. 3025–3034.

Hamari, Juho and Jonna Koivisto (2015). "Why do people use gamification services?" In: *International Journal of Information Management* 35.4, pp. 419–431.

Herzig, Philipp, Kay Jugel, Christof Momm, Michael Ameling, and Alexander Schill (2013). "GaML-A modeling language for gamification". In: *2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing*. IEEE, pp. 494–499.

Hunicke, Robin, Marc Leblanc, and Robert Zubek (2004). "MDA: A Formal Approach to Game Design and Game Research". In: *AAAI Workshop - Technical Report* 1.

Huotari, Kai and Juho Hamari (2012). "Defining gamification: a service marketing perspective". In: *Proceeding of the 16th international academic MindTrek conference*. ACM, pp. 17–22.

Huynh, Duy, Long Zuo, and Hiroyuki Iida (2018). "An Assessment of Game Elements in Language-Learning Platform Duolingo". In: *2018 4th International Conference on Computer and Information Sciences (ICCOINS)*. IEEE, pp. 1–4.

Kapp, Karl M (2012). *The gamification of learning and instruction*. Wiley San Francisco.

Koen, Peter A, Heidi M J Bertels, and Elko Kleinschmidt (2014). "Results From a Three-Year Study". en. In: p. 10.

Kumar, Janaki (2013). "Gamification at work: Designing engaging business software". In: *International conference of design, user experience, and usability*. Springer, pp. 528–537.

Matallaoui, A., P. Herzig, and R. Zarnekow (2015). "Model-Driven Serious Game Development Integration of the Gamification Modeling Language GaML with Unity". In: *2015 48th Hawaii International Conference on System Sciences*, pp. 643–651.

McGonigal, Jane (2011). *Reality Is Broken: Why Games Make Us Better and How They Can Change the World*. Penguin Group , The.

Mora, Alberto, Daniel Riera, Carina González, and Joan Arnedo-Moreno (2017). "Gamification: a systematic review of design frameworks". In: *Journal of Computing in Higher Education* 29.3, pp. 516–548.

Morschheuser, Benedikt, Lobna Hassan, Karl Werder, and Juho Hamari (2018). "How to design gamification? A method for engineering gamified software". In: *Information and Software Technology* 95.

MPS (2019). *MPS: The Domain-Specific Language Creator by JetBrains*. url: `https://www.jetbrains.com/mps/` (visited on 05/05/2019).

Osterwalder, Alexander and Yves Pigneur (2010). "You're holding a handbook for visionaries, game changers, and challengers striving to defy outmoded business models and design tomorrow's enterprises. It's a book for the..." en. In: p. 51.

Paharia, Rajat (2013). *Loyalty 3.0: How to revolutionize customer and employee engagement with big data and gamification*. McGraw Hill Professional.

Porter's Value Chain - Strategy Skills Training from MindTools.com (2019). url: `https://www.mindtools.com/pages/article/newSTR_66.htm` (visited on 02/24/2019).

Robson, Karen, Kirk Plangger, Jan H. Kietzmann, Ian McCarthy, and Leyland Pitt (2016). "Game on: Engaging customers and employees through gamification". en. In: *Business Horizons* 59.1, pp. 29–36. issn: 00076813. (Visited on 02/17/2019).

Schmidt, Douglas C (2006). "Model-driven engineering". In: *COMPUTER-IEEE COMPUTER SOCIETY*- 39.2, p. 25.

Swacha, Jakub (2018). "Representation of Events and Rules in Gamification Systems". In: *Procedia Computer Science*. Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 22nd International Conference, KES-2018, Belgrade, Serbia 126, pp. 2040–2049. issn: 1877-0509. (Visited on 10/29/2018).

Walters, David and Geoff Lancaster (2000). "Implementing value strategy through the value chain". In: *Management Decision* 38, pp. 160–178.

Werbach, Kevin and Dan Hunter (2012). *For the Win: How Game Thinking can Revolutionize your Business*.

Woodall, Tony (2003). "Conceptualising 'Value for the Customer': An Attributional, Structural and Dispositional Analysis". In: *Academy of Marketing Science Review* 12.

Zichermann, Gabe and Christopher Cunningham (2011). *Gamification by design: Implementing game mechanics in web and mobile apps*. " O'Reilly Media, Inc."

Appendix A

Illustration of the Analytic Hierarchy Process

Over the course of the development of this project, the Analytic Hierarchy Process (AHP) was not utilized, but a possible use for this method could be focused on the methodology to be used on the project's implementation phase.

A.1 Step 1 – Hierarchical Decision Tree

Having both the previous statement and the objectives of the project in mind, four different elements for this decision tree were considered:

- *Metamodel* Design Correctness.
- DSL Completeness.
- DSL User-friendliness.
- Apply complex gamification strategies.

Three different alternatives were provided for this illustration (M1, M2, M3), each giving a different amount of focus between each of the four previously mentioned elements.

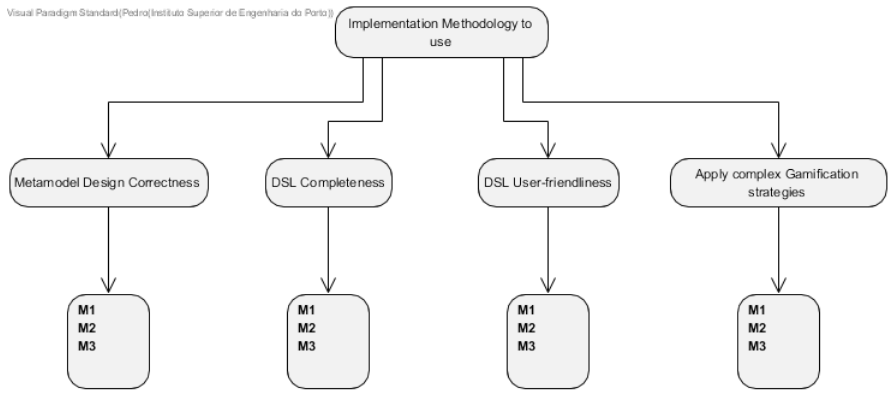


Figure A.1: Hierarchy Decision Tree

A.2 Step 2 – Comparison between Hierarchal Elements

For this next step of the AHP, the elements: *Metamodel* Design Correctness (E1), DSL Completeness (E2), DSL User-friendliness (E3), and Apply complex gamification strategies (E4), will be subject to a comparison between each other.

Table A.1: Table of comparison between hierarchal elements

-	**E1**	**E2**	**E3**	**E4**
E1	1	5	3	1/5
E2	1/5	1	1/2	1/9
E3	1/3	2	1	1/2
E4	5	9	2	1

A.3 Step 3 – Element's Relative Priority

Using the table previously created, another table with the relative priority of each element will be created by calculating the sum of each column, to later divide the values of each column by its sum. The purpose of these calculations is to equalize the values of all the elements to the same level.

Table A.2: Table of comparison between hierarchal elements with sum per column

-	**E1**	**E2**	**E3**	**E4**
E1	1	5	3	1/5
E2	1/5	1	1/2	1/9
E3	1/3	2	1	1/2
E4	5	9	2	1
SUM	98/15	17	13/2	163/90

Having the previous steps completed, the average value of each line will now be calculated, to successfully obtain the weight, or importance, of each element.

Table A.3: Table of element's relative priority

-	**E1**	**E2**	**E3**	**E4**	**R. Priority**
E1	0,153	0,294	0,461	0,110	0,254
E2	0,031	0,059	0,077	0,061	0,057
E3	0,051	0,118	0,154	0,276	0,150
E4	0,765	0,529	0,308	0,552	0,539

A.4 Step 4 – Evaluate the consistency of the relative priorities

$$\lambda_{\max} \cdot \mathbf{x} = \mathbf{A} \cdot \mathbf{x} = \begin{bmatrix} 0,153 & 0,294 & 0,461 & 0,110 \\ 0,031 & 0,059 & 0,077 & 0,061 \\ 0,051 & 0,118 & 0,154 & 0,276 \\ 0,765 & 0,529 & 0,308 & 0,552 \end{bmatrix} \cdot \begin{bmatrix} 0,254 \\ 0,057 \\ 0,150 \\ 0,539 \end{bmatrix} = \begin{bmatrix} 1,097 \\ 0,243 \\ 0,618 \\ 2,622 \end{bmatrix}$$

$$\lambda_{max} = ((1,097/0,254) + (0,243/0,057) + (0,618/0,15) + (2,622/0,539))/4 = 4,390$$

Consistency Index:

$$IC = (4,390 - 4)/(4 - 1) = 0,13 \tag{A.1}$$

Consistency Ratio:

$$RC = IC/IR = 0,13/0,9 = 0,144 = 14,4\% \tag{A.2}$$

A.5 Step 5 – Evaluate each alternative by each element

Table A.4: Table of comparison between alternatives about the *Metamodel* Design Correctness element

E1	**M1**	**M2**	**M3**
M1	1	2	1/3
M2	1/2	1	1/6
M3	3	6	1

Table A.5: Table of comparison between alternatives about the DSL Completeness element

E2	**M1**	**M2**	**M3**
M1	1	3	1/2
M2	1/3	1	1/4
M3	2	4	1

Table A.6: Table of comparison between alternatives about the DSL User-friendliness element

E3	**M1**	**M2**	**M3**
M1	1	1/2	3
M2	2	1	6
M3	1/3	1/6	1

Applying the same calculations as before to obtain the relative priority of each element, the following vectors of priority were obtained:

$$\mathbf{V}_{\text{E1}} = \begin{bmatrix} 0,222 \\ 0,111 \\ 0,667 \end{bmatrix}$$

Table A.7: Table of comparison between alternatives about the Apply complex gamification strategies element

E4	**M1**	**M2**	**M3**
M1	1	1/3	2
M2	3	1	4
M3	1/2	1/4	1

$$\mathbf{V}_{\text{E2}} = \begin{bmatrix} 0,320 \\ 0,123 \\ 0,557 \end{bmatrix}$$

$$\mathbf{V}_{\text{E3}} = \begin{bmatrix} 0,3 \\ 0,6 \\ 0,1 \end{bmatrix}$$

$$\mathbf{V}_{\text{E4}} = \begin{bmatrix} 0,239 \\ 0,623 \\ 0,137 \end{bmatrix}$$

A.6 Step 6 – Obtain composed priority for the alternatives

"A" is the matrix with the priority values of the alternatives, and "x" contains the priority values of each element of the hierarchy. R_{alts} is the vector with the results of the overall value of each alternative.

$$R_{\text{alts}} = \mathbf{A} \cdot \mathbf{x} = \begin{bmatrix} 0,222 & 0,320 & 0,3 & 0,239 \\ 0,111 & 0,123 & 0,6 & 0,623 \\ 0,667 & 0,557 & 0,1 & 0,137 \end{bmatrix} \cdot \begin{bmatrix} 0,254 \\ 0,057 \\ 0,150 \\ 0,539 \end{bmatrix} = \begin{bmatrix} 0.248 \\ 0.461 \\ 0.290 \end{bmatrix}$$

A.7 Step 7 – Choose the best alternative

With the given results, the alternative M2 seems to be the best between the rest of the alternatives, since it focuses on the elements DSL User-friendliness (E3), and Apply complex gamification strategies (E4), both deemed important for the project.

Appendix B

Illustration of a Value chain

Having both Porter's generic Value Chain (*Porter's Value Chain - Strategy Skills Training from MindTools.com* 2019) and Porter's steps to use the value chain in mind, sub-activities were defined for each of the primary and support activities represented on the Value Chain. It is important to note that this is an illustration of the Value Chain's use, by attempting to create an organization around the service presented in this project.

Primary Activities:

- Inbound logistics:
  - From the Company's servers.
- Operations:
  - Service Innovation.
  - Quality Assurance.
- Outbound Logistics:
  - Company's web page for users.
- Marketing & Sales:
  - Advertising.
  - Pricing.

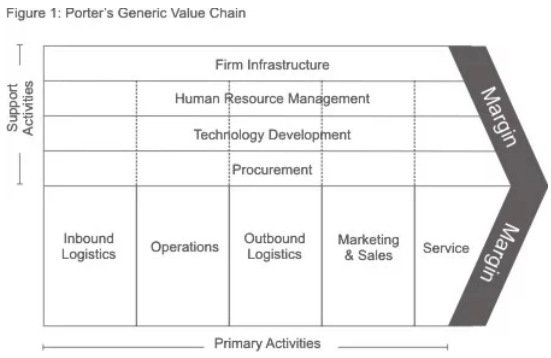


Figure B.1: Generic Value Chain by (*Porter's Value Chain - Strategy Skills Training from MindTools.com* 2019)

- Service:
  - Customer Service.

Support Activities:

- Firm Infrastructure:
  - Accounting management.
  - Legal management.
  - Administrative management.
  - General management.
- Human Resource Management:
  - Ethical.
  - Compensation for good performances.
- Technology Development:
  - Minimizing unnecessary costs.
  - Always on the lookout for potential innovation opportunities.
- Procurement:
  - Relationships with key partners.

As for possible investments to increase the service's value, investing in Operations would increase the overall service value. Investing in the Customer service department could be useful since it may have a link with decreasing possible frustration using the service, therefore increasing value.

Appendix C

Evaluation Method: Questionnaire

C.1 Questionnaire Details

To assess one of the problems set in the evaluation phase of the project (see Chapter 6), the questionnaire present in this appendix was created. Through it, participants are asked to test and to provide their personal opinion about the Domain-Specific Language developed. It is important to note that the language used regarding technical terms within the domain of Model-Driven Engineering was simplified, to avoid confusing participants who lack experience in the said domain.

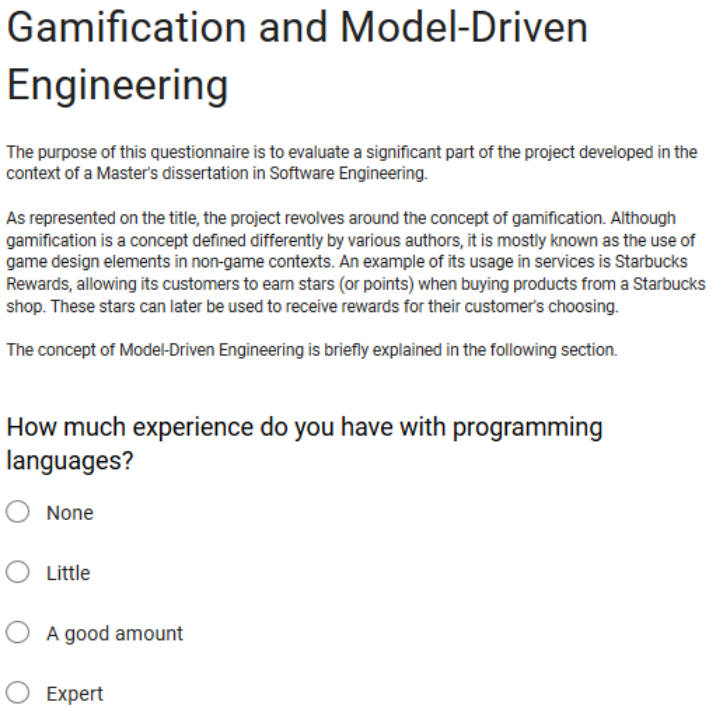
The questionnaire was created using Google Forms, and it contains 3 sections:

1. Introduces the project and the concept of gamification to the participants and inquires about their knowledge regarding programming languages.
2. Presents the the Model-Driven Engineering approach and the purpose of Domain-Specific Languages. Provides a figure consisting of the guidelines developed for the entity "Item" and its subtypes, followed by a question to evaluate whether or not the combination of the guideline and the language's simplicity are enough to allow the participants to write correctly a small portion of a design strategy.
3. Asks of the participants to give their personal opinion regarding the language's simplicity and the overall usefulness of the guidelines.

Figure C.1 represents the first section of the questionnaire. The question present in this section functions as a method to split the population between participants who have little to no knowledge about programming and programming languages, and participants who are, at least, somewhat proficient with programming.

Figure C.2 and C.3 represent the questionnaire's second section. At this stage, participants can learn more about Domain-Specific Languages before attempting to understand the guidelines used for the evaluation. The entity "Item" was chosen due to it being one of the simpler available options to design, hence allowing the focus to be on both the utility of the guidelines as well as the readability and writability of the language.

Figure C.4 presents the final section created in the questionnaire. Participants are informed of a possible answer to the previous question before being inquired about whether or not the language feels intuitive and simple to use. Since the participants are susceptible to make mistakes in the previous question, in this section they are allowed to read one of the possible answers and to realize if the language feels appropriate or if there are improvements to be made. The following question is directed to the usefulness of the guideline presented, giving



Gamification and Model-Driven Engineering

The purpose of this questionnaire is to evaluate a significant part of the project developed in the context of a Master's dissertation in Software Engineering.

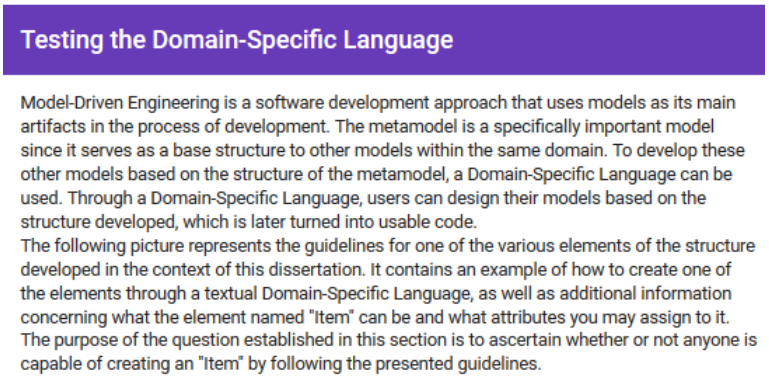
As represented on the title, the project revolves around the concept of gamification. Although gamification is a concept defined differently by various authors, it is mostly known as the use of game design elements in non-game contexts. An example of its usage in services is Starbucks Rewards, allowing its customers to earn stars (or points) when buying products from a Starbucks shop. These stars can later be used to receive rewards for their customer's choosing.

The concept of Model-Driven Engineering is briefly explained in the following section.

How much experience do you have with programming languages?

- ○ None
- ○ Little
- ○ A good amount
- ○ Expert

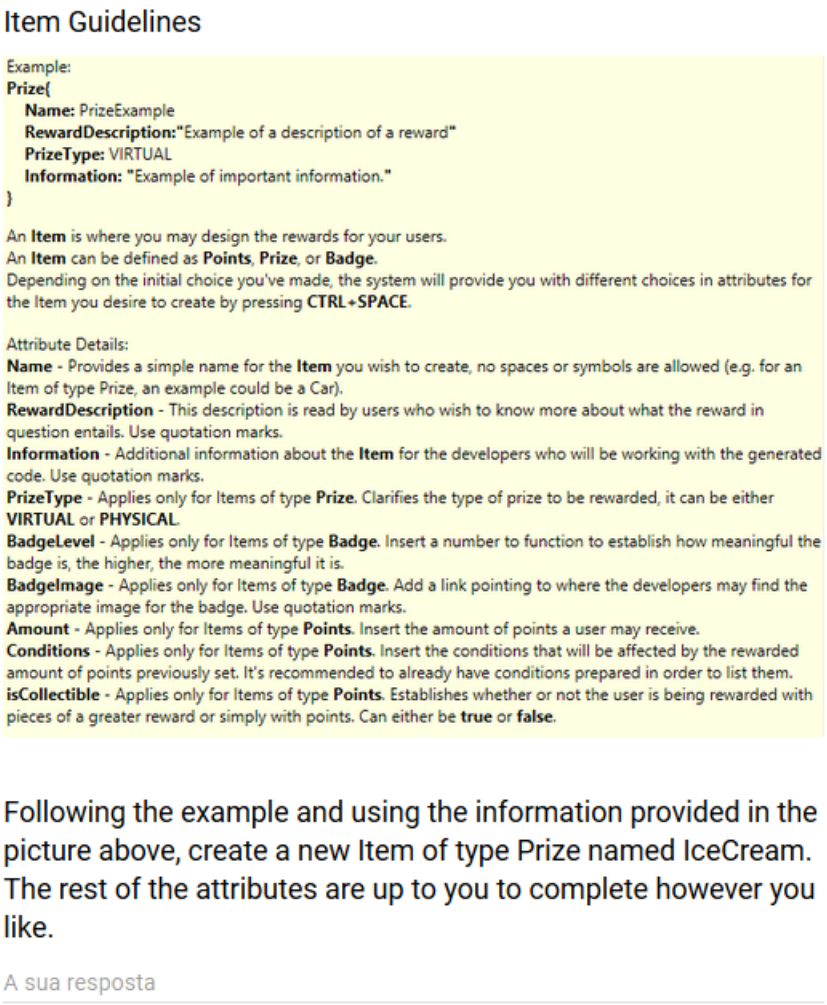
Figure C.1: First Questionnaire Section



Testing the Domain-Specific Language

Model-Driven Engineering is a software development approach that uses models as its main artifacts in the process of development. The metamodel is a specifically important model since it serves as a base structure to other models within the same domain. To develop these other models based on the structure of the metamodel, a Domain-Specific Language can be used. Through a Domain-Specific Language, users can design their models based on the structure developed, which is later turned into usable code.
The following picture represents the guidelines for one of the various elements of the structure developed in the context of this dissertation. It contains an example of how to create one of the elements through a textual Domain-Specific Language, as well as additional information concerning what the element named "Item" can be and what attributes you may assign to it.
The purpose of the question established in this section is to ascertain whether or not anyone is capable of creating an "Item" by following the presented guidelines.

Figure C.2: Part 1 - Second Questionnaire Section



Item Guidelines

Example:
Prize{
Name: PrizeExample
RewardDescription:"Example of a description of a reward"
PrizeType: VIRTUAL
Information: "Example of important information."
}

An **Item** is where you may design the rewards for your users.
An **Item** can be defined as **Points**, **Prize**, or **Badge**.
Depending on the initial choice you've made, the system will provide you with different choices in attributes for the Item you desire to create by pressing **CTRL+SPACE**.

Attribute Details:
Name - Provides a simple name for the **Item** you wish to create, no spaces or symbols are allowed (e.g. for an Item of type Prize, an example could be a Car).
RewardDescription - This description is read by users who wish to know more about what the reward in question entails. Use quotation marks.
Information - Additional information about the **Item** for the developers who will be working with the generated code. Use quotation marks.
PrizeType - Applies only for Items of type **Prize**. Clarifies the type of prize to be rewarded, it can be either **VIRTUAL** or **PHYSICAL**.
BadgeLevel - Applies only for Items of type **Badge**. Insert a number to function to establish how meaningful the badge is, the higher, the more meaningful it is.
BadgeImage - Applies only for Items of type **Badge**. Add a link pointing to where the developers may find the appropriate image for the badge. Use quotation marks.
Amount - Applies only for Items of type **Points**. Insert the amount of points a user may receive.
Conditions - Applies only for Items of type **Points**. Insert the conditions that will be affected by the rewarded amount of points previously set. It's recommended to already have conditions prepared in order to list them.
isCollectible - Applies only for Items of type **Points**. Establishes whether or not the user is being rewarded with pieces of a greater reward or simply with points. Can either be **true** or **false**.

Following the example and using the information provided in the picture above, create a new Item of type Prize named IceCream. The rest of the attributes are up to you to complete however you like.

A sua resposta

Figure C.3: Part 2 - Second Questionnaire Section

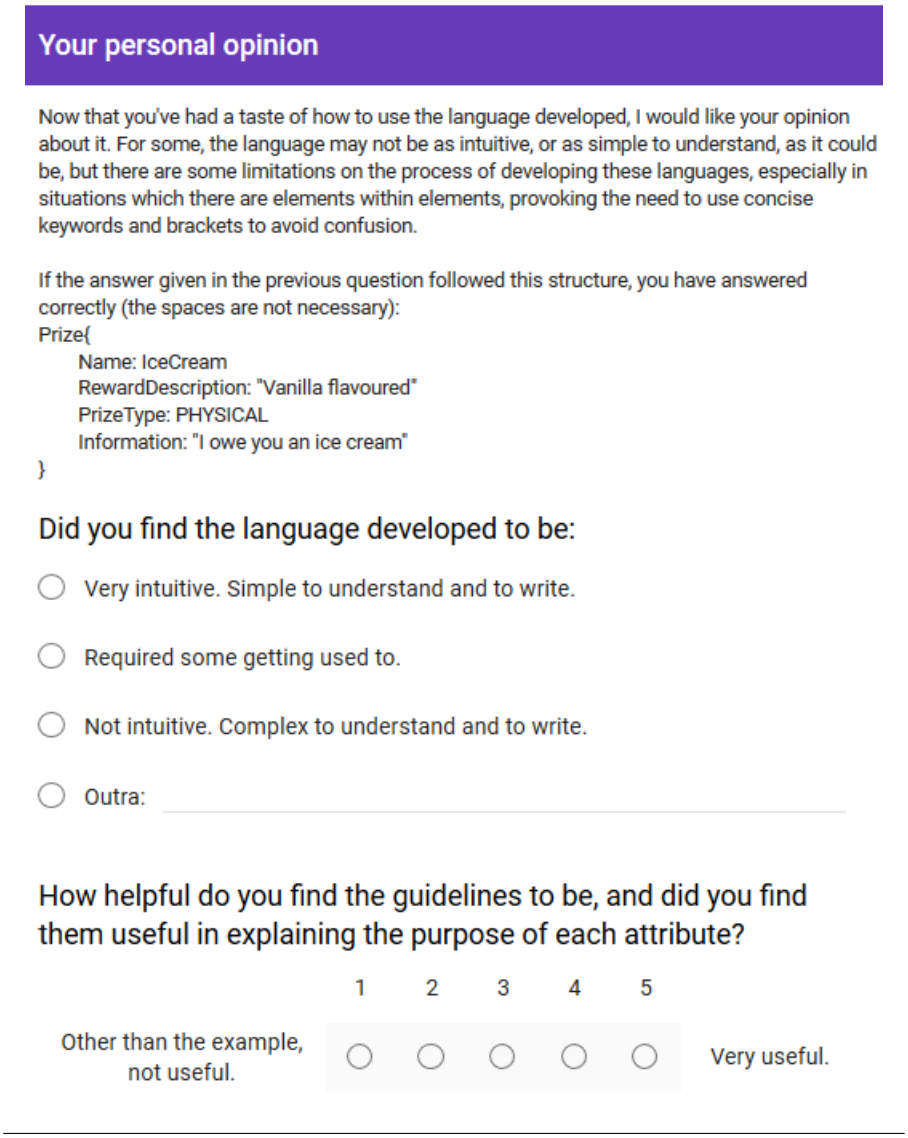


Figure C.4: Third Questionnaire Section

the participants the option to evaluate how useful they felt the guideline was when answering the question in section 2.

C.2 Question 2: Results

In this section, the answers obtained in question 2 of the questionnaire are listed alongside their respective scores, using the evaluation method explained in Chapter 6.

```
NE - Non-experienced participant.
E - Experienced participant.

Answer List
A1(NE):
```

```
    Prize(IceCream, "The reward is gained weight", PHYSICAL, "The item should be tasty and fresh")

Score = 0.5 (Structure) + 4 (Variables) = 4.5

A2(NE):
    Prize{
        Name: IceCream
        RewardDescription: "Perfect for those hot Autumn days"
        PrizeType: PHYSICAL
        Information: "A prize"
    }

Score = 5 (Structure) + 4 (Variables) = 9

A3(NE):
    Prize{
        Name: IceCream
        RewardDescription:"You earned a free Ice Cream"
        PrizeType: PHYSICAL
        Information:"The client may choose one of the various Ice Creams available for sale without having to pay. There is no restrictions for the client's choice as long as it is an ice cream"
    }

Score = 5 (Structure) + 4 (Variables) = 9

A4(E):
    Prize{
        Name: IceCream
        RewardDescription:"Congratulations, you've won an edible ice cream!"
        PrizeType:PHYSICAL
        Information:"Be sure to add allergen alert when notifying users that won the prize"
    }

Score = 5 (Structure) + 4 (Variables) = 9

A5(E):
    Prize{
        Name: IceCream
        RewardDescription: IceCream of 2 flavors
        PrizeType: PHYSICAL
        Information: Don't let it melt!
    }

Score = 5 (Structure) + 3 (Variables) = 8

A6(NE):
    Name: Ice Cream
    RewardDescription: "win an ice cream"
    PrizeType: physical
    Information: "it's chocolate ice cream"

Score = 4 (Structure) + 3.5 (Variables) = 7.5

A7(E):
    Prize{
```

```
60          Name: IceCream
61          RewardDescription:"Offer a ball of any flavoured ice-cream"
62          PrizeType: PHYSICAL
63          Information: "By going 5 times to the store, offer this prize"
64      }
65
66  Score = 5 (Structure) + 4 (Variables) = 9
67
68  A8(E):
69      Prize {
70          Name: IceCream
71          RewardDescription: "frozen delight"
72          PrizeType: PHYSICAL
73      }
74
75  Score = 4 (Structure) + 3 (Variables) = 7
76
77  A9(E):
78      Prize {
79          Name: IceCream
80          RewardDescription: "1 month sub on World of Warcraft"
81          PrizeType: VIRTUAL
82          Information: "You are not prepared"
83      }
84
85  Score = 5 (Structure) + 4 (Variables) = 9
86
87  A10(E):
88      Prize {
89          Name: IceCream
90          RewardDescription: "Frozen deliciousness"
91          PrizeType: PHYSICAL
92          Information: "Careful you don't make a mess"
93      }
94
95  Score = 5 (Structure) + 4 (Variables) = 9
96
97  A11(E):
98      Prize{
99          Name: IceCream
100         RewardDescription: "YOU GET AN ICE CREAM!"
101         PrizeType: PHYSICAL
102         Information: "Need to build an ice cream machine"
103     }
104
105 Score = 5 (Structure) + 4 (Variables) = 9
106
107 A12(NE):
108     Badge{
109         Name: IceCream
110         RewardDescription:"MORE ICECREAM"
111         BadgeLevel:10
112         BadgeImage:www.no.com
113         Information:"ICECREEEEEEEEEEEEEEEEEEEEEEAM"
114     }
115
116 Score = 5 (Structure) + 3.5 (Variables) = 8.5
117
```

```
Justification: Although this participant did not create an Item entity
    of type Prize, as was requested, they followed the structure to
    create a Badge correctly. Since the purpose of this question was to
    prove that the participants could write their desired entities
    through the DSL provided, this was not considered an incorrect answer
    .

A13(NE):
    Prize{
        Name: IceCream
        RewardDescription: "Chocolate ice-cream for winners"
        PrizeType: VIRTUAL
        Information: "waffle cone with chocolate swirl on top, unlocked
    after winning"
    }

Score = 5 (Structure) + 4 (Variables) = 9

A14(NE):
    Prize{
        Name: IceCream
        RewardDescription:"IceCream shirt for your character"
        PrizeType: VIRTUAL
        Information: "Black and white shirt with horizontal stripes."
    }

Score = 5 (Structure) + 4 (Variables) = 9

A15(NE):
    Name: IceCream
    RewardDescription: "you win a prize after doing your homework"
    PrizeType: VIRTUAL
    Information: "do homework after school"

Score = 4 (Structure) + 4 (Variables) = 8

A16(NE):
    Prize{
        Name: IceCream
        RewardDescription: "the reward is an icecream"
        PrizeType: PHYSICAL
        Information: "it's chocolate icecream"
    }

Score = 5 (Structure) + 4 (Variables) = 9

A16(NE):
    Prize{
        Name: IceCream
        RewardDescription: "the reward is an icecream"
        PrizeType: PHYSICAL
        Information: "it's chocolate icecream"
    }

Score = 5 (Structure) + 4 (Variables) = 9

A17(E):
    Prize {
        Name: IceCream
```

```
171         RewardDescription: "The Ice Cream badge, awarded after beating
    the Snow Golem boss"
172         Information: "After beating the Snow Golem boss, the city mayor
    awards the Ice Cream badge to the player in the glacial ceremony"
173         PrizeType: VIRTUAL
174         BadgeLevel: 10
175         BadgeImage: "https://i.etsystatic.com/15185102/r/il/93c093
    /1614485587/il_570xN.1614485587_2002.jpg"
176     }
177
178 Score = 5 (Structure) + 4 (Variables) - 2 (Penalty)= 7
179
180 Justification: This participant received a 2 point penalty by adding
    Badge-related attributes to a Prize type.
```

Listing C.1: List of answers for question 2

Appendix D

Evaluation Method: Running Simulator

To prove the hypothesis set on the second problem established in the evaluation phase, a Unity application was developed using the generated code from a previously created gamification strategy. Listing D.1 represents the strategy designed for the application.

```
Gamify{
    Name: RunningSimulator
    System{
        Attributes:NEW
        UserType:CUSTOMER
        UserAttributes:ACHIEVERS
    }
    GameDynamics{
        GameDynamic{
            Name:ValueReview
            DynamicType:REVIEW
            GameMechanics{
                GameMechanic{
                    Name:NewThresholds
                    MechanicType:PROGRESSBAR
                    Events{
                        Event{
                            Name:Running
                            EventType:MSUCCESS
                            PointGain:10
                            TriggerConditions(BeginnerRunner.PointSoaker, JourneymanRunner.FurtherBeyond)
                            Information:"Insert a progression event to further reward users"
                        }
                    }
                    Information:"Link game mechanic with service code"
                }
            }
            Achievements{
                Achievement{
                    Name: JourneymanRunner
                    Hidden:true
                    RequiredAchievements(BeginnerRunner, Dasher)
                    Rewards{
                        FixedReward{
                            Item:AddPoints
```

```
                        }
                    }
                    Conditions{
                        Condition{
                            Name:FurtherBeyond
                            AmountRequired:500
                            ConditionType:EQUAL
                            Information:"Achieved 500 meters of distance
"
                        }
                    }
                    Information:"Implement Achievement"
                },
                Achievement{
                    Name: BeginnerRunner
                    Hidden:true
                    Rewards{
                        FixedReward{
                            Item:AddPoints
                        }
                    }
                    Conditions{
                        Condition{
                            Name:PointSoaker
                            AmountRequired:100
                            ConditionType:EQUAL
                            Information:"Gain points from each second
running and by completing achievements"
                        }
                    }
                    Information:"Implement Achievement"
                },
                Achievement{
                    Name: Dasher
                    Hidden:true
                    Rewards{
                        FixedReward{
                            Item:AddPoints
                        }
                    }
                    Conditions{
                        Condition{
                            Name:QuickDash
                            AmountRequired:10
                            ConditionType:EQUAL
                            Information:"User reaches top speed."
                        }
                    }

                }
            }
            Information:"Implement dynamic code"
        }
    }
    Items{
        Points{
            Name:AddPoints
            RewardDescription:"Gain 100 Achievement Points"
            Amount: 100
```

```
93              Information: "Information"
94          }
95      }
96      Information:"Implement connectors between the service and the
        generated strategy elements"
97  }
```

Listing D.1: Gamification Strategy for RunningSimulator

The application developed in Unity consists of simulating mobile running apps. Whereas a running app would detect how fast it is traveling, in the simulator users can set the speed of which they would be traveling. Figure D.1 represents the interface set for the application.

Figure D.1: User Interface of Running Simulator

Once users begin the simulation and set their desired speed, the Event entity entrusted with recording activity begins notifying its respective list of conditions, resulting in the completion of an achievement. These achievements may vary between the maximum speed ran, or the distance traveled.

Listing D.2 represents the three main functions developed to handle the button in the application as well as notifying the created Event entity once the user begins to run.

```
1  void Start ()
2  {
3    distance = 0;
4    second = 0;
5    main = new Main ();
6    running = main.getRunning ();
7    beginnerRunner = main.getBeginnerRunner ();
8  }
9
```

```
10 // Update is called once per frame
11 void Update ()
12 {
13   if (GameObject.Find ("Start").GetComponentInChildren<Text> ().text ==
       "Stop") {
14     float value = GameObject.Find ("Slider").GetComponent<Slider> ().
       value * (float)(System.Math.Round(Time.deltaTime, 3));
15     distance = distance+(float)(System.Math.Round(value, 2));
16     distance = (float)System.Math.Round (distance, 2);
17     GameObject.Find ("DistanceLabel").GetComponent<Text> ().text =
       distance + " meters";
18     second = second + Time.deltaTime;
19     if (second >= 1) {
20       second = 0;
21       running.onTrigger ();
22       quickDash.updateAmount ((int) GameObject.Find ("Slider").
       GetComponent<Slider> ().value);
23       foreach (var item in achievements) {
24           item.checkCompleted ();
25         if (item.getCompleted() && !item.getAchieved()) {
26           item.setAchieved ();
27           String rewardText = "Completed "+ item.ToString() +": ";
28           foreach (var reward in item.getRewards()) {
29             rewardText = rewardText + reward.getItem().getDescription()
       + "\n";
30           }
31
32           GameObject.Find ("RewardText").GetComponentInChildren<Text> ()
       .text = rewardText;
33           GameObject.Find ("RewardPlane").GetComponentInChildren<
       MeshRenderer> ().enabled = true;
34           GameObject.Find ("RewardText").GetComponentInChildren<
       CanvasGroup> ().alpha = 1;
35         }
36       }
37     }
38   }
39   if(GameObject.Find ("RewardText").GetComponentInChildren<CanvasGroup>
       ().alpha == 1)
40       achieveSecond = achieveSecond + Time.deltaTime;
41   if (achieveSecond >= 5) {
42     GameObject.Find ("RewardPlane").GetComponentInChildren<MeshRenderer>
       ().enabled = false;
43     GameObject.Find ("RewardText").GetComponentInChildren<CanvasGroup>
       ().alpha = 0;
44     achieveSecond = 0;
45   }
46 }
47
48 public void ButtonAction ()
49 {
50   if (GameObject.Find ("Start").GetComponentInChildren<Text> ().text ==
       "Start") {
51     GameObject.Find ("Start").GetComponentInChildren<Text> ().text = "
       Stop";
52   } else {
53     GameObject.Find ("Start").GetComponentInChildren<Text> ().text = "
       Start";
54   }
```

```
55 }
```

Listing D.2: Integration with Running Simulator

Using the code presented in Listing D.2 and the code generated, users are now notified each time they complete an Achievement, as demonstrated in Figure D.2.

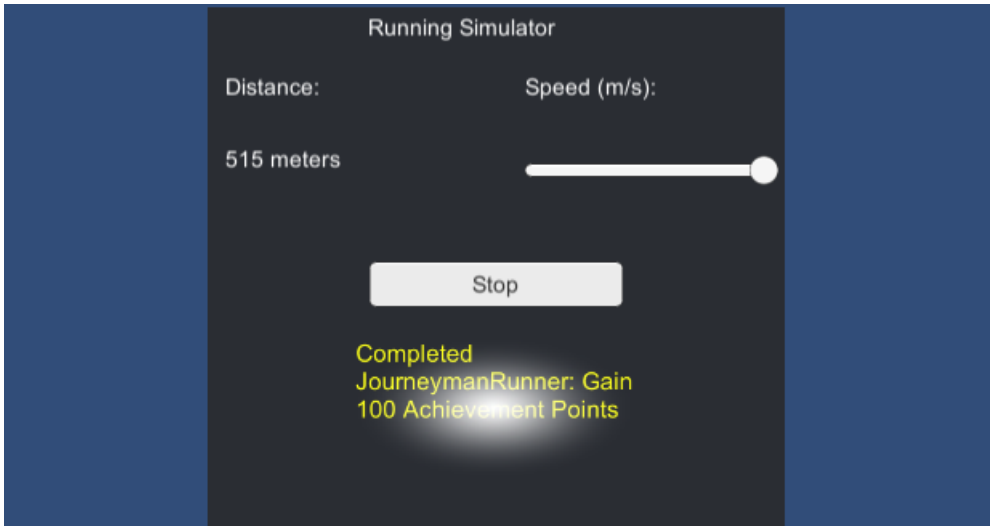


Figure D.2: Notification for JourneymanRunner Achievement