INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA

Facilitação de Gamificação em Contextos de Seguros

CARLOS DIOGO DA SILVA TAVARES

Outubro de 2018

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Enabling Gamification in Insurance Contexts

Carlos Diogo da Silva Tavares

Dissertation for Master's Degree in Informatics

Specialization in Software Engineering

Advisor: Isabel de Fátima Silva Azevedo

Supervisors: Nuno Ferreira, Manuel Ribeiro

Porto, October 2018

Acknowledgements

I would like to thank my Advisor, Isabel Azevedo, for continued support and guidance during the development of this project and document.

My Supervisor Nuno Ferreira for heavy support during the initial stage of this project as well as helping my acclimation to the company environment where this project was developed.

My Supervisor Manuel Ribeiro and Catarina Tomé for aiding me in the design of the system by evaluating my proposals and giving many suggestions for improvement.

Special thanks should also be extended to my colleagues Pedro Felgueiras for greatly aiding me in the development of the project by suggesting useful frameworks and methodologies as well as helping me with many code related issues.

Finally, I would like to thank my colleagues at i2S for providing a friendly and comfortable work environment during my almost 8 months of presence.

Resumo

A Gamificação é um método amplamente utilizado para aumentar o investimento de utilizadores, que tem evoluído constantemente na última década. As suas áreas de aplicação são diversas e, no caso específico dos produtos de seguro, a Gamificação começou a ser adotada para resolver vários problemas, sendo os mais relevantes a necessidade de aumentar as interações entre a empresa e o segurado e a redução da quantidade de dados não relatados à Companhia de Seguros, que causam cálculos de prémio menos exatos.

A aplicação da Gamificação na área de seguros em geral foi analisada para fornecer uma solução de software que permita a sua implementação numa ampla variedade de áreas e produtos, para isso, analisaram-se Frameworks para Gamificação, assim como as possíveis áreas de aplicação no setor de seguros. Um design para um conjunto de aplicações é então apresentado, seguido por uma descrição detalhada da sua implementação.

O trabalho foi desenvolvido na i2S de meados de janeiro a setembro de 2018 e tentou fornecer uma solução, com uma abordagem ampla, que fosse adaptável a vários tipos de apólices, utilizando seguro automóvel de tipo telemático, para fornecer um exemplo de uso.

Este projeto foi utilizado para avaliar a viabilidade do desenvolvimento de um módulo independente para permitir a Gamificação de produtos de seguro, para comercialização pela i2S. Este módulo diferenciaria a empresa no mercado, já que as seguradoras estão a considerar começar a utilizar a Gamificação a médio prazo, para enriquecer os seus produtos.

Palavras-chave: Gamificação; Mercado Segurador;

Abstract

Gamification is a widely utilised method to increase of user engagement that has been steadily evolving over the last decade. It's areas of application are diverse and in the specific case of insurance products Gamification has started to be adopted to solve various problems, the most prominent of which is the need to increase interactions between company and policy holder, reducing the amount of data unreported to the insurance company causing less exact premium calculations.

The application of Gamification and the insurance area in general were analysed in order to provide a software solution to enable its implementation in a wide variety of areas and products. Frameworks for Gamification were analysed as are the possible areas of application in the insurance sector. A design for a set of applications is then presented followed by a detailed description of their implementation. The work was developed in i2S from mid-January to September 2018 and it attempted to provide a solution with a broad approach, being adaptable to many types of policy, utilizing auto insurance of a telematic kind to provide an example of usage.

This project was utilized to assess the viability of developing an independent module to enable Gamification of insurance products to be commercialized by i2S. this would differentiate the company in the market as insurance companies are looking to utilize Gamification in the medium term to enhance their products.

Keywords: Gamification; Insurance Market;

Table of Contents

Table of Figures

Table of Tables

Table of Code Excerpts

Acronyms and Initialisms

AHP	Analytic Hierarchy Process
API	Application Programming Interface
DB	Database
DSL	Domain Specific Language
DTO	Data Transfer Object
GDPR	General Data Protection Regulation
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
OS	Operating System
QEF	Quality Evaluation Framework
REST	Representational State Transfer
SDA	Safe Driver Association
JSON	JavaScript Object notation
TOPSIS	Technique for Order of Preference by Similarity to Ideal Solution
TRL	Technology Readiness Level
UI	User Interface

1 Introduction

Chapter Contents

1.1 Context

Insurance companies provide widely used services, some of them required by law, to the general market, however, the interaction with their customers is extremely limited, “an astonishing 71% of customers never or rarely (only once per year) interact with their principal distributor” (Genesys, 2008).

This lack of contact causes issues for the insurance company as they lack information about the customer that has the potential to be relevant to the insurance coverage provided, as risk factors may be lower or higher than assessed and, inevitably, change as time passes. On the customer’s side another issue presents itself; in many cases, information about the insurance coverage conditions and factors of risk is lacking, making the customer not only more likely to be in a situation that creates the need for a claim but also to claim something that is not actually covered by the plan, something that could have been resolved if the policy details were consulted occasionally.

Gamification is a possible method of addressing this issue as its effectiveness in increasing user engagement has been noted over multiple implementations in a multitude of environments. The problem presented here is how Gamification can be applied to this area.

This project was developed at i2S Insurance Knowledge, founded in 1984, possessing more than 220 collaborators which has its headquarters in Porto. i2S supplies and supports, internally developed, software solutions to the various members of the insurance industry; insurance companies, pension fund managing entities insurance agents and insurance brokers. In i2S’s customer portfolio possesses insurance companies that total approximately 60% of the premiums processed in Portugal. i2S also possesses delegations in Lisbon, Angola and Madrid help to support the customer base that spreads by Portugal, Spain, France, Poland, Mozambique and Cape Verde.

Interest in utilizing gamified approaches has also been seen of late with various companies, mostly on the international level, utilizing gamification on their products, mostly in the auto-insurance area.

1.2 Problem

Derived from the context provided previously the question arises “How can the interaction between insurance company and client be increased providing benefits for both parties?” One method of motivating factors that lack a significant drive behind them is the utilization of game elements to enhance it, Gamification.

The use of Gamification methods in this area is, however, negligible, or at least extremely obfuscated behind the insurance company’s walls, and so it is necessary to analyse what can be gamified to solve the problem presented. Insurance companies can have different approaches to this and must be allowed to create their own Gamified solutions, the proposed solution is the creation of a module to be included into the i2S software which would allow for a guided approach to Gamification, leading to the problem presented:

"Is it viable to develop a module to facilitate the implementation of Gamified approaches in for insurance companies?"

1.3 Objectives

The work described in this report has the following major objectives:

1. Identify and analyse possible applications of gamification in insurance software, with the selection of one or more according to a set of criteria to be identified.
2. Analyse and select one or more gamification frameworks to be used in the gamification design of the applications.
3. Design a gamification approach in the context on Insurance.
4. Design a highly configurable system which can be utilized to define different gamified approaches and execute processes necessary to their functioning.
5. Based on the designed system, implement a prototype of technology readiness level[1] 6 (Technology demonstrated in relevant environment), for a product to be commercialized by i2S.

The following restrictions also apply:

- The design prototype must be independent of other i2S modules to function.
- Development technologies are limited to those utilized by i2S, requiring the development to be made in Java 1.8 for service and backend applications and Angular for any applications requiring Web based interfaces.

Another objective, secondary and personal to the author, is to increase the body of knowledge in the general area of gamification usages, including in the insurance domain.

1.4 Methodology

The development of this project was comprised of two stages which were done separately, the first one consisting on research, analysis and initial design, and the second one on development and the evolution of the design solution.

1.4.1 Stage 1 – Research, Analysis and Initial Design

In Stage one the process literature is procured, analysed, selected and documented regarding Gamification in a conceptual manner, utilization of game elements and methods of applying Gamification to a system as well as documentation regarding the functioning of the insurance

[1] Technology Readiness Level – Technology readiness level is a value on a scale that represents the maturity of a technology, details of the meaning of each TRL can be seen in Annex E.

industry and the utilization of gamification in the industry. The localization of possible areas of utilization of gamification in the insurance is made and the detected options evaluated and compared, resulting in the choice of a single option between the presented ones.

An initial design of the system is then created with the features necessary to address the approach created.

1.4.2 Stage 2 – Development, Design Improvements and Review

Stage two of the process is then initiated, this time inserted into a technical team of i2S utilizing the SCRUM methodology and, inserted into the team's sprints, the designed system is implemented according to the specifications determined in the previous step. Alterations to the design do to unforeseen details of implementation were done during this stage where entire areas had to be redesigned, created or were dropped completely

The last part of the process rests in evaluating the developed prototype according to a set of criteria determined initially during the definition of the project itself, then conclusions are drawn based on the results of the prototype's evaluation and the general process of the project.

During this entire process internal presentations (to i2S) are planned for broader and broader groups starting, with the team the author will work with, then to the entire area, the product direction, in open presentations to the company's employees and finally to clients of i2S.

An initial work plan determining the order of this process, including deadlines and milestones can be seen in annex A, this work plan was followed only until April when most milestones were deemed unviable. The development schedule proceeded following the established pattern and most milestones were dropped.

1.5 Value Analysis

“Value Analysis can be defined as a process of systematic review that is applied to existing product designs in order to compare the function of the product required by a customer to meet their requirements at the lowest cost consistent with the specified performance and reliability needed” (Rich and Olweg, 2000).

1.5.1 Process of Value Analysis

The process of value analysis can be divided into five stages, Orientation, Functional Analysis, Creative Brainstorming, Analysis and Evaluation and Implementation (Rich and Olweg, 2000).

Stage 1 - Orientation

The Orientation stage of the value analysis process consists in forming the Value Analysis team, selecting the product and a preparation stage. Although the constitution of the team varies the typical team consists of:

- Designers, the people who design the product itself;
- Manufacturer engineers and production engineers, the ones who have the duty of taking the designed product and determining how to execute it in detail;
- Purchasing specialists, the ones who deal with the sourcing the material necessary for the production on the product;
- Operational staff, the ones who produce the product or deliver the product.

An extended team may be utilized under certain conditions this team may extend to individuals outside the company itself such as customers, suppliers and subcontractors. (Rich and Olweg, 2000)

For the selection of the product some criteria exist, these have the objective of making the Value Analysis process more commercially important and increase the potential profits generated some of these criteria may include known problems, products whose volume sales are due to rise and products whose value margin is poor (Rich and Olweg, 2000).

After the team is assembled and the product selected the team should then proceed with a preparation stage studding both the product itself during every part of the production, the waste produced by the products development and an analysis of competitor products. Other information should also be presented to the team including the briefing of the process of the original design, material to substantiate the understanding of the product, accounting information, purchase specific, manufacturing schematics, quality loss charts and other available information (Rich and Olweg, 2000).

During this stage a team was formed to develop this project consisting of the author of this report with support from Nuno Ferreira, Manuel Ribeiro e Catarina Tomé, who hold the position of experts in the areas of development of this project. Utilizing the i2S Solutions product as a basis of analysis the innovation process then proceeded.

Stage 2 – Functional Analysis
For the second stage an analysis of the products functions is made. The functions must be determined and described, then be ranked against each other utilizing the perceived importance of every task as a comparison value. By the end of the stage a detail of all the functions of the process and their perceived importance has been created (Rich and Olweg, 2000).

During the analysis of the product the lack of the feature provided by a system that enables gamification and increases interaction with the customer was detected and so this area became the target of the innovation process.

Stage 3 – Creative Brainstorming
In this stage the team now must attempt to generate ideas for the improvement of the product's functions and/or the reduction of costs in manufacture for this process a multitude of techniques may be utilized to aid the generation of ideas (Rich and Olweg, 2000).

Utilizing the available literature and the knowledge possessed by the members of the team a set of ideas, documented in 3.1.1 was determined for the development of this project.

Stage 4 – Analysis and Evaluation
During this stage a determination of the 'cost' versus 'worth' of each function is made generating then a 'value potential' determined by the difference between the two figures. At this point the options to modify the product design consists of eliminating a part, in the case of the value provided is not perceived to be enough to justify its cost or replacing, substitute or modifying it lowering the cost by making an improvement to the part (Rich and Olweg, 2000).

Utilizing the ideas determined previously a meeting of the team was held where concepts were voted on and the decision of the idea to proceed with was taken 3.1.2. A second meeting was held in which the features to be present in the new product were determined information on these features is present in 3.2.1.

Stage 5 – Implementation
The final stage of the process consists of presenting the results of the Value Analysis process to the responsible authority; the new product will then enter the stage of preparation for the initiation of production (Rich and Olweg, 2000).

After the selection of the features the process of designing the solution and finally developing a prototype of the design system was initiated, this is documented in section 4 and 5 of this report.

1.5.2 Value, Customer Lifetime Value and Perceived Value

For the determination of the Value of a product three value types can be considered, general Value, Customer Lifetime Value and Perceived Value.

Value

The value of a product is interpreted in different ways according to the viewpoint or the interpreter and the presented context, but the common characteristic is a high level of the desired attribute/s relative to the cost of the product.(Crow, 2002) This can be represented by the simple formula that follows in which q means "Quality" and represents any desired quality.

$$Value = \frac{\sum_{i=1}^{n} qi}{Cost}$$

Taking this broad definition of Value, it is possible to expand towards more specific types of value referring to specific contexts by segmenting this definition according to viewpoint and context.

Within this broad Value; however, it can said that this project's value rests on the reasonable novelty of the utilization of the concept of Gamification in the context of insurance providing not only a good way of increasing the interaction between the insured party and the insurance company, which is one of the main driving forces behind this project, and marketing potential for the insurance companies to show new systems they use within their business, but also for the insured parties that utilize the resulting product to manage their own risk in a multitude of policies they may have, not only decreasing their chance of having to file a claim, that, even when covered does not only not cover the entirety of the damage take but also is a stressful process, but also providing them with a more accurate perspective of their potential risk factors. Lastly it also adds value to the i2S company as it provides them with a new product, which has reasonably customization aspects.

Customer Lifetime Value

Customer Lifetime Value can be defined as "any demand-side, personal perception of advantage arising out of a customer's association with an organisation's offering, and can occur as reduction in sacrifice; presence of benefit (perceived as either attributes or outcomes); the resultant of any weighed combination of sacrifice and benefit (determined and expressed either rationally or intuitively); or an aggregation, over time, of any or all of these" (Woodall, 2003).

As such this Value is a combination of five customer-related concepts, Marketing, Sale, Derived, Net and Rational Customer Lifetime Value that do not only vary from individual to individual but also on a temporal scale. This is essentially a way of subdividing the concept of Value to the customer side and then proceed to subdivide this value in a multitude of smaller composing parts of overall Value (Woodall, 2003).

Perceived Value

This type of value is based on how different customers perceive the value of the same product or service but also the perception the company has of the client's value perception. This can be represented by a multidimensional construct of the second order, on one axis the benefit and sacrifice domains are presented and the other possesses the product, service and relationship scopes. This construct and the value-based drivers corresponding to each combination can be seen in Table 1 (Lapierre, 2000).

Table 1 – Value Based Drivers Table

	Product	Service	Relationship
Benefit	Alternative Solutions Quality Customization	Responsiveness Flexibility Reliability Technical Competence	Image Trust Solidarity
Sacrifice	Price		Time/Effort/Energy Conflict

As can be concluded by this short analysis all these types of Value are complementary and are simply parts of the same whole, and only by analysing them together an actual Value estimation can be reached.

Project Value

Based on the three perceptions of Value presented the interested parties in this project can be subdivided into three, i2S, the company providing a product, the Insurance Companies, the party acquiring the product and then providing the customized product to their customers as part of a service (insurance policy) and the clients purchasing insurance from the companies and utilizing the customized product; as such the determined value can be aligned to these parties separately.

For i2S: Provides them with a new product to add to their line-up that is both innovative in the market and highly configurable to a variety of situations, this does require the usage of more human resources form development and maintenance of the product as well as adding more work to the marketing of these new products.

For i2S' Clients (Insurance Companies): Creates new marketing options of products being offered by the company, creates a way of easily tracking the evolving risk of individual clients in a cost effective way, have a highly configurable system to do this with, possibly even being bundled with the core product they purchase for i2S, this does require a new process of marketing the new product to their clients and the dedication of human resources to manage the product and to treat the information supplied by their clients, which, hopefully will be a substantial amount.

For the Insurance Companies' Clients: Provides them a way of understanding and determining their risk factors for given insurance policies they may have and work to reduce them, possibly to some reward, at low or no extra cost and in an interesting, engaging and fun way that can also save time by not requiring the client to go to a physical location to provide new information about their risk state to the company, this does require going to the process of accessing the application on a regular basis and proving the time necessary to learn and utilize it.

1.5.3 Innovation Process

Innovation, in any area, although important to the development of any company providing significant competitive advantages, comes with a risk as this process requires significant resources and failure may cause financial and image problems to the individual/group/company taking said risk.

To minimize the risk of failure, techniques for the process of innovation are constantly developed and improved. One such technique, sometimes referred as the Peter Koen Model, divides the innovation process in three areas: Fuzzy Front End(FFE); also referred to as FEI, Front End of Innovation (Koen *et al.*, 2001), New Product Development(NPD) and Commercialization (Koen *et al.*, 2002) .

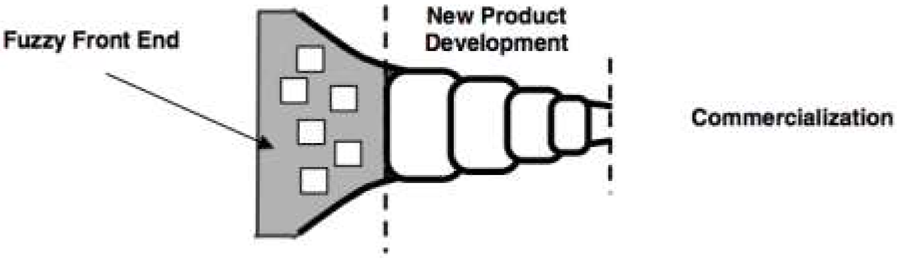


Figure 1 - Innovation Process (Koen *et al.*, 2001)

New Concept Development
In an effort to give a common language and vocabulary to the FFE process the New Concept Development (NCD) theoretical construct was developed in 2001 by Koen and his team. As can be seen in Figure 2, the New Concept Development method divides the FFE process into five key elements: Opportunity Identification, Opportunity Analysis, Idea Genesis, Idea Selection and Concept & Technology Development; these elements are then surrounded by the Influencing Factors and being fuelled by the Engine (Koen *et al.*, 2001).

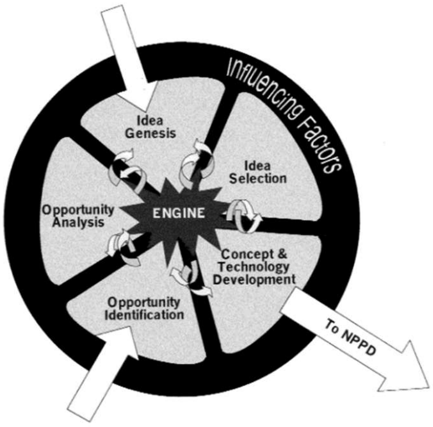


Figure 2 – New Concept Development Framework Representation (Koen *et al.*, 2001)

Opportunity Identification

At this point the opportunities that might be pursued are identified, typically generated by the current goals of the innovator party; which can be anything from an entirely new direction for the business to expand or an upgrade for an existing product. The essential part of this element is the methods and sources that can be used to identify possible opportunities. Amongst the possible methods of identifying are creativity tools and techniques such as brainstorming, mind mapping and lateral thinking; problem solving techniques such as causal analysis, fishbone diagrams, process, mapping; be the result of something as complex as market analysis or something as simple as idle chat (Koen *et al.*, 2001)

Gamification was detected as a possible viable option to respond to the problem of the lack of interaction between insurance company and client by an analysis of the insurance market in a global sense relating new approaches being utilized by companies in the insurance sector and classifying their viability. This analysis and the details of it are; however, confidential and may not be discussed in this document.

Opportunity Analysis

The ideas detected in the original stage must be analysed and evaluated so as their viability to be turned into specific business and technology opportunities can be determined. This often involves making uncertain assessments about the market and technologies. The effort into this analysis varies according to a multitude of factor such as the company's business strategy and culture. Due to the nature of this stage competitive intelligence and trend analysis are extensively used (Koen *et al.*, 2001).

Utilizing analysis methods unknown, data from consultants, both publicly available information (Chatterjee, Pathak and Kumar, 2017; Manoharan, Agarwal and Shukla, 2017) and documentation internal to i2S but also via discussions between experts on the industry at i2S, pointed to the viability of the opportunity.

Idea Genesis

This step consists on utilizing the opportunity detected previously and maturing it into a concrete idea, at this point ideas are worked upon extensively until they reach a point of sufficient detail, this can be enhanced by collaboration with other teams, companies or even direct contact with the customer. This process can be made with the utilization of brainstorming sessions and idea banks but may also result from a spontaneous, informal process. This section may also feed Opportunity Identification as the non-linear nature of the NCD process allows for a free flow between elements (Koen *et al.*, 2001).

In this project the process of Idea Genesis involved mostly a study of the available literature and an extrapolation of ideas from the present knowledge, but also discussion with experts in the industry and some literature relating to ideas for the utilization of gamification in the insurance industry (Manoharan, Agarwal and Shukla, 2017), resulting in the seven possible ideas presented in 3.1.1.

Idea Selection

The idea genesis process never generates a single idea but a multitude, as such it is necessary to select between the many ideas, and this process can be anything from a simple personal choice to a more formalized process. At this point an extensive formalized process is difficult due to the limited information available at this point and definition of the financial return of the idea itself is often speculation. This process should however not be too restrictive as a number of ideas should be allowed trough as to improve the chances that one of them will succeed (Koen *et al.*, 2001).

After the Idea Genesis stage, a group within i2S was presented the seven possible ideas and, utilizing a Weighted Decision Matrix, compared them utilizing a set of criteria, this process is documented in 3.1.2.

Concept & Technology Development

This final element of the model consists the development of a business case based on the ideas selected and a set of environment estimations; the formality of this process varies according to business but also the nature of the opportunity, the available resources and the organizational requirements for this process. Due to the nonlinear functioning of the NPD model this element may even be the first stage of the process. At the end of this element the usual result is a formal product proposal for the new concept (Koen *et al.*, 2001).

With the selected idea, the process of substantiating it into a product proposal was started, utilizing Deterding's Lens of Intrinsic Atoms, and documented in 3.2.

1.5.4 Business Model Canvas

A widely used tool for representing a company's business model is the "Business Model Canvas", this tool subdivides the parts of a business into several areas that are represented in the model from the back-end of the business to the customer side from left to right. This BMC has a total of nine "building blocks" that allow the possibility to describe any business in a simple way. These building blocks are interconnected in various ways that are represented in the layout of the BMC.(Osterwalder *et al.*, 2010a) The BMC can be consulted in annex B.

Customer Segments

Customer segments are all the people or organizations for which you are creating value, this does not necessarily have to be paying customers, it can also be the people that simple use your product (Osterwalder *et al.*, 2010b).

In this case there is only once customer segment Non-Life Insurance Companies, as the project proposes to create a product that they can use for gamification but not a product that is ready to distribute to their clients as is.

Value Proposition

Value proposition is what value your business provides to the customer segments you are selling to (Osterwalder *et al.*, 2010b).

This project's purpose is to create a highly configurable application that contains inbuilt gamified aspects for the sale to Insurance Companies. This will provide the Companies with a new tool they may utilize to increase the, extremely lacking, communication between client and company.

With this application Insurance companies will be provided with a, relatively, cheap method of engaging with their clients in new ways and being provided with more relevant information for the process of risk calculation, not only improving the relationship between them and the customer but also improving the customer's experience with their interactions, creating a more engaged and loyal client base.

Channels

Channels describe trough what means the value the business creates reaches the customer segments (Osterwalder *et al.*, 2010b).

The channels used for this project would be the same as the ones possessed by i2S, the client network already present, direct presentations from personal of i2S to prospective clients, mail marketing systems, the i2S website, social network advertising and the communication to the customer of new features and products via Client Managers.

Customer Relationship

Customer relationship outline the type of relationship the business create and foster with their customers (Osterwalder *et al.*, 2010b).

In the case of i2S the interaction with the customer is made by personnel assigned to each specific customer in the case of major customers and it is done this way in the effort to create a closer relationship between businesses.

Revenue Streams

Revenue streams represent the methods which the business uses to obtain revenue from the value proposition.

In the case of i2S this is done by the initial licensing of the product, as well as a usage-based model and services provided to the customer on request.

Key Activities

Key activities are, quite simply, the activities that the business must execute to function properly and create the value proposed (Osterwalder *et al.*, 2010b).

In the case of this project the key activities are the design of the system to be developed, the development of the system and the adaptation of existing systems to comply with the new system's parameters.

Key Resources

Key resources are the indispensable assets for the business model and are the tools that allow it to execute its key activities for the creation of the value proposed (Osterwalder *et al.*, 2010b).

To execute the activities properly a multitude of resources are required, software is required to execute the development and design of the system, hardware is required to run said software, i2S' human resources are necessary for aiding with development and designing of the system and finally the customer base is necessary for testing and feedback on the prototype.

Key Partnership
Key partners are other entities that execute key activities and provide key resources for the business, as not all key activities are executed by the business being modelled and the resources must have a source (Osterwalder *et al.*, 2010b).

For this project the key partners can are: Microsoft, for providing tools for discussion, design and documentation of the development process, Oracle and Google for the development and support of the programming languages utilized in the development of the system and Docker for internal testing and usage.

Cost Structure
Cost structure is the set of costs necessary to maintain the entire business from communications with customer segments to the maintaining of key resources and, as such, it is the last point in the BMC to be filled out (Osterwalder *et al.*, 2010b).

In i2S the cost structure is as follows: costs related to human resources such as salaries and other personnel expenditures, costs due to hardware and software licencing and purchase, and general organizational overhead, costs that the company must sustain for its regular activities to continue functioning properly.

1.6 Document Structure

This document is divided into 7 chapters, a references section and multiple annexes.

Chapter 1 Introduction

- Introduces the context of the project being documented, presents the problem to be addressed, outlines objectives for the project, generally describes the methodology followed during the entire protect and provides a pre-emptive value analysis of the developed solution.

Chapter 2 State of the Art

- Describes the concept of Gamification and underlying concepts, provides information about elements utilized in gamification, lists a set of existing frameworks and provides a detailed description about the ones being utilized for this project.
- Provides information on the insurance industry in broad terms, concepts utilized in the industry relevant to the project and a small overview of the state of the national insurance market.
- Finally, it outlines a select few utilizations of gamification in the insurance industry or in areas that can be related to the insurance industry.

Chapter 3 Gamification Design

- A lifting of possible areas of application in the insurance industry is made, to obtain a broad perspective over the entire group of prospective gamification targets only one of these areas being selected for exploration according to a set of criteria.
- The selected idea is made into a concept and then, following the described gamification framework, gamification concepts are evaluated and selected for implementation into the system.
- A more focus approach is then designed for implementation and evaluated utilizing the criteria outlined in the Octalysis framework.

Chapter 4 Analysis and Design

- Provides an overview of the solution to be developed, including use case, high level components and deployment diagrams, a listing of the features of the solution beings design with detailed information about processes and system APIs, constraints and dependencies as well as some additional information considered pertinent.
- A themed approach to an auto-insurance policy utilizing telematics information is then designed for implementation in the solution.

Chapter 5 Development

- This chapter will document the entire development process of the solution, including how the design processes were addressed, problems detected during the implementation of each process are also documented as well as implementation alternatives.

Chapter 6 Evaluation

- Testing hypotheses, necessary to evaluate the results of the gamification system being developed, are determined, evaluation metrics are assigned, methodology of determining said metrics is described and a description of the methods utilized to evaluate these metrics are determined.
- A testing model utilizing the QEF (Quality Evaluation Framework) is described for usage in the evaluation of the state of completion of the system being developed and then applied to the developed solution.
- The design of a test suite of Unit and Integration tests is detailed, and its implementation details are determined and the results of their execution on the code base are presented.

Chapter 7 Conclusion

- A synthesis of the document contents, proceedings and results is made.
- The conclusion generated from the results of the project is presented
- Future work and limitations regarding the entirety of the project and discriminated and detailed.

Annexes

- Contains support information that, although not deemed necessary to be included in the main body of the document, clarifies or supplements content that is.

2 State of the Art

Chapter Contents

2.1 Gamification

Rumoured to have been used in first in the yearly 2000's, Gamification is a term which most sources agree can be generally defined as "the use of game elements and mechanics in non-game contexts" (Deterding *et al.*, 2011). This definition however does not consider its theoretical foundations, purposes and standards for its practice. (Seaborn and Fels, 2015).

After study of the area the author opted for utilizing their own definition of gamification it being: "Gamification is the granting of game like qualities to a system." By utilizing this definition it is possible to encapsulate a broader array of concepts than the ones put forth in Deterding's definition by utilizing the more abstract wording of "game like qualities" instead of "game elements and mechanics", "non-game contexts" was considered an unnecessary specification of area.

Gamification, as previously stated, heavily involves the utilization of game elements and mechanics in non-game contexts, what was not stated is why these elements are used. Games are powerful at motivating individuals utilizing methods of driving their desires and doing it in a predictable way that, if done right, will provide enjoyment to the individual. (Zichermann and Cunningham, 2011) As the information present pointed that these ways of motivating individuals if games can do this for virtual situations, would it not be possible to take the elements that make games and engaging and apply them to other contexts that do not cause individuals to become engaged with such ease? The exploration of this question gives rise to the concept of Gamification.

2.1.1 Viability of Gamification

The use of Gamification has been increasingly explored in this decade, but little research has been made about the viability of its use. A study dated 2014 which compared the results of 24 studies on Gamification, with evaluations in various areas, considers that "the literature review suggests that, indeed, gamification does work, but some caveats exist." (Hamari, Koivisto and Sarsa, 2014)

Not all studies evaluate the same type of outcomes and the lack of a standardized evaluation method made the results not always comparable, most of the reviewed studies revealed that Gamification produces positive aspects in general but there are several findings that this is not true for all aspects and all areas. Studies found that the effects of Gamification may not be applicable in the long-term, attributing behaviour change to the novelty aspect provided but also that removing Gamification from a previously gamified system may have detrimental effects on the still engaged users. (Hamari, Koivisto and Sarsa, 2014)

In the end, Gamification appears to have some impact on the engagement and enjoyment of the users of a system, but the lack of a standard methodical approach to the evaluation puts the results in question as proper evaluation parameters were not used across the board. (Hamari, Koivisto and Sarsa, 2014)

2.1.2 General Objectives of the Gamification Process

Gamification can be applied to a vast array of environments and situations, each process of gamifying a system having objectives in mind as well as their own methodology of reaching said objectives. No matter what the context of the application of the gamification process its main objective is usually one of the five presented; with special note to Behaviour Change as it is core to most applications of gamification even if they intend to deal with the any of other objectives.

Relieving Tedium and Increasing Satisfaction
A problem in many workspaces is that workers sometimes get tired of their tasks as they are many times monotonous, long or even just difficult. During the 20th century it was found out that many workers, of their own accord, would ether play some sort of game on breaks, or even integrate games within their work context to combat the felt tedium of factory work; therefore it was, eventually, realized that utilizing games in work contexts could increase both their employees satisfaction, adding additional motivation for the workers to complete their tasks, and, in the process, combating the tedium of the regular monotonous workday prevalent in allot of employees tasks. With added motivation worker's tedium lowered and their satisfaction in the workplace increased. This process lead, obviously, to the next objective in the list. (P.Waltz and Deterding., 2015)

Improving Performance
The objective of any company is to improve worker performance, providing additional value to the company operations; as such workplace competitions, prizes and rankings have long been part of the regular functioning of many enterprises. Contests and rewards such as workers of the month, performance bonuses, among others can be seen as a gameful approach to a performance increase. This process however is extremely complex as varied psychological, social and economic factors, must be taken into account when creating these gameful approaches since different groups and organizational types respond differently to incentives. Gamification can offer much more than simple competition and basic reward based systems, providing an effective system of feedback to the individual, not only related to a single task, but considering a wider variety of factors, providing various ways of workers succeeding and dynamically assign rewards and motivation for improvement in distinct areas. (P.Waltz and Deterding., 2015)

Encouraging Unremunerated Work, Internally and Externally
A point that inhabits a grey area in gamification is the encouraging of unremunerated work, this is usually done by taking simple tasks that the company needs done and creating a game out of completing them, engaging internal or external individuals and having them provide the work without direct remunerated compensation. (P.Waltz and Deterding., 2015)

Internally, this can be done in an effort to improve relationships and cooperation between workers motivating them to participate more actively in tasks such as sharing information between individuals in different parts of the company that might obtain tangible benefits from it, but, as there is not tangible benefit for the information sharer, may not be shared otherwise. (P.Waltz and Deterding., 2015)

Externality, the objective may be to obtain information from clients that would not otherwise be given to the company in a spontaneous way or create content for the use of an application owned by the company. In essence attempting to use external resources to solve problems or obtain information that may be useful to the entity that supplies the gamified solution. (P.Waltz and Deterding., 2015)

Bolstering Training and Recruiting

The most studied of all these uses of gamification in enterprise contexts it is also the one that causes the most enthusiasm. Games created within this context may be used to evaluate potential recruits in possible work contexts tasks, facilitating the recruiting process; not only that but these games may also be used in training, providing the trainee with a safe and monitored environment where they can easily learn the basics and details of their tasks, as well as be quickly corrected on their mistakes and guided into the correct approach, this process can also be integrated into the general work process, aiding not only new workers to develop their skills but also aiding regular workers in improving the level of quality of their work. (P.Waltz and Deterding., 2015)

Behaviour Change

It is important to mention, if not only due to that fact that it is the core objective of gamification, everything that gamification attempts to achieve is a change in the behaviour of a target group in a way to align their interests and actions with the interest of the gamifying entity. In the end, if people are convinced to believe that there is a personal advantage to follow the preferred behaviour incentivized by the gamified system they will follow it.

2.1.3 Theoretical Foundations of Gamification

To understand how the process of gamification works, it is important to have some context of the theoretical foundations that are used to elaborate gamification frameworks. Although few sources determining theoretical foundations and basis of analysis for gamification systems exist, seven were detected by Seaborn and Fels in their survey about the state of gamification in 2015 (Seaborn and Fels, 2015): Self-Determination Theory; Intrinsic and Extrinsic Motivation; Situational Relevance; Situated Motivational Affordance; Universal Design for Learning; User-centred Design and the Transtheoretical Model of Behaviour Change.

Self-Determination Theory

“Self-Determination Theory is an approach to human motivation and personality that uses traditional empirical methods while employing an organismic metatheory that highlights the importance of human’s evolved inner resources for personality and behavioural self-regulation” (Richard M Ryan and Deci, 2000).

Utilizing this theory as a basis for a gamification framework leads to an obvious focus on the user’s self-motivation, creating an approach to the process that focuses on engaging the user with the task presented as the main objective of the gamification process; utilizing mechanics that reward the user with perceived self-improvement and allow for a degree of control on the part of the user. A set of motivation target areas and the example of some mechanics that are theorized to be effective at supporting them can be seen in Table 2.

Table 2 - Motivational areas and corresponding mechanics refering to Self-Determination Theory (adapted from Aparicio *et al.*, 2012)

Motivation Area	Mechanics
Autonomy	profiles, avatars, macros, configurable interface, alternative activities, privacy control, notification control
Competence	positive feedback, optimal challenge, progressive information, intuitive controls, points, levels, leaderboards
Relation	groups, messages, blogs, connection to social networks, chat

A notable fact is that Self-Determination Theory provides a basis on which most following theories are built upon, making it a constant throughout the most of the analysis of gamification foundations.

Intrinsic and Extrinsic Motivation

"Intrinsic motivation is defined as the doing of an activity for its inherent satisfactions rather than for some separable consequence." (Richard M. Ryan and Deci, 2000) and "Extrinsic motivation is a construct that pertains whenever an activity is done in order to attain some separable outcome" (Richard M. Ryan and Deci, 2000).

With these definitions the determination can be made that the utilization of this foundation rests mainly on the user reward system of a framework, intrinsic rewards subsisting on how motivating the task is to the user due to their enjoyment of the task or the inherent satisfaction felt by its completion; extrinsic motivation however subsists on rewards external to the task at hand, one such clear example is the salary that comes with most jobs which provides not a reward on the doing of the work itself but a monetary reward external to it. In Figure 3 the classifications of different types of motivation, including Amotivation, the lack of motivation itself, can be seen.

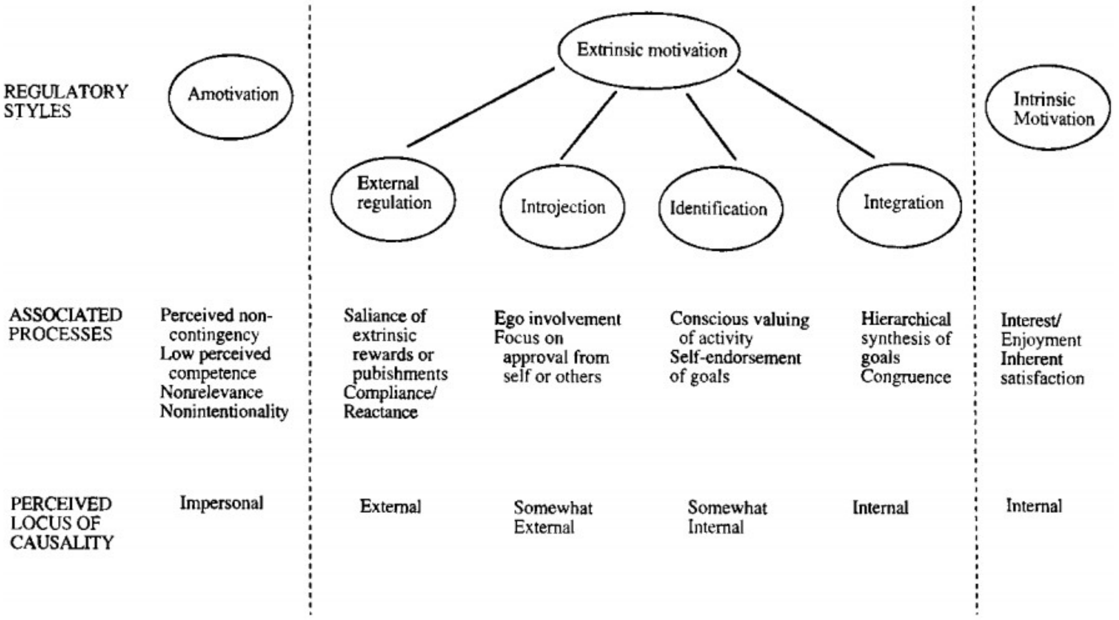


Figure 3 – A taxonomy of human motivation (Richard M. Ryan and Deci, 2000)

Situational Relevance

A broad reaching topic, Situational Relevance has mostly dealt with relevance in information retrieving, namely the determining of the effectiveness of search tools and algorithms; usually creating a query and having judges deciding on the relevance of the results according to the query made. This is a problem since distinct individuals with different backgrounds will, without a doubt, consider results differently, making the judgement inherently subjective. As such, with extensive studies in the area it was determined that a true objective better method is incredibly hard/next to impossible to achieve and, due to the inherent subjective nature of the problem, it is better to just ask the user how relevant something is (Nicholson, 2012).

Situated Motivational Affordance

"Situated motivational affordances describe the opportunities to satisfy motivational needs provided by the relation between the features of an artefact and the abilities of a subject in a given situation, comprising of the situation itself, and the artefact in its situation-specific meaning and use." (Deterding, 2011)

Situational Motivational Affordance is an attempt to build upon the concept of Situational Relevance, but with the specific goal of providing a way to analyse the effectiveness of a certain artefact in increasing an user's motivation based on said user's background and their perception of the validity of the element in the particular context as well as in the broader context both are contained in. (Deterding, 2011; Nicholson, 2012)

One such example is the inclusion of a leaderboard type system, that provides a competitive environment between users of the application; although in game contexts this kind of system is, in the vast majority of cases, chosen by the user and free of consequences outside the context it is presented it, the same system utilized to represent a classification amongst the employees of a company according to their performance is "neither voluntary not free of consequence" (Deterding, 2011), as these values may be used for the calculations of monetary rewards, therefor having been seen as exploitative and controlling by the people they were intended to motivate. (Deterding, 2011)

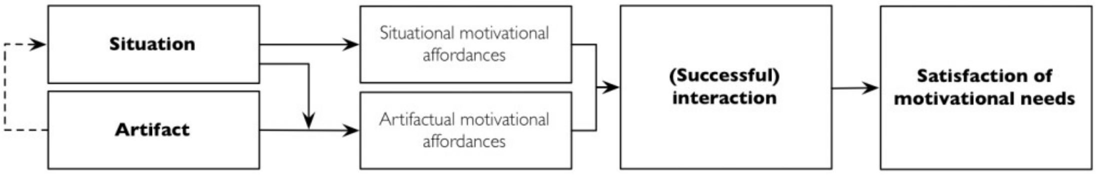

Situation
Artifact
Situational motivational affordances
Artifactual motivational affordances
(Successful) interaction
Satisfaction of motivational needs

Figure 4 - Situated Motivational Affordance Representation (Deterding, 2011)

Universal Design for Learning
Universal Design for learning as a basis for gamification rests mainly on an attempt to increase the spectrum of the ways the process being gamified can be completed. The main objective of adopting this theory is to increase the engagement of the end user by allowing them to select the way they want to complete the objective. The wider the options available the more users will find ways to engage with the process. (Nicholson, 2012)

Universal Design for Learning's concepts when a applied to gamification strongly align with the concepts of Self-Determination Theory, defending that each user has different methods they may want to use to tackle a specific objective and, by enabling these methods, possibility of engaging with an user is increased.

User-centred Design
User-centred Design advocates the following points (Norman, 1988):

- Possible actions at any moment should be easy to determine;
- Everything should be visible to the user including the conceptual model of the system, possible alternative actions and the results of actions taken;
- The current state of the system should be easy to evaluate;
- Mappings between intentions and required actions, actions and the resulting effect and the visible information and the state of the system should be natural.

Through the process of gamification, the constant objective is always the motivation of the end user as such focusing the entire process of gamification on what would be better to more easily motivate the user is a desirable and important way of reaching a successful gamified solution. (Nicholson, 2012)

2.1.4 Game Elements and Utilization

In the process of gamification, elements that are most common present in games are used in various ways with distinct objectives. These elements are varied and inconsistent between sources as they provide different basis for their qualifications of elements, some of these can be seen in Table 3 and Table 4.

Table 3 - Selection of game mechanics support (Aparicio *et al.*, 2012)

Autonomy	Competence	Relation
Profiles, avatars, macros, configurable interface, alternative activities, privacy control, notification control.	Positive feedback, optimal challenge, progressive information, intuitive controls, points, levels, leaderboards.	Groups, messages, blogs, connection to social networks, chat.

Table 4 - Game-design elements and motives (Blohm and Leimeister, 2013)

Game Element: Mechanics	Game Element: Dynamics	Motives
Documentation of Behaviour	Exploration	Intellectual Curiosity
Scoring Systems, badges, trophies	Collection	Achievement
Rankings	Competition	Social recognition
Ranks, levels, reputation points	Acquisition of Status	Social Recognition
Group Tasks	Collaboration	Social Exchange
Time pressure, tasks, quests	Challenge	Cognitive Stimulation
Avatars, Virtual Worlds, Virtual Trade	Development, Organization	Self-determination

Some sources attempted to create a standard terminology group for easy of reference to elements as can be seen in Table 5 however some of the higher level elements seem to be missing from this definition, such as those which would determine thematic.

Table 5- Legend of game element terminology (Seaborn and Fels, 2015)

Term	Definition	Alternatives
Points	Numerical Units indicating progress	Experience points; score.
Badges	Visual icons signifying achievements.	Trophies.
Leaderboards	Display of ranks for comparison.	Rankings, scoreboard.
Progression	Milestones indication progress.	Levelling, level up.
Status	Textual Monikers indicating progress.	Title, ranks.
Levels	Increasingly difficult environments.	Stage, area, world.
Rewards	Tangible, desirable items.	Incentives, prizes, gifts.
Roles	Role-playing elements of a character.	Class, character.

Another terminology group by a different source in the context of a study of the perceived effectiveness of gamification in different contexts detects ten elements but does not properly define them: Points, Leaderboards, Achievements/Badges, Levels, Story/Theme, Clear Goals, Feedback, Rewards, Progress and Challenge. (Hamari, Koivisto and Sarsa, 2014)

As can be seen there is some discrepancy on what terms mean what between different sources, due to this a group of elements was chosen for a deeper detailing in this report: Points, Levels, Leaderboards, Badges, Challenges; as they appear to be the most prominent elements to appear across all evaluated sources and Customization as it seems to be an integral part of self-motivation theory that appears widely overlooked in most cases.

Points

Points are by far the most overt type of gamification element and can be seen in almost all aspects of society representing an extremely vast array of important values. As such Points are a mandatory requirement for any process of gamification since most other elements depend on them to function even at the most basic level, although not necessarily in a way apparent to the user. The utilization of this element is paramount to the success of any gamified system and details regarding the rate and amount at which Points are awarded to the user are highly important. These Points can serve different purposes and in terms of gamification can be divided in some types: Experience Points, Redeemable Points, Skill Points, Karma Points and Reputation Points. (Zichermann and Cunningham, 2011)

Experience Points

Experience Points are considered the most important of the five types of points presented and have the particularity of not serving as any kind of currency within the system, unlike most others, being a way of tracking the status of the user within the system. Most actions taken within the context of the system will earn the players this kind of points, and in general they are never debited and cannot be redeemed. This type of Points can be utilized to align the user's behaviour with the objectives of the system designer. Ideally this type of Points never reaches a cap, instead continuing to increase forever. (Zichermann and Cunningham, 2011)

Experience Points in gamification are utilize in much the same way as videogames and many examples of their usage exist, for instance the internet game/chore tracker Chore Wars utilizes experience points as a standard RPG game. When the player archives 200 points they raise their level and their stats change according to what chores they most did, changing their values up and down. This allows the system to more closely represent the user's strengths in tasks and changes in behaviour allowing for a sense of progress and specialization.

In most systems Experience Points are directly tied to a player level usually unlocking something, rewarding the player or simply changing a title. This is most likely due to their original utilization in old school pen and paper Role-Playing Game (RPG) Dungeons and Dragons in 1974 and popularized by further games over the years.

Redeemable Points

Redeemable Points are, as the name suggests, points that can be redeemed within the system for things, and therefore tend to fluctuate over time, as they are awarded and spent. In loyalty programs and social games, the loop of earning and spending this points is "gain and burn". Generally Redeemable Points, are the foundation of a virtual economy within the system and require monitoring and pose significant problems if related to real world monetary transactions. (Zichermann and Cunningham, 2011) A danger of Redeemable Points is the way they are perceived by the user according to the possible rewards they can be exchanged by, if the options are not desirable or realistically obtainable the risk of the user being lost is increased (Zichermann and Cunningham, 2011).

The obvious comparison and origin of Redeemable Points is currency, many systems utilize this type of Points to allow the users to redeem them for rewards of various types. A basic example of gamification utilizing Redeemable points is Loyalty Cards, every time the user makes a purchase at a shop they obtain a certain amount of loyalty points which then can then exchange for a reward at the same store, effectively having a monetary value.

Skill Points

Skill Points can be seen lower level Experience Points and are rewarded along Experience and Redeemable Points, and instead of representing overall progress as Experience Points do, they represent progress in a subset of the system (Zichermann and Cunningham, 2011).

Karma Points

Karma Points can also be perceived as a subset of Redeemable Points with a specific particularity, while users may be encouraged to save Redeemable Points but Karma points have no reason to be saved as their main use is the sharing of these points with others, or investing into some sort of subsystem (Zichermann and Cunningham, 2011).

This kind of points is harder to find in gamified approaches though it is possible to find in some games with social aspects easily, one such example are Gifts from the mobile game Pokémon GO, each user may have up to 10 Gifts, but they are worthless by themselves, the only use for Gifts is to give them to friends, hereby giving said friends useful items and increasing your friendships level. This isn't a straight Karma Point system as we are talking about the item gift and not "gift points" but it does behave as one.

Reputation Points

Reputation Points are a complex point system to determine trust between parties within the system and their complexity is usually derived from the way these point systems are designed and the determination of activities that provide a meaningful value to these points as their value is of great importance. (Zichermann and Cunningham, 2011)

In the online platform Stack Overflow the points called Reputation has a dual utilization, first it works as Reputation Points, which help to denote the quality level of your questions and answers allowing other users to immediately see the likelihood of your contributions being valid and good; however, they also work as Experience Points unlocking new privileges of website usage with higher reputation points. Karma in the online platform reddit is also a type of Reputation Points, and not Karma Points, they are attributed by other members of the platform however they can never be redeemed or traded to someone, being instead freely given and taken by others.

Levels

Levels are usually indicators of progress but are not exclusively used for that purpose and, although their use in gamification is different that their traditional use in regular games, they are still useful as a method of marking the user's standing on the gamified system. Normally and classically levels represent an increase in the difficulty of achieving the provided goals, as the Level rises so does the difficulty of the goal and, sometimes, so does the reward; however, they can also be used as an enduring representation of how far the user is in the system, requiring higher and higher requirements to reach the following goals but never leaving the possibility of a user returning to a previous level. (Zichermann and Cunningham, 2011)

Leaderboards

Leaderboards have the objective of providing users with a comparison, they are usually simple and do not require much explaining to most users and are normally represented in a list fashion which contains all users of the system ordered by certain criteria, normally some a point system, in which each user is given a rank number according to their position in the list. (Zichermann and Cunningham, 2011)

Leaderboards provide an issue related to privacy of information, this is especially prevalent in the process of gamification since the information is not a fabrication of a game but, usually, actual personal information of the user itself. As such special care must be taken when implementing Leaderboards in gamified systems such as utilizing abstract point systems (Zichermann and Cunningham, 2011).

Another care to take when utilizing Leaderboards is the case where they negatively impact the motivation of the users, this is normally more prevalent with competitive Leaderboards, one idea to deal with this kind of problem is the use of social or relative leaderboards, comparing users only with their friends, a group they belong to or only from that day, among others, in order to allow every user to have a place in a leaderboard (Zichermann and Cunningham, 2011).

Badges

Badges can represent various things and can affect the user in various ways by targeting different motivations such as the surprise of getting an unexpected reward or the simple desire to collect all badges. Within the gamified system Badges are used generally as a way to signal progress to the user and may even take the place of levels (Zichermann and Cunningham, 2011).

Again like Leaderboards there are concerns regarding the utilization of Badges when it comes to the retention of users, Badges need to be meaningful as an endless parade of Badges that are given out within a very short span of time will only leave the user with the impression that each Badge has a relatively low value and are less inclined to be motivated to obtain them, on the other hand extremely few and too hard to obtain Badges will make the user demotivated as obtaining them seems too difficult or time consuming. Another concern has to do with the obfuscation of the requirements to obtain Badges, as this both has the impact of possibly increasing the enjoyment of the user that obtains an unexpected Badge and the possibility of frustrating the user that is attempting to obtain that Badge and is not able to determine the requirements; a possible solution is to have both types of Badges present, predictable and unpredictable (Zichermann and Cunningham, 2011).

Another aspect to take into account when creating a gamified system utilizing Badges is the visual aspect of the badge as it has an impact on the enjoyment of the user receiving it as well as the collection aspect of badges (Zichermann and Cunningham, 2011).

Challenges

Challenges are the elements that allow the directing of users within the gamified system by providing the user with tasks to perform that, when completed, will reward them in some manner. Challenges also do not need to be overtly stated, such as the description of how to obtain a Badge. Challenges also do not need to be restricted to requiring a single user to complete them, social challenges involving multiple users can be used to not only allow the user to progress in the gamified system but also to nurture a social sense to the system increasing the factors of social interaction (Zichermann and Cunningham, 2011).

Customization

One aspect that increases a user's investment in a gamified system is the ability to customize their experience, this can take many forms, from fully customizable 3D avatars, to just allowing a user to place a headshot and choosing a screen name. By having the ability to customize their experience, to even the slightest level, users become more committed to the gamified system and are more likely to stay invested. One thing to have in mind while having some customization is that while some customization is engaging too many choices cause a drop in satisfaction drastically so a reasonable amount of customization is preferable over an expansive and over-varied one (Zichermann and Cunningham, 2011).

2.1.5 Gamification Frameworks

According to Morschheuser *et al.*, (2018), the designing of gamified software rests on thirteen design principles that are reflected within most gamification frameworks that are inserted into seven distinct stages:

- **Design Principle 1**: Understand the user needs, motivation and behaviour, as well as the characteristics of the context.
- **Design Principle 2**: Identify project objectives and define them clearly.
- **Design Principle 3**: Test gamification design ideas as early as possible.
- **Design Principle 4**: Follow an iterative design process.
- **Design Principle 5**: Profound knowledge in game-design and human psychology.
- **Design Principle 6**: Assess if gamification is the right choice to achieve the objectives.
- **Design Principle 7**: Stakeholders and organizations must understand and support gamification.
- **Design Principle 8**: Focus on user needs during the ideation phase.
- **Design Principle 9**: Define and use metrics for the evaluation and monitoring of the success, as well as the psychological and behavioural effects of a gamification approach.

- **Design Principle 10**: Control for cheating / gaming-the-system.
- **Design Principle 11**: Manage and monitor to continuously optimize the gamification design.
- **Design Principle 12**: Consider legal and ethical constraints in the design phase.
- **Design Principle 13**: Involve users in the ideation and design phase.

Morschheuser *et al.*, 2018, then proceeds to attribute these principles to the determined seven phases of the gamification method, this distribution can be seen in Table 6.

Table 6 - Mapping of identified design principles to phases of contributed gamification engineering method (adapted from Morschheuser *et al.*, 2018)

Method Phase	**Design Principles reflected in the method phase**
1. Project Preparation	2, 6, 7, 9
2. Analysis of Context and Users	1
3. Ideation	8, 13
4. Design	3, 4, 5, 12, 13
5. Implementation of Design	4, 11, 13
6. Evaluation	9
7. Monitoring	9, 10, 11

Gamification methods and their comparative evaluation are somehow limited in sources as such much of this section is taken from an analysis made by Deterding (Deterding, 2015), the reasons for this choice were, the previously made comparison of existing gamification methods by this author, the determination of various characteristics and issues of said methods and a reasonable suggested gamification method by the author. It is of note that some of the characteristics determined by Deterding are mentioned in other reviews of the available methods for gamification which also share the opinion of a lack of a generalized method for the gamification process (Mora *et al.*, 2015). More recently, Morschheuser *et al.* atempted to construct a newer, more complete method to clear these limitations.

Since the beginning of the decade the attempt to create a standard method of gamification has been made by scholars; however, these attempts rested mostly on theoretical basis for the development of a method and how to interlock them than in determining a method. A small four stage method was actually created (Aparicio *et al.*, 2012) however it proved simplistic and too high concept to be of much use as the method of choosing the right elements for tasks was completely ignored as was with most of the academic research. (Deterding, 2015)

In the industry side something more substantial is attainable, based on a method by Amy Kim (Kim, 2010). This method consists of identifying the “engagement style” of the target audience/s, then creating “player journeys” for each of the detected audiences with three stages, novices, experts and masters; this is followed by a mapping of engagement mechanics to each phase of these journeys and connected to reward systems; customization for the user is then chosen and finally engagement loops are devised regularly: call to action, social action, system feedback, and underlying motivation. All this process leaning heavily on social engagement methods and actions. The majority of later methods amend this one and their aggregate, according to Deterding, reaches a process similar to the listed below (Deterding, 2015):

1. Identify system owner goals.
2. Identify trackable behaviours of end users that support these goals; quantify their relative contribution in a metric.
3. Profile and segment end users using their player typologies (usually "Bartle Types" (Bartle, 1996)).
4. Select and specify game design patterns:
   - 4.1. Translate the quantified system owner value of end user behaviour into point values displayed back to users.
   - 4.2. Articulate an ordered sequence of explicit goals for end users, consisting of sets of behaviours or point thresholds (quests, challenges, levels).
   - 4.3. Define feedback to display upon single user actions ("engagement loop") as well as reaching point thresholds or goals (advisements, badges, leaderboards), including potential rewards (virtual items, customization options).
   - 4.4. Chose additional game design patterns.
5. Playtest.
6. Build and Deploy.
7. Use analytics of user behaviours to monitor system performance and guide the improvement and release of new content and features.

Also according to Deterding the available methods possess a range of characteristics that made them unsuitable or insufficiently developed. The characteristics determined are (Deterding, 2015):

- Lack of formative research, with poor detail on data collecting methods or simply suggesting the use of educated guesses.
- Reliance on player topologies, mostly Battle Types, which can lead to a loss of context when determining user personas and the mechanics that work towards their engagement, also ignoring further expansion on those types and the inherent context in which these types were determined (Multi-User Dungeon players) extrapolating that the data determined in this specific context applies to all contexts.
- Appealing to motivational psychology, with self-determination theory being the most widespread theoretical method utilized across frameworks, as mentioned earlier, and the mixing of various theoretical models leading to questionable and untested models of motivation.
- Inherent addictive, pattern-based approach, namely the use of the same subset of game patterns whose centrality to games is questionable at best, with a general concert of utilizing a repeatable pattern-based approach to the design of any gamification process with a focus on addictive, iterative patterns.

- Lacking guidance in game design pattern choice, namely the lack of guidance of adapting the supplied patterns in specific contexts, with suggestions being vague or even non-existent, with a lack of reference to previously determined player personas or types for the evaluation of pattern use, although with some exceptions.
- Lack of interactive prototyping, as no methods detail the need of an explicit evaluation phase, focusing on post-launch monitoring if at all.
- Data-driven design, as the methods suggest tracking and monitoring after deployment only despite the major value that data from user behaviours has to the development of a gamified system.

Deterding then determines that the reviewed methods simply fail the requirements of gameful design, previously determined in the same source. Due to this, Deterding then proposes a new model for gamification which is the one selected for usage in this project even though it processes some limitations that are discussed in 7.1.1.

Presented by Morschheuser *et al.* there is also a comparison between various methods, many of which are, however, not available to the author, this evaluation can be seen in annex F as it was too large to present here properly.

Although the method proposed by Morschheuser *et al.* is of more general application it did not offer an example of its application as Deterding's did, hence the choice. And despite listing various methods in its paper, including Deterding's, most were unavailable to the author of this report, some were part of removed papers and the few available were specific to unrelated areas and Deterding's method (Morschheuser *et al.*, 2018).

The Lens of Intrinsic Atoms Gamification Method

After evaluating and dismissing the viability of the available approaches, Deterding provides his own method based on Design Lenses, a method developed originally for game design by Schell (Jesse Schell, 2008) but, later, an adaptation was suggested by Scott (Scott, 2010) for general interactive design, but very lightly developed. The method of Design Lenses utilizes memorable names, a design principle statement and a set of questions in an attempt to align the designer's thinking to the lens, and on Skill Atoms, a way of attempting to develop a game grammar, skill atoms "describe a feedback loop between user and system that is organized around a central challenge or skill", they are composed of goals, actions, objects, rules, feedback, challenge and motivation. By combining these aspects Deterding creates what he calls "The Lens of Intrinsic Skill Atoms" which can be seen in Table 7. (Deterding, 2015)

To answer the questions presented in this lens, Deterding proposes a five step process: "first, the designer determines goals and parameters of the design. Formative research identifies user needs, motivations, and hurdles underlying the target activity. If gameful design is appropriate, the designer moves on to synthesis and ideation. In innovating mode, triplets of activity, challenge, and motivation are teased out and translated into brainstorming triggers for new skill atoms. In evaluating mode, the lens of intrinsic skill atoms is used to tease out the skill atoms already entailed in the existing system. Further design lenses are then used to identify issues and generate ideas for improvement. In both modes, the designer storyboards (new or revised) skill atoms and iteratively builds, tests, and evaluates prototypes of them.” (Deterding, 2015)

Table 7 - The Lens of Intrinsic Skill Atoms (adapted from Deterding, 2015)

The Lens of Intrinsic Skill Atoms
In pursuing their needs, a user’s activity entails certain inherent, skill-based challenges. A gameful system supports the user’s needs by both (a) directly facilitating their attainment, removing all extraneous challenges, and (b) restructuring remaining inherent challenges into nested, interlinked feedback loops of goals, actions, objects, rules and feedback that afford motivating experiences.
1. What Motivations energize and direct the activity? 2. What challenges are inherent in the activity? What challenges can be removed though automation on improving usability? What challenges remain that the user can learn to get better at? 3. How does my system articulate these inherent challenges in goals? (How might it articulate them to connect to the user’s motivations?) 4. What actions can users take in my system to achieve these goals? 5. What objects can users take in my system to achieve these goals? 6. What rules does my system articulate that determine what actions are allowable and what system changes and feedback they result in? 7. What feedback does my system provide on how successful the user’s actions were and how much progress they have made towards their goals? (How might I make this feedback clear, immediate, actionable, speaking to the user’s motivations, affording a sense of competence?)

- **Strategy**: Initialized by clearly defining the targets of the intervention and the metrics that will be used to evaluate the obtained results, target audience and activity are then determined due to some form of data gathering method, the resulting list is then prioritized based on the presume effect and the best target for motivation improvement finally the standard constraints are detailed as well as the technology requirements. (Deterding, 2015)
- **Research**: The activities detected previously in the Strategy step are deconstructed into behaviour chains or a similar representation and analysed; utilizing this a set of user needs, motivations and hurdles is to be identified this can be done in various methods; however, interviews with target users are usually the best way to do this determination; finally, it is determined if graceful design is in fact a useful approach to the previously determined target. (Deterding, 2015)
- **Synthesis**: Activity-challenge-motivation triplets are identified by determining what challenges are inherent to the activity, what challenge can be automated, which challenges can remain and be utilized and what possible motivation can be used to

correspond to the activity that is nurtured by the challenge; at this point the Lens of Intrinsic Atoms is used to aid in the determination of the triplets. (Deterding, 2015)

- **Ideation**: Generation of ideas on how to gamify the system at hand, this can be done in various ways, the one suggested by Deterding is the use of innovation steams, simple sentences with predetermined gaps that can be filled with ideas from the participants in a brainstorming process to generate a quick and structured set of possibly useful ways of gamifying, designers can use designs lens to evaluate skill atoms and generate ideas utilizing these evaluations. The generated ideas are then to be prioritized according to criteria determined by the group, then sketched into initial concepts in various ways, these concepts are then to be evaluated and refined with design lens. (Deterding, 2015)
- **Iterative Prototyping**: Creating low-fidelity prototypes and then playtesting them, discarding failed concepts and iterating on working ones. At this point the important element to test is not a finished interactive process but a rough draft consisting only of the core challenge of each skill atom. The entire process is then repeated until a deemed viable core experience is found and only then does it proceed to actual development. (Deterding, 2015)

Octalysis Framework

Created Yu-Kai Chou the Octalysis framework is based on 8 core drives that enables the classification and analysis of a gamification design solution in each of these axis, in areas classified as Left Brain and Right Brain core drives and in White or Black Hat Gamificiation and is then classified with an “Octalysis Score”. the specific process of addressing these eight core drives is not publicly documented and does not appear to follow one based one relying more on a semi-subjective evaluation of how well addressed areas are by the implemented design solution, this is, however, pure speculation. Despite all this the Octalysis framework is an important framework in the field of Gamification and the provided public details shall be listed as follows. (Chou, 2018)

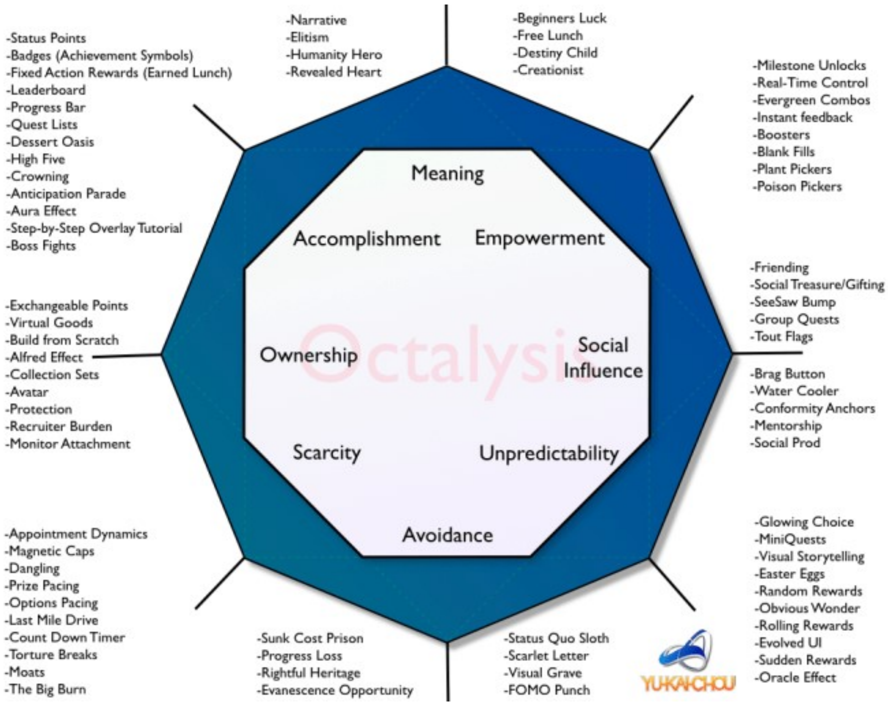


Figure 5 - Eight Core Drives Octalysis (Chou, 2018)

The Eight Core Drives

As mentioned previously and can be seen in Figure 5 the Octalysis framework rests upon 8 Core Drives (Chou, 2018):

1. **Epic Meaning & Calling**: Allowing the player to feel a sense of being greater than himself the system revolves around the player not the other way around.
2. **Development & Accomplishment**: The feeling of progress, this drive is usually accomplished utilizing challenges combined with badges and its where most gamification approaches begin and end.
3. **Empower of Creativity & Feedback**: This drive rests on allowing the user to get creative with the system and be allowed feedback of the results of their creativity usually instantly or very quickly.
4. **Ownership & Possession**: The feeling of ownership of something in a system can be cultivated in many ways such as the allowing of customization or the utilization of virtual currencies that allow the player to perceive value in the continuation of utilization of the system as it’s no longer someone else’s but their own.
5. **Social Influence & Relatedness**: Investing in social interaction allows the player to compare, brag, compete and learn from other players, also it relates to how relatable a system is to its users**.**
6. **Scarcity & Impatience**: Not allowing immediate satisfaction keeps the need inside the players mind, the player then feels a need to utilize the system to attempt to obtain what they don’t have or maintain what they achieved.
7. **Unpredictability & Curiosity**: Not allowing the player to know exactly what happens next is a strong drive as the necessity of knowing what isn’t know is core to humanity.

8. **Loss & Avoidance**: This drive is based on the wanting to avoid a negative result, the player doesn't wait to lose the game, nor achieving a result that is closer to losing the game. This can also be combined with the Scarcity drive.

Left Brain / Right Brain Drives

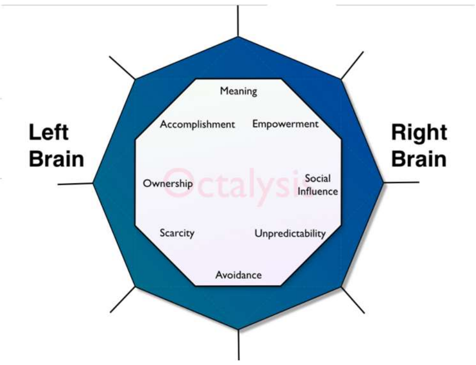


Figure 6 - Octalysis Left vs Right Brain (Chou, 2018)

The drives are divided into Left and Right brain drives, the Left-Brain drives relating to logic, calculations and ownership, generally Extrinsic Motivators and the Right Brain drives relating to creativity, self-expression and social aspects, generally Intrinsic Motivators. Experiences related to Right Brain Drives are generally preferable. Their distribution can be generally seen in Figure 6, but specific areas to which each drive belongs to is vague regarding Meaning and Avoidance (Chou, 2018).

White Hat vs Black Hat Gamification

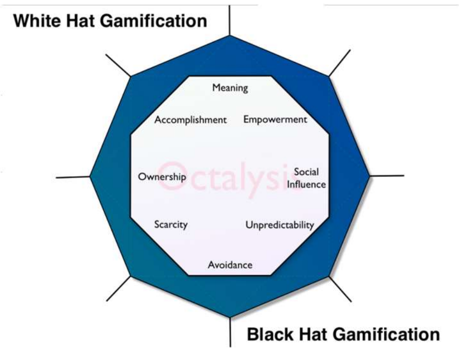


Figure 7 - Octalysis White vs Black Hat

Gamification can be White Hat and Black Hate according to the Octalysis framework, the top set of drives is classified as White Hat, drives based on personal creativity and enjoyment of the usage of the system in general, and the bottom ones as Black Hat, drives based on damaging the player for not coming back for the system with generally unpredictable results. Again, the details are vague regarding Ownership and Social Influence to which set they belong to (Chou, 2018).

Octalysis Score

According to Chou the system can then be classified with the Octalysis score to see the state of gamification, this score is calculated by attributing a value from 0 to 10 to each drive and then summing the square of this values creating an Octalysis score that ranges from 0 to 800 (Chou, 2018).

Octalysis Gamification Levels

There are different levels of the application of the Octalysis Framework (Chou, 2018).

- **Level 1**: Level one is a simple general application of the different drives to a system (Usually gamification stops here)
- **Level 2**: Adds the phases of a Player's Journey (Discovery, On-boarding, Scaffolding and Endgame) creating 4 different approaches for the application of the framework for each stage.
- **Level 3**: On top of the 4 stages add Battles Player Types (Achievers, Explorers, Socializers and Killers) creating a combined set of 16 applications.

Levels up to 5 are mentioned but not publicly presented, therefore details are unknown.

2.2 Insurance

From the early days of humankind that the feeling of safety is one of the basic necessities of the human being, from shelter to tools humans continue to improve on ways to guarantee personal and general safety. When it comes to financial safety, the Phoenicians created an association that would replace the ship to ship-owners that lost a ship at sea and animals lost due to this event, a rudimentary form of insurance. In Ancient Greece, a premium for risk was already calculated for the cargo carried on said ships, utilizing the ships route, the ship's class and the type of cargo being transported. Eventually the Romans assimilated this method and improved upon it.

The first officially recorded insurance contract was made in Italy in 1347, where the insurer took upon itself the risk of the transported cargo in exchange for a set amount of money; this method then expanded towards other European countries, suffering modification along the way. With the English industrial revolution, the insurance sector received a development boost as new and increasingly modern ways of management were introduced. During the 17th century in the coffee shop of Edward Lloyd, where ship captains regularly met, a naval insurance ring-fenced fund[2] was created and would then lead to the creation of the *"Lloyd's of London"* in 1720, as the world's naval insurance. (Gilberto, 2008)

"Insurance is a contract between an individual (the policyholder) and an insurance company. This contract provides that the insurance company will cover some portion of a policyholder's

[2] *Ring-Fenced Funds – "Assets restricted in relation to certain liabilities on a going-concern basis, leading to the restriction of these funds within the business of an undertaking."* (Tower, Frankfurt and Tel, 2015)

loss if the policyholder meets certain conditions stipulated in the insurance contract. The policyholder pays a premium to obtain insurance coverage. If the policyholder experiences a loss covered by insurance, such as a car accident or a house fire, the policyholder files a claim for reimbursement with the insurance company. The policyholder will pay a deductible to cover part of the loss, and the insurance company will pay the rest." (Pareto, 2008b)

Insurance is, essentially, the transfer of risk in exchange for a premium an individual's risk of financial loss is transferred to an insurance company and in the case of a loss the individual can then request a compensation or service from the insurance company. This allows a way to reduce uncertainty, avoiding the possibility of a substantial, potentially devastating, loss in exchange for a relatively small amount by dividing it amongst a larger group. For the entire process to work the concept of mutualisation is essential as it allows the division of the costs by a substantial group; this causes unique and rare risks to have a substantially higher premium cost than other, more generic, coverages. (Associação Portuguesa de Seguradores, 2012)

2.2.1 Insurance Classes

There are two major classes of insurance, Life and Non-Life, each with its own branches. Each of these branches deals with incredibly distinct situations and although they share some terminology and concepts they both have situations only applicable to each specific class.

Life Class
The Life branch simply put deals with the life and death of individuals and consists of life insurances, wedding / birth insurance, unit linked insurance, and capitalization contracts. Unlike the Non-Life branch most of the products in the Life branch can be cancelled by the insurance holder with no need for just cause on their part as normally these kinds of insurance are set with a longevity limit at the point where the insurance holder receives a reimbursement dictated by interest rates and the invested value.(Autoridade de Supervisão de Seguros e Fundos de Pensões, 2015)

Surrender is the act of receiving the provision owned by the insurance provider causing a cease of contract and is usually calculated at the time of the contract stipulation utilizing the dictated premiums. That calculation should be annexed to the policy by the insurance provider. This process is normally started by the policy holder. (Autoridade de Supervisão de Seguros e Fundos de Pensões, 2015)

Reimbursement: at the end of a Life contract the policy holder is reimbursed the value due according to the contract stipulations previously agreed on. (Autoridade de Supervisão de Seguros e Fundos de Pensões, 2015)

Profit Sharing: is the right of every policy older, insured person or beneficiary to receive part of the results generated by the insurance contract; the insurance provider must inform the policy holder of the participation value distributed to them, when the contract ends the policy holder receives the value that is attributed to them but not yet distributed. (Autoridade de Supervisão de Seguros e Fundos de Pensões, 2015)

Reduced paid-up insurance/policy: consists in the lowering of the previously agreed upon reimbursement and surrender values and it can be initiated by the policy holder or the insurance provider normally being caused by unpaid premiums, a new calculation of the values for surrender and reimbursement is now made and annexed to the policy. (Autoridade de Supervisão de Seguros e Fundos de Pensões, 2015)

Lines

Life Insurance

"Life Insurance guarantees, as its main coverage, the risk of death and/or the risk of survival of one or more insured people and may include, as additional coverage, the risk of invalidity, accident or unemployment." (Autoridade de Supervisão de Seguros e Fundos de Pensões, 2015)

The deferent modalities of life insurance function differently. In the case of death risk coverage, the insurance provider pays he value to a previously named beneficiary if the insured person dies during the time period determined in the contract while in the case of survival insurance the beneficiary get payed, not if the insured person dies during the determined time but, if the person still lives when the contract ends, being usually used as saving's accounts; a mix of both these types is also exists where a value gets payed in any of the above situations but utilizing differing values for survival and death. (Autoridade de Supervisão de Seguros e Fundos de Pensões, 2015)

Wedding / Birth Insurance

"Insurances that pay some capital and/or rent in case of wedding or the birth of children." (Autoridade de Supervisão de Seguros e Fundos de Pensões, 2015)

Unit Linked Insurance

"Unit linked insurance consist of life insurance with variable capital in which the value to receive by the beneficiary depends, fully or partially, of a reference value constituted by one or more participation units." (Autoridade de Supervisão de Seguros e Fundos de Pensões, 2015)

In this type of insurance, an individual establishes a unit linked insurance contract and pays a certain premium; this premium is pooled with the premium from other individuals (or not) and then invested, then, after some predetermined amount of time the individual can then liquidize the investment fund and take his profits, losses or even just the original value invested, based on the terms of the contract and how the related investment behaved. (Autoridade de Supervisão de Seguros e Fundos de Pensões, 2015)

Capitalization Contracts

"Capitalization contracts are contracts in which the insurance provider pays a certain predetermined value, after a certain number of years in exchange for a single or periodic premium payment." (Autoridade de Supervisão de Seguros e Fundos de Pensões, 2015)

"Unlike Life Insurance, Capitalization contracts are not connected to a risk related to death or survival of the insured person; upon the operation of funding the contract the insurance provider obligates itself to pay a determined value at the end of the contract independently of any event relating to the life of the policy holder.” (Autoridade de Supervisão de Seguros e Fundos de Pensões, 2015)

Non-Life Class

Non-Life Class insurance is any other type of insurance not present in the Life Class These types of insurance mainly focus on covering property instead of dealing with the life of the insured party; despite this Health Insurance is an insurance line also present in this Class.

Lines

Although the number of insurance lines present in the Non-Life Class is incredibly high some of the major insurance lines, at least in the Portuguese market, are Accidents & Health, Workmen’s Compensation, Health, Fire & Other Damage; Motor; Marine & Transport; Air; Carriage of Goods and General Third-party Liability. As with areas providing the vast majority of the Portuguese national insurance market special distinction must be given to Workmen’s Compensation and Motor Insurance (Autoridade de Supervisão de Seguros e Fundos de Pensões, 2015; Associação Portuguesa de Seguradores, 2017).

Workmen’s Compensation Insurance

Workmen’s Compensation Insurance is mandatory for any employing entity to have as to cover any consequences of accidents suffered by their employees during the execution of their duty; the lack of this type of insurance is, therefore, punished by a heavy fine and in case of an accident by an employee the employing entity is responsible of all payments that would be made by the mandatory insurance coverage (Autoridade de Supervisão de Seguros e Fundos de Pensões, 2015).

This type of insurance covers any contracted workers, in training personnel, administrators, directors, managers and the like that are remunerated for that activity and the service workers of that entity. This insurance line classifies as a work accident any that happen within a wide range of circumstances including, but not limited to, the daily journey of workers and the provision of services outside the workplace dictated or consented by the employing entity. This line provides medical and monetary assistance to the injured person in case of accident (Autoridade de Supervisão de Seguros e Fundos de Pensões, 2015).

Motor Insurance

Motor Insurance is one of the mandatory types of insurance in Portugal, at least when referring to damage to third parties, that accounts for over a third of the national insurance market. Its lack being an infraction punished by law which may include the apprehension of the vehicle, the proprietor of the vehicle being fined and in the case of accident the proprietor being fully responsible for the reimbursement of all damage to the injured party. (Autoridade de Supervisão de Seguros e Fundos de Pensões, 2015).

This insurance must cover at least damage to third parties and the passengers of the insured vehicle, not including the driver; however, optional insurances exists that can be added to the default coverage, namely: an increase of the standard max capital value covered, travel assistance in case for the vehicle and passengers, legal protection during a possible judicial or administrative process and temporary unavailability of the use of the insured vehicle, among others (Autoridade de Supervisão de Seguros e Fundos de Pensões, 2015).

2.2.2 Risk, Premium and Combined Ratio

To understand the workings of the Insurance industry the concepts of Risk and Premium are essential, and, in the case of Non-Life insurance, the concept of Combined Ratio is also important as it allow a metric of evaluating the viability of an insurance product.

Risk
Risk is the chance of an insurance claim being filed with a company by a certain individual and ultimately largely determines the value of the premium paid to the insurance company. Although the actual method of calculating risk is different from company to company they all do it by considering "risk factors". These risk factors are attributes or behaviours of an individual that are perceived or known to cause situations where an insurance claim will be presented to the company. (Associação Portuguesa de Seguradores, 2012) A possible formula is presented below where "RF" represents the risk value of a specific risk factor.

$$Risk = \sum_{i=0}^{n} RFi$$

There are various types of risk when it comes to insurance: preventable, minimizable, avoidable and unforeseeable; preventable risks refer to risks that are easily preventable if basic precautions are taken; minimizable risks are risks that, not being completely preventable, the chances can be reduced if certain actions are taken, or risk increasing behaviours are avoided; avoidable risks are those that are completely avoidable not requiring action or lack thereof to be necessary; finally unforeseeable risks are those who are not apparent, or impossible to prevent against. (Pareto, 2008a) As such the risks generally considered when calculating the total risk of a client are the preventable, minimizable and avoidable types.

Finally, it is important to mention that risk should be random as it is impossible to insure something that is guaranteed to happen as this make risk transfer completely impossible, as such insurance claims generated from intentional damage of the insured party or a subcontractor of theirs do not fall within the scope of the claim.

Premium
The premium is the value that the client pays to the insurance company for their adhered policy to a specific insurance product. This value is calculated in a way to reasonably be able to respond to any claims that are made during the insurance contract with a margin of profit to the insurance company to maintain the company's long-term viability. This value is generally based on the factor of the probable value of loss in a claim, Claim Value, multiplied by the chances of a claim occurring, adding on to the result the expenditures of the company in managing the insurance product, a safety margin for the event that an unusually high number of claims does not completely empty the fund for claim response and a profit margin for the purpose stated

previously. (Associação Portuguesa de Seguradores, 2012) This calculation can be presented as follows:

$$Premium = (Claim\,Value\ x\ Risk) + Expenditure + Profit\ Margin + Security\ Margin$$

To compete with each other in terms of premium a company can change some values of this formula; the Claim Value is not controlled by the company itself as such it is not malleable, when it comes to Risk some change can be made, namely having a higher risk tolerance would let a company lower the cost of premiums by a certain value, Expenditure must be dealt with internally by optimizing the processes to manage insurance contracts; however, this becomes increasingly difficult.; the Profit Margin and Security Margin are the most malleable parts of the premium calculation, this however does pose some more concerns to the company as lowering the Security Margin may cause problems due to the possibility of a large group of claims possibly heavily lowering the purse for that type, the Profit Margin, if too low can also slow down the development of the insurance company.

Combined Ratio

Determining that the premiums are in an acceptable value compared to the claims filed with the company is of vital importance; one of the ways to determine this in non-life insurance is the Combined Ratio. This measure is calculated utilizing the value of the premiums collected and comparing it to the total expenses and the costs of paying off claims, the ratio may then be used to determine where to cut costs or if the premium itself should be raised; if the Combined ratio is below 100% the premium is enough to cover all expenses and claim payments, being lucrative to the company, otherwise if the Combined Ratio reveals a value higher than 100% the expenses outweigh the gains presenting a loss for the company. (Associação Portuguesa de Seguradores, 2012) A possible formula for the combined ratio can be seen bellow.

$$Combined\ Ratio = \frac{Expenses + Claim\ Costs}{Gross\ Written\ Premium}$$

2.3 Gamification in Insurance

Gamification is a broad reaching topic and, if taken far enough, almost any aspect of life may be seen as being gamified due to the fact that game or game like elements are present in practically everything from education to work passing through government systems and class based societies; as such, attempting to detect all such uses, even limiting it to only insurance industry contexts, would be a prohibitively extensive task, as such seven main opportunity areas are mentioned which are be detailed in section 3.1.1:

- Sales Enhancement
- Product Targeting
- Collaboration Promoting
- Internal Knowledge Dissemination
- Self-Risk Management Encouragement
- Client Self-Service Encouragement
- Gamified Insurance Product Creation

2.3.1 Current uses of Gamification in Insurance

Although its spread is not far reaching, Gamification has started to make its way into the insurance market. The clearest examples can be seen in North America and in the United Kingdom. In the Portuguese market gamification is only present in a very limited manner, even stretching the concept's definition to the limit.

This list is limited to the usage of Gamification in interactions with the user as its existence in internal areas is extremely transversal being applicable to almost any business in the same basic way.

Aviva Drive Challenge
The Aviva insurance company released a "nifty (and free) app that monitors your driving skills" (Aviva, 2017) for policy holders who pay more than £200. This app monitors the user's driving for a journey selected by the users and rates user's skill from 0 to 10 after 200 miles, providing up to 28% discount in premiums over £400 but also rewarding the user with a multitude of badges. (Aviva, 2017)

John Hancock Term with Vitality
John Hancock insurance teamed up with Vitality to create a product that gives users a way to monitor and improve their health situations utilizing a system of personal goals, rewarding users Vitality Points as they complete tasks and providing them with premium discounts and a variety of benefits and discounts in other areas. (John Hancock Insurance, 2017)

Ok! Teleseguros OK! GPS
Ok! GPS by Ok! Teleseguros is the rare exception of an insurance policy with some game-like elements present in the national market, at least in an overt way. This policy requires the user to place a device in the vehicle that records some parameters of the usage of the vehicle that are then used to calculate discounts on the policy renewal. This can be perceived slightly as gamification as the defined margins on the parameters provide a goal for the policy holder to reach, but this information is not given to the policy holder on a regular basis which limits his capability of targeting the presented margins accurately. Therefore, while it may be classified as a gamified solution, this requires a stretch of the concept to its extreme. (Ok! Teleseguros, 2017)

2.3.2 Gamification in related areas

Although not inherently tied to insurance policies some existent gamification approaches may be utilized by insurance companies to track customers life habits.

Wellness / fitness is a very prominent area when it comes to gamification which can be attached to an insurance product in the health area with relative ease. Gamified systems such as Fitbit (Fitbit Inc., 2018) and CaféWell (Welltok Inc., 2018) already collect a high amount of data from the user and provide a gamification approach, emulating or adapting one of these systems and utilizing the information collected for a health insurance policy.

Fitbit
Fitbit rests upon a physical product, the Fitbit smartwatch, this watch offers health tracking capabilities, able to obtain a vast amount of health information, including weight, activity, exercise, sleep and food habits, providing rewards in terms of achievement badges and progress notifications. It also provides a competitive and social aspect by utilizing leaderboards to compare players with family and friends and allowing users to issue each other challenges, cheers and taunts. Fitbit also possesses integration capabilities with a multitude of other fitness applications (Fitbit Inc., 2018).

CaféWell
As mentioned, CaféWell's availability is locked behind a reference by an employer, hospital or health provider. The CaféWell system focuses on allowing the user to determine their health goals, providing personalized goals for each user, having them complete them for monetary rewards, suggesting methods of increasing wellness and allowing real-time tracking of personal progress (Welltok Inc., 2018).

2.4 Existent Gamification Enabling Solutions

A few solutions to provide aid and ease in gamifying systems already exist in the marketplace, two of which will be detailed in terms of publicly available information, Badgeville and Captain UP.

2.4.1 Badgeville

Badgeville claims to be the leader in enterprise gamification and digital motivation. Badgeville offers a platform to their clients that can be utilized for a multitude of solutions, this product is called Badgeville Enterprise Plus and can be extended by two additional products, GameViews and MotivationMetrics^{TM}.

Badgeville Enterprise Plus
The Badgeville Enterprise Plus product serves as the base platform to implement gamification approaches and is composed of 8 modules:

1. Rules Engine: Allows for the inclusion of complex rules to attribute rewards and work with platform data by combining sets of events within the system, it also allows connection with events from exterior systems.
2. Engagement Engine: Serves as a method of monitoring engagement at multiple levels and adjusting rewards to maximise it.
3. Guided Experience: Utilized to attribute sets of tasks or "missions" to users that need to be performed to accomplish a task also defining priorities between these tasks.
4. Advocacy Engine: Gives enterprises a way to reward advocacy by the target group utilizing their contributions to various social networks.
5. Impact Engine: Serves as a way to track the impact of users have on others allowing the promotion of behaviour which provides a positive impact.
6. Reputation Engine: Is utilized to attribute users with visible rewards of their activity, accomplishments and hit milestones to allow for an identity within the system to be created.
7. Behaviour Library: A set of pre-configured common behaviours collected from multiple deployments of Badgeville is available to allow for quicker creation of gamification approaches.
8. Integrated Gamification: The platform offers an API, application connectors and blueprints to quickly allow the addition of gamification to existing system.

GameViews

GameViews is an extension to Badgeville Enterprise Plus that allows the visualisation of user's reputation engine standings, it also allows the presentation of various leaderboards that provide a ranking between user performance.

MotivationMetrics^{TM}

MotivationMetrics is another extension to the Badgeville Enterprise Plus that offers metrics on multiple factors of the system that can be used to create information to be presented in user dashboards or for analysis of the gamification system's success metrics and user behaviour.

2.4.2 Captain UP

Capitain UP offers a set of features divided into four classes Gamification, Social, Communication and Back Office, details on these features is however not clear. The features enumerated by each class are as can be seen in From this grouping, which can be seen in

Table 8, we can assume that the Gamification portion includes all gamification elements available, Social the ways utilized to increase the social and ownership aspects of the applications, Communication available systems to communicate with the users of the application and Back Office, information gathering, user administration and gamification configuration (Capitain Up, 2018).

Table 8 - Captain UP Features/ Classes (adapted from Capitain Up, 2018).

Gamification	Badges, Levels, Trophies, Points, Multi-currency, Rewards
Social	User Profile, User Area, Activity Feed, Share, Leaderboards, Community
Communication	Inbox, Segmented Messages, Notifications, Customized Pop-ups, Bulk Messages, Welcome Message
Back Office	Game Mechanics, User Management, A/B Testing, Rule Engine, Segmentation, Insights & Analitics

3 Gamification Design

Chapter Contents

3.1 Gamification Idea Genesis and Selection

To decide where the utilization of gamification would be useful a set of ideas for the usage of gamification in insurance were determined and then, subsequently evaluated and selected. The determination of the ideas being determined by a general study of the insurance market and the processes of the industry as well as resorting to some documents that made this analysis previously. Then, utilizing a weighted comparative matrix the ideas were evaluated and one was selected to continue development.

3.1.1 Ideas

Being limited to the Insurance market and processes a few possible targets for gamification were detected during the study of the market and in discussion with experts at i2S. The following areas for possible gamification identified:

- Sales Enhancement
- Product Targeting
- Collaboration Promotion
- Internal Knowledge Dissemination
- Self-Management of Risk
- Client Self-Service
- Creation of New Gamified Products

Sales Enhancement
Gamification may be used in areas related to the sale of insurance products, this can be done by providing goals and friendly competition between the salespeople employed at an insurance company. This has the particularity of being expandable to areas such as support infrastructure and client support with relatively low difficulty and cost as most methods are easily transferable to different contexts. The main issue with this area is the ease of creation of a system perceived as unfair or discriminatory, possibly lowering the performance of the elements that do not feel as though they have a chance of competing.

Product Targeting
The usage of gamification when it comes to product targeting offers the possibility of creating a gamified system to engage users and determine groups for a new line of insurance is a distinct possibility; the same process can also be used to help suggest to the client the best product to suit their needs. This was partially adapted from Chatterjee, Pathak and Kumar, 2017.

Collaboration Promotion
Collaboration can be useful in the insurance context by utilizing a system not unlike those existent in websites such as StackOverflow, in which users reward other users by other users on the validity and accuracy of their answer to an asked question. In insurance a possible utilization is the exchange of information between underwriters, but it can also have possible

uses related to facilitating the training of new collaborators in all areas by providing the possibility of finding previous answers to a problem they may be having or the ability to pose their own question for it to be answered by someone who knows the answer quickly.

Internal Knowledge Dissemination

The insurance industry deals with a wide range of products, a high amount of legislation and processes protocols, security or otherwise, that must be adhered to for its proper functioning and disseminating this knowledge internally is a possible issue, as the process of doing so is usually tedious and has associated costs for the company, as such the possible use of a gamified system to promote internal knowledge of product and procedure can be detected. This was partially adapted from Chatterjee, Pathak and Kumar, 2017.

Self-Management of Risk

This use of gamification is already in use, especially when referring to health risks, even in ways not connected to insurance contexts, and, in the case of health, it provides a way to manage a client's risk when it comes to health and life insurance by utilizing sets of goals and tasks in an effort to improve their health situations, at the same time lowering risk and improving the well-being of the customer utilizing the system. Examples of this use can be seen in section 2.3. This can then also be applied to the context of other insurance services such as motor insurance by providing the clients with goals and incentives to improve their driving and lowering their risk of accident, among others.

Client Self-Service

Gamification can be used in aiding customers, and potential customers, to make more informed decisions on which products to buy, improving their knowledge on the process of reporting changes of their risk status and even with the reporting of claims, lowering the costs of the company in dealing with unknowledgeable customers and providing customers with a deeper engagement with the company. This was partially adapted from Chatterjee, Pathak and Kumar, 2017.

Creation of New Gamified Insurance Products

A possibility presented itself of maybe creating new insurance products with gamification elements woven into them from the start, this can also be seen as an extension of the gamification of the Client-Side Insurance Risk Management aspect presented previously as the utilization of telematics data in the creation of the insurance product is an enabler of the process of gamification, this point however states that the gamification system could be, not an addition on top of an insurance product, but woven into the design of the product itself.

3.1.2 Idea Comparison

For the comparison of the ideas presented previously a meeting was held and they were evaluated on a weighted comparative table, Table 9, to verify which options would be the most applicable for the purpose of the project. For this comparison some criteria were determined beforehand: The criteria for the selection of the area of application were determined via meeting with a group of specialists at i2S and are:

- Promotion of proximity of the Client with the Insurance Company
- New Customer Acquisition/Maintenance
- Dependence on previous information
- Value Potential for the Insurance Company
- Innovation Level for i2S
- Ease of Integration with i2S solutions
- i2S Core Independence
- Market Potential of i2S
- Insurance Company Marketing Potential
- Ease of Development
- Insurance Sector Relevance

Table 9 – Idea Weighted Evaluation Matrix

	Weight	I1-SE	I2-PT	I3-CP	I4-KD	I5-MR	I6-SS	I7-NP
Promotion of proximity of the Client with the Insurance Company	5%	1	3	1	1	5	3	3
New Customer Acquisition/Maintenance	5%	1	4	1	1	5	4	2
Dependence on previous information	5%	5	5	5	5	2	3	1
Value Potential for the Insurance Company	5%	3	5	2	3	5	3	4
Innovation Level for i2S	10%	2	5	1	3	4	3	4
Ease of Integration with i2S solutions	15%	3	1	1	1	3	1	2
i2S Core Independence	10%	4	2	5	4	2	3	2
Market Potential	10%	2	5	2	4	5	3	2
Insurance Company Marketing Potential	10%	1	5	1	1	4	2	4
Ease of Development	20%	3	2	4	4	3	3	1
Insurance Sector Relevance	5%	3	5	2	2	4	2	3
Final Classification	100%	2.6	3.35	2.4	2.75	3.65	2.6	2.35

I1-SE	**S**ales **E**nhancement	**I5-MR**	Self-**M**anagement **R**isk
I2-PT	**P**roduct **T**argeting	**I6-SS**	Client **S**elf-**S**ervice
I3-CP	**C**ollaboration **P**romotion	**I7-NP**	Creation of **N**ew Gamified Insurance **P**roducts
I4-KD	Internal **K**nowledge **D**issemination		

Progress of the Discussion Meeting

From the initial presentation of the ideas it was obvious that the ideas *Sales Enhancement*, *Collaboration Promotion, Internal Knowledge Dissemination* were not going to be chosen due to an extremely broad area of application and the fact that other solutions in areas of *Sales*

Enhancement and *Collaboration Promotion* were already widely available, some even for free, *Internal Knowledge Dissemination* although interesting did not seem to give much value to the client (Insurance Company) in opposition to the other ideas.

After that *Creation of New Gamified Insurance Products* was also excluded due to the prohibitive difficulty of development and necessity of extensive market studies to properly implement, combined with the perceived lack of willingness of insurance companies to make radical changes and the necessity of extensive amounts of information required to configure and personalized the products created by this idea.

Lastly *Client Self-Service Encouragement* was excluded as it not only required a new interface with the end customer (insured person) to be developed as it is currently inexistent, the need to develop an entirely new system for the support of this feature, as well as the relatively average marketability and value to the client.

Leaving then ideas *Product Targeting* and *Self-Management of Risk* as the main candidates for development, with comparable scores; this choice was then delegated to the author of this report opting for *Self-Management of Risk* based on the results of the comparison table (Table 9), personal preference and perception of being the most aligned idea with the original goal of the project.

3.2 Conceptualization of Self-Management of Risk

Utilizing Deterding's Lens of Intrinsic Skill Atoms framework for gamification design, the chosen idea, *Self-Management of Risk*, for the system was developed into a more mature concept. However, for purposes of ease of analysis a general concept description was drafted to allow for a basis for the first discussion. The ideas proposed on the general concept description were validated and the process of taking the selected gamification method and applying it to the system was executed.

3.2.1 Initial Concept

In this Gamification Approach the user is shown a screen with a list of his policies, which are named and flagged according to the type of policy. This screen contains only basic information about the policy, a risk level ranging from None to Extreme (Table 10) based on a percentage system, a themed name of the policy, the type of policy it is (represented by an icon) and the premium being paid for that policy along with the time period of payments.

Upon entering the screen for a specific insurance type the user is presented with the risk level, the risk factors, how each risk factor is a affecting them, and, possibly an evolution of the premium they are paying and reasons for its decrease or increase. On each specific risk an option is present that shows how to lower the risk of that factor. The user then is given ways of combating those risks such as filing documents proving health living and work safety reports., that are they analysed by the insurance company and the risk reanalysed.

Table 10 - Risk Levels Name, Range and Colours

Name	Risk Percentage	Colour
None	0%	Not Applicable (Bar does not exist)
Very Low	1% to 10%	Light Blue
Low	11% to 30%	Blue
Medium	31% to 60%	Green
High	61% to 90%	Red
Very High	91% to 99%	Dark Red
Extreme	100%	Black

Telemetric based insurances would give the users a set of information about the data collected by the sensors and allows them to set personal goals, these goals then reward the user in a small way when completed.

The user is also compared with other users utilizing the system in a variety of areas and is rewarded with badges based on his performance. Upon receiving a badge the user may then share it in various social media.

Reward System

For this system three groups of rewards were planned, was subject to change if necessary during the application of the Gamification method and that is the stage where the gamification elements are supposed to be determined; however, for the purposes of illustrating the functioning of the system it was deemed necessary for a rudimentary description of the reward system to be created, monetary rewards, regarding the paid premium as this could evolve by due to the changes in risk factors; badges, given out in a broad manner for various achievements in the system, these amounts would be limited in a way not to devalue them; and other rewards provided by the insurance companies such as discounts on other products or some sort of prize.

Major Limitations

The major limiting factors of the proposed concept are the lack of leeway regarding the premium values, and the difficulty of engaging the player over long timespans.

The national markets premium prices are already very low for the most widespread insurance lines so the lowering of premiums is very difficult, for this system to work a slight increase of the prices of insurance policies is a distinct possibility which would allow additional range to the monetary rewards offered.

The difficulty of creating an engagement loop over the long timespans that some lines possess is also of prime importance. Timespans of months or years between necessary interactions will cause the system to be forgotten by the user, a method to combat these high timespans needs to be implemented.

3.2.2 Gamification Using Lens of Intrinsic Atoms

The idea selected, and the initial concept of the system determined the process of applying Deterding's gamification method is ready to start. In this section the method will be applied, one step at a time. First determining the key activities of the system, then the insentient challenge and motivations to these activities and finally the development of ideas to promote the activities based on their perceived challenges and motivational factors.

Strategy
The target outcomes of this proposed system are the reduction of risk in clients of the insurance companies, measured by the results of the risk calculation, and the increase of communications between the client and the insurance company, measured by total number of communications in a year.

This system's target audience is the people who possess insurance policies, this can be segmented however into three groups:

- Users with only dynamic-risk insurance policies (DPH: Dynamic-Risk Policy Holder)
- Users with only regular insurance policies (RPH: Regular Policy Holder)
- Users with both types of policy (MPH: Multi-type Policy Holder).

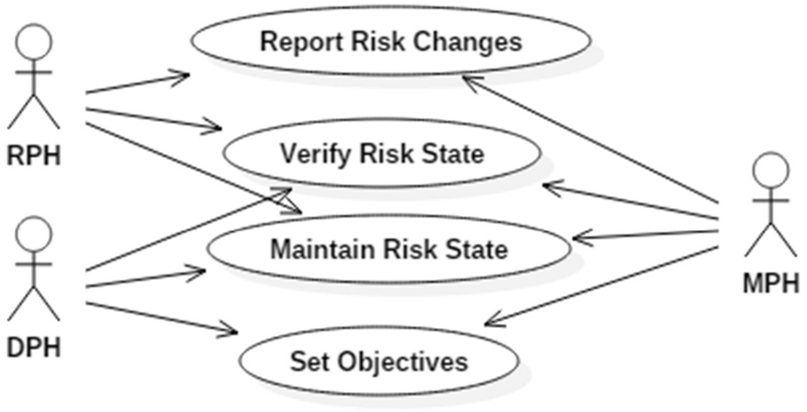


Figure 8 – System Core Activities

As can be seen in Figure 8, Multi-policy holders can execute any of these activities however members in Dynamic-Risk Policy Holders cannot execute "Report Risk Changes" as this is an automatic process for them and Regular Policy Holders cannot execute "Define Objectives for a Risk Factor" as this process is limited to dynamic risks as non-telematics data is rarely changed.

Unfortunately, as this system does not yet exist first person information on usage is unavailable for this first concept, therefore these tasks were deduced based on the system concept presented in 3.2.1.

The main constraints detected for this system are: information availability and consistency; difficulty of propagation of this system into the end user; flexibility of the system to adapt to different insurance lines and, possibly, classes; legal issues regarding the utilization of end user information; limited timeframe of approximately 4 months for design and development with limited manpower for the prototype.

Research

Since information is lacking for the research section, suppositions have been made to provide an answer to the requirements of this stage of the method. For each activity a profile, detailing goals, evaluation metrics, behaviour chains, motivations and hurdles were created. This was made for the major activities of the proposed system, all others being subservient to the selected ones. These activity profiles can be seen in Table 11, Table 12, Table 13 and Table 14.

Table 11 – Verify Risk State Activity Profile

Activity:	Verify Risk State
System owner goal:	Frequent risk monitoring
Metric:	Number of days/month users consult the system
User Need:	Maintain vigilance over their risk details
Behaviour chain:	Look at presented General Risk values.
Motivators:	Wish to keep track of their risk state (Autonomy)
Hurdles:	The evolution of the risk in some insurance lines takes a significant amount of time to happen as such the need for this regular verification is low in such cases.(Boredom)

Table 12 – Report Risk Changes Activity Profile

Activity:	Report Risk Changes
System owner goal:	Obtain information about policy holders
Metric:	Number of valid reports sent
User Need:	Inform company of mitigating factors
Behaviour chain:	Select specific insurance policy > Select risk factor > Select Report > Fill out requirements > Send report
Motivators:	Potential rewards in premium (Monetary)
Hurdles:	Inability to understand/complete the reporting process (Incompetence)

Table 13 – Define Objectives for a Risk Factor Activity Profile

Activity:	Define Objectives for a Risk Factor
System owner goal:	Increase user self-management
Metric:	Number of goals set/achieved
User Need:	Determine personal, auto-imposed goals
Behaviour chain:	Select Policy > Select Risk Factor > Determine Goal and Timeframe
Motivators:	Goal Setting and Completion (Competence / Autonomy)
Hurdles:	Possibility of Failing Goals, Overestimating capabilities (Incompetence)

Table 14 – Improve/Maintain Risk State Activity Profile

Activity:	Improve/Maintain Risk State
System owner goal:	Mitigate the costs of covering claims
Metric:	Risk state value
User Need:	Increase self-security
Behaviour chain:	Verify Risk Factor > Resolve to Improve/Maintain It > Act on Resolve
Motivators:	Sensation of capability to maintain safety (Competence)
Hurdles:	Fear of sudden increase/demotivation due to unavoidable increase (Impotence) Inability to maintain stable states (Incompetence)

Synthesis

The activities the give result to a set of triplets to be used for the ideation stage. In Table 15 the triplets for the defined activities can be seen.

Table 15 – Activity Triplets

Activity	Challenge	Motive	
		Positive	Negative
Verify Risk State	Maintaining a habit of checking risk state over a long period of time	Autonomy	Boredom
Report Risk Changes	Reporting correct information in the proper way	Monetary	Incompetence
Define Objectives for a Risk Factor	Creating reasonably and constant goals	Competence Autonomy	Incompetence
Improve/Maintain Risk State	Maintaining reasonably risk levels or reducing them	Competence	Impotence Incompetence

Ideation

Utilizing the defined triplets' methods of gamifying these methods would then be generated, this however does not fit very well as during this development process an initial concept was proposed that already utilized gamification approaches. Therefor the ideation stage shall be used to not only attempt to catch details missed in the initial concept but also to justify choices made during that stage.

Verify Risk State

As can be seen in the previous section the issue regarding task A1-VGS is mainly the unwillingness of the user to return to the application to verify their risk state. While long periods of time (months) are an acceptable frequency for non-dynamic risk policy holders the target situation is to have the user visit the system at least once a month, ideally even once a week. As such it is necessary to incentivize the user to return to the application to verify the risk state on a regular basis. The question to be answered as "How can the user be incentivized into returning on a regular basis?".

A possible method is the one utilized by the leaning platform "Duolingo" (*Duolingo*, 2018) which provides you something called "Streaks" each time you complete your daily goal. This cannot be directly applied into the proposed system as the concept of daily goals in not exactly present. The simplest way to apply this to the system is its utilization to reward frequent returning, proposing a similar system to "Duolingo" and providing a reward for the returning of the user on a determined basis, let us call it "Watcher Points", his would be configurable for each type of policy, if multiple policies are present this value would be the lowest value present in the existing policies.

This would then be fortified by providing the users with rewards regarding the increase of their "Watcher Points"; these rewards being badges, title changes or some other type of reward that can be determined.

Report Risk Changes

Possibly the most important activity of this system this is only present in non-dynamic risks as those are automatically sent to the insurance company. This point is a complex one and cannot be tackled only by the usage of gamification, information about the risk factor must be present between the risk factor and the risk reporting area, even possibly being present in the same place. The question to answer is not however regarding the information presence but the correctness and the providing of incentive to the reporting process. The question being "How can the reporting of risk changes in an accurate way be incentivized?".

The difficulty of gamifying this point is that the method of reporting the risk changes can be different from policy to policy therefore an "in report process" approach has to be light, not even mentioning the fact that most of this processes rest in the simple sending of a document. Never the less the utilization of a reward system is again necessary, consisting of badges only would be a limited approach, not focusing on the motive detected in the Research and Synthesis steps this method; the user has to be rewarded with both feedback about the report they have filed and also provide them with a monetary reward, even if negligible.

When it comes to the information distribution about the correct procedure for change reporting, reasons to reduce the risk factor and possible rewards, this does not appear to require gamification, and it should be presented between the risk factor and the reporting process during system navigation, regular accessing of his information seems unnecessary to incentivize as it is only relevant for users already being incentivized to report the risk changes.

Define Objectives for a Risk Factor

Definition of objectives is possibly the easiest method to gamify as some of the gamification work is done by the users themselves and part of the reward is intrinsic in the way that the user is obtaining a sense of accomplishment by achieving the self-set goals. The motivational need therefore lies upon the incentive to set goals, set them regularly and set them reasonably. The question being presented therefore is "How can the user be motivated to set more and better goals for themselves?".

Firstly a number of goals set, goals completed and a completion percentage should be present in the area where goals are set and their status verified, this provides the user with valuable feedback to the quality of their goal setting, this can be however changed to just the goals completed, however it does not completely inform the user of their goal setting quality, this information can be, however; divided by some timescale (yearly, monthly, weekly) that can be used to track the users goal setting quality over time, also not letting the user be stuck in a state where the coveted hundred percent completion is impossible, this also serves the promotion of the user coming back and setting more goals, by instilling a mind-set of "I want to get 100% in this timescale." or "I want to reach 100 goals completed!"; that are reinforced by the inclusion of badges for the completion of these goals; a minimum threshold of goals for classification in a timescale should be set, as its lack would make the completion of a single, extremely simple goal a hundred percent completion ratio for the timescale. This can also be utilized by a leaderboard system where users are ranked against each other in the aspects mentioned before.

Improve/Maintain Risk State

The core activity of the entire system the usage of gamification in this activity rests on enhancing existing intrinsic motivations of having low risk levels, the question present here is "How can the need of having a low risk for users be enhanced?".

This activity, being core to the system, requires special care and special methods. Firstly the presented value should not be seen as "Risk" but as "Safety", the value going from a representation of "Unsafe" to "Fortified" as an example. Making the change in risk a positive outcome instead of a mitigation outcome, reasonable ranges need to be defined for these factors however. The other part of this is the feedback of what effect changes in risk have in the premium calculation, this must however be tested in prototype as the discrimination of these values may heard the user to less important risk factors in punctual cases, giving the user a clear idea in the impact of their actions, increasing the feeling of competence and power over their situations. This can also be classified in the communal leaderboards referenced in A3-DOF and be rewarded in the same way.

Again, the utilization of a rewards system to complement the intrinsic value is necessary and as such benefits should be given out on risk lowering, the utilization of badges and titles are cost effective means but again this should be followed by a monetary reward, even if negligible, as noted previously when mentioning premium calculations.

Prototyping
Prototyping will be completely described in the rest of this document as the goal is to create just that. Due to the lack of pre-existent data and customer groups, as well as time and resource constraints, low level prototyping will be skipped, and a complete system prototype will be developed instead which represents the bulk of the following sections of this document.

3.2.3 The Safe Driver Agency

An important part of Gamification is to provide a theme to the game being designed, for this purpose the initial idea of Self-Management of Risk was trimmed down to a single type of insurance, one that had the dynamism necessary to gamify, telematics-based automobile insurance.

Being one of the most common insurance policies due to its mandatory nature automobile insurance is extremely wide-spread, and the, relatively, recent addition of telematics to this type of insurance presents a clear opportunity of gamification, allowing the user to both obtain clear data about their performance, allowing visibility of improvement and improving transparency of the benefits of this improvement.

For this area the "Safe Driver Agency" idea was put in motion, utilizing not only the previous determined information utilizing the Lens of Intrinsic Atoms method but also acquiring some help from general game design lens from the book "The Art of Game Design" (Jesse Schell, 2008) and guidance from the 8 Core Drives in the Octalysis Framework to attempt to reach a better solution. Details on the answer given to the lens deemed applicable can be seen in annex G.

Theme

The first part of the design of the Safe Driver Agency was to generally define the theme of the Agency. This was done by deciding on the general style of the system, terminology used and general mechanical approach. A simplistic logo for the theme was also drawn for usage the UI and can be seen in Figure 9.

Figure 9 - Safe Driver Agency Logo

For the Safe Driver Agency all users are Agents, classified with an Agent Rank(Or Level), which have to accomplish operations(challenges) given by the Agency in an effort to combat unsafe driving behaviour in their town and, ultimately, country.

The drivers are to go along their normal lives and be classified on their driving skill according to parameters collected during the voyage by a previously installed telematics system in their vehicle.

This thematic approach was done to influence Core Drive 1 from the Octalysis framework, Epic Meaning & Calling by attempting to provide a deeper sense of purpose to the player when using the system, thinking themselves not only as one more policy holder but as a Special Agent tasked and trusted with an important meaning. Impact on Core Drive 5 Social Influence may also be considered as the actions within the system can be perceived as having influence on society, although not recognized by peers, recognized by the game itself.

Reinforcing Aspects

A few details of the system were chosen in order to enforce this theme, some already mentioned previously.

Names

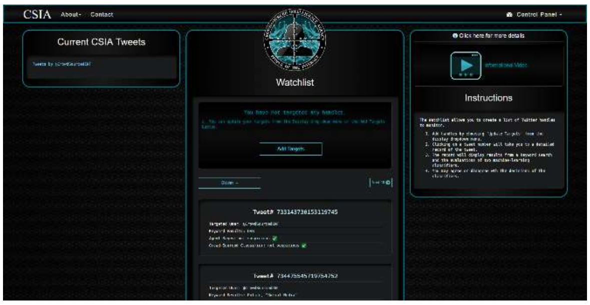
The names of various elements are to be set to represent the overall theme, a few such examples are Missions whose name is to be changed to Operations to simulate sets of operation ran by the Agent, obviously the utilization of Agent in all appropriate areas, Agent Tavares, when the user is referring to the author.

The Overseer

The Overseer is a hypothetical AI that runs the Agency, its usage is as a character in the system that provides advice and distributes Operations to the Agents, this is used to simulate that the Agency is secret so the Overseer is the only one that knows the full extent of the Agency and what Operations are being ran.

Control Panel Design

The Agent Control Panel's design including controls and information presentation style need to be themed to reinforce that it should be something hidden and not available to all, proper design of this is needed to allow for the user to feel immersed in the experience, as such an interface that holds a theme similar to the CSIA website (CSIA, 2015) which can be seen in Figure 10 would is the goal.



You have not targeted any handles.

The Watchlist allows you to create a list of Twitter handles to monitor.

1. Add handles by choosing 'Update Targets' from the display dropdown menu.
2. Clicking on a tweet number will take you to a detailed record of the tweet.
3. The record will display results from a keyword search and the evaluations of two machine-learning classifiers.
4. You may agree or disagree with the decisions of the classifiers.

Tweet# 733143738153119745

Tweet# 734475545719754752

Figure 10 – CSIA Website (CSIA, 2015)

Operations

To provide the players with short term and long term goals a players are attributed operations by the Overseer. These operations are determined by their Agent Rank, Attributed Score and the stage of the process they are in. This is done so Operations can be targeted to the proficiency of a player as well as allowing guidance to be given to the user during critical times.

Three types of Operations are determined: Period based(Weekly with max of 3, Monthly with max of 2 and Yearly with max of 1) operations that are renewed every time period as the player completes them, they rise in difficulty and reward according to the period and player performance, weekly being the easiest to achieve and yearly the hardest, these operations are replaced every period; Information Operations, this type of operation is given to the players in two cases, to aid the On-boarding process by being used as a Tutorial of how to utilize the Control Panel and to inform the player that it is necessary to update information in their personal page when the company needs it; and finally Challenge Operations, these Operations work much like Period Based, but they are selected by the user or given automatically and pertain to a single collected parameter, are used to allow the player to improve their performance in the parameter, but they can also attributed in the case of a significant drop in performance of an area. As players complete operations they receive both Experience points, used to increase their Agent Rank, and Reputation points, used to claim rewards from the Provision Store.

This tackles Core Drive 2 Development and Accomplishment, by only attributing the rewards by completion of a change tailored for the player's skill and performance, ideally allowing for a smooth curve of challenge that also incentivizes and rewards skill development. As Periodic operations are also reset every period Core Drive 8 Loss and Avoidance is also impacted as if the player does not complete them within the designated timeframe they are gone forever. The Limited amount of operations present and any time, specifically periodical may also contribute to Core Drive 6: Scarcity and Impatience as only a limited amount of missions may be completed within that period and a ceiling of experience and reputation points is shown.

Global Operation

As mentioned before the objective of the Agents in the Safe Driver Agency is to improve driving safety in their area, city and ultimately country, this is called the Global Operation, and serves to give users a perception of their impact.

This global operation possesses its own screen where various data is presented, but the most important detail is a map centred on the player's town that shows the risk level of areas in a colour coded scheme with the following range blue, green, yellow, red, from safest to most dangerous. This map would be update as regularly as possible and provide percentages on the risk level change in the areas. Players are also rewarded based on their impact to lower the risk in zones.

The Global Operation can tackle Chore Drive 1 and Core Drive 5 the most as it both gives the perception of the Epic Meaning by showing player effects on the overall world and having the application recognize the player's effect on causing this change. It also may affect Core Drive 7 Curiosity and Unpredictability as the effects of other players in the risk is not public to the user and the change effect is only shown on a periodic basis it cannot be predicted.

Agent Rank

As agents complete Operations their Agent Rank increases, each rank is represented by a medal that can be seen at all times while using the control panel and a designation, a list of these designations and how many levels exist within each one can be seen in Table 16. Agent rank is utilized to calculate Operation difficulty and rewards and a simple measure of progress within the system.

As with Operations this serves to increase the effect of Core Drive 2 Development and Accomplishment, however the Agent Rank is something personal to the player and something they have achieved, having a slight impact in Core Drive 4 Ownership and Possession.

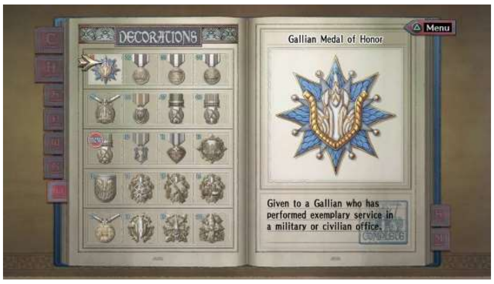
Table 16 – Agent Rank Designation and Levels

Rank Designation	Levels
Trainee	1
Rookie Agent	5
Special Agent	5
Adept Special Agent	5
Senior Special Agent	5
Master Special Agent	5
Elite Special Agent	Infinite

Ribbons

As players achieve milestones in the Safe Driver Agency system they are rewarded with Ribbons that serve as a marking of their capabilities and proficiency. These provide no other effect than decorative and should be displayed in a showcase allowing the user to see their achievements in display. Videogames regularly take this approach, and medals are themed according to the type for game, in the case of the Safe Driver Agency the ribbon design can be militarized as secret agencies generally are. A good example to serve as the base for the showcase in the Safe Driver Agency can be taken from the game Valkyria Chronicles created by SEGA (Sega, 2008) (Figure 11) which displays medals earned during gameplay in a book page in general terms but can be selected for details on why they are earned, for the purpose of the Safe Driver Agency the book detail can be omitted but keeping the general outline and idea., other games such as Star Wars: Battlefront II have a slight variation of this, unlocking instead 3d images on a virtual Diorama according to player achievements.

Ribbons' main goal is to again show a player's progress and accomplishments within the system therefor it falls on Core Drive 2 Development and Accomplishment yet again, but as agent rank they are also personal to the player and can be seen as impacting Core Drive 4 Ownership and Possession.



Given to a Gallian who has performed exemplary service in a military or civilian office.

Figure 11 - Valkyria Chronicles Decorations (Sega, 2008)

Provision Store

The Provision Store is where the player can exchange their earned currency to Upgrade their Agency Vehicle or obtain equipment to aid in their Operations, at least thematically. Effectively the Provision Store allows the player to exchange earned reputation points for extensions to their policy themed as upgrades to the Agency Vehicle, the insured vehicle. For instance there are two extensions to your window protection, 1000€ and 2000€ coverage, these would be named Enhanced Windows, and Reinforced Windows, proper information needs to be explained in the description of the reward to avoid confusion. Other rewards such as small objects themed to the Safe Driver Agency could also be sold in the store, giving another way to spend Reputation, which if not properly implemented can quickly become excessive.

The Provision Store has a direct impact on the value of the insurance policy, allowing the user to obtain additional coverage for little or no price to themselves simply for utilizing the system, this can be utilized, the rewards offered in the provision store can be seen as having an impact on Core Drive 6 Scarcity and Impatience as some rewards may be desirable but require considerable investment in the system before being achievable, owning the rewards themselves can impact Core Drive 4 Ownership and Possession as they may be perceived as something achieved with a degree of effort and are the property of the player.

Agent Bonus

As the main incentive to utilize the Safe Driver Agency system the user is given an Agent Bonus which translates as a discount on their insurance policy each month. This is calculated internally by the insurance company and not within the system itself. However the details of how the calculation is achieved are passed to the Safe Driver Agency control panel where the player can see what influenced the bonus they received.

The detail of the Agent Bonus is only a "dressing up" of the discount obtained from the usage of the system and doesn't appear to have any significant impact in any of the Core Drives, with the exception of Core Drive 8 Loss and Avoidance, as a low Agent Bonus is technically a loss on the side of the player as they will pay more for their policy in that period.

Agency Leaderboards

In an effort to provide a sense of competition to the game the introduction of Agency Leaderboards is utilized to classify players on their performance against each other due to the confidentiality inherent to the insurance policies this cannot show other players' information and therefor the leaderboards only show the position of the player and how many are above them in the rating instead of a general leaderboard, the information of other players is kept in the background at all times. Players that do well on the leaderboard will receive badges for their performance as well as an experience and reputation point reward. Leaderbords would be divided into Periodic, following the same pattern as Challenges and Area, Local, Regional and Global leading to a total of 9 distinct leaderboards where a player can compete.

Leaderboards can usually be attributed to Core Drive 2 Development and Accomplishment but also to Core Drive 5 Social Influence and Relatedness but being anonymous damages their capability of addressing this Drive.

Octalysis Score

Finally, we can calculate the Octalysis Score of the designed approach. Table 17 shows how the score for each core drive was determined and what influenced it and Figure 12 shows a representation of the calculated scores in an Octalysis Diagram.

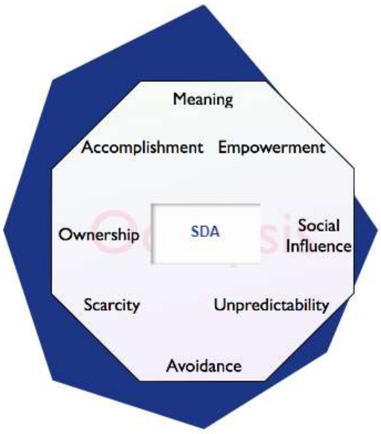


Figure 12 - Octalysis Diagram for SDA

Table 17 – Octalysis Core Drive Score

Drive	Contributing Elements	Determined Score
CD1: Epic Meaning and Calling	Theme, Global Operation	6
CD2: Development and Accomplishment	Operations, Agent Rank, Ribbons, Leaderboards	5
CD3: Empowerment of Creativity and Feedback		0
CD4: Ownership and Possession	Agent Rank, Ribbons, Provision Store	4
CD5: Social Influence and Relatedness	Theme, Global Operation, Leaderboards	2
CD6: Scarcity and Impatient	Operations, Provision Store	5
CD7: Curiosity and Unpredictability	Global Operation	2
CD8: Loss and Avoidance	Operations, General System Context	4

Three details about Table 17 must be discussed in further detail. Core Drive 3's score of 0 is easily determined by the lack of any contributing elements as there is little room to allow creativity on a system whose ultimate goal is the calculation of insurance premiums, not much can be done about this; the low value of Core Drive 5 despite the amount of present elements is due to the anonymity of all social aspects, so their overall impact is classified as being smaller; finally the element General System Context, refers to the environment that leads to the system, the insurance aspect, not utilizing this system is inherently detrimental as it leads to a higher price on the premium paid.

From these two this we can determine that this design was heavily Left Brain focused with a balanced White Hat / Black Hat approach, the Octalysis score is at 126 out of a possible 800. The system will lose most of its impact if the player loses interest in completing the Operations presented as they provide the only way of the player to be rewarded from utilizing the system.

4 Analysis and Design

Chapter Contents

4.1 Solution Design Overview

In this section a general overview of the solutions' design is outlined with a definition of its scope (section 4.1.1), functions for all modules(section 4.1.2), the description of the intended operational environment (section 4.1.3), the dependencies on other systems (section 4.1.4), the overall constraints (section 4.1.5) and a brief introduction of the Data Models utilized to configure the system (section 4.1.6). Additional information regarding relevant topics and a list of terms utilized during the rest of the solution design are also present.

4.1.1 Scope

Before the definition of the requirements for this solution it is important to frame what its actual scope is. Thus, a general overview of the solution as well as what the intended scope is can be seen in the high-level component diagram in Figure 13.

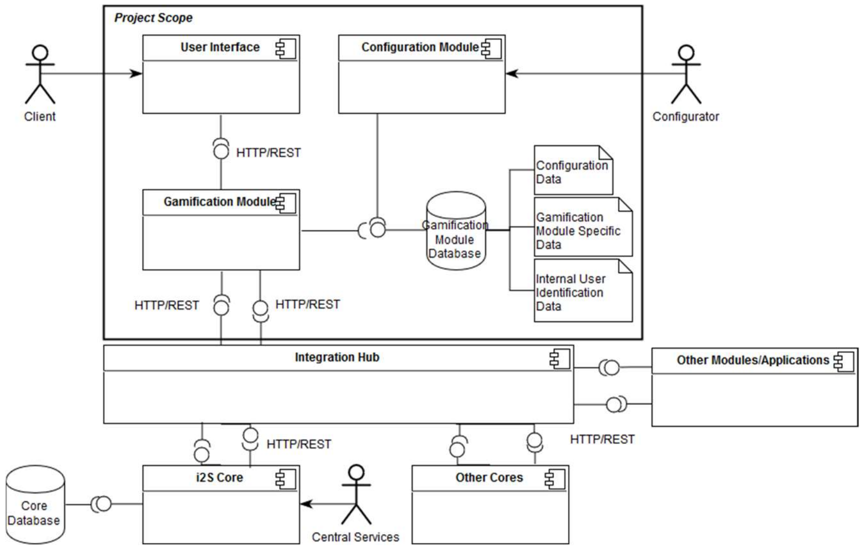


Figure 13 – High Level Component Diagram

As can be seen in the previously mentioned diagram the project's scope will consist of four elements, a Gamification Module, containing all the logic for the gamified approach to be utilized; a Database for this module where all persistent information that is only pertinent to the Gamification Module is stored; an User Interface Application that will supply the user with an interface that represents of the rules and elements defined in the Gamification Module's configuration, this application will be ultimately just a representation of a possible use of the Gamification Module. All the communications between modules id made utilizing REST services via the HTTP protocol and communication with the databases id done utilizing JPA.

4.1.2 Functions

As the solution is divided into various parts to clarify the responsibilities of each one the functions will be mentioned in relation to the part that has the responsibility. The Database will not be mentioned as its only responsibilities are to store data and respond to queries, as no other activities will be performed on the database's side, nor will the functions of the Configuration Module as this Module already exists and is only within the scope as some additions to it may be required to answer the new needs of the Gamification Module.

As the Gamification approach ran from the more general term to the more specific, the functions regarding the User Interface Application changed, this was not the case for the other applications which maintained the required functions, for the purpose of this report both lists of requirements will be shown.

User Interface Application:

- Basic Login and Logout options
- Allow messages from the "Overseer" to be shown at any time.
- Allows for the visualization and completion of Operations of multiple types.
- Allow visualization of the Global Operation, including risk zones, changes and personal impact.
- Provide constant visualization of Agent Rank, Experience Points and Reputation Points.
- Allow for the presentation of a Ribbon Showcase showing all ribbons earned by the player.
- Allow a Store where rewards can be visualized and claimed utilizing Reputation Points.
- Allow the visualization of Statistics collected by the system.
- Allow the visualization of earned Agent Bonus with breakdown of how it was calculated.
- Allow the visualization of a Score history.
- Allow the visualization of Leaderboards.

Gamification Module:

- Loading, Temporarily Storing and Managing Concepts, Concept Templates, Rules and Tables
- Generating DTO's for exposure to external APIs
- Respond to requests by external applications utilizing a REST interface.
- Connect to the (SQL or NoSQL) Gamification Database containing all the configuration elements for the module to function.
- Allow the connection to, configured, REST based interfaces, which return which function as Secondary Data Sources.
- Periodically updating prebuilt components.
- Periodically saving changes in memory to the Database.

Configuration Module:

- Viewing, Editing, Creating and Deleting Concepts, Templates, Rules and Tables, in obfuscated and pure JSON formats.
- Allow a Game Element view to allow configuration based on Game Elements instead of a direct format.

4.1.3 Environment

To help illustrate the operational environment of the system to be developed a low detail deployment diagram was drawn, including not only the system being developed but the systems it will connect to including both the relevant parts of the pre-existent i2S solution and the client's access point.

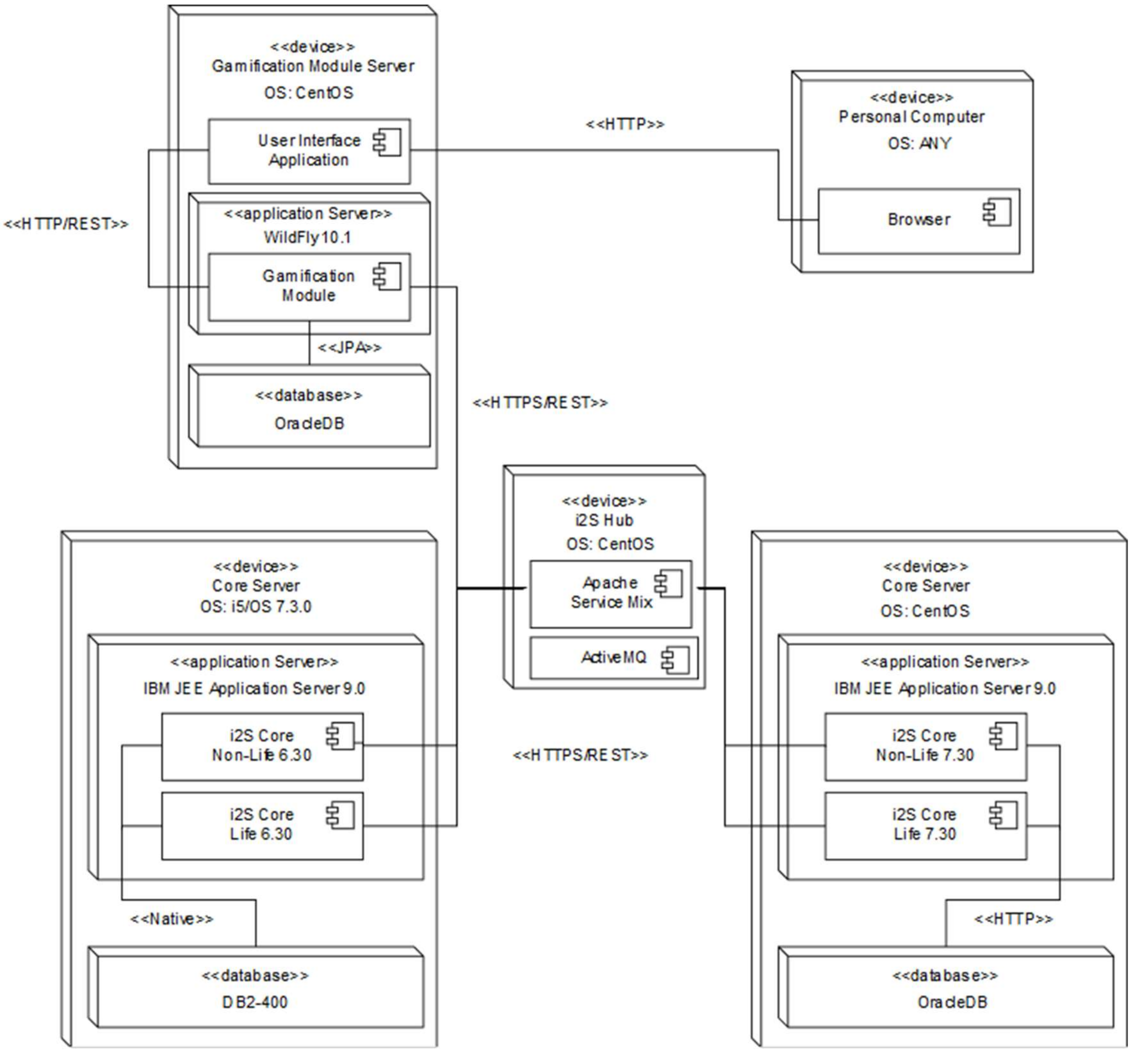


Figure 14 - Deployment Diagram

As can be seen in Figure 14 the developed system will reside in a machine utilizing the CentOS Linux distribution (The CentOS Project, 2017) as its operating system, where both applications developed will be executing.

The Gamification Module is coded in Java 8 and will run in a WildFly 10.1 application runtime environment while the User Interface Application will be developed utilizing Angular 4. The database will be an Oracle Database.

Communications with the i2S Hub's Service Mix will be done utilizing HTTPS/REST as it is done with all other connections to this component, communications between the Gamification

Module and the User Interface will be done utilizing HTTP/REST and finally communications between the Gamification Module and the Oracle Database will be done utilizing JPA.

Eventually it was decided that for this instantiation a MongoDB solution would be used instead of the OracleDB however they system is to be made ready to use any kind of DB to function.

4.1.4 Dependencies

As can be seen in Figure 13 this solution, only encompassing a limited part of the system, can only work if an application that supplies insurance data to the user is present to interface with the Gamification Module this is known as the Secondary Data Source, and although the system is prepared to utilize only the data present in its database, or Primary Data Source, no updating of user data can be made without the secondary source.

4.1.5 Constraints

As the design of this solution involves interfaces with various other systems this causes some constraints regarding design and, consequently, implementation of the proposed solution. There are two issues detected:

It must be data source agnostic, the source of the data provided to the system can be any application at all so the binding between applications and the Gamification Module must be configurable, the main issue presented is not the reception of data by the Gamification Module though as this remains constant but the supplying of data by it.

It must be user interface agnostic, meaning the data provided by the Gamification Module must be able to be used by any application that wishes to utilize it, this requires the definition of a standard data structure to be consumed by the application utilized as interface.

4.1.6 Modelling System

To allow for the flexibility required for the system a set of modelling systems had to be created so elements and rules could be represented and edited freely. As such the Concept System and the Rule System were created to respectively model system elements and rules, allowing the creation elements utilizing an OO-Like design, based on attributes and rules. These two systems are complex and will be detailed before the Modules themselves as they are used across all of them.

4.1.7 Terms Utilized

A few terms are utilized within the description of Module Design and Data Models that need to be clarified to avoid confusion.

Gamification Database: The Gamification Database is a database of any type that holds all information about the Gamification Configuration, for the purpose of the initial prototype it also holds Authentication Data. This Database can be of any distribution in the scenario.

Primary Data Source: Refer to Gamification Database.

Secondary Data Source: Refers to the API accessed by the DAL of the Configuration Module to obtain data from the i2S Core System, this data is deemed to come with proper JSON structure for the concept it refers to.

System: System refers to the conjunction of the Gamification Module and Configuration Module but not the User Interface Application as this application is a client to the System and not part of it. This is also referred to as the **Gamification System**.

Module: is one of the three major parts of the Scope, Gamification Module, Configuration Module and User Interface Application.

Module Section: Is one of the large organizational blocks within the Module which can be seen in the initial description of each Module in both textual description and an UML Component Diagram.

Gamification Approach: A designed Gamified solution, two exist and can be seen in report section 3.2, the **Original Approach** refers to the Initial concept and **Selected Approach** to the **SDA Approach**.

Base Element: The four Models described in sections 4.2 and 4.3.

Game Element: A speculative structure that would combine Base Elements into a new type of Element with a level of abstraction. These would represent more complex concepts such as Badges, Missions and Leaderboards utilizing one or multiple Base Elements.

4.2 The Concept Model

During the design of the system the necessity for a high amount of configurability was immediately detected as a requirement and as such the abstraction of a Concept was created to allow flexibility in defining the elements of the system.

Concepts can be divided into two simple parts, a Concept Template, that defines what fields the concept can have, including their values by default, as well as general rules to run for all the concepts that share the defined CTK (Concept Type Key) of the Concept Template and the Concept Instance, normally referred to only as Concept, which contains the data for the fields, a field called the CK (Concept Key) composed of the previously mentioned CTK and a CIK (Concept Instance Key) as well as any additional triggers wanted and sub concepts that are related to that instance and can be accessed from it.

In Figure 15 a Class Diagram of how a Concept (Instance) is to be implemented in the system. The Concept Template is to be implemented in the same way with the only changes being the field "ck:int[]" which is replaced by "ctk:int" and the lack of the "composingConcepts" and "stack" fields which are not part of the Template.

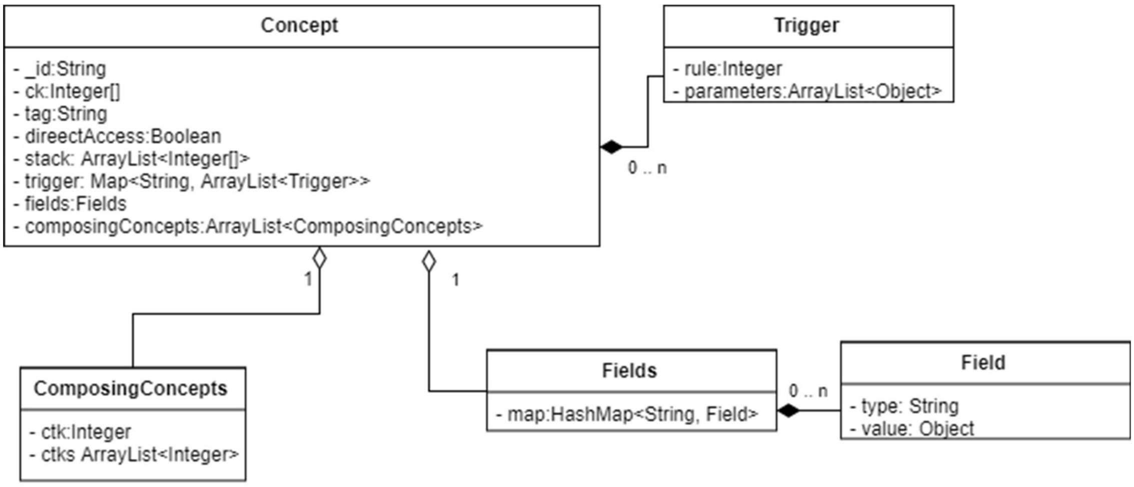

Concept
- _id:String
- ck:Integer[]
- tag:String
- direectAccess:Boolean
- stack: ArrayList<Integer[]>
- trigger: Map<String, ArrayList<Trigger>>
- fields:Fields
- composingConcepts:ArrayList<ComposingConcepts>
Trigger
- rule:Integer
- parameters:ArrayList<Object>
0 .. n
1
1
ComposingConcepts
- ctk:Integer
- ctks ArrayList<Integer>
Fields
- map:HashMap<String, Field>
0 .. n
Field
- type: String
- value: Object

Figure 15 - Concept Class Diagram

An example of the JSON for a Concept Template and Instance can be seen in Code 1 and Code 2 which is used to represent a Badge (Ribbon) in the implemented prototype.

The evolution of the JSON for the Concept Template and Concept can be seen in annexes G.1 and G.2 which shows all versions of the JSON utilized until this final version was finally decided upon.

As can be seen in the Concept Template JSON in Code 1 there is a tag called "trigger", this tag is where rules that have to be ran are located as well as the parameters required to execute them, they are however no rules that are executed by the Badge Template, nor by the Badge Instance in fact as in the prototype Badges are attributed by rules present in the Statistics Template.

```
{
    "_id" : ObjectId("5b17c2ee07246539c811a0de"),
    "ctk" : NumberInt(3),
    "tag" : "badge",
    "directaccess" : true,
    "trigger" : {
        "auto": [ ],
        "explicit" : [ ],
        "prebuild" : [ ]
    },
    "fields" : {
        "name" : {
            "type" : "String",
            "value" : "Badge Name"
        },
        "image" : {
            "type" : "String",
            "value" : "default"
        },
        "image_type" : {
            "type" : "String",
            "value" : "svg"
        },
        "description" : {
            "type" : "String",
            "value" : "Description"
        }
    }
}
```

Code 1 - Concept Template JSON

```
{
    "_id" : ObjectId("5b3a62675cd0c40ae4893e70"),
    "ck" : {
        "ctk" : NumberInt(3),
        "cik" : NumberInt(10)
    },
    "tag" : "badge",
    "directAccess" : false,
    "stack" : {  },
    "trigger": {
        "explicit": [ ],
        "auto": [ ],
        "prebuild": [ ]
    },
    "fields" : {
        "image" : {
            "type" : "String",
            "value" : "default"
        },
        "name" : {
            "type" : "String",
            "value" : "Platinum Completist"
        },
        "description" : {
            "type" : "String",
            "value" : "Compteted 500 Missions"
        },
        "image_type" : {
            "type" : "String",
            "value" : "svg"
        }
    },
    "composing" : [  ]
}
```

Code 2 – Concept Instance JSON

In the Instance itself presented in Code 2 we can see the two new fields that weren't present in the Template, composing and stack, again the Badge does not have any value in ether, representing that this concept has no concepts under it and no concepts above in a hierarchy.

4.2.1 Concept Tags

Before moving forward it is important to define what all the Tags in the Concept JSON mean so a clear understanding of the structure can be determined for implementation within any application that needs to deal with concepts.

_id
The "_id" tag is generated by the MongoDB when a file is introduced and has no bearing on the system, it does however need to be present to allow for proper CRUD functionality as it functions as the base, immutable Id in the database.

CK (CTK and CIK)
As previously mentioned CK is the combined key that represents which object is being utilized, being the *de facto* id in the system, allowing the quick visibility of what type of Concept we are dealing with via de CTK and which specific instance of that Type. This also aids in keeping the numbers of Id's lower as the amount of data grows within the system.

Tag
The "Tag" tag is utilized only in the configurator for quick verification by the user of what a Concept or Concept template is supposed to be, the rest of the system should only use the CTK to make this identification.

DirectAccess
The DirectAccess tag determines if a Concept can be obtained by itself or requires a certain "stack" (more on "stack" later) to be passed via the request or present within the Concept by default. This is set to true only in root concepts, which can function without a parent in mind such as the Badge Concept and the Player Concept.

Stack
The Stack tag is a tree of concepts that are parent, or higher up in the hierarchy than the concept it belongs to, it can have multiple CKs within it and it's sorted from closest parent to furthest parent, the last element on the array being known as the root parent. In

Code 3 an example of a Stack with 2 CKs can be seen. Stacks are utilized within Rules to allow navigation between Concepts and are not necessarily present within a Concept by default, a temporary Stack can be inserted into a Concept for the purpose of a single Rule execution.

```
"stack" : [
    [
        NumberInt(5),
        NumberInt(3)
    ],
    [
        NumberInt(1),
        NumberInt(1)
    ]
]
```

Code 3 – Stack Example

Trigger

The Trigger tag has 3 different sub-tags, which are used to define rules that run at different times. The rules within the “auto” tag run whenever the concept is loaded into the system, calculating or verifying states which are always necessary, members of the “explicit” tag are only executed when explicitly called from the API and are used to execute rules such as the using of points to buy a reward, finally rules within the “prebuild” tag are executed periodically according to the configured timeframe and are utilized to periodically calculate concepts that only need to be calculated once for all users such as leaderboards.

In

Code 4 we can see an example of Trigger tag with actual data this one from the Mission Template in the prototype. It only possesses values within the “explicit” sub-tag however the same structure is valid for any of the other arrays. A trigger call is composed by an object with two tags, “rule” which determines the Rule which will be called and “parameters” which lists the values for all parameters the Rule requires.

Parameters can be of two types, if a string surrounded by square brackets (“[ ]”) they refer to a field within the Concept Instance where the Rule is being executed from, if the parameter does not have the square brackets it is a literal value, which can be a number, string or Boolean value.

```
"trigger" : {
    "auto" : [],
    "explicit" : [
       {
         "rule" : NumberInt(1),
         "parameters" : [
            "[completed]",
            "[unlocked]",
            "[exp]",
            "[reputation]"
          ]
       },
       {
          "rule" : NumberInt(2),
          "parameters" : [
             "[completed]",
             "[unlocked]",
             "[type]",
             "[reputation]"
          ]
       }
   ],
   "prebuild" : []
 }
```

Code 4 – Trigger Example

Composing

The Composing tag represents sub-concepts to the current concept, these serve to relate dependent Concepts to an overall parent Concept, in the case of the prototype developed we have the example of the Player Concept which holds, the sub-concepts Statistics, Policy, Mission and Badge, an example of the Composing tag of this exact concept can be seen in

Code 5.

```
"composing" : [
    {
        "ctk" : 2,
        "cik" : [1]
    },
    {
        "ctk" : 11,
        "cik" : [1]
    },
    {
        "ctk" : 10,
        "cik" : [1,3,4,5,6]
    },
    {
        "ctk" : 3,
        "cik" : [1,11,12,13,14,15]
    },
    {
        "ctk" : 9,
        "cik" : [1]
    }
]
```

Code 5 - Player Composing Tag

Fields

The Fields tag works as the attributes of a Concept and serves different purposes in Instances and Templates. In the case of Templates it serves to determine which fields are mandatory for every concept that shares that Template to have, as well as to define default values for those fields. In Instances these values are inevitably changed and serve as the Concept's method of storing data.

Fields may have any name with the notable exception of "ref" which is a reserved field name. The presence of a "ref" field tells the system that the Concept has an external and when loading a concept with this field it must request its data from the external source defined within the system by sending the "ref" value to the provided REST interface.

4.2.2 Auxiliary Elements

One auxiliary element, UInfo was originally developed to insert data relevant to the user interface, this was deemed to be a responsibility of the user interface and so it was dropped, however, not before being designed and implemented. Details on the JSON and more explicit inner workings can be seen in annex G.4.

UInfo
To prepare information to be utilized by the API it was deemed necessary to enrich the Concept themselves with additional information and as such the UInfo (**U**ser **I**nterface **Info**) elements were created.

Each Concept Template has its own associated UInfo Template which contains details on how to enrich concepts that adhere to that Concept Template when requested by an external source. The lack of this Template simply returns the Concept Instance as is, without any kind of treatment. The UInfo Template holds 0 to n UInfo for each field of the Concept Template, which can be of multiple types. An example of a UInfo Template's structure utilizing an UML Class Diagram can be seen in Figure 16.

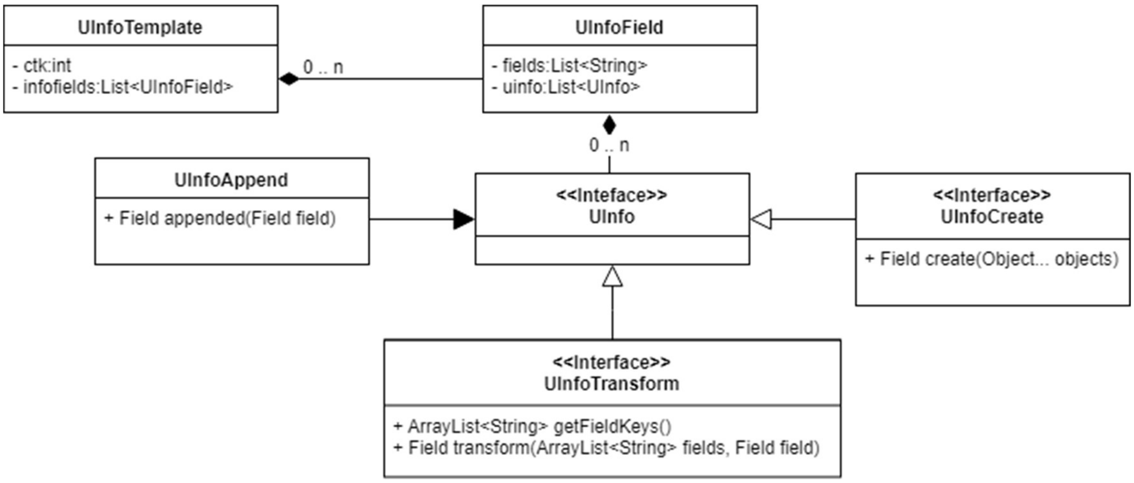

UInfoTemplate
- ctk:int
- infofields:List<UInfoField>
0 .. n
UInfoField
- fields:List<String>
- uinfo:List<UInfo>
0 .. n
UInfoAppend
+ Field appended(Field field)
<<Inteface>>
UInfo
<<Interface>>
UInfoCreate
+ Field create(Object... objects)
<<Interface>>
UInfoTransform
+ ArrayList<String> getFieldKeys()
+ Field transform(ArrayList<String> fields, Field field)

Figure 16 - UInfo Class Diagram

UInfo Types

Four types of UInfo exist, each of them having multiple sub-types, three of them are present in Figure 16 but one was omitted as it is considered special. These types are UInfoAppend, UInfoTransform, UInfoCreate and the omitted one UInfoBlank.

Append
Append their information to the existing data, a good example is the DisplayText which is used to add an array of [language key, localized text] to one of the fields, not replacing the information present within it. The full list of Append Type UInfos is: NfoBar, NfoColour, NfoDisplayText and NfoRange.

Transform
Completely transform one or more fields, the original field or fields they are associated with do not appear as an effect of these UInfo's but a new transformed field. The full list of Append Type UInfos is: NfoImage and NfoTransformText.

Create

Create a completely new field with predetermined information therefor type is not associated with a specific field. The full list of Create Type UInfos is: NfoDerivedNewText and NfoNewText.

Blank

Technically used as a simple flag UInfo the, hardcoded, Blank type serves only to force the addition of the default value of a field to the final Concept object.

UInfo Template Application

When a concept is called via the API the Concept is obtained from the database (or secondary data source) and then the UInfo Template is applied to the Concept in the order set in the Template, with the suggested order being Create -> Transform -> Append -> Blank.

4.3 The Rule Model

A system for Gamification inherently requires a way to execute actions upon the present values, and leaving this responsibility to an outside entity creates a great deal of security risk, therefor the inclusion of a Rule Engine was necessary. Rules defined have to be sufficiently flexible to do whatever necessary run the Gamified system. The initially the responsibility of this was to be attributed to the i2S IMBL (the internal domain specific language utilized in other modules of the company) however this proved not to be a viable option so a small Rule engine was designed utilizing the Spring Expression Language (SpEL) after some experimentation with a mathematical operation parser.

So that Rules could be easily defined by a configurator a JSON file structure was also created. A simple rule defined in this structure can be seen in Code 6, this is a Rule taken from the prototype instance. Details about the evolution of the Rule JSON file can be seen in annex G.3.

```
{
    "_id" : ObjectId("5b30eca25cd0c43530dd8ecf"),
    "key" : NumberInt(4),
    "name" : "AddAgentBadge",
    "facts" : [
        {
            "key" : "fuel",
            "type" : "String",
            "value" : "[0]"
        }
    ],
    "condition" : {
        "operation" : "true",
        "resultType" : "Boolean",
        "resultVar" : "result"
    },
    "then" : [
        {
            "operation" : "tableCompare(3, #fuel)",
            "resultType" : "Number",
            "resultVar" : "badgeId"
        },
        {
            "operation" : "#concept.getParent().addComposingConcept(3, #badgeId)",
            "resultType" : "Number",
            "resultVar" : "result"
        }
    ],
    "else" : []
}
```

Code 6 - Rule JSON

In Figure 17 the representation of the rule structure can be seen in an UML Class Diagram.

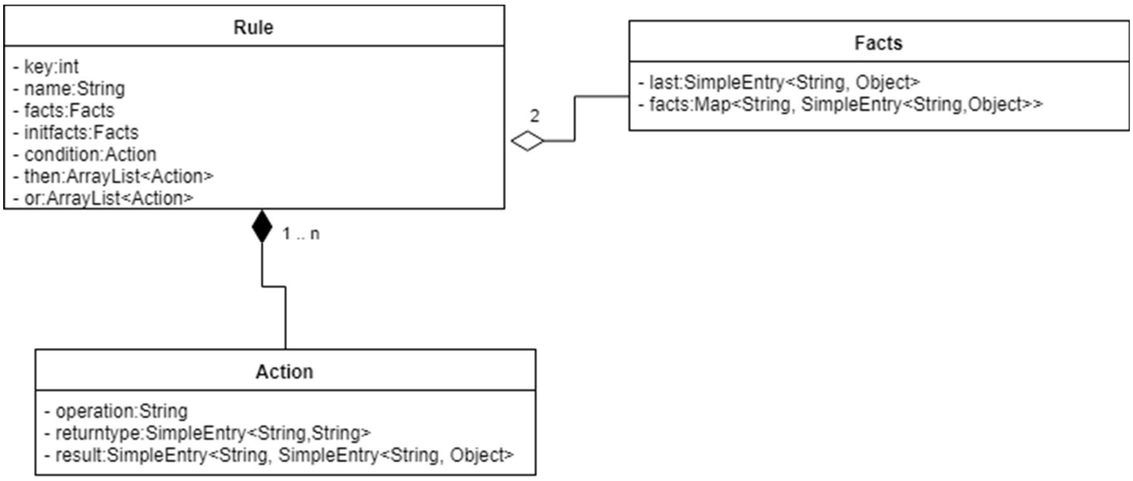

Rule
- key:int
- name:String
- facts:Facts
- initfacts:Facts
- condition:Action
- then:ArrayList<Action>
- or:ArrayList<Action>
Facts
- last:SimpleEntry<String, Object>
- facts:Map<String, SimpleEntry<String,Object>>
2
1 .. n
Action
- operation:String
- returntype:SimpleEntry<String,String>
- result:SimpleEntry<String, SimpleEntry<String, Object>

Figure 17 - Rule Class Diagram

4.3.1 Rule Tags

The Rule JSON is composed for seven top level tags, one of them the "_id" tag which serves as an ID for MongoDB.

Key
The Key tag serves as the ID for the system to recognize the rule, any mention of the rule in the system will be made by referring to the present key value.

Name
The Name tag serves the same purpose as the Tag tag in the Concept, serving only for ease of navigation among rules in the configurator.

Facts
Facts are utilized to define what exterior variables need to be inputted into the Rule for it to be executed. Fact objects are composed of three tags, "key", "type" and "value": the "key" tag refers to what variable name will be used in the expressions to follow, the "type" tag informs the type of value held in the variable and the "value" tag which value the variable possesses. The value can be, as showed in Code 6, a number surrounded by square brackets (Ex. "[0]"), this refers to parameters passed to the rule by numerical order starting at 0, values may also be literal strings, numbers or booleans.

Condition, Then and Else
The tags Condition, Then and Else are grouped together here as they all have the same type of content, Actions, as can be seen in Figure 17. The Condition tag only contains a single action that must return a boolean value with a variable name of result, why Then and Else may have any number of actions that return whatever intended, the final value of the rule is always discarded.

Actions consist of three tags, “operation”, “resultType” and “resultVar”. “operation” contains the set of instructions to be executed by the SpEL, consisting of Java code, “resultType” the type of the returning variable and “resultVar” the name of said variable.

The “operation” tag possesses a limited set of instructions not tied to the objects themselves, the full list of these instructions and instructions planed for the future can be seen in 4.4.4.

4.3.2 Auxiliary Elements

To address some limitations, present in the Rule Model another element was necessary. Currently the Rules could not utilize a set of values to obtain a dependent value, for instance, knowing what ribbon to give the player based on their Score. For this purpose the Table element was created.

Tables
Tables allow the search of a value in a predefined table by 1 to n values and function by defining a set of Axis, with a minimum of 1 and no set maximum, and then providing values for every combination of values of the defined axis in a n-dimensional matrix allowing any combination of these Axis values can be searched within the table. An UML Class Diagram for the implementation of tables can be seen in Figure 18.

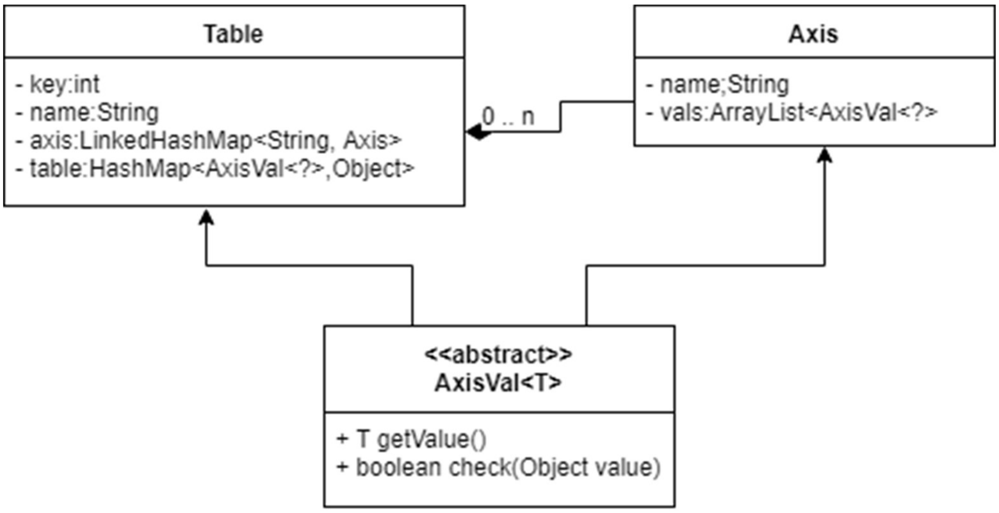


Table
- key:int
- name:String
- axis:LinkedHashMap<String, Axis>
- table:HashMap<AxisVal<?>,Object>
0 .. n
Axis
- name;String
- vals:ArrayList<AxisVal<?>
<<abstract>>
AxisVal<T>
+ T getValue()
+ boolean check(Object value)

Figure 18 - Table Class Diagram

As the JSON files for Tables are extensive it was decided not to present one within this section as it requires minimal explanation of tags. It can however be consulted in annex **Error! Reference source not found.**.

AxisVal Types

A detail of Figure 18 that is worthy of note is the AxisVal<T> abstract class which refers to values within the table Axis that are also applied to the table itself to allow for quick search. To allow for max flexibility of values within tables it was determined that the utilization of a simple value to compare against was unsuitable and four types of AxisVal were created to address all possible cases of configuration detected.

AxisValValue

AxisValValues are the simplest of all types, referring to a static value of any type, during the verification state this type of AxisVal returns true if the value is exactly the same.

AxisValBEdge

AxisValBEdges are Bottom Edge values, these work only with numerical values and will return true if the value is lower or equal to the axis value.

AxisValTEdge

AxisValTEdges are Top Edge values and are the antithesis of AxisValBEdge returning true when the value is equal or greater than the axis value.

AxisValRange

AxisValRange also only works with numerical values and verify that the value is between the two axis values, inclusively.

4.4 Gamification Module

The Gamification Module possesses five internal components.

- **API Provider**: exposes the Gamification REST API and deals with the preliminary stage of the response process by verifying its validity and accesses.
- **Concept Manager**: Saves and manages Concept, Concept Templates and Auxiliary Elements as well as triggering their rules.
- **Data Access Layer**: Performs CRUD operations with the Gamification Module Database and deals with communications with other applications such as Secondary Data Sources.
- **Rule Handler**: Loads and saves Rules and auxiliary elements, also deals with orchestrating the flow of rule execution processes.
- **Task Executer**: The Task Executer component is where the periodic events of the gamification module are executed. These events run on parallel threads and are triggered on a predetermined timescale, which is configured before the launch of the application.

A representation of the relations between these modules and exterior interfaces can be seen in Figure 19.

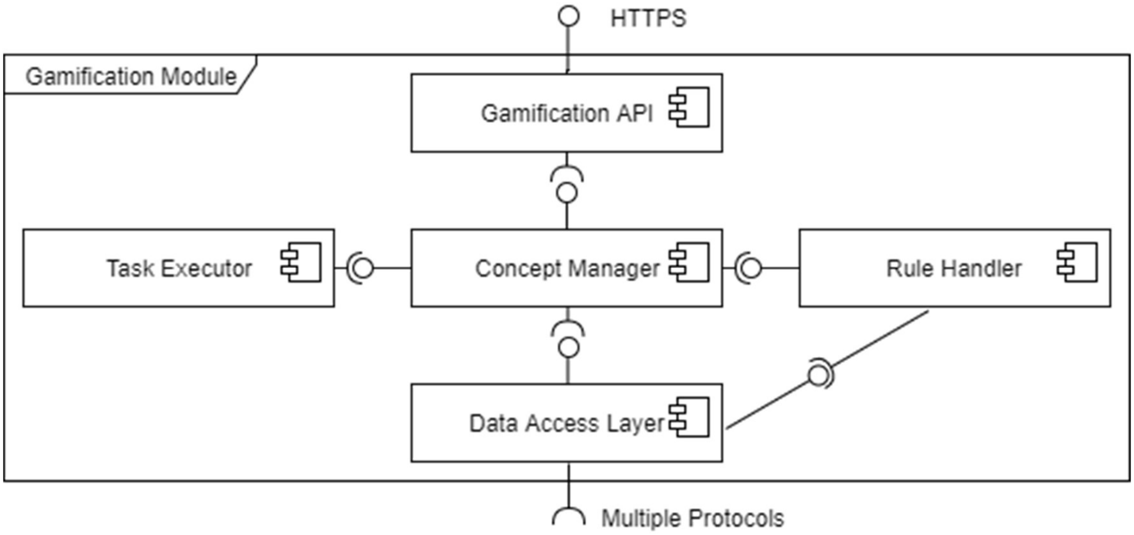


Figure 19 - Gamification Module Organization

4.4.1 Features

As the core of the entire System the Gamification Module possesses the main features of the entire system namely:

- Loading, Temporarily Storing and Managing Concepts, Concept Templates, Rules and Tables
- Generating DTO's for exposure to external APIs
- Respond to requests by external applications utilizing a REST interface.
- Connect to the (SQL or NoSQL) Gamification Database containing all the configuration elements for the module to function.
- Allow the connection to, configured, REST based interfaces, which return which function as Secondary Data Sources.
- Periodically updating prebuilt components.
- Periodically saving changes in memory to the Database.

4.4.2 Processes

Start-up

The start-up process consists in verifying that the system's dependencies are available for utilization. This process is mostly handled by the Spring Boot as it will fail initialization if it cannot connect to the Primary Source, the Gamification Database. After that connection attempts will be made to secondary data sources and their availability or lack of will be registered in the system. A flow diagram representation of this process can be seen in Figure 20.

The Start-up process is not tied to any specific Module as it is executed upon the first launch of the application.

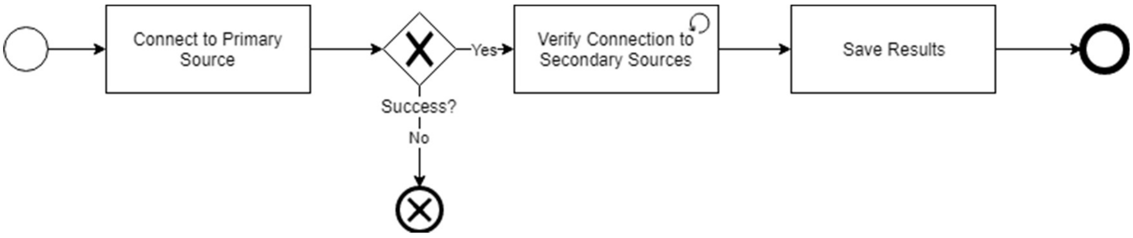

Connect to Primary Source
Yes
Success?
No
Verify Connection to Secondary Sources
Save Results

Figure 20 - Start-up Process Diagram

General Update

This process is made periodically to update the concepts in memory within the database itself and clear memory space. Upon a configured amount of time the system saves all the concepts currently stored in memory and saves them on the Primary Data Source, completely wiping the memory afterwards. This process can be seen in Figure 21.

As the general update is executed according to a predetermined timeframe, parallel to normal execution, it is a responsibility of the Task Executor.

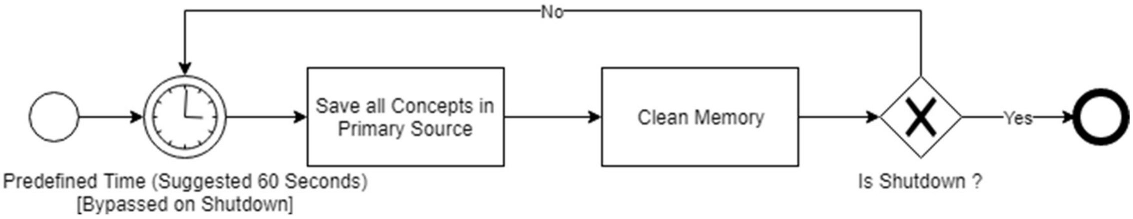

No
Save all Concepts in Primary Source
Clean Memory
Yes
Predefined Time (Suggested 60 Seconds) [Bypassed on Shutdown]
Is Shutdown ?

Figure 21 - General Update Process

Shutdown
The shutdown process is executed only when explicitly called. This does not consider events such as unscheduled shutdowns, in any case this process only executes the General Update process then closes the application, the Shutdown process cannot fail and will not save any data if the data source cannot be reached.

Request Response
The Gamification Module can respond to external requests via the Gamification API which, although though up for use by the User Interface Application also has occasional use requests to force triggering of rules, these have limited access permissions that must be met for their call.

In any case the Request Response process executes in the exact same way for all requests, first request validity is tested followed by authorization and only then is the Request executed, in case of failure the error is softened and in case of success the result is returned. In any case the interface responds with proper, standard HTTP Status codes for all cases.

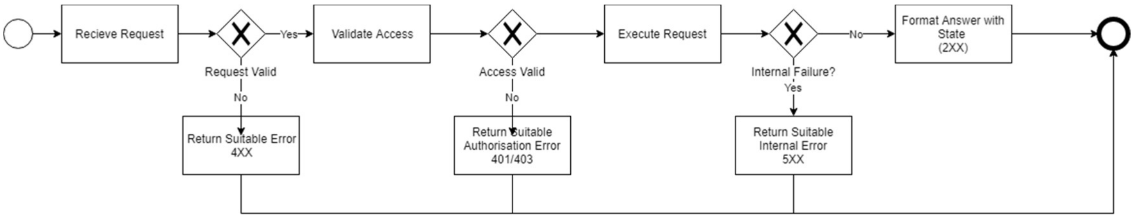

Recieve Request
Yes
Validate Access
Execute Request
No
Format Answer with State (2XX)
Request Valid
No
Access Valid
No
Internal Failure?
Yes
Return Suitable Error 4XX
Return Suitable Authorisation Error 401/403
Return Suitable Internal Error 5XX

Figure 22 - Request Response Process

Pre-Builder
The Pre-Builder must trigger occasionally to execute the update of the data of Concepts who possess Prebuild triggers, the frequency of this is global and defined in the application configuration files. The Pre-builder process is launched on a secondary thread to the main application which checks for concepts with the “prebuild” tag and executes them.

Upon the launch of the application the Pre-Builder should be executed immediately and only then transfer to the waiting state. The Pre-builder is handled by the Task Executer and works parallel to the rest of the system but involves all sections of the module with the exception of the API Provider as it necessitates loading of Concepts and execution of Rules.

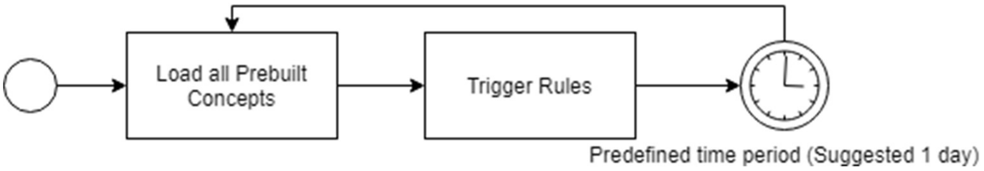

Load all Prebuilt Concepts
Trigger Rules
Predefined time period (Suggested 1 day)

Figure 23 – Pre-build Process

Concept Loading

Concept loading is done when a concept is requested by the API or internally by the Pre-Builder. When this happens the system first detects if the requested concept is already present in memory, if so it returns the loaded concept, otherwise it attempts to locate the concept in the Primary Data Sources, if the concept is present within the data source, it triggers the concepts template's rules, triggers the concepts rules and saves it into memory, otherwise it attempts to obtain it from the secondary data source if not available the entire process fails. In Figure 24 the general flow of this process can be seen.

This process involves both the DAL and the Concept Manager, also making a call to the Rule Handler in case rules of the "auto" variety are present.

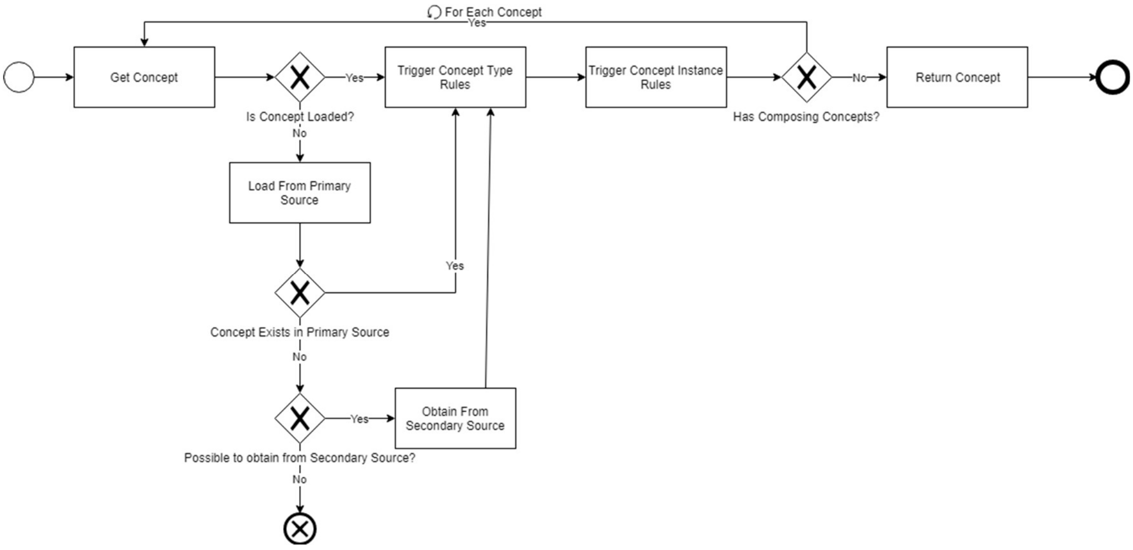

For Each Concept
Yes
Get Concept
Yes
Is Concept Loaded?
No
Trigger Concept Type Rules
Trigger Concept Instance Rules
Has Composing Concepts?
No
Return Concept
Load From Primary Source
Yes
Concept Exists in Primary Source
No
Yes
Obtain From Secondary Source
Possible to obtain from Secondary Source?
No

Figure 24 - Concept Loading Process

Rule Triggering

When a Concept is loaded or changed the associated rules must be triggered to allow for proper presentation of the available data. As such Rules are regularly triggered on change and load, however, utilizing the API it is possible to force the execution of Rules, utilizing a "/trigger" path to a concept with the POST method, this however is only necessary dependent on the timeframe defined for General Update as a very short general update will cause the information of Concepts in memory to be volatile at best, and as such the concept must be obtained from the database much more often causing the Rule to trigger every time.

This process can be seen, in general terms, in the diagram present in Figure 25.

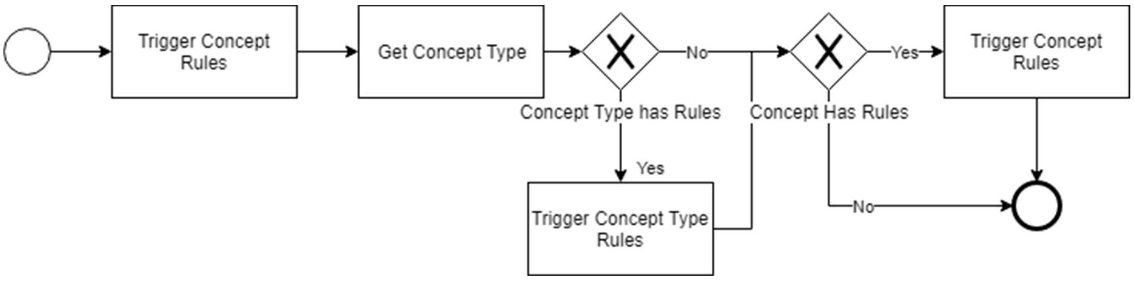


Figure 25 - Rule Trigger Process

Rule Execution

When rules are triggered they follow a strict execution pattern that is always followed and work much like an "if" statement in any programming language, it calculates the result of the available condition and then proceeds to execute one of two instruction paths depending on the result of said condition. This process is purely done by the Rule Handler and operations are processed by the SpEL.

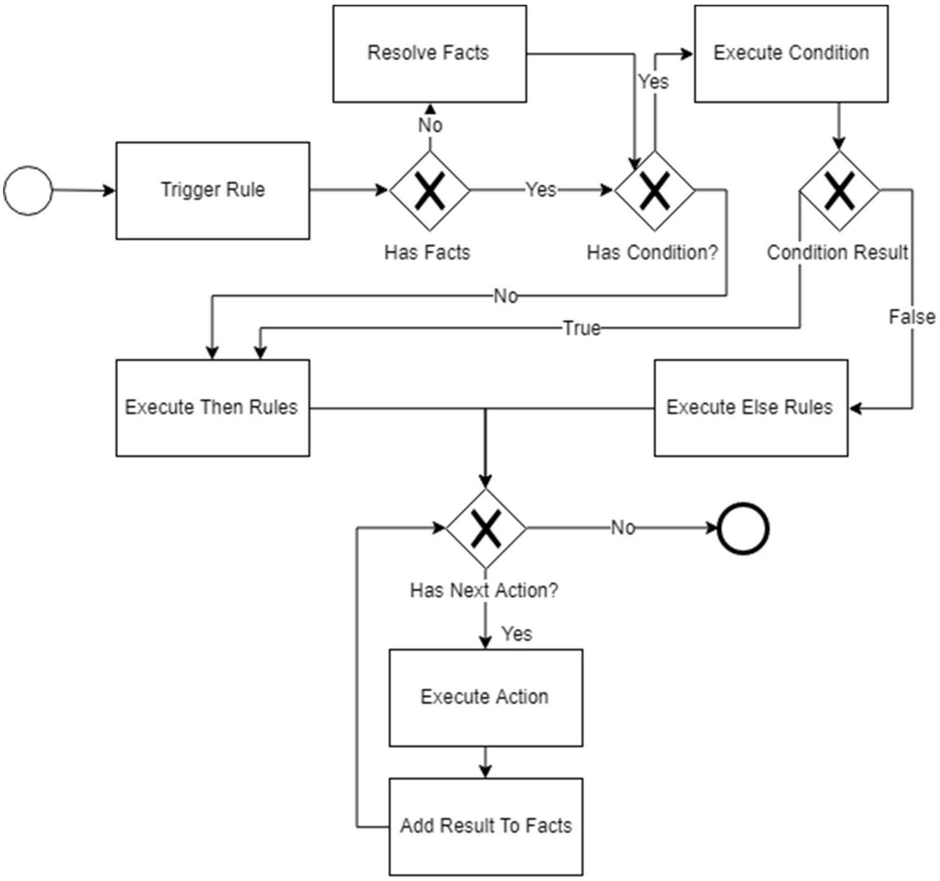


Figure 26 - Rule Execution Process

4.4.3 Software Interfaces

The Gamification Module provides only one exterior API, the Gamification REST API, which is to be utilized by any User Interface Applications. As seen in Table 18 this API contains five methods. No method requires anybody, however POST methods may receive a Stack<Int[]> to override the obtained Concept's Stack temporarily.

Table 18 – Gamification API

Method	Path	Description
GET	/concept/user	Obtain the Player Concept for the authenticated user.
GET	/concept/{ctk}	Get all Concepts of a CTK.
GET	/concept/{ctk}/{cik}	Get a specific Concept.
POST	/concept/{ctk}/{cik}	Get a specific concept with Stack Override
POST	/concept/{ctk}/{cik}/trigger/{rule}	Trigger a specific explicit Rule of a Concept, Stack Override Optional

4.4.4 Additional Information

SpEL Support Variables and Methods

Three additional parameters are always present during execution of a Rule by default, the Concept that called the rule and the individual CTK and CIK of that Concept and are referred to as #concept, #ctk and #cik.

SpEL only allows for the execution on methods from the passed variables and generally available libraries that do not require explicit imports in a Java class. To allow for some extra flexibility Context Class is created with additional methods to allow for executions out of these scopes.

The additional methods present in the Context Class are can be seen in Table 19, some whose necessity was detected but whose exact functioning was not determined can be seen on an additional Table 20, members of this last table are due to rise as development proceeds due to the unquestionable presence of rule needs undetected at this time.

Table 19 – Defined Support Methods

Method	Description
getConcept	Only utilized in the initial parameter definition serves as a method of obtaining the concept that called the rule via the ctk and cik.
tableCompare	For use within operations tableCompare serves as a way to obtain a value from a Table based on a set of Axis values.
invertList	A utility function as a method for list inversion cannot be called from SpEL invertList reverses the order of a given list object.
runRule	A rule may only need to be executed if a certain condition is valid in another, this method allows the execution of said rule but the parameters for it must all be sent from the operation command.

Table 20 – Undefined Support Methods

Method Designation	Description
Multi Rule Execution	Some processes may need to execute the same rule multiple times with different values or even multiple rules the only way to do so is to execute them individually, a method to allow this needs to be implemented to facilitate this.
Concept Creation	Features like Leaderboards need to be able to dynamically create their placement based on a set of data, but even if possible to calculate they cannot create new concepts by means of operations.
Batch Calculations	If the need to calculate something according to a large quantity of data, as there is no proper method of dealing with lists the suppling of Batch Calculation functions is necessary to do properly address it.
List Sorting	This necessity also comes from Leaderboards, lists need to be possible to sort according to a parameter, as list functions are not available auxiliaries must be added.

4.5 Configuration Module

With the purpose of manipulating the System Models the Configuration Module is where all of the work regarding the Gamified Approach's structure and rules is made. The Configuration Module is composed by two different Applications, a Backend, server side, application and a Frontend client side one. The Backend trims down the original database information to make it easier for utilization by the interface and deals with necessary calculations to properly index new data objects. The Frontend offers an interface that allows for a visual manipulation of Concepts, Templates, Rules and Tables (with a maximum of 2 axis) and an overview that allows the addition of Game Elements to the system, obfuscating the Base Element structure.

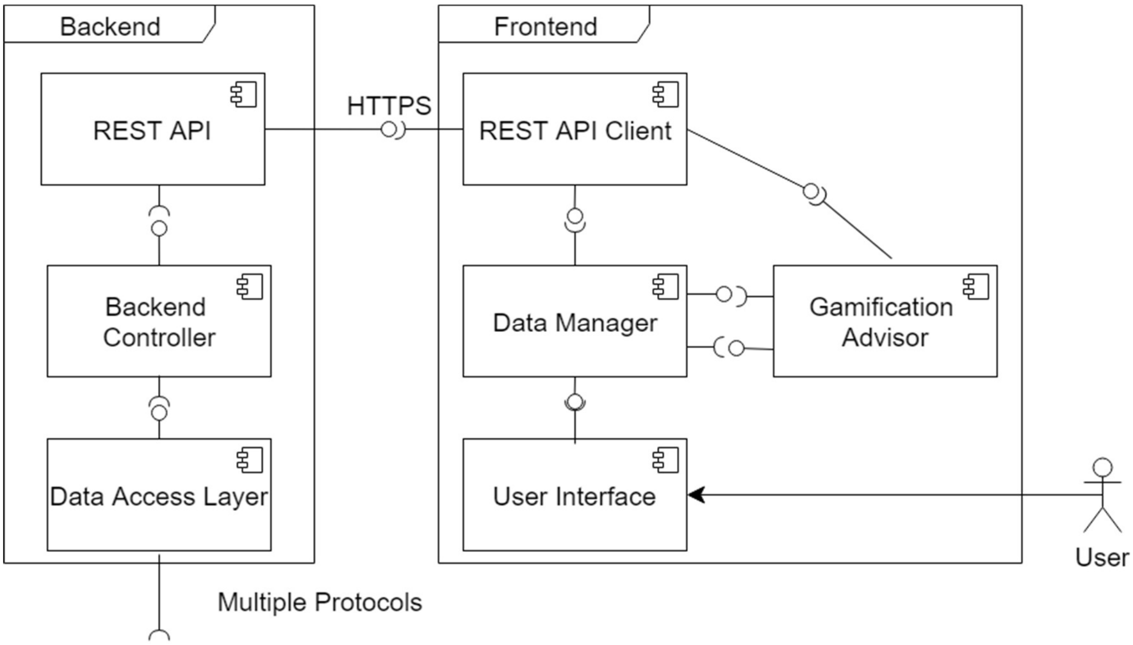


Figure 27 - Configuration Module Organization

4.5.1 Features

To properly represent the list of features in the case of the Configuration Module it is important to define two sets, the one referring to the Frontend Application and those referring to the Backend Application. Beginning with the Frontend we have the requirements regarding the visual representation and general features of the Configuration Module.

- Viewing, Editing, Creating and Deleting Concepts, Templates, Rules and Tables, in obfuscated and pure JSON formats.
- Allow a Game Element view to allow configuration based on Game Elements instead of a direct format.

Regarding Backend, the features reflect its job as the manager of data integrity and translation task management and background work to support the mentioned Frontend features. The implementation of these features takes precedence.

- Guarantee that Templates exist before allowing creation of a new Concept Instance.
- Generate sequential CTK, CIK and Key values for new objects during the creation process.
- Populate new Concept Instances with the fields from Concept Templates and their default values.
- Delete related Instances when a Template is deleted.
- Simplify data structures to allow for easier manipulation in the editor.
- Verify correctness of Rule operations and parameter definitions.
- Verify correctness of Rule calls in Concept Template and Instance triggers.

As the project's development never reached the point where Game Elements were in question structures to determine these game elements were not defined. The initial ideal is that these would contain a list of premade Concept Instances, Rules and Tables with base Concept Templates necessary already inserted into the database. Details about regarding Game Elements are to be taken as speculation and unproven to be viable due to the lack of structure definition. Game Element Templates are to, theoretical, static members of the System hardcoded into the Database, for this reason creating, editing or deleting is not an option.

4.5.2 Processes

The Configuration Module is a pure user driven application, as such none of these processes are automatic and are all executed by a request made by the Frontend. Some processes in this group basic and have no exception options besides complete failure of the entire process due to undue use, as such diagrams will not be present for these unless they possess a very specific exception.

Get Element Type Overview
Utilized to provide overview of a certain Element this process first obtains all elements of that type from the database filtering only the necessary fields and create a DTO with only the necessary data for transfer to the UI. Base Elements and Game Elements behave in the same way regarding this Process.

Get Base Element
This process requires the previous existence of an Element to proceed, if this is not the case this process is not accessible and as such this situation will be omitted from the description. In this case the reference for the Element will be sent to the server (CTK, CK or Key), the server will then respond with the Edit DTO for that specific Element.

Create New Base Element
When the process of creating a new element is started the system must first determine what the Id of that element will be within the database, in the case the database does not know how to generate it. After this first step the element is generated in its empty state and the process respond with the Edit DTO. The case of the Concept Instance Element is an exception, first the

request must contain a valid CTK to determine which Template the Instance will adhere to and before the Edit DTO can be returned it must be initialized with the default values present in the Template.

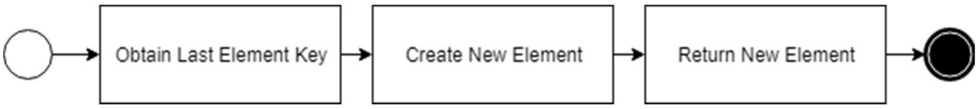


Figure 28 - Create New Base Element Process

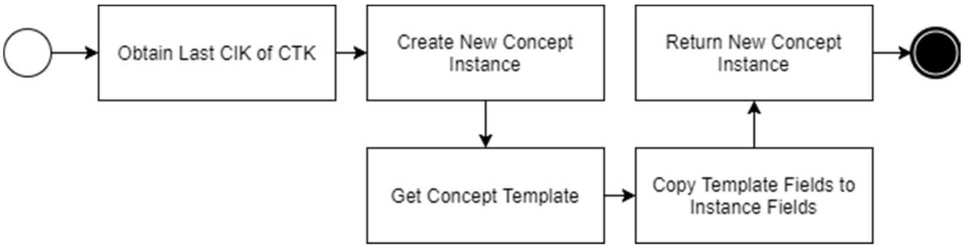

Obtain Last CIK of CTK
Create New Concept Instance
Return New Concept Instance
Get Concept Template
Copy Template Fields to Instance Fields

Figure 29 - Create New Concept Instance Process

Insert/Update Base Element

When the UI sends an element to the Backend to be updated this is a simple process of transforming the Edit DTO to a normal Object, a conversion which checks for invalid states, and then the converted Object is inserted into the database, in case of failure a 415 "Unsupported Media Type" error should be returned.

Delete Base Element

The Deletion process works exactly as the Update Element Process changing only the method executed on the Database, please refer to the Update Element Process for details on how this method works.

Get Game Element

Getting a Game Element involves first obtaining the Game Element definition from the Gamification Database this then is followed by obtaining the Concept Instances who's CK is mentioned by the Game Element, these Instances are then appended to the Game Element and an Edit DTO is created and returned.

Create Game Element

Creating a new Game Element begins by acquiring the corresponding Game Element Template, the Base Elements mentioned within this template are then generated a new Game Element is created and the Base Element data appended to this new Game Element, finally the Edit DTO is created and returned, as represented in Figure 30.

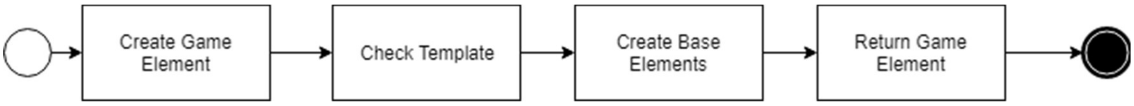


Figure 30 - Create New Game Element Process

Insert/Update Game Element
The Game Element Edit DTO is sent by the UI to the Backend. The Edit DTO is then deconstructed and the Base Elements split and updated. The main body of the Game Element is then updated in the database.

Delete Game Element
To delete a Game Element the UI must again send an Edit DTO to the Backend where the process will be the same as the Updating of a Game Element with the command being delete.

4.5.3 Software Interfaces

Spring Data
A Spring Data connection is necessary to access the Gamification Database and should contain commands for all Base Elements CRUD functionality. The following Commands are deemed necessary, auxiliary commands may be added according to implementation need.

- Select (All / One)
- Update (One)
- Insert (One)
- Delete (One)
- Obtain Last Index (NoSQL Databases)

Frontend to Backend API
The API contained within the Backend Application is utilized to manage CRUD operations triggered by the Frontend and contains a large amount of methods, in Table 21 an abbreviated list of methods can be seen, the Concept Element has a different GET call from the rest of the elements, these are added at the end of the list. POST and PUT commands require the Element to be sent in the request's body.

Table 21 – Configuration Backend API

Method	Path	Description
GET	/{element}	Get All Elements
GET	/{element}/new	Get a New Empty Element
GET	/{element}/overview	Get Overview of all Elements
GET	/{element}/{key}	Get a specific Element (Except Concept)
PUT	/{element]	Create New Element
POST	/{element}	Update Element
DELETE	/{element}/{key}	Delete Element (Except Concept)
GET	/concept/{ctk}	Gets all Concepts of a CTK
GET	/concept/{ctk}/{cik}	Get a specific Concept
DELETE	/concept/{ctk}/{cik}	Delete a Concept

4.5.4 User Interfaces

As the Configuration module serves as the method to allow a human to configure the Gamification approach within the system it requires a set of User Interfaces to allow the proper working of the Module. Three main areas exist to the Configuration Module, Raw JSON, Base Element and Game Element. The User Interface descriptions will be divided by these areas in their definition and a general interface area.

General Interfaces

Base Navigation

Composed of a top fixed Navigation Bar the Base Navigation part of the interface will always be present on all screens and it possesses links that allow for navigation to the three areas of the application. By default the application opens on the Base Element area.

Search Screens

Every single Element, Base or Game, must have an initial Search Screen. In this screen it should be possible to view a list of all elements of that type within the database as well as some basic information about them. The Raw JSON area possesses the same Element sections as the Base Element area.

This screen should allow navigation towards the Creation/Edition Screens as well as contain a button to Delete a selected Element.

It must also be possible to provide search filters to refine the presented list of elements according all parameters included in the Element's overview.

Raw JSON Interfaces

Creation/Edition Screen

Whatever option is selected the Edition and the Creation Screen possess the same design with the exception of the name of the "Save" button which should be named "Create" in the creation screen.

This screen should contain a single Textbox containing the entire JSON of the Element, properly coloured for a Light Theme editor, the colour pallet found in Studio 3T for JSON files (Every JSON file in this document utilizes this colour scheme) is suggested and indented.

Buttons

Three buttons should be present at the top of the Screen, a "Create"/"Save" button which is only available when the contents of the Textbox are valid JSON, a "Cancel" button which returns to the Element's Search Screen, with an alert message requiring confirmation of the intent to leave and a "Visual Editor" button which transfers to the corresponding Element's Base Element Creation/Edition Screen, maintaining the present JSON data.

Base Element Interfaces

General Creation/Edition Screen

Whatever option is selected the Edition and the Creation Screen possess the same design with the exception of the name of the “Save” button which should be named “Create” in the creation screen.

At the top the Metadata of the Element should be present and, by default, immutable. The data that should be present in the metadata section can be seen in Table 22.

Table 22 - Creation / Edition Screen Metadata

Element	Metadata Section
Concept Template	CTK, Tag(Editable), Direct Access(Editable)
Concept Instance	CK, Tag(Editable), Direct Access(Editable)
Rule	Key, Name(Editable), Condition(Editable)
Table	Key, Table Size (Calculated), Name(Editable), Return Type (Editable)

For each one of the Elements different, collapsible, sections need to be available.

Concept Template and Concept Instance Sections

Two sections are shared by the Concept Template and Concept Instance, the Fields Section and the Trigger Section.

The Fields Section contains data on all the fields of the Instance/Template, in the case of the template one Checkbox is at the top, determining the inclusion or not of the “ref” special field in the concept. The rest of the section contains a list, each line containing the name, the type and the value (Default value in the Template’s case) and a button to remove a field. At the end a button should be present to allow the creation of a new field.

The Triggers Section is the same in both cases and is composed by three subsections “Auto”, “Explicit” and “Prebuild”. All contain a list containing the key of the rule to trigger, the parameters to send to the rule and a button to delete the trigger from the list. At the end of each subsection a button allowing for the creation of a new trigger is present.

Concept Instance Sections

Besides the sections shared with the Concept Template concept Instance possesses two additional ones, the Stack Section and the Composing Section.

The Stack Section contains a list in which the lines have, the position of the concept within the Stack, the CK of a concept and a button to delete a concept from the Stack. A button is present at the bottom to add a new CK to the Stack.

The Composing Section contains a list in which the lines have, a CTK and a list of CIK’s of that CTK, at the end a button to remove the CTK, CIK’s pair from the list is present. At the bottom of the list a button to add a new Composing Concept group is available.

Rule Sections

Rule has three sections, the Parameters Section the Then Actions Section and the Else Actions Section, these last two sections being identical.

The Parameters Section holds a list of parameters consisting of a key, the parameter type in a Dropbox, a checkbox to notify that the parameter is received or static with the name “Received”, the parameter value which is disabled in the checkbox “Received” is checked, and a button to delete the parameter from the list. Also, a button that allows the addition of a new parameter to the list is present at the end.

The Then and Else Action Sections consist on a list of Actions with the fields Operation, Result Type and Result Variable and a button to remove and action, A button is also present at the bottom of the list to add a new Action.

Table Sections

Table has three sections, an Axis Overview Section, an Axis Sections and a Value Section.

The Axis Overview Section lists all axis with their Name and Size as well as a delete button if more than one Axis is present. The number of axis is limited to two in the Base Element Screen to allow for proper representation of values. If the number of axis is lower than two a button is present under the list to add a new Axis of 0 size.

The Axis Section possesses a Dropbox with all the axis defined within the Axis Overview and shows a list of all the AxisVal’s within that Axis under the dropdown. This list consists of line with the type of AxisVal, selected from a Dropbox and the Value associated with the AxisVal as well as a button to remove that AxisVal from the list. A button exists at the end of the list that allows the addition of a new AxisVal.

The Values Section shows a list of all values within the table to a max of 2 Axis in a matrix fashion. Values can be changed to the intended value in this Section, but their type is enforced to be the one defined in the Return Type Metadata option.

Buttons

Three buttons should be present at the top of the Screen, a “Create”/”Save” button which is only available when the contents of the Textbox are valid JSON, a “Cancel” button which returns to the Element’s Search Screen, with an alert message requiring confirmation of the intent to leave and a “Raw Editor” button which transfers to the corresponding Element’s Raw JSON Creation/Edition Screen, maintaining the present Element data.

Game Element Interfaces

Game Element Interfaces are defined on a Game Element basis and don’t allow transition to the other two as they generate multiple Base Elements, dependent on the case. The project never reached the stage where these interfaces were necessary to be defined therefor definition does not exit.

4.6 User Interface Application

Not technically part of the System the User Interface Application was the original objective of the project and everything else is built only to provide support to this application. It was done to allow and interface for the user for a Gamification Approach.

4.6.1 Features

After modification of the Original Approach to the SDA Approach the set of features changed, as can be seen in the Overview section of this document. Omitting the old requirements and focusing on the new we have the present list:

- Basic Login and Logout options
- Allow messages from the "Overseer" to be shown at any time.
- Allows for the visualization and completion of Operations of multiple types.
- Allow visualization of the Global Operation, including risk zones, changes and personal impact.
- Provide constant visualization of Agent Rank, Experience Points and Reputation Points.
- Allow for the presentation of a Ribbon Showcase showing all ribbons earned by the player.
- Allow a Store where rewards can be visualized and claimed utilizing Reputation Points.
- Allow the visualization of Statistics collected by the system.
- Allow the visualization of earned Agent Bonus with breakdown of how it was calculated.
- Allow the visualization of a Score history.
- Allow the visualization of Leaderboards.

The Backend Application to support the interface has an additional set of features regarding in memory storage of Concepts and managing all user sessions to avoid frontend failures, it also works to obfuscate more complex interactions with other systems:

- Validate User Session with the Gamification Module.
- Store session data in memory temporarily.
- Supply an API that obfuscates REST requests to the Gamification Module.

4.6.2 Processes

For this list only complex processes that involve manipulation of data will be presented, processes regarding simple data display with no user interaction are mentioned in the User Interface section.

Login
When first entering the application the user will be asked to enter their credentials as to unlock their entry into their pages. The user will then send their username and password via the form.

The Backend of the UIA will then send this username and password to the proper authentication server from which it will receive a session token. With this token the backend will create a session for the user, retrieving their associated Concept Instances and returning the session token to the Frontend application.

Logout

When the user presses the logout button present in the interface or after a predetermined amount of time the Logout Process begins. This process consists on removing the user’s session from the system and erasing all saved Concept Instances present in memory specifically related to that user.

Present Message

To address the Overseer character this process that has to be triggered by utilizing an Observer is to be implemented. In this case the observer will see a message has been added to an Observed list, the observer will the present a message in the middle of the main window of the UIA with the content contained within the message.

Claim Operation Reward

When an Operation is deemed complete a button that allows the user to claim it is shown. When this button is pressed the concept regarding that Operation is obtained and all explicit Rules defined by that case are explicitly called by the Backend Application. After this all the affected Concepts are reloaded from the Gamification Module to reflect the changes in system state.

Purchase Reward

Purchasing a reward from the store works exactly as claiming the rewards from an Operation, with the only difference being the rules executed.

Present Global Mission Data

The method of how to execute this process was never determined due to the need for a mapping software where an overlay would be displayed utilizing previously calculated data which would be obtained from the Secondary Data Source.

4.6.3 Software Interfaces

As single interface exists to allow communication between the Backend and Frontend of the application, however, the Backend connects to the Gamification Module to obtain all the presented data.

Table 23 – Gamification API

Method	Path	Description
POST	/auth	User Login
POST	/auth/logout	User Logout
GET	/player	Get the user’s Player Concept
GET	/player/policy	Get the user’s Policy Concept
GET	/player/stats	Get Player’s Statistics Concept
GET	/player/policy/scores	Get Policy’s Score Concepts
GET	/player/badges	Get Player’s Badge Concepts
GET	/player/rewards	Get Player’s Reward Concepts
GET	/player/rewards/{cik}	Claim a Reward
GET	/player/missions	Get Player’s Attributed Missions
GET	/player/missions/{cik}	Complete a Mission
GET	/badges	Get all Badges
GET	/rewards	Get all Rewards

4.7 SDA Model Mapping

This section contains the details of how the SDA Approach was implemented within the defined Models. This includes an overview of the present Concept Templates, Concept Template details and expected Composing Concepts of Instances.

4.7.1 Template Overview

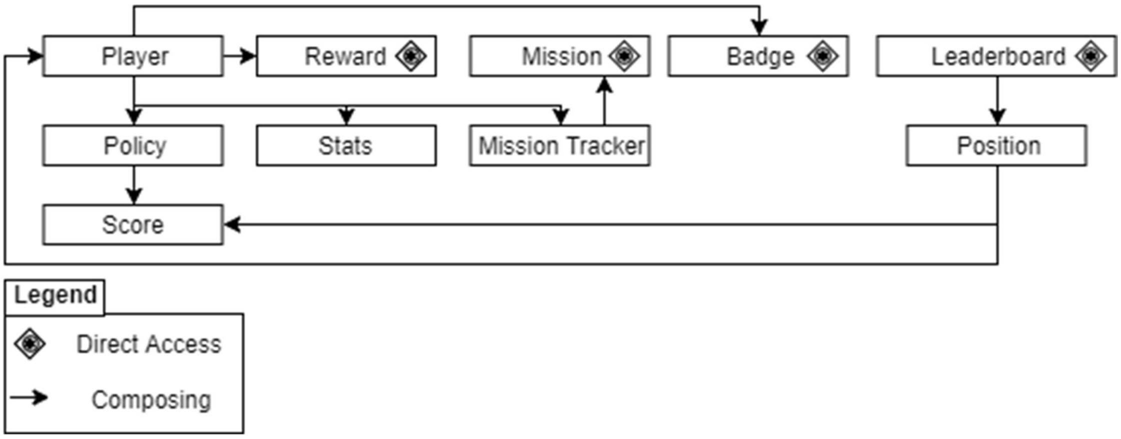


Figure 31 - SDA Template Diagram

As can be seen in Figure 31 a total of ten Templates exist within the system. More extensive models were lightly experimented with but were immediately discarded. These ten templates are then utilized to define the gamification approach.

As noted in Figure 31 four Templates are classified as “Direct Access”, Templates classified in this method mean any concepts belonging to this Template can be obtained in bulk by requesting all concepts of that Template. The reasons for this classification are:

- **Reward**: Rewards are required to be presented in bulk to the player without previously requiring the rewards to be obtained therefore no references to the group of Rewards exist.
- **Badge**: See Reward.
- **Mission**: Missions are to be attributed by the SDA Application utilizing a semi random system which require the previous knowledge of all missions.
- **Leaderboard**: Leaderboards are mostly independent and generally public, the only private information is the Position, which must be filtered by the SDA backend to guarantee it refers to the user requesting the information before being shown.

4.7.2 Template Details

Each Template contains information required to determine the concepts of that type, the specific fields were described previously in section 4.2. Here the specific utilization in each template.

Player
The Player template represents the player within the system, the details of can be seen in Table 24. It holds a reference to the customer that can be utilized to connect the gamification module information to a customer in the overall system, this is never utilized inside the gamification module.

Table 24 – Player Template

Tag	**player**	
Trigger	**Rules**	
auto	Check Level(exp)	
explicit	-None-	
prebuild	-None-	
Fields	**Type**	**Default Value**
ref	String	“”
level	Number	0
name	String	“”
reputation	Number	0
exp	Number	0

Policy
Policies present some small details about the actual policy that can be shown in the system, this includes values regarding premiums but also a display name for the policy object, in this case a car, which is represented by the licence plate in the example.

Table 25 - Policy Template

Tag	**policy**	
Trigger	**Rules**	
auto	Badge [Fuel Type]	
explicit	-None-	
prebuild	-None-	
Fields	**Type**	**Default Value**
ref	String	“”
base_value	Number	0
policy_object	String	“”
max_discount	Number	0
fuel_type	String	“Electric”
start_date	Number (Saves in Epoch)	0
end_date	Number (Saves in Epoch)	0

Score

Each time period, Week, Month or Year, a score is generated within the core system that is transferred together with the policy within the composing concepts by transferring the reference. This score is then utilized within the system to provide the user with statistics regarding their performance, to attribute badges and to calculate completion of missions.

Table 26 – Score Template

Tag	**score**	
Trigger	**Rules**	
auto	-None-	
explicit	-None-	
prebuild	-None-	
Fields	**Type**	**Default Value**
ref	String	“”
date	Number (Saves in Epoch)	0
period	String	“Week”
score	Number	0
km	Number	0
abrupt_breaking	Number	0
sudden_accelerations	Number	0
average_speed	Number	0
max_speed	Number	0
night_time_hours	Number	0
average_zone_risk	String	“Medium”

Stats

Stats are utilized to keep track of the player’s progress through the game, here spending, top scores and total of o completed missions and obtained badges are shown. This is also where rules to attribute most badges are present.

Table 27 – Stats Template

Tag	stats	
Trigger	**Rules**	
auto	5([missions]), 6([max_score])	
explicit	-None-	
prebuild	-None-	
Fields	**Type**	**Default Value**
reputation_earned	Number	0
reputation_spent	Number	0
ribbons	Number	0
missions	Number	0
missions_week	Number	0
missions_month	Number	0
missions_year	Number	0
avg_score	Number	0
max_score	Number	0
total_paid	Number	0
total_discount	Number	0

Mission

Missions work as objectives for the player to achieve they have a name and a description, as well as a time period, weekly, monthly and yearly as a timeframe to be achieved. Default values can be seen in Table 28.

Table 28 – Mission Template

Tag	mission	
Trigger	**Rules**	
auto	-None-	
explicit	-None-	
prebuild	-None-	
Fields	**Type**	**Default Value**
name	String	“Mission Name”
description	String	“Mission Description”
reputation	String	0
exp	String	0
type	String	“Weekly”

Mission Tracker

As Missions are shared by all players therefore, to allow for multiple missions to be attributed to multiple players, Mission trackers exist. Mission trackers track if the mission was complete or not and also contain the rules to check if a mission has been completed or not. A difficulty field is also present, but it is purely cosmetic and only utilized for presentation to the user, the difficulty of the mission is derived from the values set in the “Check Mission” rule.

Table 29 – Mission Tracker Template

Tag	**Mission_tracker**	
Trigger	**Rules**	
auto	-None-	
explicit	Check Mission (requirements) Complete Mission (completed, unlocked)	
prebuild	-None-	
Fields	**Type**	**Default Value**
completed	Boolean	false
unlocked	Boolean	false
dificulty	String	“Basic”

Badge

Badges are extremely simple, requiring only a name, description and image, as can be seen in Table 30. They possess no internal rules as all rules to attribute badges are set in other templates, in this example the Statistics and the Policy Templates.

Table 30 – Badge Template

Tag	**badge**	
Trigger	**Rules**	
auto	-None-	
explicit	-None-	
prebuild	-None-	
Fields	**Type**	**Default Value**
name	String	“Badge Name”
image	String	“Default”
description	String	“Description”

Reward

Rewards are, in this example only policy extensions, their purchase within the game doesn’t change any mechanic as the process with which they have to be treated has to be done by the Core system. In the module the only thing rewards have is a cost, name and description, as well as a rule that allows the rewards to be purchased by the player.

Table 31 – Reward Template

Tag	**reward**	
Trigger	**Rules**	
auto	-None-	
explicit	Buy Reward (cost)	
prebuild	-None-	
Fields	**Type**	**Default Value**
cost	Number	0
name	String	“Reward Name”
description	String	“Description”

Leaderboard

Leaderboards are shared by all the players, leaderboards contain basic information such as dates and scope, all the specific information is stored in the Position elements, which are available to the player. Leaderboard templates contain a single prebuild trigger, which serves to calculate the leaderboard positions.

Table 32 – Leaderboard Template

Tag	**leaderboard**	
Trigger	**Rules**	
auto	-None-	
explicit	-None-	
prebuild	-None-	
Fields	**Type**	**Default Value**
date	Number (Saves in Epoch)	0
period	String	“Week”
area	String	“Local”
level	String	“Global”

Position

Utilized as part of leaderboards Positions serve to provide only a partial view to the player, not allowing the sharing of players information amongst each other but still allowing a sort of competition, these elements utilize the internal ref to identify the user.

Table 33 – Position Template

Tag	**position**	
Trigger	**Rules**	
auto	-None-	
explicit	-None-	
prebuild	-None-	
Fields	**Type**	**Default Value**
ref	String	“”
score	Number	0
position	Number	0

4.7.3 Rules

Rules are built as seen in 4.3, however, for brevity in this section, only overall information about the rules can be seen in Table 34 representing which rules are deemed necessary in this system.

Table 34 - Rule Table

Rule Name	Parameters	Description
Claim Mission	completed unlocked	Claim the rewards from a mission in the event it is completed and unlocked.
Update Statistics	completed unlocked type reputation	If the mission is completed and unlocked update the details on the statistic concept.
Calculate Statistics	-None-	Calculate statistics fields based on present information (does not calculate completed missions)
Attribute Badge	Variable	Required to verify if a badge has can be claimed or not, parameters will vary according to each specific badge but it will always add a badge or do nothing if conditions are not met.
Check Mission	Variable	Missions required to verify if a badge has been completed on not, will set the mission as completed if true
Buy Reward	-None-	Spend reputation points to claim a reward, will only add the rewards if enough reputation exist.
Calculate Leaderboard	-None-	Calculates the leaderboards based on the data of all users.
Check Level	Exp	Checks what level the player is according to the exp.

4.7.4 Tables

A few Tables are also necessary for the SDA example, these are used to store various information. A list of Tables, and their description can be seen in Table 35.

Table 35 - Tables Table

Table Name	Parameters	Description
Km / Score Title	Km, Score	Determines what title will be given to a score dependent on the score and total km travelled within the timeframe.
Fuel Badge	Fuel Type	Returns which badge to attribute according to what fuel type the players vehicle uses.
Level / Exp	Exp	Returns what level a player should be according to their exp value.

5 Development

Chapter Contents

5.1 Implementation Overview

Before implementation could be initiated the technologies and tools had to be selected. This section provides a listing of the technologies and tools utilized, justifying the reason for their choice and what they were utilized for within the system. The listing of technologies and tools utilized can be seen in Table 36 and Table 37

Table 36 - Utilized Technologies

Technology	Usage
Java 8	Utilized to implement the Backend applications and gamification module
Angular 5/6	Utilized to implement the SDA application's and the configuration's interfaces.
Maven	Utilized to manage dependencies on Java 8 projects.
Docker	Utilized to create containers to simulate production environment.
JSON	Utilized as the basic data transfer structure between applications and to save data in the database.
MongoDB	NoSQL database engine utilized to store all information of the module.
Spring Framework	Utilized to facilitate implementation of the backend systems.
Swagger	Utilized to document all RESTful APIs across the project.
Tomcat	Embedded into the Spring Boot project which encapsulates the Spring Framework it is used to allow the execution of Spring applications without deployment to a sever.
Wildfly	Utilized in docker containers to simulate production environment.

Table 37 - Utilized Tools

Tool	Usage
Eclipse IDE (Oxigen/Photon)	IDE utilized for development of Java 8 projects.
Visual Studio Code	IDE utilized for developing Angular projects
MongoDB Compass Community	GUI supplied by MongoDB to visualize databases, eventually swapped to the more feature complete 3T Studio.
3T Studio	GUI to allow manipulation of NoSQL databases, eventually swapped for NoSQLBooster as licences expired.
NoSQLBooster for Monog DB	GUI to allow manipulation of MongoDB databases. Free Version utilized.
Docker for Windows	Docker for Windows was utilized to create and manage the Docker containers
Kinematic	Kinematic allows a GUI to manage Docker containers.

5.1.1 Utilized Terminology

Before each of the descriptions of module implementations each of the processes was summarily evaluated and their implementation state and testing stage determined. In this section the definitions for each of the values noted in these evaluations is explained.

Implementation State
Refers to the completeness of a certain process, these are based on a subjective assessment of this completeness state and can be seen in the following list:

1. Not Implemented
2. Partially Implemented
3. Implemented

Testing Stage
Refers to the range of the tests executed in a certain process to guarantee functional correctness, these classifications are based on the usual test levels found in software applications, the latter levels of test, requiring the previous utilization of the posterior level:

1. Untested – No Testing was done
2. In-Development Tested – Testing was done during the development of the process by utilizing the application. This can be classified as being Acceptance Testing, however, due to the unstructured nature of this process it will be set under any other type of testing.
3. Unit Tested
4. Integration Tested
5. System Tested
6. Acceptance Tested

5.2 Implementation Process

As the proposed system utilized a significant amount of applications a process of development was outlined following mostly a mostly standard, although striped down software development process, composed of three parts, Analysis, Design and Implementation. This process didn’t contain the maintenance nor planning stages as the planning was mostly to be done during the Analysis process and Maintenance was unnecessary at this stage.

This type of flow allowed a method of development were design flaws detected during the implementation of the system could be quickly fixed and implemented.

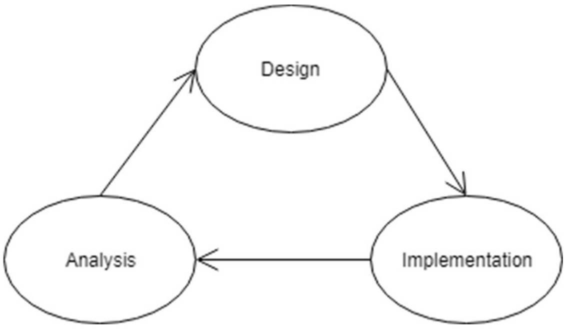


Figure 32 - Software Development Process

During the development of this a standard cycle of 2 weeks was used for that process, the first 2 days of a week being the analysis and design stages and following 7 days implementation and the last day an analysis of the implementation stage. At the start of the first week the meeting with the company was held. This process was, however, preceded with an initial analysis and design.

5.3 Implementation Stages

For the development of the design solution certain implementation stages were determined, as mentioned previously, adaptations to the stages were made during the development to address requests by the company. These stages mimic the parts of a classic temple Foundation, Pillars and Pediment. Foundation being the implementation of the core functions of the system, Pillars the general overall development and finally Pediment the final touches such as UI improvements and extensive verifications over all applications. Between each stage a week was taken to properly integrate the loose documentation created into this document.

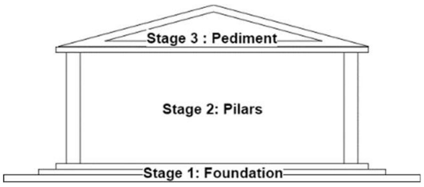


Figure 33 - Implementation Stages

5.3.1 Stage 1: Foundation

The Foundation stage is where the base of the design is implemented, at the end of the stage a basic implementation of the concepts is done, this stage presents minimal testing parameters and low failure tolerance, its only objective is to present a basis of the 3(4 if considering backend/frontend) applications and 1 database that compose the scope.

Objectives

1. Core Gamification Module System in a minimum viable working stage.
2. Structure of all different data types determined, defined and validated.
3. Reasonably testing battery for Gamification Module. (50% Coverage goal)

Stage Review

Taking around 2 Months to conceptualize, implement, change and test, the development of this initial stage of the gamification module was delayed by a lapse in the scope comprehension during the initial design, a lack of understanding of the technologies being used, namely the Spring Framework and JPA, which required study delaying the entire implementation process.

The resulting project at this stage has some implementation problems, a reasonably amount of unstable code and poor authentication systems implemented. The result is a bare minimum prototype of the system that, although it contains all necessary features, requires significant extension to be 100% functional. Eventually a refactor of this system will be necessary to provide a more stable application without unnecessary code and proper verification of data all along the way. This will be done in a future stage.

During the process of the implementation of Stage 1 the top level design of the system was very much unchanged as it proved good enough for implementation, a number of features and data was however missing from the more specific design level, processes required some slight alterations from the initial design as more requirements were detected but in general the sweeping changes came to due to mistakes made during the implementation of the chosen design or the introduction of new frameworks and technologies into the project which required major refactoring.

Objective Completeness

Core Gamification Module System in a minimum viable working stage (Completed)

Although the system appears to work completely fine the lack of tests in many areas don't allow for certainty of complete system stability, however, it can be used without errors for the most part.

Structure of all different data types determined, defined and validated (Competed)

Although the structure of most data types in the system will not change, requirements regarding the function of the Configuration Module were still uncertain during the writing of this paragraph and as will be stated in the detailed implementation report these structures may require change.

Reasonably testing battery for Gamification Module. (Incomplete 25%)

Test coverage is extremely low, future tests will be developed but as of now tests are functional only, not hitting limit values and only a very small amount runs, this is due to the fact that a great deal of tests made before the switch to Spring Framework stopped being viable, and the changes in many class structures broke much of the rest. A new battery of tests needs to be made.

5.3.2 Stage 2: Pillars

The Pillars stage is the longest stage and is where the features, determined previously, are implemented and tested. By the end of this stage most of the features of the core systems, gamification module, UI Logic/Data levels and Configuration Module Logic/Data levels are complete.

Objectives

1. Configuration Module Backend and Frontend implemented and working.
2. Seamless interaction with the database by both modules
3. Extensive test battery for both modules. (70% Coverage goal)
4. Deployment Test.
5. Rudimentary User Interface Application implementation.

Stage Review

Taking quite longer than Stage 1 was expected, however, some unexpected situations did occur that changed the tacking of objectives causing an imbalance in completion. Initial slow development due to the use of Angular 5 short taking only about one week but missing information within the configuration required refactoring of old code in the Gamification Module and the restructuring of Concept and Template JSON files.

The Configuration focus on functioning as a visual way of editing concept JSON first proved perfectly feasible and by the end of the first month all types of concepts with the exclusion of UITemplates were configurable by the Configuration Module. Considerable time was, however spent in attempting to realize the Configuration of UITemplates, a process that took two weeks after which it was left uncompleted as starting work on the User Interface Application was paramount to the success of this stage.

After the definition of the SDA, which was done in a week's time the process of implementing the User Interface Application was started with mixed results. Interface implementation took the time of the first two weeks with minimal functional evolution as it was mostly focused on the backend while the front end was laid out and prepared to receive values. Following this displays and controls for missions and rewards, with accompanying configurations took around another week time. At this point needs to alter the Concept and Template files arose once more and the remaining time was spent updating the System to accommodate these changes and making changes to allow the needed flexibility of the Rule Handler to accommodate the needs of the SDA Approach.

Objective Completeness

Configuration Module Backend and Frontend implemented and working. (Completed 35%)

Although the Configuration Module allows for edition of all the Model structures the completion rate is estimated at 35% for two main reasons. First it still requires development for proper forms for a few fields, mainly the rule definitions which does not hold enough space to even represent an entire operation at once. And secondly the obfuscation of the configuration to the Game Element perspective was never implemented, which accounted for a great part of the Configuration Module Completeness according to the set requirements.

Seamless interaction with the database by both modules (Completed 90%)

Both Modules can access and operate the database without any issues with one exception that preferably should be avoided but it may occur. There is a possibility that, if the Configurator and Gamification Modules are both being executed simultaneously, and the Gamification Module loads a Concept, then, before the General Update task is executed, the Configurator makes an alteration in said Concept and saves it to the database, during the eventual General Update the Configurator's changes will be overwritten by the Gamification Module.

Extensive test battery for both modules. (Completed 50%)

Unit testing was done to the Model classes in both modules, for all other parts of the processes testing was made during development or utilizing Postman to make predefined calls to the defined APIs.

Deployment Test (Completed 40%)

The prototype was deployed in Docker containers made to simulate the deployment environment. Although all applications and databases were deployed successfully to the Wildfly Containers not all of them functioned properly, multiple errors in communication regarding all modules exist, these were mostly caused by the authentication procedure and were not successfully fixed.

Rudimentary User Interface Application implementation (Completed 90%)

The User Interface Application is almost complete, having only three features unimplemented, two of which aren't part of the set determined for a rudimentary level (Global Operation Display and the Overseer Messages), therefor leaving only the Leaderboards feature not implemented regarding this rudimentary implementation.

5.3.3 Stage 3: Pediment Definition

The final Stage, Pediment, revolves around improving both the User Interface Application and the Configuration Module Frontend to a decently looking state, preferably conforming to the interface standards at i2S, adding extensive verification and error handling to the entire system as well as extending the data access systems of both applications to more databases/database types and improving the gamification aid present in the Configuration Module.

Objectives

1. Extensive test battery (≈90% Coverage goal)
2. Extensive error handing and prevention.
3. Improved UI
4. Extended Gamification aid.
5. Prebuilt Connections to most widely used SQL and NoSQL databases.

Stage Review

This stage was never initiated as errors in the design stage caused revisions to be made in the original applications, requiring additional time to refactor great parts of the developed code.

5.4 Gamification Module Implementation

The first to be implemented the Gamification Module is the most complex of all the applications on display in this report, handling all the logic of the system and having the most failure points of all applications. As such it required additional care having been completely refactored three times during development.

5.4.1 Configurations

For this module it was necessary to configure database access as well as security and tasks, since this application was created utilizing the Spring Framework this was done with Spring Configuration classes.

For this configuration to work annotations must be added to the Entry class which can be seen in Code 7. An important detail to note is that these annotations only consider database access utilizing MongoDB, other NoSQL databases will not function without alterations nor will SQL databases.

The configuration of the MongoDB database required no additional files to work but problems regarding the conversion of data from database to application required a configuration file to be made that registered a conversion from Object to Object[].

```
@SpringBootApplication
@EnableMongoRepositories
@EnableAutoConfiguration(exclude= {DataSourceAutoConfiguration.class})
@EnableAspectJAutoProxy
@EnableScheduling
public class GamificationModuleApplication
```

Code 7 – Code Excerpt from GamificationModuleApplication.java

To prepare for additional databases to be utilized in the future a Spring Framework factory was created. This required the creation of a configuration that implements the logic to select the proper component to instantiate. This component will follow an interface called *DataAccessStrategy* and be tagged with a Component annotation that designates the handled database.

Regarding the configuration of task execution, it required a simple configuration class which created a *TaskExecutor*, which is part of the Spring Framework. Tasks were then attributed a @Scheduled annotation which determined their periods of execution.

The security configuration utilizes JWT authentication and was the most complex! this required both an extensive configuration class and an API though which credentials and checked and tokens retrieved. In the configuration class it was also necessary to instantiate a PasswordEncoder, in this case BCryptPasswordEncoder was used as it was the one presented in the example studied and its presence is to be removed from this application and delegated to another application finally it is necessary to configure the HttpSecurity, which details where authentication will be executed from, session creation policy and paths to require authentication and WebSecurity, which allows specific requests or resources to be ignored when accessing requests.

To allow for easier customization by part of the user all details that could be configured are present in an *application.properties* file, this file can be seen in its totality in annex I. In Table 38 it is possible to see what each configuration annotation's purpose is.

Table 38 – Property Description

Property	Definition
data.access.database	Database to use.
spring.data.mongodb.host	Location host of the database.
spring.data.mongodb.port	Port to connect to the database.
spring.data.mongodb.database	Name of the database to connect to.
ref.provider.url	Url of the ref provider.
ref.provider.token	Token to identify the application in the ref provider.
task.[taskname].delay	Initial delay before a task starts execution.
task.[taskname].rate	Delay between task executions.
jwt.header	Header where the JWT token is present.
jwt.secret	Application's secret.
jwt.expiration	The expiration date for tokens.
jwt.route.authentication.path	Path for authentication requests.
jwt.route.authentication.refresh	Path for token refresh requests.

5.4.2 Data Access

This application has two methods of accessing data, the first is utilizing the Repositories to access a database and the second via a REST Client that connects to the Insurance Core applications which contains the specific data of clients and policies.

The number of Repositories to deal with the access of data is not predetermined as different database may warrant a different type of database access. In this case utilizing MongoDB five repositories were created, four (ConceptRepo, ConceptTemplateRepo, RuleRepo and TableRepo) to deal with model data and one (UserRepo) to allow access to the user credentials by the security layer. These Repositories are utilized, exclusively, by a class that implements *DataAccessStrategy,* whose methods cfTesan be seen in Table 39.

Table 39 – Gamification Module DataAcessStrategy methods

Method	Return	Description
User		
getByUsername(String)	User	Get a user by its username.
insertUser(User)	User	Creates a new user in the database.
Concept		
getAllConcept()	List<Concept>	Get all concepts.
getAllConceptsOfType(int)	List<Concept>	Get all concepts of a type.
getConcept(int,int)	Concept	Get a specific concept.
getConceptByRef(String)	Concept	Get a concept by the Ref field.
insertConcept(Concept)	Concept	Creates a new concept.
updateConcept(Concept)	Concept	Updates a concept
getLastCikOfCtk(int)	int	Obtains the last index of a type.
ConceptTemplate		
getAllTemplates()	List<ConceptTemplate>	Get all concept templates-
getTemplate(int)	ConceptTemplate	Get a specific template.
getAllPrebuildTemplates()	List<ConceptTemplate>	Get all templates with at prebuild field.
Rule		
getAllRules()	List<Rule>	Get all rules.
getRule(int)	Rule	Get a specific rule.
Table		
getAllTables()	List<Table>	Get all tables.
getTable(int)	Table	Get a specific table.

The REST Client has two main features, requesting a new Concept with a specific ref and requesting an update to an existing Concept. This is executed whenever a concept with a ref field is created or loaded, respectively.

5.4.3 Processes

The Processes detailed for this Module in section 4.4.2 suffered some alterations due to implementations requirements, and some were not able to be fully implemented. Overall most Processes were completely implemented and had testing, even if done only during development and not in a structured manner. In this section a detailing of the implementation of each process will be done. In Table 40 it is possible to see an overview of the state of each Process.

Table 40 – Gamification Module Process Implementation Evaluation

Process	Implementation State	Testing Stage
Startup	Partially Implemented	In-Development Tested
Shutdown	Implemented	In-Development Tested
Request Response	Implemented	In-Development Tested
General Update	Implemented	Unit Testing
Pre-Builder	Implemented	Unit Testing
Concept Loading	Implemented	In-Development Tested
Rule Triggering	Implemented	In-Development Tested
Rule Execution	Implemented	In-Development Tested

Startup

The Startup process is handled entirely by the Spring Framework, causing a fatal failure if the database connection is impossible to set up. As this is done during the Spring Framework the stack of objects and functions called is too large for proper documentation here.

This process is qualified as Partially Implemented since the checking of secondary data sources is not made.

Shutdown

Utilizing the Spring Framework Shell a single command was added, “shutdown”, allowing for a single parameter which decides if the shutdown is forced or not. This is done by creating a class with the @ShellComponent tag.

```
@ShellComponent
public class ShellCommands {
        @Autowired
        public ConceptEngine engine;
        @ShellMethod("Shutdown")
        Public void shutdown(@ShellOption(defaultValue="false") boolean force,
@ShellOption(defaultValue="false") boolean f) {
                if(force || f) {
                        GlobalLogger.Log(this, "Executing Hard Shutdown");
                }else {
                        GlobalLogger.Log(this, "Executing Soft Shutdown");
                        engine.generalUpdate();
                }
                System.exit(0);
                return;
    }
}
```

Code 8 – Shutdown Command Class

As can be seen in Code 8 the console command calls a System.exit(0), classifying it an error free shutdown, this process cannot fail, even if the GeneralUpdate, which will be detailed in future down, does.

Request Response

Request response refers to the four methods present in the Gamification API. All of these are handled individually and require methods of execution and require individual descriptions.

getPlayer

First, we have the getPlayer() method. This returns the player authenticated with the supplied cardetials. In Code 9 we can see that this is done by obtaining the JWT user from the Principal object, then a call to the Concept Engine is made to obtain a concept via a “ref” field. This field is set in the User’s data for this exact purpose and is also, as documented in 4.7.2, present in the Player Concept. It is also of note that the return value is not the Concept itself but an APIConcept, which removes unnecessary information deemed unnecessary outside the Gamification Module namely database ids and triggers.

```
@RequestMapping(value = "/user", method = RequestMethod.GET)
public ResponseEntity<?> getPlayer(Principal principal, @RequestHeader HttpHeaders headers) {
        try {
                JwtUser currentUser = (JwtUser) (UserDetails) ((Authentication)
principal).getPrincipal();
                Concept concept = engine.getConceptByRef(currentUser.getRef());
                APIConcept api = ConceptAPIConverter.Normal2API(concept);
                if (concept == null) {
                        return ResponseEntity.status(HttpStatus.NO_CONTENT).build();
                } else {
                        return ResponseEntity.status(HttpStatus.OK).body(api);
                }
        } catch (Exception e) {
                e.printStackTrace();
                return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).build();
        }
}
```

Code 9 – API getPlayer() Method

getConceptsOfCtk / getConceptOfCk

Utilized to obtain the listing of the elements mentioned in 4.7.1, the getConceptsOfCtk() method does not necessitate to have its code presented as its work is simply checking if the template has the “direct_access” flag as true, obtaining all concepts of the specific Ctk from the database, converting them to APIConcepts, and returning the list, surrounded by a similar try…catch block to the one present in Code 9.

The getConceptOfCk() method was also included in this paragraph due to its similarity to the getConceptOfCtk() without the check for the “direct_access” flag and loading only a single concept.

postTriggerRule

The postTriggerRule() method is one of the more complex, while not necessarily a proper REST method, as it calls for the execution of code upon a resource and then the resulting resource instead of simply the resource, this method is utilized to run explicit Rules. As can be seen in Code 10, this method requires the ctk and cik of the concept and the rule to trigger, this rule is checked to belong to the use when calling the executeExplicit() method as this will fail as well as an optional Stack in the body.

This Stack is utilized to override a Concept’s default Stack to allow it to possess a different parent during the execution of that specific rule, this was originally used for Mission Trackers to allow the Mission to replace the Mission Tracker as the Rules for attribution were stored in the Mission itself, it is no longer necessary but due to its possible usefulness in a future case it was left as a possibility. This Stack override utilizes a different Rule execution method specific to this situation before running the rule and can be seen in Code 11. It functions by storing the Concept's original Stack in a temporary variable, replacing it with the Stack override during execution and restoring it after the Rule has been executed.

More details on Rule execution can be seen further along this section.

```
@RequestMapping(value = "/{ctk}/{cik}/trigger/{rule}", method = RequestMethod.POST)
public ResponseEntity<?> postTriggerRule(@PathVariable("ctk") int ctk, @PathVariable("cik") int cik,
                @PathVariable("rule") int rule, @RequestHeader HttpHeaders headers,
                @RequestBody(required = false) Stack<Integer[]> stack) {
        try {
                Concept concept = engine.getConcept(ctk, cik);
                if (concept == null) {
                        return ResponseEntity.notFound().build();
                } else {
                        if(stack != null) {
                                ruleengine
                                .executeExplicit(concept.getTemplate(), concept, rule, stack);
                                APIConcept api = ConceptAPIConverter.Normal2API(concept);
                                return ResponseEntity.ok(api);
                        }else {
                                ruleengine
                                .execute(concept.getTemplate(), concept, rule, TriggerType.explicit);
                                APIConcept api = ConceptAPIConverter.Normal2API(concept);
                                return ResponseEntity.ok(api);
                        }
                }
        } catch (Exception e) {
                e.printStackTrace();
                return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).build();
        }
}
```

Code 10 – API postTriggerRule() Method

Error! Objects cannot be created from editing field codes.

Code 11 - Stack Overide Execute

General Update

The General Update task is executed every set timeframe and deals with saving alterations and wiping the memory data in the Concept, Rule and Table Engines. This is done to allow alterations to be persisted in memory for a certain amount of time but release memory when concepts are not currently in use.

The General Update is done in a class tagged with the @Service annotation, this then contains a method with the @Scheduled tag.

```
@Service
public class GeneralUpdateTask{
        @Scheduled(fixedDelayString="${task.generalupdate.delay}", initialDelayString = 
"${task.generalupdate.delay}")
        public void scheduledGeneralUpdate() {
                while(Locks.checkDataAccessLocks());
                Locks.generalUpdateLock.lock();
                try {
                        conceptEngine.generalUpdate();

                        conceptEngine.wipeMemory();
                        ruleEngine.wipeMemory();
                        tableEngine.wipeMemory();
                }catch(Exception e) {
                        e.printStackTrace();
                }finally {
                        try {
                                Locks.generalUpdateLock.unlock();
                        }catch(Exception e) {
//Only instance of failure would be if the lock was never obtained and can be safely ignored.
                        }
                }
        }
}
```

Code 12 – General Update Task

As can be seen in Code 12 this is done by calling the generalUpdate() method from the ConceptEngine, and then proceeding to call the wipeMemory() methods of all engines. This also requires a concurrency lock, and a timeout for the attempt to lock is not present so, in a high load environment, this method may become deadlocked.

The generalUpdate() method itself consists of calling the DataAccessStrategy utilized in that specific runtime and calling for an update of all Concepts stored in memory, the exact code is present in Code 13.

```
public void generalUpdate() {
        try {
                loadedConcepts.forEach((key, value) -> {
                        this.dataAccess.updateConcept(value);
                });
                GlobalLogger.Log(this, "Updated " + loadedConcepts.size() + " Concepts");
        } catch (Exception e) {
                GlobalLogger.Log(this, "Unhandled Exception Caught", Level.SEVERE);
                e.printStackTrace();
        }
}
```

Code 13 – Concept Engine generalUpdate() Method

Pre-Builder

The Pre-Builder works as an automatic task and like the General Update, including the attempt to obtain a lock, which may cause problems in high load environments. This task has only two important instructions, first it calls for the ConceptEngine to load all concepts which have the prebuild tag, second it calls the general update method to save all data, the code of this task is present in Code 14. This is done to avoid possible losses of data due to unscheduled shutdowns or errors.

```
@Scheduled(fixedRateString="${task.prebuild.rate}")
public void scheduledPrebuilderTask() {
        try {
                engine.loadPrebuild();
                while(Locks.checkDataAccessLocks());
                Locks.prebuildLock.lock();
                engine.generalUpdate();
        }catch(Exception e) {
                e.printStackTrace();
        } finally {
                try {
                        Locks.prebuildLock.unlock();
                }catch(Exception e) {
//Only instance of failure would be if the lock was never obtained and can be safely ignored.
                }
        }
}
```

Code 14 – Pre-Builder Task

The method loadPrebuild() works by loading all templates that contain Pre-build triggers and this then proceeding to execute the prebuild rules of all concepts of that template type. The exact code of this can be seen in Code 15.

```
public void loadPrebuild() {
        List<ConceptTemplate> templates = this.dataAccess.getAllPrebuildTemplates();
        templates.forEach((template) -> {
                this.loadedTemplates.put(template.getCtk(), template);
                Integer ctk = template.getCtk();
                List<Concept> concepts = this.dataAccess.getAllConceptsOfType(ctk);
                ruleEngine.execute(template, concepts, TriggerType.prebuild);
        });
}
```

Code 15 – Concept Engine loadPrebuild() Method

Concept Loading

The loading of a Concept is done when requested by the API or the Pre-Builder and is the only Element whose loading requires more than a simple query via the DataAccessStrategy. In Code 16 its possible to see that a check to the Data Access lock group is made, as seen before this is locked during General Updates and Pre-Builds to avoid changing of data while these processes are executing.

The Concept is not the first thing to be loaded, first the existence of a valid Template for the supplied type needs to be checked. In that case the Concept’s existence in the loaded list is checked, if it doesn’t exist it is loaded from the DataAccessStrategy. The existence of the “ref” special field is then checked and if present its data is update from the RefProvider which connects to the Insurance Core. Finally, the Concept is saved into memory and the Rules executed.

```
public Concept getConcept(int ctk, int cik) {
        try {
                if (Locks.checkDataAccessLocks()) {
                        throw new LockedAccessException();
                }
                SimpleEntry<Integer, Integer> ck = new SimpleEntry<Integer, Integer>(ctk, cik);
                // Obtain Template
                ConceptTemplate template = this.getTemplate(ctk);
                if (template.equals(ConceptTemplate.INVALID)) {
                        return Concept.INVALID;
                }
                // Try to load from memory
                if (loadedConcepts.containsKey(ck)) {
                        return loadedConcepts.get(ck);
                }
                // Try to load from database
                Concept concept = this.dataAccess.getConcept(ctk, cik);
                // Check if concept exists
                if (concept == null) {
                        return Concept.INVALID;
                }
                // Update if concept has ref special field
                if (concept.getFields().containsKey("ref")) {
                        Concept updated = updateFromRefProvider(concept);
                        concept = updated;
                }
                // This has to be done before execute the rules or it will cause an infinite loop
                loadedConcepts.put(new SimpleEntry<Integer, Integer>(ctk, cik), concept);
                ruleEngine.execute(template, concept, TriggerType.auto);
                return concept;
        } catch (LockedAccessException lae) {
                return Concept.INVALID;
        } catch (Exception e) {
                e.printStackTrace();
                return Concept.INVALID;
        }
}
```

Code 16 - Concept Engine getConcept() Method

Rule Triggering

The Rule Triggering process suffered some changes from its design, being split as part of the rule execution and the is inherently tied to loading elements as the loading implies the checking of the template and the explicit call to execute rules, the rules are then joined by the Rule Engine just before execution, as such refer to Element Loading, and Rule Execution.

Two exceptions to this exist, firstly the Pre-build process calls for rule execution by utilizing the same general method as the Element Loading as can be seen in Code 15 and when the explicit rules are called via the API, which again utilizes a slightly different method by obtaining the Concept first and then calling for the Concept's Template, as can be seen in Code 10.

Rule Execution

Rule Execution was implemented mostly as determined in the design stage. The exception was the addition of executing a initial action to add the Concept itself to the fact list, so it could be utilized by the posterior actions without an explicit call.

Rule execution initiates when a call is made to the Rule Engine's execute() method present in Code 17, which proceeds to parse all existing parameters to the rule and set up a list to be sent to the Rule's execution proper.

```
public Concept execute(ConceptTemplate template, Concept concept, TriggerType type) {
        HashMap<Integer, ArrayList<Object>> triggers = template.getTrigger().get(type);
        triggers.putAll(concept.getTrigger().get(type));
        triggers.forEach((rulekey, triggerparameters) -> {
                try {
                        ArrayList<Object> parameters = new ArrayList<>();
                        Rule rule = this.getRule(rulekey);
                        // Add Default Parameters
                        parameters.add(concept.getCk()[0]);
                        parameters.add(concept.getCk()[1]);
                        // Resolve other parameters
                        triggerparameters.forEach(parameter -> {
                                if (parameter instanceof String) {
                                        String str = (String) parameter;
                                        if (str.contains("[")) {
                                                String sub = str.replace("[", "").replace("]", "");
                                                Field val = concept.getFields().get(sub);
                                                parameters.add(val.getValue());
                                        } else {
                                                parameters.add(parameter);
                                        }
                                } else {
                                        parameters.add(parameter);
                                }
                        });
                        rule.run(parameters.toArray());
                } catch (Exception e) {
                        e.printStackTrace();
                }
        });
        return concept;
}
```

Code 17 – RuleEngine execute() Method

In Code 18 we can see the method utilized for rule execution, as can be see this follows the designated process set in the design stage with the changes mentioned previously. The first command runs the premade Action THIS, this action obtains the Concept that called the Rule, this is an artefact of a previous development cycle but could have been changed so that when the run() method is called the concept is already present in the args[] array.

The condition Action is the execution to evaluate which branch is to be ran, and the proper execution is made, for reasons of clarity one of two methods, with similar code is executed executeThen() or executeElse(), their difference being the list iterated though and could have been merged into a single method execute(List<Action>). Due to the method's simplicity it was deemed more convenient to provide two different functions to increase readability, the exact code of the executeThen() function can be seen in Code 19.

```
public SimpleEntry<String, Object> run(Object[] args) throws Exception{
        resolveFacts(args);
        SimpleEntry<String, Object> retval;
        Facts thisfacts = Action.THIS.execute(facts);
        facts.append(thisfacts);
        Facts conditionresult = condition.execute(facts);
        retval = conditionresult.getLast();

        Boolean result = (Boolean) retval.getValue();
        facts.append(conditionresult);

        if(result.booleanValue()) {;
                retval = executeThen().getLast();
        }else {
                if(or.size() > 0 ) {
                        retval = executeElse().getLast();
                }
        }
        return retval;
}
```

Code 18 – Rule’s run() Method

```
private Facts executeThen()  throws Exception {
        Facts effects;
        for (Action action : then) {
             effects  = action.execute(facts);
                facts = effects;
        }
        return facts;
}
```

Code 19 – Rule’s executeThen() Method

Finally, Action execution should be mentioned, as can be seen in

Code 20, to execute each action an expression is built in the SpELExecutor, which handles the set up of environments for execution of SpEL Expressions. The expression is then internally executed in the SpELExecutor and the method getResult() is called to obtain the result of the action. This is then returned as a new Fact. This result is then backpropagated and added to the fact list as can be seen in Code 19.

```
public Facts execute(Facts facts) {
        Facts resultfacts = new Facts();
        try {
                SpELExecutor.getInstance().buildExpression(this.operation, facts);
                Object result = SpELExecutor.getInstance().getResult();
                this.result = new SimpleEntry<String, SimpleEntry<String,
Object>>(this.returntype.getKey(), new SimpleEntry<String, Object>(this.returntype.getValue(),
result));
                if(!Objects.isNull(facts)) {
                        resultfacts.append(facts);
                }
                resultfacts.put(this.result.getKey(), this.result.getValue());
        } catch (NoSuchMethodException | SecurityException e) {
                e.printStackTrace();
        }
        return resultfacts;
}
```

Code 20 – Action’s execute() Method

5.5 Configuration Module Implementation

The Configuration module is composed of two applications, one for the backend programmed in Java that handles database access and management of concepts and a frontend in Angular that provides the interface for editing. This module was not complete due to the necessity of moving on to the User Interface Application.

5.5.1 Configurations

The Configuration Module utilizes similar configurations to those present in the Gamification Module, removing the interaction with the Insurance Core. The only configuration present is that regarding the MongoDB database and DataAccessStrategy, as the Model Classes for both projects are the same the configurations and parameters are the same in both applications. Rehashing this information is unnecessary therefor for information regarding these details refer to sections 5.4.1. Regarding the Angular application no configuration files are available, as such details such as connections to the Internal API are hardcoded into the program.

5.5.2 Data Access

One detail that is different from the Gamification Module is the DataAccessStrategy, as this Module deals with the CRUD functions of the database. In this Module the DataAccessStrategy has six methods for three Elements (Concept Templates, Rules and Tables), and seven for Concepts. In Table 41 it is possible to see each of the seven methods and their description.

Table 41 - Configuration Module DataAcessStrategy methods (Abbreviated)

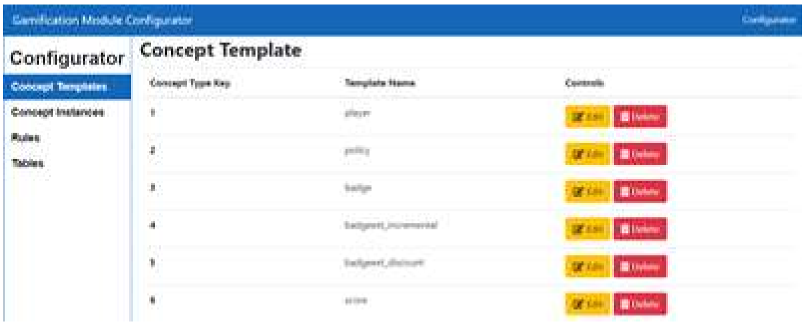
Method	Description
GetAll	Obtains all Elements.
GetOne	Obtains a specific Element.
UpdateOne	Updates one Element
InsertOne	Inserts a new Element.
DeleteOne	Deletes one Element.
GetLastIndex	Obtains the final index of an element. (Altered in Concepts to get the last index of the specific CTK)
GetAllOfType	Unique to Concept to get all Concepts of a Template.

5.5.3 User Interfaces

Described in section 4.5.4 all interfaces were created as planned. A total of five distinct interfaces exist, Overview, which is similar for all Base Elements and then the Editing interface for each of the Elements.

Overview Screen

Starting with the Overview, present in Figure 34, it contains a single table which lists all entries of that Element. Elements have different information presented here the details, but all have the Key at the start and two controls at the end. A button is also present at the end of the list to allow the addition of a new entry.



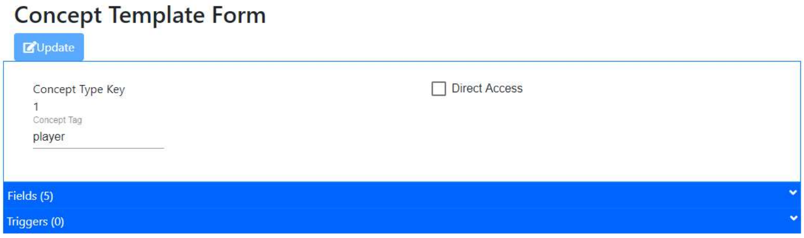
Concept Template

Concept Type Key	Template Name	Controls
1	player	Edit Delete
2	policy	Edit Delete
3	badge	Edit Delete
4	badgeset_incremental	Edit Delete
5	badgeset_discount	Edit Delete
6	score	Edit Delete

Figure 34 - Concept Template Overview Screen

Concept Template Form

The Concept Template form is composed of the basic information and a Fields and Triggers section as seen in Figure 35. In the basic information the CTK, concept type Tag and Direct Access Flag are present.



Concept Template Form

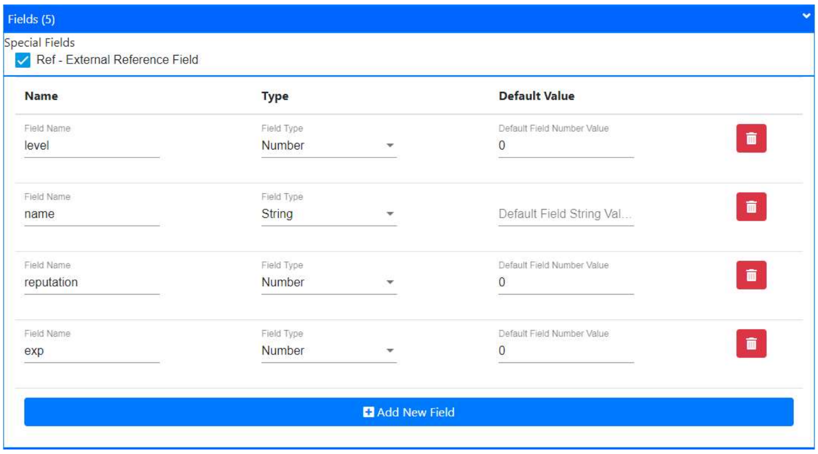
Concept Type Key
1

Concept Tag
player

□ Direct Access

Figure 35 - Colapsed Concept Template Form

The Fields Section, in Figure 36, is a table which contains the field name, type and default value as well as a button to remove the field. Before the Fields list a section for special fields exists which only holds a checkbox to add or remove the external reference field “ref”. At the bottom of the scree a button to add an empty field to the list is present, this new field will invalidate the form until its fully detailed.



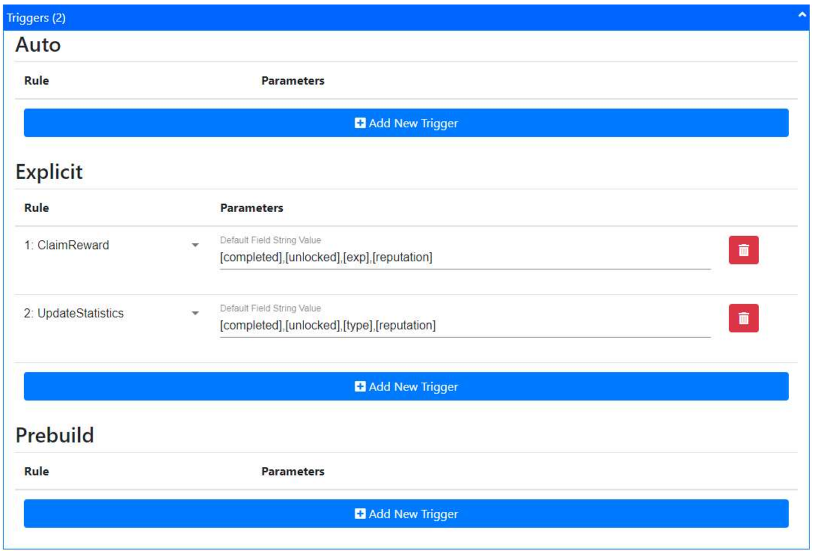
Special Fields

☑ Ref - External Reference Field

Name	Type	Default Value
Field Name level	Field Type Number	Default Field Number Value 0
Field Name name	Field Type String	Default Field String Val...
Field Name reputation	Field Type Number	Default Field Number Value 0
Field Name exp	Field Type Number	Default Field Number Value 0

Figure 36 – Fields Section

The Triggers section holds three subsections, one for each type of trigger (auto, explicit and prebuild) these sections can hold multiple Rule triggers, as seen in Figure 37. Each of these sections contains a table in which each row contains the Rule to trigger, a list of parameters and a button to remove the trigger from the list, bellow a button is also present to add a new trigger.



Auto

Rule	Parameters

Explicit

Rule	Parameters
1: ClaimReward	Default Field String Value [completed],[unlocked],[exp],[reputation]
2: UpdateStatistics	Default Field String Value [completed],[unlocked],[type],[reputation]

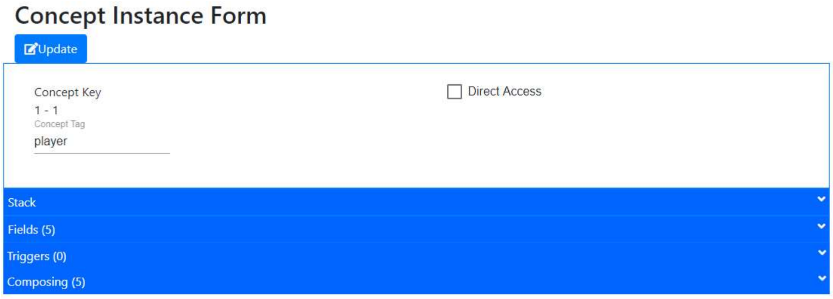
Prebuild

Rule	Parameters

Figure 37 - Triggers Section

Concept Instance Form

The Concept Instance Form contains much the same data as the Concept Template Form, including the same basic details (with CTK swapped by the CK) and a Fields and Triggers Section. This form also contains two additional sections, the Stack and Composing Section. The Fields section has its default values inherited from the Concept Template and these fields cannot be removed, the Triggers section is not inherited from the Concept Template and may be used to add Instance specific Triggers. This form can be seen in Figure 38.



Concept Instance Form

Update

Concept Key
1 - 1

Concept Tag
player

□ Direct Access

Stack

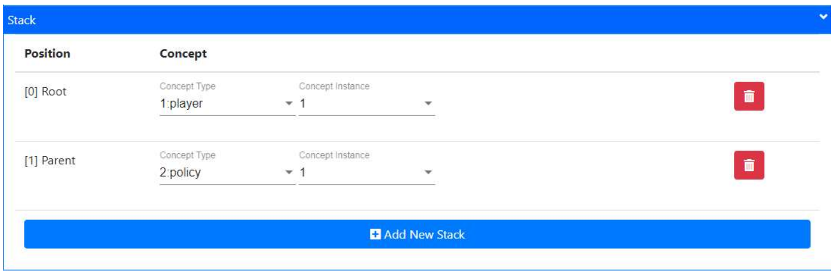
Fields (5)

Triggers (0)

Composing (5)

Figure 38 - Colapsed Concept Form

The Stack section serves to determine Concept hierarchy, in Figure 39 it is possible to see the Stack of a Score. Scores are belonging to a Policy and Polices belong to a Player, in this case this Score belongs to Player 1-1’s Policy 2-1. Stacks can be changed by removing Stack members with the delete button and adding new Stack members by pressing the button bellow the list.



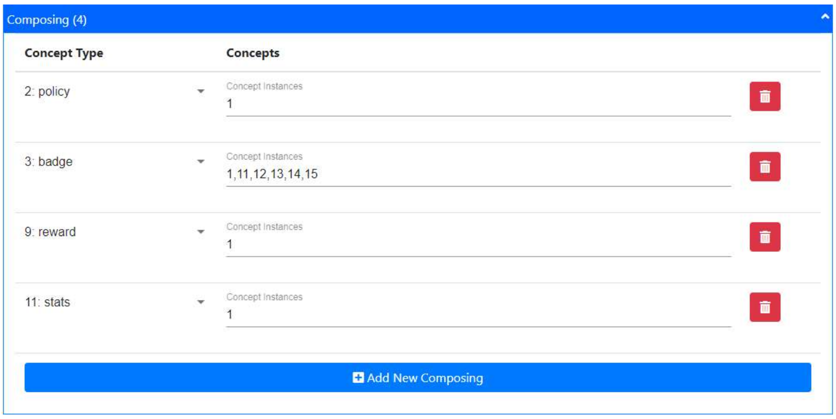
Stack

Position	Concept	
[0] Root	Concept Type: 1:player Concept Instance: 1	
[1] Parent	Concept Type: 2:policy Concept Instance: 1	

Add New Stack

Figure 39 – Stack Section

The Composing Concepts section serves to determine which Concepts belong to a parent Concept, in this case the Concept is the Player 1-1, who possesses the Policy 2-1, Badges 3-1, 3-11, 3-12, 3-13, 3-14 and 3-15 and the Stats 11-1. The Composing Concepts cha be changed by deleting entries using the button on the right or by adding new entries using the button bellow.



Composing (4)

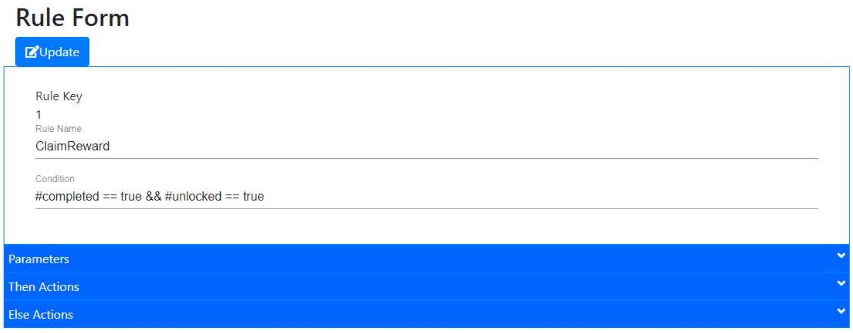
Concept Type	Concepts
2: policy	Concept Instances: 1
3: badge	Concept Instances: 1,11,12,13,14,15
9: reward	Concept Instances: 1
11: stats	Concept Instances: 1

Add New Composing

Figure 40 – Composing Concepts Section

Rule Form

The Rule Form contains a basic details section, a Parameters section, a Then Actions section and a similar Else Actions section. The basic details section contains the Rule's key, name and the condition Action to evaluate before executing the Rule, in the case of the Rule Form for Rule 1, ClaimReward present in Figure 41 the condition checks if the parameters "compted" and "unlocked" are both true.



Rule Form

Rule Key
1

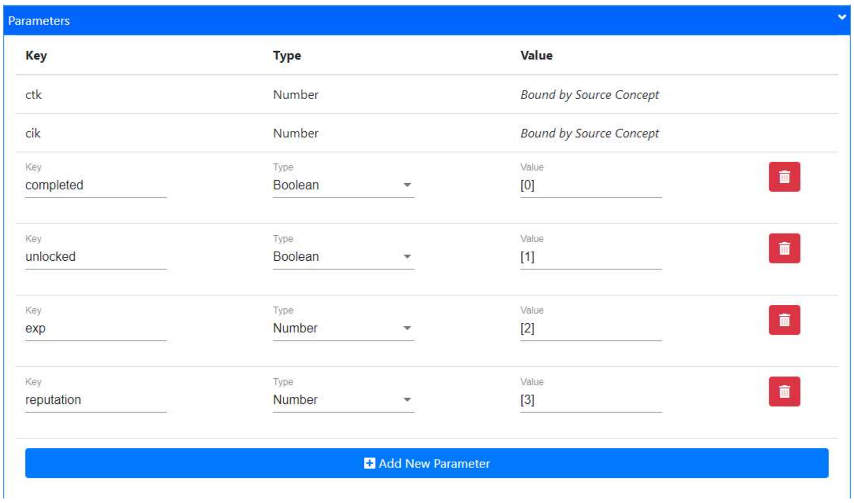
Rule Name
ClaimReward

Condition
#completed == true && #unlocked == true

Figure 41 - Colapsed Rule Form

The Parameters section contains all parameters that must be sent to the Rule for execution. In Figure 42 it is show that two other non-editable parameters exist, "ctk" and "cik", these parameters are autowired into a Rule's execution during runtime and always represent the CK of the Concept running the rule, a third, unrepresented parameter is the "concept" which points to the source Concept itself. Otherwise parameters can be deleted and added utilizing the buttons on the right of each entry and the button on the bottom respectively.

Editing parameters requires the choice of a Type, the available types are Number, Boolean and String, the value present in the Value row must be of the selected type or a number surrounded by "[]", these brackets represent that this value is sent to the Rule, otherwise the value is static and may not be received from outside sources.

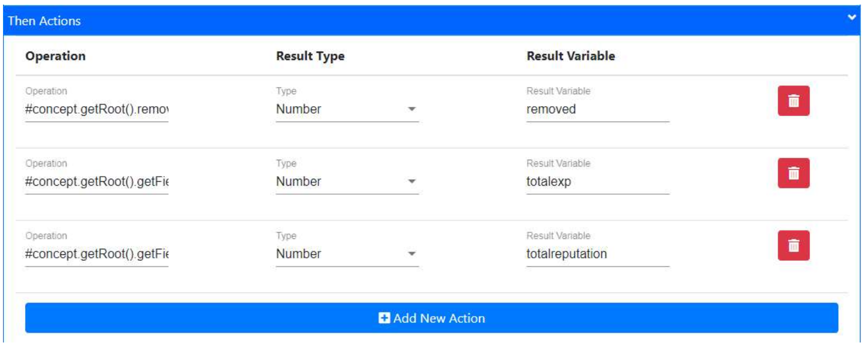


Parameters

Key	Type	Value
ctk	Number	*Bound by Source Concept*
cik	Number	*Bound by Source Concept*
completed	Boolean	[0]
unlocked	Boolean	[1]
exp	Number	[2]
reputation	Number	[3]

Figure 42 - Parameters Section

The Then Actions and Else Actions sections are equivalent, each of them contains a list of entries containing an Operation to run, the return value type of said Operation and the result variable where the Operation's result is stored.



Then Actions

Operation	Result Type	Result Variable
Operation #concept.getRoot().remo\	Type Number	Result Variable removed
Operation #concept.getRoot().getFie	Type Number	Result Variable totalexp
Operation #concept.getRoot().getFie	Type Number	Result Variable totalreputation

Figure 43 - Then Actions

Table Form

The Table Form contains the Basic Information as well as an Axis Overview section and Axis Section and a Value Section. In figure Figure 44, Table "Score by Km – Badge" with Key 1, utilized to determine which badge to attribute to a score, this table as a size of 7x5 and returns a Number.

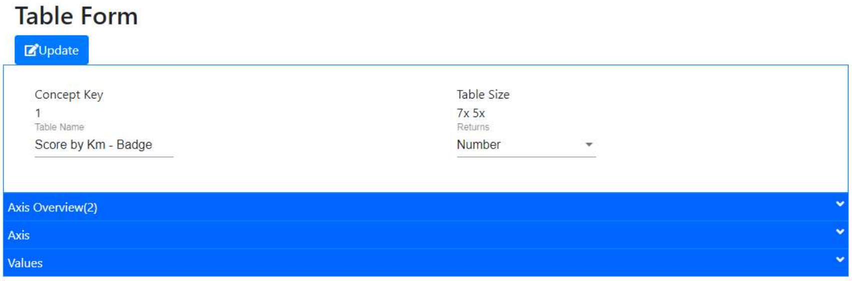


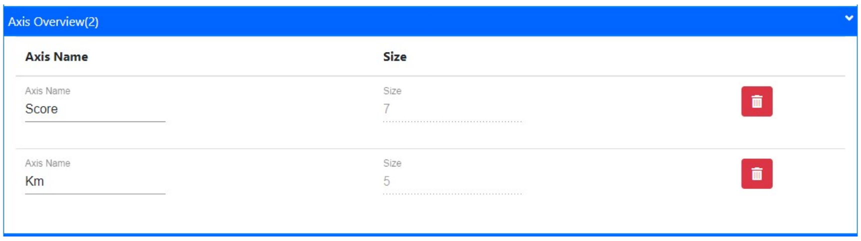
Table Form

Concept Key
1
Table Name
Score by Km - Badge

Table Size
7x 5x
Returns
Number

Figure 44 – Colapsed Table Form

In the Axis Overview section, the name of each Axis is determined, and the total size of the Axis can be seen. As can be noted Figure 45 this Section doesn't allow the creation of more than two Axis, tables aren't limited to this size, as noted in previous section they can be N dimensional this limitation was set on the interface to facilitate representation. Axis can be deleted utilizing the right button and added if the bottom button is present.

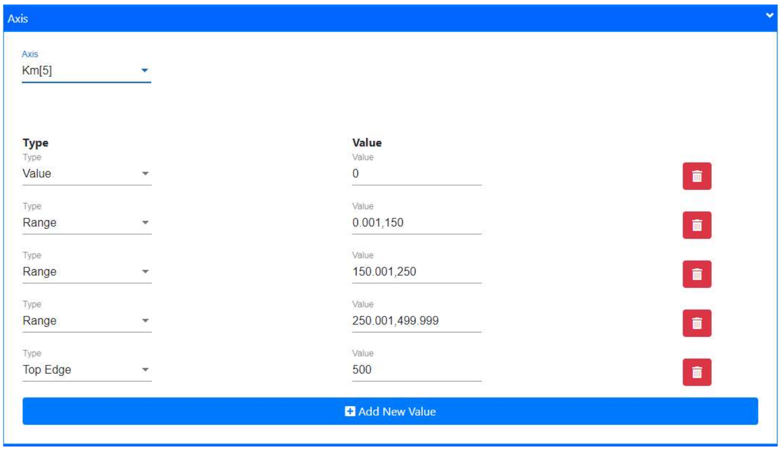


Axis Overview(2)

Axis Name	Size
Axis Name Score	Size 7
Axis Name Km	Size 5

Figure 45 - Axis Oveview Section

In the Axis section the first element present is a dropbox that allows the selection of an Axis, only one Axis may be edited at once in this section. Elements can be added to an axis utilizing the bottom button and removed utilizing the button on the right of each entry. For each entry in the Axis a type must be selected (Bottom Edge, Range, Top Edge or Value) and a value determined, only the Value option supports non-numerical options values.



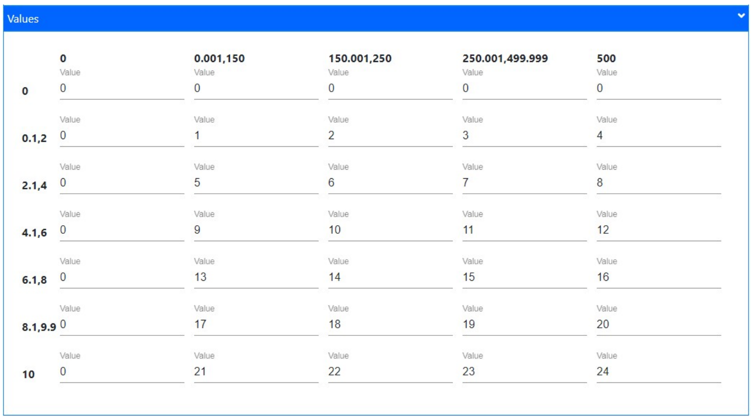
Axis

Axis: Km[5]

Type	Value
Value	0
Range	0.001,150
Range	150.001,250
Range	250.001,499.999
Top Edge	500

Figure 46 - Axis Section

In the Values section the Table as a table, if only one axis exist this table is a single column otherwise it looks as presented in Figure 47. Each of the Table values must be individually defined in this section before the table can be saved.



Values

	0	0.001,150	150.001,250	250.001,499.999	500
0	0	0	0	0	0
0.1,2	0	1	2	3	4
2.1,4	0	5	6	7	8
4.1,6	0	9	10	11	12
6.1,8	0	13	14	15	16
8.1,9.9	0	17	18	19	20
10	0	21	22	23	24

Figure 47 - Table Values Section

5.5.4 Processes

The Configuration Module requires additional work to be completed and a such some of the processes were never implemented. Despite this it is still possible to configure the gamification approach with the present editors, although with some additional difficulty. All the processes described here are done by the interaction of two applications, the Frontend, coded in Angular and the Backend coded in Java, interaction is done via a REST API whose methods will be described when necessary. All Elements behave the same in all Processes, having only differences in data checks, as such they will described in general terms.

In Table 42 it is possible to see an overview of the state of each Process. Processes classified as Not Implemented will not be described.

Table 42 – Configuration Module Process Implementation Evaluation

Process	Implementation State	Testing Stage
Get Element Type Overview	Partially Implemented	In-Development Tested
Get Base Element	Implemented	In-Development Tested
Create New Base Element	Implemented	In-Development Tested
Insert/Update Base Element	Implemented	In-Development Tested
Delete Base Element	Implemented	In-Development Tested
Get Game Element	Not Implemented	Untested
Insert /Update Game Element	Not Implemented	Untested
Delete Game Element	Not Implemented	Untested

Get Element Type Overview

This process is classified as Partially Implemented as it lacks support for the determined Game Elements, however its implementation with base Elements is complete.

This process is initiated upon entry into one of the Overview screens of the Frontend UI. All interface classes in the Fronted UI utilize the OnInit interface supplied by Angular, this allows a method to be executed when the loading of the page is initiated. In Code 21 we can see that in the ngOnInit() method supplied from the interface a call is made to a RESTService which contains a getTemplateOverviews() Method, whose value is attributed to an Observable variable. This allows for the call to the rest service to be asynchronous so the page will load even before the response of the rest call.

```
export class ConceptTemplateOverviewComponent implements OnInit {
  templateoverviews: Observable<TemplateOverview[]>;

  constructor( private rs: RESTService, private router : Router,
                 private route : ActivatedRoute) {}

  ngOnInit() {
    this. templateoverviews = this.rs.getTemplateOverviews();
  }
}
```

Code 21 – ConceptTemplateOverviewComponent Class Excerpt

In Code 22 a partial view of the RESTService class can be seen. The RESTService class is an injectable meaning it can be inserted into the constructor of other classes as can be seen in the previous Code 21. Focusing on the getTemplateOverviews() method it is possible to see it utilizes a HttpClient, supplied by the Angular Http libraries to execute a getMethod which returns an Observable of TemplateOverview[] which will update when the request is completed.

```
@Injectable()
export class ComponentrestService {

  backend : String = "http://localhost:9536";
  httpOptions = {
    headers: new HttpHeaders({
      'Access-Control-Allow-Origin': '*',
      'Content-Type': 'application/json'
    })
  }

  constructor(private client: HttpClient) { }

  getTemplateOverviews(): Observable< TemplateOverview[]> {
    return this.client.get<TemplateOverview[]>(backend + “/template/overview", this.httpOptions);
  }
}
```

Code 22 – RESTService getTemplateOverviews Excerpt

This request is answered by the ConceptTemplateRESTAPI of the Backend application which then obtains all Templates and then converts them to Template Overviews to reduce the amount of data transferred between applications. The Contents of the Overview classes are different between Elements and their contents can be seen in Table 43.

```
@RequestMapping(value = "/overview", method = RequestMethod.GET)
public ResponseEntity<Object> getOverview(@RequestHeader HttpHeaders headers) {
        List<TemplateOverview> templateoverviews = new ArrayList<>();
        List<ConceptTemplate> t = ctc.getInternal();
        for(ConceptTemplate ct : t) {
                templateoverviews.add(new TemplateOverview(ct.getCtk(), ct.getTag()));
        }
        return new ResponseEntity<Object>(templateoverviews, HttpStatus.OK);
}
```

Code 23 – ConceptTemplateRESTAPI getOverview Method

Table 43 – Overview Contents

Element	**Overview**
Concept	CK, tag, non-template field count, non-template trigger count, Composing Concepts count
Concept Template	CTK, tag
Rule	Key, name, facts count, condition action expression, then action count, else action count
Table	Key, name, return type, axis sizes (Ex. 7x8)

Get Base Element

Get Base Element is called whenever the Edit option is selected in the Overview screen. Upon entering the Edit screen, the REST call is made to the Backend as it was in the Get Element Type Overview Process. After this the webpage requires a form to be created to allow for edition of the Element. In Code 24 an excerpt from the Concept Template edit page shows the first part of the ngOnInit method.

First the CTK is obtained from the route parameters, the request is then made to the RESTService. Unlike the Overview this form requires the loading of the Template before presenting data, this is done by utilizing the subscribe() method of the Observable object. Initially the template value from the observable is transformed into a Template object, then a set of FormGroups and FormArrays are created to allow binding of data from fields in the HTML page.

Of note in Code 24Code 23 is the presence of a request for all Rules, this is done to provide a selection box in the Triggers Section which can be seen in Figure 37, the same logic applies for Concept Templates in the Composing Concepts Section for the Concept element visible in Figure 40.

On the Backend side the Elements obtained from the database are cleaned by removing their database specific information, in this case a removal of the id Field, and creating an Editor variation of the Model class which is then returned.

```
ngOnInit() {
    this.sub = this.route.params.subscribe(params => {
      this.ctk = +params['ctk'];
      this.template = this.rs.getConceptTemplate(this.ctk);
      this.template.subscribe(t => {
. . .
          let template = new ConceptTemplate(t.id, t.ctk, t.tag, t.trigger, t.fields);
          this.rules = this.rs.getRules();

          let triggercontrolautoarr = new FormArray([]);
          template.trigger.get(TriggerType.auto).forEach(trigger => {
. . .
          })
. . .
          this.form = this.formBuilder.group({
            id: [''],
            ctk: [''],
            tag: ['', Validators.required],
            directaccess: ['', Validators.required],
            fields: this.fieldscontrol,
            trigger: this.triggerscontrol
          })
          this.form.patchValue({
            id: template.id,
            ctk: template.ctk,
            tag: template.tag,
            directaccess: template.directaccess
          });
}}}
```

Code 24 – ConceptTemplateEditComponent Excerpt

Create New Base Element

Creating New Base Element is simply the creation of a new empty Object of that element with the Key, CTK or CK set to one more than the last present in the database with the notable exception of the Concept Element.

When creating a Concept Element, it is necessary to copy the field data from the Concept Template it refers to, therefore the concept has a set of Fields with default values when created, everything else is empty. In Code 25 the creation of the new Concept in the Backend is visible.

```
public EditorConcept getNew(int ctk) {
        ConceptTemplate template = das.getConceptTemplate(ctk);
        Concept c = new Concept(template);
        int last = das.getLastCikOfCtk(ctk);
        c.setCk(new CK(c.getCk().getCtk(), last+1));
        EditorConcept ec = new EditorConcept(c.get_id(), c.getCk(), c.getTag(), c.isDirectAccess(),
c.getStack(), c.getTrigger(),c.getFields(),c.getComposing());
        return ec;
}
```

Code 25 – ConceptControler getNew() Method

Insert/Update Base Element

After the Get Base Element or Create New Base Element Processes are completed the UI can be utilized to edit the presented Element. After this editing is complete the Insert / Update Process is initiated by calling the Submit method of the page, seen in Code 26.

If the form possesses valid values according to the validation details set up on the FormGroups a DTO is created from the values of the form, this data cannot be directly used in Concepts and Concept Templates due to FormArrays not being compatible with the used format of Triggers and Fields.

A REST request is then made to the Backend to Insert(PUT) or Update(POST) the Element where the relevant DataAccessStrategy method is called.

```
submit() {
        if (this.form.valid) {
        let dto = ConceptFactory.generateConceptDTO(this.form.value)
        let x = this.rs.putConcept(dto).subscribe(x =>
                this.router.navigate(['../../../..'], { relativeTo: this.route })
        )
}}
```

Code 26 – ConceptCreateComponent submit() Method

Delete Base Element

The Delete process is started from the Overview Screen, on the far right of each Element's overview there is a red button with a trashcan icon. Upon clicking the button the removeElement() method is called which, as noted in Code 27, contains a single instruction to the RESTService.

In the specific case of the removeElement() methods an instance of the current Component is sent. This serves to allow the reloading of the Overview Screen when the element is effectively removed in the Backend by calling the original Compoent's ngOnInit() method.

```
removeConcept(ctk : Number, cik :Number){

   this.rs.deleteConcept(ctk, cik, this)

}
```

Code 27 – ConceptOverviewComponent removeComponent() Method

5.6 User Interface Application Implementation

The User Interface Application was implemented, as the Configuration Module, with a Backend in Java and a Frontend in Angular. This module was developed in a limited amount of time and lack the complexity of the other two.

5.6.1 Configurations

The Backend of the User Interface Application utilizes only data originating from the Gamification Module as such dispenses any configuration regarding database access the only option present in the configuration file is the url string to the Gamfication Module as REST Clients require it to obtain information. The Frontend also requires the url to the Backend to allow its REST Client to execute requests, this is, as noted in the Configuration Module, hardcoded.

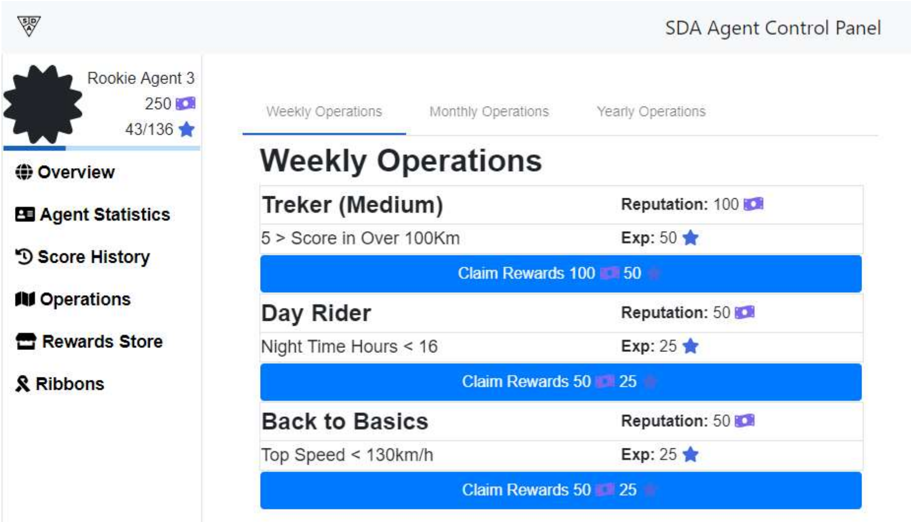
5.6.2 User Interfaces

Interface details for the User Interface Application were partially described in section 3.2.3 while describing the theme of the SDA, these outlines were not followed for the development of this module as its development time was limited.

A total of five screens exist (excluding the Login Screen) in the application. The Operation Screen, the Score Screen, the Ribbon Screen, the Reward Store Screen and the Agent Statistics Screen. Across all screens a sidebar is present with buttons to all screens, including an Overview Screen that was never developed, and a Logout button. This Sidebar also contains the Agent Rank, amount of Reputation owned by the Agent and Exp progress to the next Agent Rank.

Operations Screen
The Operations Screen, seen in Figure 48, lists the currently owned missions of the Player, there are three groups of operations, Weekly, Monthly and Yearly that are presented in specific group Tabs. Each tag contains a listing of all missions each entry of which contains: a name, a description outlining the requirements and completion reputation and exp rewards.

When a mission is deemed complete the Claim Rewards button becomes active, otherwise it shows as an inactive button, in the presented example all missions were completed and show as active. Clicking the button removes the mission from the list and attributes the player the determined rewards.


SDA Agent Control Panel

Rookie Agent 3
250
43/136 ★

- Overview
- Agent Statistics
- Score History
- Operations
- Rewards Store
- Ribbons

Weekly Operations | Monthly Operations | Yearly Operations

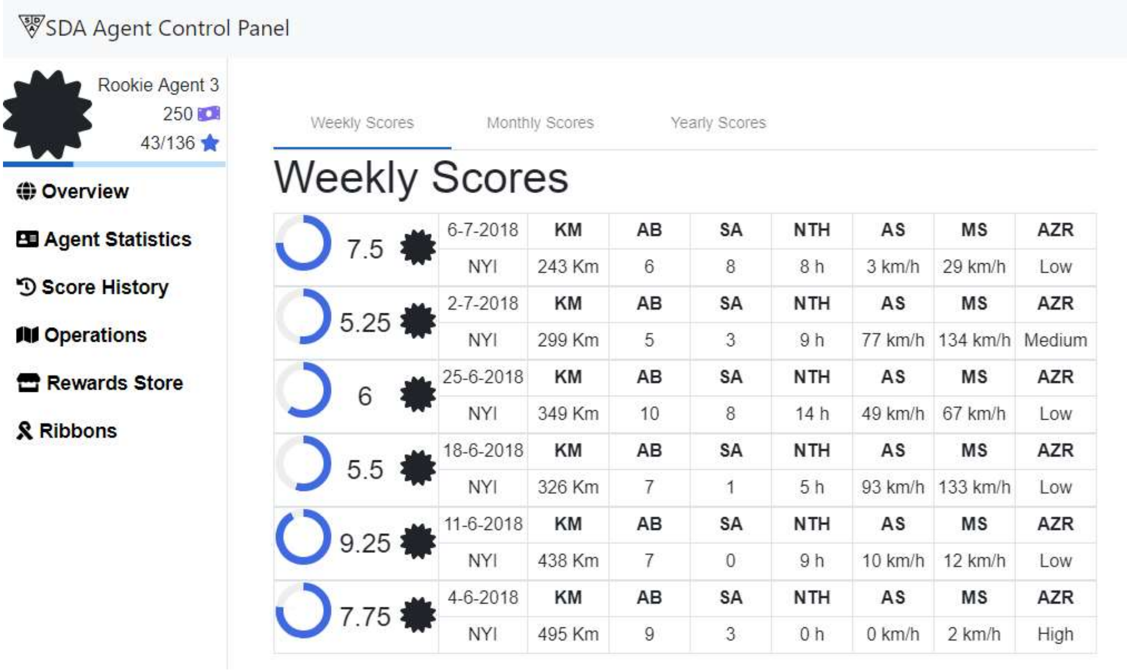
Weekly Operations

Treker (Medium)	Reputation: 100
5 > Score in Over 100Km	Exp: 50 ★
Claim Rewards 100 50	
Day Rider	Reputation: 50
Night Time Hours < 16	Exp: 25 ★
Claim Rewards 50 25	
Back to Basics	Reputation: 50
Top Speed < 130km/h	Exp: 25 ★
Claim Rewards 50 25	

Figure 48 - Operations Screen

Score Screen

The Score Screen lists the players score over a determined time period which and are, like operations, divided into Weekly, Monthly and Yearly tabs. In Figure 49 the Score screen was filled with semi-random performances which cause odd values. The Scores are presented in a List format ranging from newest to oldest. These Scores contain the actual Score (from 0 to 10), and the parameters collected from the player's vehicle as noted in 4.7.2, as well as the week they were calculated on and the badge attributed to this score (this feature was not implemented).


SDA Agent Control Panel

Rookie Agent 3
250
43/136 ★

- Overview
- Agent Statistics
- Score History
- Operations
- Rewards Store
- Ribbons

Weekly Scores | Monthly Scores | Yearly Scores

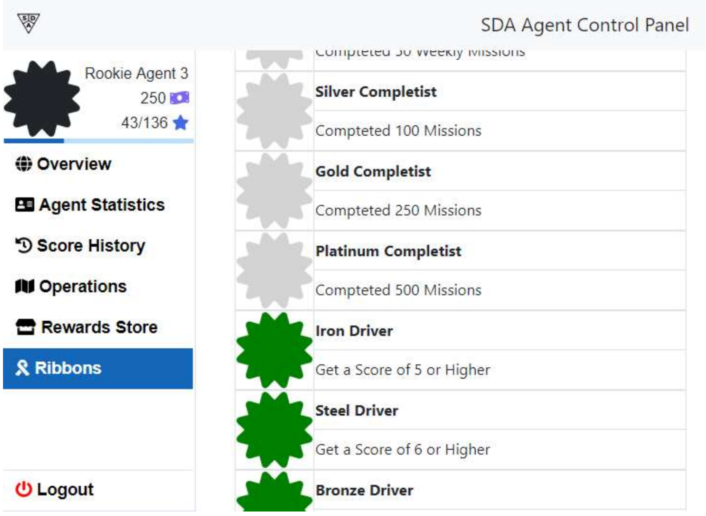
Weekly Scores

7.5	6-7-2018	KM	AB	SA	NTH	AS	MS	AZR
	NYI	243 Km	6	8	8 h	3 km/h	29 km/h	Low
5.25	2-7-2018	KM	AB	SA	NTH	AS	MS	AZR
	NYI	299 Km	5	3	9 h	77 km/h	134 km/h	Medium
6	25-6-2018	KM	AB	SA	NTH	AS	MS	AZR
	NYI	349 Km	10	8	14 h	49 km/h	67 km/h	Low
5.5	18-6-2018	KM	AB	SA	NTH	AS	MS	AZR
	NYI	326 Km	7	1	5 h	93 km/h	133 km/h	Low
9.25	11-6-2018	KM	AB	SA	NTH	AS	MS	AZR
	NYI	438 Km	7	0	9 h	10 km/h	12 km/h	Low
7.75	4-6-2018	KM	AB	SA	NTH	AS	MS	AZR
	NYI	495 Km	9	3	0 h	0 km/h	2 km/h	High

Figure 49 - Score Screen

Ribbon Screen

The Ribbon Screen provides a listing of all Ribbons (Badges in the Configurations) that are possible to obtain. This listing presents already obtained badges in green and unobtained badges in light gray. This screen was not implemented as intended as noted in section 3.2.3. In this implementation no further interaction exists and badges all utilize a generic image.



Completed 50 Weekly Missions

Silver Completist
Compteted 100 Missions

Gold Completist
Compteted 250 Missions

Platinum Completist
Compteted 500 Missions

Iron Driver
Get a Score of 5 or Higher

Steel Driver
Get a Score of 6 or Higher

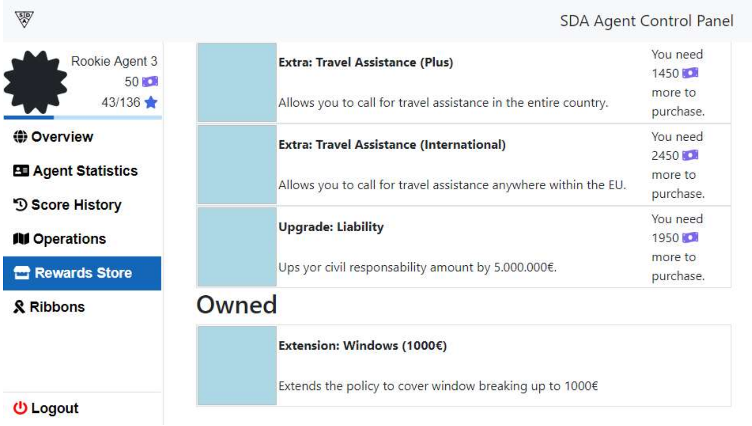
Bronze Driver

Figure 50 - Ribbon Screen

Reward Store Screen

The Rewards Store is where the Player trades in their Reputation for an actual reward, in this example all rewards are optional details or extensions of a standard auto insurance policy. Rewards are presented as a listing, each entry showing a representative image, a name and a short description.

When a Player possesses enough reputation to claim a reward a button, showing the message "Buy Reward" with the amount required for purchase is presented, when pressed it adds the rewards to the Player and debits the stated amount of reputation, the newly purchased reward is then added to a list at the end of the reward set. If not enough Reputation is available the additional required amount is show next to the reward entry.



Extra: Travel Assistance (Plus)
Allows you to call for travel assistance in the entire country.
You need 1450 more to purchase.

Extra: Travel Assistance (International)
Allows you to call for travel assistance anywhere within the EU.
You need 2450 more to purchase.

Upgrade: Liability
Ups yor civil responsability amount by 5.000.000€.
You need 1950 more to purchase.

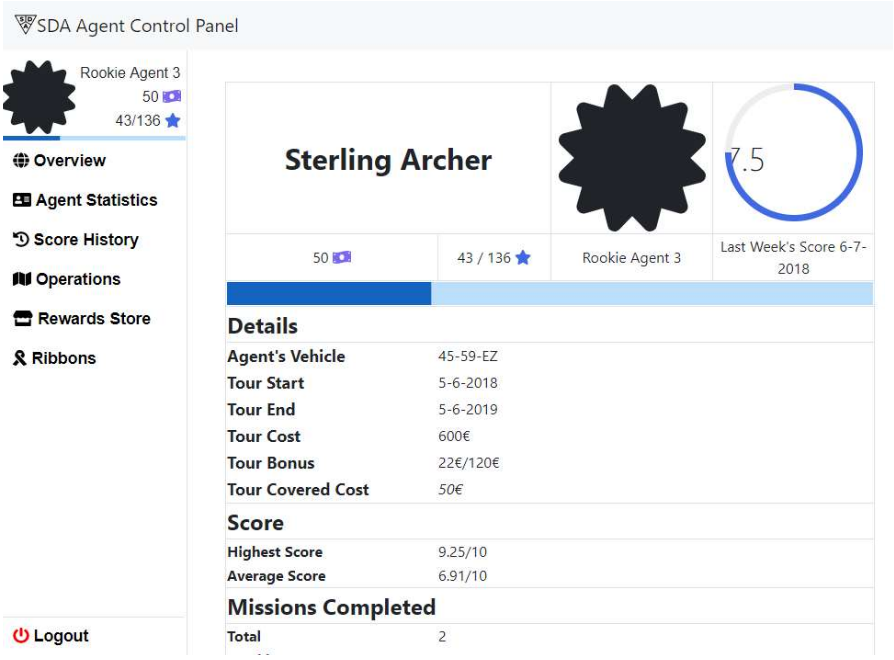
Owned

Extension: Windows (1000€)
Extends the policy to cover window breaking up to 1000€

Figure 51 - Rewards Store Screen

Statistics Screen

The Statistics Screen provides a general view of the Player's progress in the system. On top it shows the players name, reputation, experience, rank badge and the previous week's score. Then a set of information related to the Policy, high scores, mission completion and reputation spending. This screen has no interaction.



Details

Agent's Vehicle	45-59-EZ
Tour Start	5-6-2018
Tour End	5-6-2019
Tour Cost	600€
Tour Bonus	22€/120€
Tour Covered Cost	50€

Score

Highest Score	9.25/10
Average Score	6.91/10

Missions Completed

Total	2

Figure 52 - Statistics Screen

5.6.3 Processes

The User Interface application was never complete as such some of the more complex processes were not implemented. Most of this application serves only as a display of data to the user. In Table 44 the state of the Processes can be seen.

A noteworthy detail is that the data from the Gamification module is filtered before being send to the Frontend and transformed into a simpler format by creating a map of field name, value which is simpler to work in the interface.

Table 44 – User Interface Application Process Implementation Evaluation

Process	Implementation State	Testing Stage
Login	Implemented	In-Development Tested
Logout	Implemented	In-Development Tested
Present Message	Not Implemented	Untested
Claim Operation Reward	Implemented	In-Development Tested
Purchase Reward	Implemented	In-Development Tested
Present Global Mission Data	Not Implemented	Untested

Login

Logging in into the User Interface Application is necessary to access any of the other pages. This is done via a REST Request to the Backend, and, as seen in Code 28, the response token is then stored in one of the Services and will be utilized for future requests. The username and password values are sent to the Backend in the body of a post request as seen in Code 29.

```
public auth(username : String, password : String){
        this.bcs.postAuth(username, password).subscribe(x => {
                this.bcs.setToken(x.token)
                if(this.isAuthed()){
                        this.router.navigate(['../main'], { relativeTo: this.route })
                }
        })
}
```

Code 28 – AuthService auth() Method

```
postAuth(username: String, password: String): Observable<Token> {
        this.httpOptions.headers = new HttpHeaders({
                'Access-Control-Allow-Origin': '*',
                'Content-Type': 'application/json'
        })
        var user = {
                "username":username,
                "password":password
        }
        return this.client.post<Token>(url+"auth/", user, this.httpOptions)
}
```

Code 29 – BackendConnectionService postAuth() Method

In the Backend an AuthRESTClient is created and the information set to the Authentication API of the Gamification module, this is to be changed with the addition of an authentication specific application. As seen in Code 30 after the REST call to the Authentication API the token is utilized to register a new Session. These sessions hold all the players specific Concepts when obtained from the Gamification Module.

```
@RequestMapping(value="/auth", method = RequestMethod.POST)
public ResponseEntity<?> auth(@RequestHeader HttpHeaders headers, @RequestBody AuthUser userinfo) {
        try {
                AuthRESTClient arestc = new AuthRESTClient(userinfo);
                JSONObject token = arestc.execute();
                if(token.containsKey("token")) {
                        session.startSession((String) token.get("token"));
                        return ResponseEntity.status(HttpStatus.OK).body(token);
                }
        } catch (Exception e) {
                return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).body(e.getMessage());
        }
        return ResponseEntity.status(HttpStatus.UNAUTHORIZED).build();
}
```

Code 30 – UIAPI auth() Method

Logout

When the Logout button a GET request is set to the Gamification API, as seen in Code 31, when this request is received by the Backend the session is ended and all data regarding that token is removed by closing the user's section represented by the token, this can be see in the API method present in Code 32.

```
logout(): void {
        this.updateHeaders();
        this.client.get(url+"/auth/logout/");
        this.token = null;
}
```

Code 31 – SidebarComponent's logout() Method

```
@RequestMapping(value="/auth/logout/", method = RequestMethod.GET)
public ResponseEntity<?> logout(@RequestHeader HttpHeaders headers) {
        try {
                String token = headers.getFirst(HttpHeaders.AUTHORIZATION);
                session.endSession(token);
                . . .
                return ResponseEntity.ok(json);
        }catch(UnexistantSessionException | ParseException | IOException e) {
                . . .
        }
        return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).build();
}
```

Code 32 – UIAPI logout() Method

Claim Operation Reward

The when claiming the rewards from a successful mission a request is sent to the Backend whose code is present in Code 33. In this method two calls are executed to the Gamification REST API to run the relevant rules to both give the rewards to the player and update the Stats Concept. The result of this call is then return to the Frontend and the page is loader.

```
@RequestMapping(value="/player/missions/{cik}", method = RequestMethod.GET)
public ResponseEntity<?> getClaimMission(@RequestHeader HttpHeaders headers, @PathVariable("cik") int
cik) {
        try {
                String token = headers.getFirst(HttpHeaders.AUTHORIZATION);
                ...
                Concept player = this.session.getSessionDataConcept(token, 1);
                RESTClient grc = new RESTClient("concept/10/"+cik+"/trigger/1", "[["+player.getCTK() +
"," + player.getCIK()+"]]",reqheaders,  HttpMethod.POST);
                Concept result = grc.execute();
                grc = new RESTClient("concept/10/"+cik+"/trigger/2", "[["+player.getCTK() + "," +
player.getCIK()+"]]",reqheaders,  HttpMethod.POST);
                result = grc.execute();
                if(result == null) {
                        throw new NullPointerException();
                }
                return ResponseEntity.ok(result.getJsonObject());
        }catch(Exception e) {. . .}
        return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).build();
}
```

Code 33 - UIAPI getClaimMission() Method

Purchase Reward

Much like the Claim Operation Reward Process the Purchase Reward Process calls the Gamifcation API to run the rule to add the Reward to the Player, the Reward concept is then returned, as presented in Code 34.

```
@RequestMapping(value="/player/rewards/{cik}", method = RequestMethod.GET)
public ResponseEntity<?> getClaimReward(@RequestHeader HttpHeaders headers, @PathVariable("cik") int
cik) {
        try {
                String token = headers.get(HttpHeaders.AUTHORIZATION).get(0);
                . . .
                Concept player = this.session.getSessionDataConcept(token, 1);
                RESTClient grc = new RESTClient("concept/9/"+cik+"/trigger/13", "[["+player.getCTK() +
"," + player.getCIK()+"]]", reqheaders,  HttpMethod.POST);
                Concept result = grc.execute();
                if(result == null) {
                        throw new NullPointerException();
                }
                return ResponseEntity.ok(result.getJsonObject());
        }catch(Exception e) {. . .}
        return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).build();
}
```

Code 34 - UIAPI getClaimReward() Method

5.7 Deployment

As originally detailed the system is to be deployed in multiple servers running the CentOS operating system utilizing Wildfly 10.1 application server, however these were not available during the development of the project. To simulate this multiple Docker containers were created to simulate in this environment.

Alterations to the standard Wilfly 10.1 configuration are necessary for the project to work as are changes in the standard build configurations of the Java applications on eclipse.

The Frontend applications are supposed to be merged into the existing Java application packaging for release requiring alterations in the standard build configurations as well, this alternative was not explored and as such the specifics of their deployment are not noted.

To allow for deployment to the WIldfly sever the applications should be packaged in .war format instead of the standard .jar. For this the pom.xml file must be altered to specify the packaging to war by altering the <package> field and by removing the embed tomcat serve by altering the spring-boot-starter-web dependency and by adding a dependency to “javax.servlet-api” as seen in Code 35. It is also necessary to alter the @SpringBootApplication class to extend “SpringBootServletInitializer” which adds a new method which can be seen in Code 36.

Running the command “mvn compile package” will compile the source classes and generate the .war for deployment which can be deployed by navigating to the file in the Wildfly management console.

```xml
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
        <packaging>war</packaging>
        <dependencies>
                <dependency>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-starter-web</artifactId>
                        <exclusions>
                                <exclusion>
                                        <groupId>org.springframework.boot</groupId>
                                        <artifactId>spring-boot-starter-tomcat</artifactId>
                                </exclusion>
                        </exclusions>
                </dependency>
                <dependency>
                        <groupId>javax.servlet</groupId>
                        <artifactId>javax.servlet-api</artifactId>
                        <scope>provided</scope>
                </dependency>
        </dependencies>
</project>
```

Code 35 – pom.xml alterations

```java
@SpringBootApplication
public class SdaApplication extends SpringBootServletInitializer {
        @Override
        protected SpringApplicationBuilder configure(SpringApplicationBuilder application) {
                return application.sources(SdaApplication.class);
        }
}
```

Code 36 - User Interface Application Entry class edited for deployment

6 Evaluation

Chapter Contents

6.1 Testing Methods

Before preparing the tests to be executed it is important to define the details of each test, what their purpose, their prerequisites, their application methodology and the way to evaluate the results supplied.

6.1.1 Student's t-test

Student's t-test is a test to determine the existence of a significant difference between two means, this is measured by the conventional statistic called Student's t, the larger the t the larger the difference between sample means. (Shier, 2004a, 2004b)

There values are necessary to determine t in both Paired t-tests and Independent t-tests:

- n the number of samples from each population.
- $\bar{d}$ is the mean difference between the two samples and is given by formula, in which x is an element of each sample.

$$\bar{d} = \bar{\mu}_1 - \bar{\mu}_2 = \frac{\sum(x_1 - x_2)}{n}$$

- s the standard deviation of the sample (s is utilized instead of σ as this is a calculation of an estimated standard deviation since the calculation of σ requires the consideration of the entire population) n-1 is also utilized as this calculation will be utilized to take further conclusions.

$$s = \sqrt{\frac{\sum_{i=1}^{n}\left(x_i - \bar{d}\right)^2}{n-1}}$$

The part of the process as well as the df[3] differ from Paired t-tests and Independent t-tests; however, the final part of the process is the same. The resulting t value is then matched against a t table according to the desired significance level (α[4]) value and if the test is one-tailed or two-tailed. (Shier, 2004a, 2004b)

Analysis of the value from the t-table follows and conclusions regarding H_0 and H_1.

Paired t-test

Paired t-tests are utilized to compare two populations means, with two samples in which observations can be paired with each other. This is the case with before and after measures of the same subject, the exact case for which the paired t-test is utilized in this document. (Shier, 2004a)

[3] Degrees of Freedom in statistics are a complex concept however in simple terms df represent the number of values in the final calculation of the statistic which are free to vary.

[4] α (Significance Level) is the probability of rejecting the null hypothesis when true, so the higher the α value the higher level of certainty of the results of the statistical test.

In the case of paired t-tests the formula that determines t is the following. This is however a composed formula as the lower part of the division ($\frac{s}{\sqrt{n}}$) is in fact the calculation of the standard error of the means difference (Shier, 2004a).

$$t = \frac{\bar{d}}{\frac{s}{\sqrt{n}}}$$

Independent t-test

Independent t-tests are done when the two samples cannot be paired; one example is when dealing with the samples of the same measure regarding two distinct groups. (Shier, 2004b)

In the case of independent t-tests the process is slightly more complex. First requiring the calculation of the pooled standard deviation and not the standard deviation of the difference between samples changing the necessary formula (Shier, 2004b).

$$s_p = \sqrt{\frac{(n_1 - 1)s_1{}^2 + (n_2 - 1)\, s_2{}^2}{n_2 + n_2 - 2}}$$

Only then may t be calculated (Shier, 2004b).

$$t = \frac{\bar{d}}{s_p\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

6.1.2 Spearman's Correlation

Spearman's Correlation is a statistical test to determine the level of relation between two variables and benchmarks it based on a monotonic relationship, which is less restrictive than a linear relationship (Laerd Statistics, 2017).

To begin values for each variable are ranked from lowest to highest. Then the difference between ranks is calculated as is the squared difference between ranks.

$$d = \sum (x_1 - x_2)$$

$$d^2 = \sum (x_1 - x_2)^2$$

After finding d^2 the formula to calculate r_s is utilized.

$$\mathrm{r}_s = 1 - \frac{6\, d^2}{n(n^2 - 1)}$$

r_s calculated assertions can now be made about the data. r_s provides a value between -1 and 1, value of 0 representing the complete lack of correlation between the two variables, a value of 1 representing a perfect positive relation (as x increases y increases), and -1 a perfect negative relation (as x increases y decreases) (Laerd Statistics, 2017).

6.1.3 Quality Evaluation Framework

The QEF was developed to aid "business process modelers and analysts to work in a systematic and generic manner when including quality factors in their BPM activities." However, it's application areas are manifold as such it can be utilized to evaluate the completion of a software solution implementation (Heidari and Loucopoulos, 2014).

The QEF requires three different concepts to be implemented, these concepts are Dimensions, Factors and Requirements. Quality Dimensions are the most general concept and represent the characteristic groups of the system we are evaluating. Factors are more specific characteristics within the dimension groups, finally Requirements are the specific requirements determined to complete each factor regarding the software solution being evaluated (Heidari and Loucopoulos, 2014).

Provided with n Dimensions and n Factors composed of n Requirements, the objective of the system is the 100% completion mark, the ideal system. But different Requirements or Factors are of varying importance to the completeness of the system, some factors may be extremely important as others only slightly as such the possess weights. Requirements are weighted on a scale of even numbers 2 to 10 while Factors share a percentage-based weight within their dimension. Dimensions do not possess a weight between themselves contributing evenly to the completion, this causes extremely small dimensions to over inflate the completion rate of the system, a weight parameter could be added increasing the correctness of this situation.

This completion is measured in a D value, this D value is the distance between the current point and the ideal system point and is calculated via a standard point Euclidean distance calculation.

$$D = \sqrt{\sum (x_a - x_i)^2}$$

6.2 Preparation

Utilizing the problem described in section 0 as well as the idea chosen in section 3.1.2, conceptualized in 3.2 and the details detected in section 4.1 of this report a set of problems that the project attempts to answer can be determined. For these problems were formalized, hypotheses determined, and the statistical test method chosen according to the data presented and the test objective. The parameters for the QEF are also determined in this section for evaluating the completion of the system and the following of the objectives outlined during its design.

6.2.1 Problem 1

The first problem is directly extrapolated from the objectives listed for the project, namely that the number interactions between user and company is to be increased.

P_1:"The usage of a gamified system increases interactions between an insurance company and a given client."

Metrics
To properly evaluate this problem, we require a set of metrics referring to the number of interactions between insurance company and client before and after the introduction of the gamified system. Three values are necessary due to the difference in types of change risk reporting, overall interactions, automatic interactions and manual interactions. Overall interactions are both automatic and manual interactions, automatic interactions are triggered by dynamic risk factors, these are sent automatically by systems exterior to the gamified system such as car sensors and fire alarms. that communicate information to the insurance company, manual interactions are those which require the client to actively seek out the company such as paying the insurance premium or reporting a change in a risk factor, claims are not to be integrated into this value.

- Overall interactions before and after the system was introduced.
- Automatic interactions before and after the system was introduced.
- Manual interactions before and after the system was introduced.

Hypotheses
Letting μ_b be the data regarding interactions before the introduction on the system and μ_a the data after we reach the following hypotheses.

$$H_0: \mu_a - \mu_b = 0$$

$$H_1: \mu_a - \mu_b > 0$$

The goal is to prove H_1 these hypotheses should be tested 3 times, once for each one of the types of metrics, overall, automatic and manual.

Methodology
This data will be obtained from database records of client company interaction. The number of samples should be n ≥ 30.

Testing
To test the hypotheses, we have one group providing two samples of data before and after, therefore we have a set of dependent variables and the statistical tests to be utilized in this situation are one-tailed paired t-tests; however, dependent on the size of the group utilizing the system, possible to utilize a paired z-test instead as dependent on the number of system testers the calculation of the standard deviation for the entire population may be viable and the sample size larger.

6.2.2 Problem 2

From P_1 we can extrapolate a second problem, namely the fact that the utilization of the system may not be the reason that the number of interactions has changed, it may be the cause of an environmental change that caused a change in the entire population of clients. So, we have P_2.

P_2: "Utilizing the system has a noticeable effect in the number of interactions of users compared to non-users."

This way we can be sure if the clients utilizing the system have a larger evolution of interactions than clients not utilizing the system.

Metrics
For this we require 2 groups with the following metrics

- Difference in overall interactions before and after the system was introduced.
- Difference in automatic interactions before and after the system was introduced.
- Difference in manual interactions before and after the system was introduced.

Hypotheses
The hypotheses to test the problem must now be determined for each problem as such we reach this. For P_2 we have the following hypotheses but first we must determine de nomenclature to use regarding datasets. Let μ_1 be the data collected from the "System Users" group and μ_2 the "Non-System Users" group.

$$H_0 : \mu_1 - \mu_2 = 0$$

$$H_1 : \mu_1 - \mu_2 > 0$$

Utilizing the difference in interactions from before and after we create two datasets named μ_1 and μ_2 utilizing these datasets we determine the hypotheses wanting again to prove H_1 users using the system causes a larger increase of interactions than the non-user population.

Methodology
This data is obtained the same way as for P_1; however, this situation utilizes data from 2 control groups, "System Users" and "Non-System Users" which will be used to evaluate if the effect (positive, negative or null) of the introduction system was based on environmental factors that impacted the entire population of policy holders or in fact caused by the system. As P_1 $n \geq 30$ is recommended.

Testing
Again, taking the nature of data, the number of groups and the samples of data for each group we have two groups each providing one set of data therefore one-tailed independent t-tests will be used to evaluate it. Z-tests are not a viable way to evaluate this due to the massive population comprised in one of the groups.

6.2.3 Problem 3

Appearing from the selected idea to gamify, self-management of risk, P_3 was constructed to attempt to evaluate the capability of the system to promote risk reduction.

P_3:"The utilization of the gamified application will reduce the overall risk calculation."

This will compare the risk calculation of clients with their risk calculations before they begin utilizing the gamified system with the ones from the period after but also compare their risk evolution calculation with users that do not utilize the system to verify how impactful the utilization of the system is.

Metrics

- Client risk calculation before the usage of the gamified system.
- Client risk calculation after the usage of the gamified system.

Hypotheses

For this problem the hypotheses mimic those of P_1 as the objective is to detect an increase, so the hypotheses presented are the same, wanting to prove H_1.

$$H_0 : \mu_a - \mu_b = 0$$

$$H_1 : \mu_a - \mu_b > 0$$

Methodology

Refer to P_1.

Testing

Information is again regarding a single group and two data collections (before and after) so the method utilized will be a Paired T-test. The usage of a Z-test is possible according to the exact reasons presented in P_1.

6.2.4 Problem 4

P_4:"The utilization of the gamified system causes a noticeable difference in risk evolution between users and non-users"

P_4 is to P_3 as P_2 is to P_1 as such the goal of this problem is to verify that the evolution in risk is in fact caused by the system and not an environmental change.

Metrics

- Client risk evolution for system users.
- Client risk evolution for non-system users.

Hypotheses

Let μ_1 be the data collected from the "System Users" group and μ_2 the "Non-System Users" group. As the goal is the same H_0 and H_1 mimic P_2's hypotheses for the same exact reasons.

$$H_0: \mu_1 - \mu_2 = 0$$

$$H_1: \mu_1 - \mu_2 > 0$$

Methodology

Refer to P_2.

Testing

Testing will, exactly as P_2 be done utilizing an Independent T-test, same reasoning applies.

6.2.5 Problem 5

Although not in the scope of the objectives of the project this problem comes from personal interest of the author, is the change in risk factor correlated to the increase of interactions between client and insurance company.

P_5:"The increase of interaction between client and insurance company is correlates to the reduction of risk factors."

Metrics

- Client overall interactions with company.
- Client risk evolution.

Hypotheses

As this is a correlation tests the hypotheses are of a different type, since the test to be used will be Spearman's Correlation the hypotheses will be relating to r_s, and by default H0 in this test is $\mathrm{r}_s = 0$ which denotes a complete lack of any correlation. As such we are trying to disprove H_0 and prove H_1.

$$H_0: \mathrm{r}_s = 0$$

$$H_1: \mathrm{r}_s \neq 0 \ \cap \ \mathrm{r}_s > 0$$

Methodology

For this test we require the data form P2 and P4, this time the data collected should be equal measures from both groups and should not be divided in these groups, sample size should, as always be n ≥ 30 with at least 15 measures from each group.

Testing

Due to doubts that the data regarding this problem could be linear, as one of the variables has a finite ceiling, the test utilized will be Spearman's Correlation.

6.2.6 Quality Evaluation Framework

To verify the completion of the developed system a QEF table is to be created, utilizing a secondary reference of the ISO 25010 standard (International Organization for Standardization, 2011; Anon, 2015) the number of dimensions is considerable compared to other representation of the QEF, as such a heavily reduced version present in **Error! Reference source not found.** is proposed for the evaluation of the software solution.

Table 45 - QEF Proposed Dimensions and Factors

Dimension	Factor	Description
Functional Completeness	Gamification Module	degree to which the set of functions covers all the specified tasks and user objectives. (Gamification Module)
	Configuration Module	degree to which the set of functions covers all the specified tasks and user objectives. (Configuration Module)
	User Interface Application	degree to which the set of functions covers all the specified tasks and user objectives. (User Interface Application)
Compatibility	Co-existence	degree to which a product can perform its required functions efficiently while sharing a common environment and resources with other products, without detrimental impact on any other product.
	Interoperability	degree to which two or more systems, products or components can exchange information and use the information that has been exchanged.
Reliability	Fault Tolerance	degree to which a system, product or component operates as intended despite the presence of hardware or software faults

6.2.7 Unit Testing

For unit testing of this project JUnit will be used to develop automated tests to address the functional correction of all non-trivial methods, other methods, such as the standard “getters and setters” will not be tested. REST API’s will be tested via Postman.

When it comes to the applications with their frontend in Angular 6 testing will be done utilizing the Jasmine testing framework, a framework that is utilized to test JavaScript code and comes appended with Angular 6 installations. For these applications, data access methods will not be tested as they are already being tested in the API’s original application.

6.3 Result Evaluation

Due to the lack of exposure of the application to the target user group the evaluation of the previously defined problems was not possible. Therefor any evaluation of the effectiveness of the system on the target user group is classified as Future Work.

6.3.1 Quality Evaluation Framework

Utilizing the Quality Evaluation Framework determined previously we can now assess the completion state of the implemented system compared with the determined design requirement. For more precise evaluation of the evaluation will be executed one Dimension at a time.

The final result of the QEF evaluation was a total of 1.72 which corresponds to a 70% completion rate.

Functional Completeness Dimension
This Dimension is based on the completion of the processes determined for each application, following the terminology set up in section 5.1.1 the values were set to 0% (Not Implemented), 50% (Partially Implemented) or 100% (Implemented). The completion rate for this Dimension is of 81% as can be seen in Table 46.

Table 46 – Functional Completeness Dimension Table

Functional Completeness		81%
Gamification Module	Weight 34%	100%
GM1 Start-up	6	100%
GM2 General Update	4	100%
GM3 Shutdown	4	100%
GM4 Request Response	10	100%
GM5 Pre-Builder	2	100%
GM6 Concept Loading	10	100%
GM7 Rule Triggering	8	100%
GM8 Rule Execution	8	100%
Configuration Module	Weight 39%	64%
CM1 Get Element Type Overview	4	50%
CM2 Get base Element	10	100%
CM3 Create New Base Element	8	100%
CM4 Insert/Update Base Element	10	100%
CM5 Delete Base Element	6	100%
CM6 Get Game Element	4	0%
CM7 Create Game Element	6	0%
CM8 Insert/Update Game Element	6	0%
CM9 Delete Game Element	2	0%
User Interface Application	Weight 26%	80%
UIA1 Login	6	100%
UIA2 Logout	6	100%
UIA3 Present Message	2	0%
UIA4 Claim Operation Reward	10	100%
UIA5 Purchase Reward	10	100%
UIA6 Present Global Mission Data	6	0%

Compatibility Dimension

The Compatibility Dimension refers to interactions between application and with shared resources. The criteria for the evaluation of the requirements in this dimension was lose and based on perception of completion. In Table 47 it is possible to detect that the completion rate of this Dimension was 91%.

Table 47 – Compatibility Dimension Table

Compatibility		91%
Co-existence	Weight 60%	85%
CE1 Applications Access the Database without conflict	6	100%
CE2 The Gamification Module allows access to multiple UI Applications	6	100%
CE3 It is possible to utilize different Database Engines	4	40%
Interoperability	Weight 40%	100%
IO1 Concepts can be transferred between GM and the UI	8	100%
IO2 Concepts created in the CM can be open in the GM	10	100%

Reliability Dimension

The Reliability Dimension deals with the fault tolerance of applications, this was classified in a binary manner, all actions relevant to the requirement ether follow it completely or fail completely. In Table 48 we can see that the completion level of this was set at 63%.

Table 48 – Reliability Dimention Table

Reliability		63%
Fault Tolerance	Weight 100%	63%
FT1 Invalid API Requests fail gracefully	6	100%
FT2 Inexistent Data in database fails gracefully	8	100%
FT3 Invalid Data In database fails gracefully	8	0%

7 Conclusion

Chapter Contents

7.1 Objective Completion State

All the objectives for the project were tacked in different timeframes and did not require the same amount of time to complete and as such each objective will be tackled separately. These objectives shall be numbered from 1 to 5 in according to the order presented in the original detailing:

1. Identify and analyse possible applications of gamification in insurance software, with the selection of one or more according to a set of criteria and restrictions.
2. Analyse and select one or more gamification frameworks to be used in the gamification design of the applications.
3. Design a gamification approach in the context on Insurance.
4. Design a highly configurable system which can be utilized to define different gamified approaches and execute processes necessary to their functioning.
5. Implement a prototype of technology readiness level 6 (Technology demonstrated in relevant environment), for a product to be commercialized by i2S based on the designed system.

Table 49 - Objective Completion State

Objective	Completion State
O1	Completed
O2	Completed
O3	Completed
O4	Completed
O5	Partial Completion

At the end of this project four of the total five objectives can be said to have been completed.

Evidence for the completion of Objective 1 can be seen in section 3.1 where the application areas are analysed and selected according to a set of criteria previously determined.

Objective 2 was determined early on Deterding's Lens of Intrinsic Atoms was chose for the initial analysis and selection, followed by a passing of the designed approach to a more specific field of auto insurance, this focused approach was then passed through all the lens of game design once and finally evaluated utilizing Octalysis standards. As such it can be said that multiple gamification frameworks as well as some aspects of pure game design were utilized in the application of game design. This all also involves Objective 3 and details on both these objectives, including the final gamification approach selected can be seen in section 3.2 where the process of applying the chosen frameworks to create the approach is documented.

After the approach was determined the development of the system necessary to support its functioning, including details on processes, interfaces and supporting data models can be seen in chapter 0 more specifically in sections 4.1, 4.2, 4.3, 4.4, 4.5 and 4.6 give answer and justification to Objective 4.

Objective 5 was not fully realized, documented in chapter 0 which describes the implementation of the features previously determined in chapter 3, not all features were included into the developed system or properly implemented according to specifications determined.

As it stands the prototype stage is at TRL 5, having been tested within a lab environment but having never been exposed to the final user, thus failing to hit the mark of TRL 6 determined within Objective 5.

7.2 Future Work and Limitations

Since in the world almost nothing can be considered perfect, complete and omnipotent the project detailed in this document needs additional work and processes a multitude of limitations that must be declared for reference in a posterior date.

7.2.1 Gamifying Risk Management

The attempt to utilize gamification in insurance leads to an interesting problem which is the lack of interaction necessary in this system. Insurance policies are extremely static and as such, without utilizing only dynamic risk factors, it appears virtually impossible to develop a generalized solution for all types of insurance, as such the initial gamified system planed possessed extremely shallow game concepts, points and badges can only go so far. The system wasn't something a user can actively use whenever wished since there are major limitations to their influence in the system. Developing an appealing environment and aesthetic is paramount to the success of this type system. Attaching other, more interactive, elements could possibly increase engagement, but it has the possibility of dethatching the user from the original goal.

To attempt to alleviate these issues the transition was taken to the more interactive area of Telematics Auto Insurance, where the data could be supplied in at a much faster rate so the dynamism necessary appeared sufficient. The SDA game appears to the author to be more appealing and more viable gamification approach than the original, as the dynamism allows the user interaction to be much more frequent.

7.2.2 Module Feature Conclusion and Improvement

Features within all three modules are still incomplete and need to be finished, robustness needs to be increased across the board as error tolerance is average at best and applications will often lock up during execution if handled without proper care. The stage of these unfinished features is different across all modules, so it is important to analyse them singularly.

Gamification Module

The Gamification Module was refactored multiple times during the lifecycle eventually having to be scraped completely and fully rebuilt. As it stands the Gamification Module is feature complete, however, limit cases may still cause undetected issue, there is also no tolerance for database errors. To finalize the Gamification Module it is still necessary to remove authentication responsibilities from this module and extend the instruction set of rules to allow for the mentioned gaps.

Configuration Module
The Configuration module is the least functional of all, it possesses only the capability of editing Base Element types directly, without any Gamification obfuscation as originally intended. For this module to be completed, various details need development including the creation of Game Element abstractions and the improvement of search parameters.

User Interface Application
The User Interface Application has its interface section severely undeveloped, an in-depth graphic design overhaul needs to be executed. Feature completion is also not present, information regarding global risk data needs to be included into the configured SDA example as well as messaging concepts to allow for the full completion of this application.

Safe Driver Agency
The Safe Driver Agency example has only undergone a single iteration hand had no exposure to a real or simulated end user, evaluation of the effectiveness of its determined mechanics was therefor never tested and its engagement level may be lower than intended.

7.3 Author's Note

To finalize this document I would like to, slightly, drop the formality and provide my honest opinion about various areas on the development of this project. Five points will be addressed. My overall appreciation of the results achieved by this project, the solution designed and the gamification approach presented, my opinions on gamification, gamification in insurance and finally a self-evaluation of my performance during the project.

7.3.1 On the projects results

As can be seen by previous sections within this chapter, and really the entire document, the project cannot be called complete, partially complete I would say. The major issue is due to lack of "on field" testing that would be necessary to refine both the designed system and gamification approach.

I do not believe I have achieved little though, I designed a system that, although extremely generic, can still execute all the features I found necessary, even if not all of them are implemented, and the majority is.

To have completed this project with all the design parameters accomplished I would have needed at least one of three things, a stronger base to start with, mainly a pre-existent rule engine ready to use from day 1, as it was the section that most time occupied to get working properly; a better understanding of the technologies coming in, I wasn't familiar with Spring Boot, or even Spring in general, or JPA for that matter, and since we are naming things Angular too. I knew almost nothing of these technologies and frameworks when I started this project, which consumed allot of time, harming the final result; a team, I developed the entirety of this project mostly alone, with periodic input from two members of the company every two weeks or so, at least 1 more team member from day one would probably have resulted on a much better and complete project, not having people second guess you it's incredibly negative when designing anything and many of the time wasting features that were eventually dropped may have been dropped much sooner, or changed to be productive work, with a second person on board.

7.3.2 On the presented solution

I honestly like the solution I designed, although the implementation does not contain the most appealing feature I wanted to develop since the start of the project, the Game Element abstraction in the Configuration Module, I believe that with this implemented (and probably someone who actually can design an interface) the entire system would be able to reach the wanted potential.

The generic approach may seem odd to some people, why not go for a more focused approach when defining things in the system. The first reason was because of the requirement of it being highly configurable, you can do pretty much anything you want, seems highly configurable to me, maybe a bit too much. But another point, probably the more important one of the two, was to cover my basis. When I started this project my idea of what I would be doing was not fully formed, if I could configurator a wide variety of options I would be able to adapt to radical changes, which happened with the SDA, without requiring any sort of massive refactoring effort, instead a few changes of JSON files got me to what I intended, with some bumps along the way granted.

7.3.3 On the presented gamification approach

The SDA was a weird concept to elaborate, and although I am reasonably happy with the end result, I feel that it wouldn't be a very successful solution if presented in the open market. It seems to me week, especially regarding the obtained Octalysis score, if 126 out of 800. I also think that the solution may appear to be extremely more black hat than predicted, my rating took account the application in only a vacuum, namely without the player being passively punished by not playing the game by paying more or by not having access to policy extensions, I did although try to tie most of the discount values to the Score system which would be attributed weather the user utilized the application or not.

7.3.4 On gamification

One thing I enjoyed most of all was my interaction with the thematic of Gamification. Gamification was only a faraway concept for me before the beginning of this project, but now after reading much about the topic and attempting my own approach, I've gained some respect for how difficult it is to do, my methodology was simplistic, going from a first pass with Deterding's method and them passing through all game design lenses of the book that inspired the method itself.

Gamification is something to do with care though, it can be utilized to incentivize negative behaviour and hook vulnerable people to exploitative situations and I can see it being very easily abused to improve a company's revenue at much cost to its employees or clients.

7.3.5 On gamifying insurance

Insurance is an area in which gamification is extremely hard, at least it is my perception. There are only a few types of insurance that can really benefit from this utilization as most are updated so infrequently that they become practically impossible to interact with in a reasonably timeframe. This was the reason for the utilization of auto insurance as the driving a car can be quickly updated and done with certain regularity.

References

Anon (2015) *Product Quality - ISO / IEC 25010 Quality in Use - ISO / IEC 25010*. Available at: https://edisciplinas.usp.br/pluginfile.php/294901/mod_resource/content/1/ISO 25010 - Quality Model.pdf.

Aparicio, A. F. *et al.* (2012) 'Analysis and application of gamification', *Proceedings of the 13th International Conference on Interacción Persona-Ordenador - INTERACCION '12*, (October), pp. 1–2. doi: 10.1145/2379636.2379653.

Associação Portuguesa de Seguradores (2012) *Como funciona o seguro*. Lisboa. Available at: https://www.apseguradores.pt/Portal/portal.aspx?MicrositeId=1&PageId=52.

Associação Portuguesa de Seguradores (2017) *Insurance in Portugal*. Lisboa. Available at: https://www.apseguradores.pt/Portal/Content_Show.aspx?ContentId=3047&PageId=43&MicrositeId=1&CategoryId=56.

Autoridade de Supervisão de Seguros e Fundos de Pensões (2015) *Guia de Seguros e Fundos de Pensões*. 3ª Edição. Edited by Autoridade de Supervisão de Seguros e Fundos de Pensões. Lisboa. Available at: http://www.asf.com.pt/NR/rdonlyres/C688F77F-0CE7-4059-9C88-5F1D53A8D92E/0/Guiawebuv.pdf.

Aviva (2017) *Take the Aviva Drive challenge - Aviva*. Available at: https://www.aviva.co.uk/car-insurance/drive/ (Accessed: 31 January 2018).

Bartle, R. (1996) *HEARTS, CLUBS, DIAMONDS, SPADES: PLAYERS WHO SUIT MUDS*. Available at: http://mud.co.uk/richard/hcds.htm (Accessed: 22 January 2018).

Blohm, I. and Leimeister, J. M. (2013) 'Gamification: Design of IT-based enhancing services for motivational support and behavioral change', *Business and Information Systems Engineering*, 5(4), pp. 275–278. doi: 10.1007/s12599-013-0273-5.

Capitain Up (2018) *Capitain Up :: Engagement Platform as a Service*. Available at: https://captainup.com (Accessed: 6 March 2018).

Chatterjee, A., Pathak, P. and Kumar, A. (2017) *Will Gamification be the Game Changer in Insurance Distribution ?* Available at: https://www.infosys.com/industries/insurance/Documents/game-changer-insurance-distribution.pdf.

Chou, Y.-K. (2018) *Octalysis: Complete Gamification Framework -Yu-Kai Chou*. Available at: http://yukaichou.com/gamification-examples/octalysis-complete-gamification-framework/#.WuiKICHw-Uk (Accessed: 23 April 2018).

Crow, K. A. (2002) 'Value Analysis and Function Analysis System Technique', *DRM Associates*. Available at: http://www.npd-solutions.com/va.html.

CSIA (2015) *CSIA*. Available at: http://www.crowdsourcedintel.org/radnets/manage_targets/%40h0use (Accessed: 6 August

2018).

Deterding, S. *et al.* (2011) ‘From game design elements to gamefulness’, in *Proceedings of the 15th International Academic MindTrek Conference on Envisioning Future Media Environments - MindTrek ’11*. New York, New York, USA: ACM Press, pp. 9–15. doi: 10.1145/2181037.2181040.

Deterding, S. (2011) ‘Situated motivational affordances of game elements : A conceptual model’, in *ACM Human-Computer Interaction*, pp. 3–6. doi: ACM 978-1-4503-0268-5/11/05.

Deterding, S. (2015) ‘The lens of intrinsic skill atoms: A method for gameful design’, *Human-Computer Interaction*, 30(3–4), pp. 294–335. doi: 10.1080/07370024.2014.993471.

Duolingo (2018). Available at: https://duolingo.com/ (Accessed: 5 February 2018).

European Association of Research and Technology Organizations (2014) *The TRL Scale as a Research & Innovation Policy Tool , EARTO Recommendations*. Available at: http://www.earto.eu/fileadmin/content/03_Publications/The_TRL_Scale_as_a_R_I_Policy_Tool_-_EARTO_Recommendations_-_Final.pdf.

European Comission (2017) *Horizon 2020 - Work Programme 2016-2017*. Available at: https://ec.europa.eu/research/participants/data/ref/h2020/other/wp/2016_2017/annexes/h2020-wp1617-annex-g-trl_en.pdf.

Fitbit Inc. (2018) *Fitbit Official Site for activity Trackers and More*. Available at: https://www.fitbit.com (Accessed: 30 July 2018).

Genesys (2008) *Customer Service Strategies for the Retail Banking Industry*, *Banking*. Available at: http://www.genesys.com/resources/brochures/customer-service-strategies-for-the-insurance-industry-strategy-guide-north-america.pdf.

Gilberto, F. (2008) *Manual Prático dos Seguros*. 1rst edn. LIDEL.

Hamari, J., Koivisto, J. and Sarsa, H. (2014) ‘Does Gamification Work? -- A Literature Review of Empirical Studies on Gamification’, in *2014 47th Hawaii International Conference on System Sciences*. IEEE, pp. 3025–3034. doi: 10.1109/HICSS.2014.377.

Heidari, F. and Loucopoulos, P. (2014) ‘Quality evaluation framework (QEF): Modeling and evaluating quality of business processes’, *International Journal of Accounting Information Systems*, 15(3), pp. 193–223. doi: 10.1016/j.accinf.2013.09.002.

International Organization for Standardization (2011) *ISO / IEC 25010:2011 - Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models*. Available at: https://www.iso.org/standard/35733.html (Accessed: 20 February 2018).

Jesse Schell (2008) *The art of game design*. 1rst edn, *The effects of brief mindfulness intervention on acute pain experience: An examination of individual difference*. 1rst edn. Elsevier Inc.

John Hancock Insurance (2017) *Frequently asked questions*, *John Hankock Insurnace*. Available at: https://jh1.jhlifeinsurance.com/jhl-ext-templating/filedetail?vgnextoid=198d7a0fbe21c410VgnVCM1000003e86fa0aRCRD&siteName=JHSalesNet.

Kim, A. (2010) ‘Gamification Workshop’. Available at: https://www.slideshare.net/amyjokim/gamification-workshop-2010.

Koen, P. *et al.* (2001) ‘Providing Clarity and A Common Language to the “Fuzzy Front End”’, *Research-Technology Management*, 44(2), pp. 46–55. doi: 10.1080/08956308.2001.11671418.

Koen, P. A. *et al.* (2002) ‘Fuzzy Front End: Effective Methods, Tools, and Techniques’, in Belliveau, P., Griffin, A., and Somermeyer, S. (eds) *The PDMA Toolbook for New Product Development*. 1st edn. Wiley.

Laerd Statistics (2017) *Spearman’s Rank-Order Correlation - A guide to use it w, what it does and what the assumptions are.* Available at: https://statistics.laerd.com/statistical-guides/spearmans-rank-order-correlation-statistical-guide.php (Accessed: 21 February 2018).

Lapierre, J. (2000) ‘Customer-perceived value in industrial contexts’, *Journal of Business & Industrial Marketing*, 15(2/3), pp. 122–145. doi: 10.1108/08858620010316831.

Manoharan, K. S., Agarwal, R. and Shukla, G. (2017) *Gamification for Insurers : A Practitioner ’ s Perspective*. Available at: https://www.cognizant.com/whitepapers/gamification-for-insurers-a-practitioner-s-perspective-codex2268.pdf.

Mora, A. *et al.* (2015) ‘A Literature Review of Gamification Design Frameworks’, in *2015 7th International Conference on Games and Virtual Worlds for Serious Applications (VS-Games)*. IEEE, pp. 1–8. doi: 10.1109/VS-GAMES.2015.7295760.

Morschheuser, B. *et al.* (2018) ‘How to design gamification? A method for engineering gamified software’, *Information and Software Technology*, 95(March), pp. 219–237. doi: 10.1016/j.infsof.2017.10.015.

Nicholson, S. (2012) ‘A User-Centered Theoretical Framework for Meaningful Gamification’, in *Games+ Learning+ Society*, pp. 1–7. doi: 10.1007/978-3-319-10208-5_1.

Norman, D. A. (1988) *The Design of Everyday Things*. Available at: Basic Book.

Ok! Teleseguros (2017) *Guia de Assistência GPS*. Available at: https://www.okteleseguros.pt/bo/files/1c1755af6bf94d9beb1860e8a064cb389ea82639.pdf.

Osterwalder, A. *et al.* (2010a) ‘Business Model Generation’, *John Wiley & Sons, Inc.*, 30(5377), p. 288. Available at: http://www.amazon.com/Business-Model-Generation-Visionaries-Challengers/dp/0470876417.

Osterwalder, A. *et al.* (2010b) *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 … written by co-created by designed by*, *Booksgooglecom*. doi: 10.1523/JNEUROSCI.0307-10.2010.

P.Waltz, S. and Deterding., S. (2015) *The Gameful World - Approaches, Issues, Applications*. Edited by S. P. Walz and S. Deterding. Massachusetts Institute of Technology. doi: 10.1017/CBO9781107415324.004.

Pareto, C. (2008a) *Intro to Insurance: Fundamentionals of Insurance, 14/10/2008*. Available at: https://www.investopedia.com/university/insurance/insurance2.asp (Accessed: 22 December 2017).

Pareto, C. (2008b) *Intro To Insurance: What Is Insurance?, 14/10/2008*. Available at: https://www.investopedia.com/university/insurance/insurance1.asp (Accessed: 22 December 2017).

Rich, N. and Olweg, M. (2000) *Value Analysis, Value Engineering*. Cardiff.

Ryan, R. M. and Deci, E. L. (2000) 'Intrinsic and Extrinsic Motivations: Classic Definitions and New Directions', *Contemporary Educational Psychology*, 25(1), pp. 54–67. doi: 10.1006/ceps.1999.1020.

Ryan, R. M. and Deci, E. L. (2000) 'Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being.', *American Psychologist*, 55(1), pp. 68–78. doi: 10.1037/0003-066X.55.1.68.

Scott, B. (2010) *Designing with Lenses*, *UX Booth*. Available at: www.uxbooth.com/articles/designing-with-lenses/ (Accessed: 22 January 2017).

Seaborn, K. and Fels, D. I. (2015) 'Gamification in theory and action: A survey', *International Journal of Human-Computer Studies*, 74, pp. 14–31. doi: 10.1016/j.ijhcs.2014.09.006.

Sega (2008) 'Valkyria Chronicles'. Sega.

Shier, R. (2004a) *Statistics: 1.1 Paired t-tests*. Available at: http://www.statstutor.ac.uk/resources/uploaded/paired-t-test.pdf.

Shier, R. (2004b) *Statistics: 1.2 Unpaired t-test*. Available at: http://www.statstutor.ac.uk/resources/uploaded/unpaired-t-test.pdf.

The CentOS Project (2017) *About CentOS*. Available at: https://www.centos.org/about/ (Accessed: 19 February 2018).

Tower, E. W., Frankfurt, W. and Tel, G. (2015) *Guidelines on ring-fenced funds*. Available at: https://eiopa.europa.eu/Publications/Guidelines/RFF_Final_document_EN.pdf.

Welltok Inc. (2018) *CaféWell*. Available at: https://www.cafewell.com (Accessed: 30 July 2018).

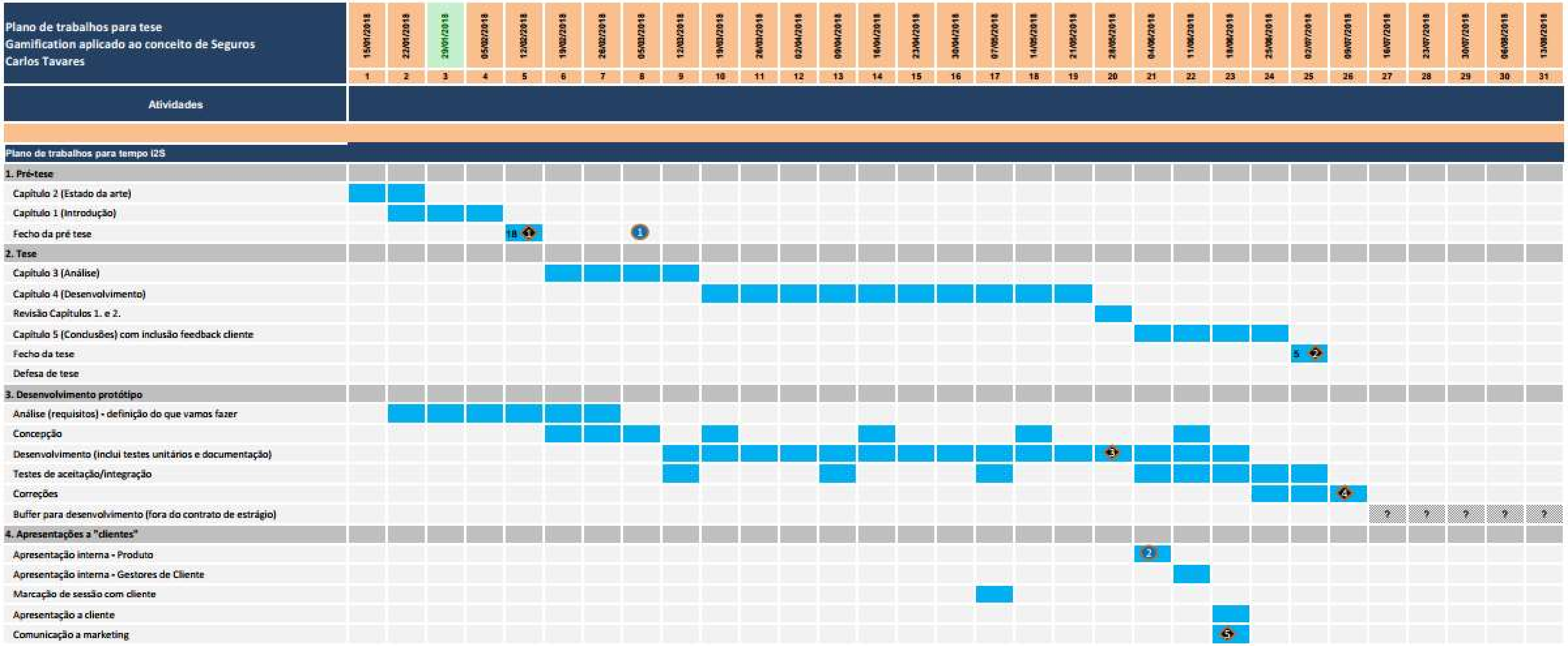
Woodall, T. (2003) 'Conceptualising "Value for the Customer'': An Attributional, Structural and Dispositional Analysis"', *Academy of Marketing Science Review*, 12(5), pp. 1–42.

Zichermann, G. and Cunningham, C. (2011) *Gamification by design; implementing game mechanics in web and mobile apps*. 1rst edn.

Annexes

Chapter Contents

A. Work Plan



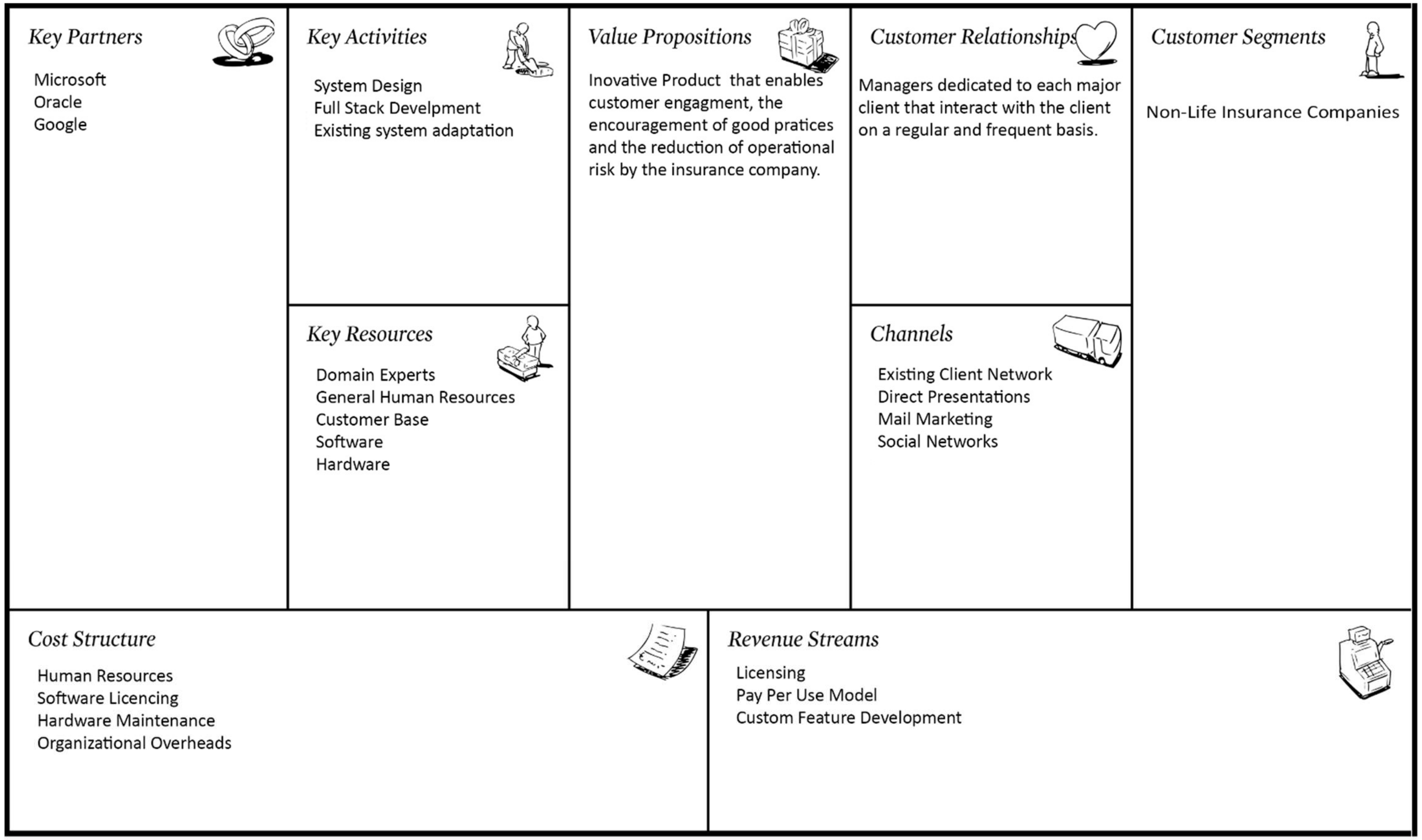
Plano de trabalhos para tese Gamification aplicado ao conceito de Seguros Carlos Tavares	15/01/2018	22/01/2018	29/01/2018	05/02/2018	12/02/2018	19/02/2018	26/02/2018	05/03/2018	12/03/2018	19/03/2018	26/03/2018	02/04/2018	09/04/2018	16/04/2018	23/04/2018	30/04/2018	07/05/2018	14/05/2018	21/05/2018	28/05/2018	04/06/2018	11/06/2018	18/06/2018	25/06/2018	02/07/2018	09/07/2018	16/07/2018	23/07/2018	30/07/2018	06/08/2018	13/08/2018
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Atividades																															
Plano de trabalhos para tempo i2S																															
1. Pré-tese																															
Capítulo 2 (Estado da arte)	■	■																													
Capítulo 1 (Introdução)		■	■	■																											
Fecho da pré tese					18 ◆1			●1																							
2. Tese																															
Capítulo 3 (Análise)						■	■	■	■																						
Capítulo 4 (Desenvolvimento)										■	■	■	■	■	■	■	■	■	■												
Revisão Capítulos 1. e 2.																				■											
Capítulo 5 (Conclusões) com inclusão feedback cliente																					■	■	■	■							
Fecho da tese																									5 ◆2						
Defesa de tese																															
3. Desenvolvimento protótipo																															
Análise (requisitos) - definição do que vamos fazer		■	■	■	■	■	■																								
Concepção						■	■	■		■				■				■				■									
Desenvolvimento (inclui testes unitários e documentação)									■	■	■	■	■	■	■	■	■	■	■	◆3	■	■	■								
Testes de aceitação/integração									■				■				■				■	■	■	■	■						
Correções																								■	■	◆4					
Buffer para desenvolvimento (fora do contrato de estrágio)																											?	?	?	?	?
4. Apresentações a "clientes"																															
Apresentação interna - Produto																					●2										
Apresentação interna - Gestores de Cliente																						■									
Marcação de sessão com cliente																	■														
Apresentação a cliente																							■								
Comunicação a marketing																							◆5								

● n Milestone
◆ n Entregável

Milestone	Descrição
1	Passagem para Product Team
2	Primeira apresentação "pública"!
3	
4	

Entregável	Descrição
1	Entrega da Pré-tese
2	Entrega da Tese
3	Protótipo demonstrável
4	Protótipo final e documentação
5	Comunicação ao Marketing para site i2S

B. Business Model Canvas



Key Partners

Microsoft
Oracle
Google

Key Activities

System Design
Full Stack Develpment
Existing system adaptation

Key Resources

Domain Experts
General Human Resources
Customer Base
Software
Hardware

Value Propositions

Inovative Product that enables customer engagment, the encouragement of good pratices and the reduction of operational risk by the insurance company.

Customer Relationships

Managers dedicated to each major client that interact with the client on a regular and frequent basis.

Channels

Existing Client Network
Direct Presentations
Mail Marketing
Social Networks

Customer Segments

Non-Life Insurance Companies

Cost Structure

Human Resources
Software Licencing
Hardware Maintenance
Organizational Overheads

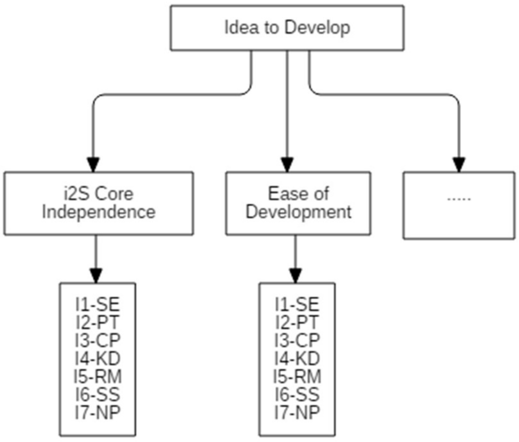
Revenue Streams

Licensing
Pay Per Use Model
Custom Feature Development

C. AHP Utilization Example

During the development of this project some evaluations have to be made, this however does not need to utilize the AHP method. As an example the evaluation of the chosen idea can be adapted to the AHP method, this was made utilizing a weighted evaluation matrix due to the amount of criteria presented being substantial but it could also be done utilizing the AHP method.

Step 1 – Hierarchical Decision Tree(Summarized)



Step 2 – Hierarchal Element Evaluation

Seeing as the evaluated method has 11 criteria, for the purpose of this demonstration only 4 will be utilized called New Customer Acquisition/Maintenance(C1), Ease of Integration with i2S solutions(C2), i2S Core Independence(C3) and Ease of Development(C4).

	C1	C2	C3	C4
C1	1	1/6	1/3	1/9
C2	6	1	3	1/3
C3	3	1/3	1	1/5
C4	9	3	5	1
Soma	19	4 ½	9 $^1/_3$	1 $^2/_3$

$$A = \begin{bmatrix} 1 & 1/6 & 1/3 & 1/9 \\ 6 & 1 & 3 & 1/3 \\ 3 & 1/3 & 1 & 1/5 \\ 9 & 3 & 5 & 1 \end{bmatrix}$$

Step 3 - Normalized table

	C1	C2	C3	C4
C1	0.05	0.04	0.04	0.07
C2	0.32	0.22	0.32	0.20
C3	0.16	0.07	0.11	0.12
C4	0.47	0.67	0.54	0.61

$$\dot{A} = \begin{bmatrix} 0.05 & 0.04 & 0.04 & 0.07 \\ 0.32 & 0.22 & 0.32 & 0.20 \\ 0.16 & 0.07 & 0.11 & 0.12 \\ 0.47 & 0.67 & 0.54 & 0.61 \end{bmatrix}$$

$$x = \begin{bmatrix} 0.05 \\ 0.27 \\ 0.12 \\ 0.57 \end{bmatrix}$$

Step 4 – Relative consistency evaluation

IR = 0.9

$$Ax \cong \begin{bmatrix} 0.19 \\ 1.09 \\ 0.46 \\ 2.38 \end{bmatrix}$$

$$D_{max} = Average\left(\frac{0.19}{0.05}; \frac{1.09}{0.27}; \frac{0.46}{0.12}; \frac{2.38}{0.57}\right) \cong 4.08$$

$$IC = (D_{max} - 4) * (4 - 1) \cong 0.24$$

$$RC = \frac{IC}{IR} = \frac{0.24}{0.9} \cong 0.26 = 26\%$$

Values should have the lowest RC possible, hopefully under 10% but 26% is reasonable in this case given the removal of a great deal of criteria for this example.

Step 5 – Matrix for each criteria

For this only ideas I2-PT, I4-KD, I5-RM and I7-NP will be tested as they represent a decent range of classification in the weighted evaluation matrix.

C1	I2	I4	I5	I7		C1	I2	I4	I5	I7	Mean
I2	1	4	1/2	1		I2	0,24	0,33	0,25	0,18	0,25
I4	1/4	1	1/5	1/2		I4	0,06	0,08	0,10	0,09	0,08
I5	2	5	1	3		I5	0,47	0,42	0,49	0,55	0,48
I7	1	2	1/3	1		I7	0,24	0,17	0,16	0,18	0,19
Sum	4 1/4	12	2	5 1/2							
C2	I2	I4	I5	I7		C2	I2	I4	I5	I7	Mean
I2	1	1	1/3	1/2		I2	0,14	0,14	0,13	0,17	0,14
I4	1	1	1/3	1/2		I4	0,14	0,14	0,13	0,17	0,14
I5	3	3	1	1		I5	0,43	0,43	0,38	0,33	0,39
I7	2	2	1	1		I7	0,29	0,29	0,38	0,33	0,32
Sum	7	7	2 2/3	3							
C3	I2	I4	I5	I7		C3	I2	I4	I5	I7	Mean

I2	1	1/2	1	1		**I2**	0,20	0,20	0,20	0,20	0,20
I4	2	1	2	2		**I4**	0,40	0,40	0,40	0,40	0,40
I5	1	1/2	1	1		**I5**	0,20	0,20	0,20	0,20	0,20
I7	1	1/2	1	1		**I7**	0,20	0,20	0,20	0,20	0,20
Sum	5	2 1/2	5	5							
C4	**I2**	**I4**	**I5**	**I7**		**C4**	**I2**	**I4**	**I5**	**I7**	**Mean**
I2	1	1/4	1/3	2		**I2**	0,12	0,13	0,09	0,20	0,13
I4	4	1	2	4		**I4**	0,47	0,50	0,55	0,40	0,48
I5	3	1/2	1	3		**I5**	0,35	0,25	0,27	0,30	0,29
I7	1/2	1/4	1/3	1		**I7**	0,06	0,13	0,09	0,10	0,09
Sum	8 1/2	2	3 2/3	10							

Criteria Matrix

	C1	**C2**	**C3**	**C4**
I2	0,25	0,14	0,20	0,13
I4	0,08	0,14	0,40	0,48
I5	0,48	0,39	0,20	0,29
I7	0,19	0,32	0,20	0,09

$$PM = \begin{bmatrix} 0.25 & 0.14 & 0.20 & 0.13 \\ 0.08 & 0.14 & 0.40 & 0.48 \\ 0.48 & 0.39 & 0.20 & 0.29 \\ 0.19 & 0.32 & 0.20 & 0.09 \end{bmatrix}$$

Step 6 – Composed Priority

$$\begin{bmatrix} 0.25 & 0.14 & 0.20 & 0.13 \\ 0.08 & 0.14 & 0.40 & 0.48 \\ 0.48 & 0.39 & 0.20 & 0.29 \\ 0.19 & 0.32 & 0.20 & 0.09 \end{bmatrix} x \begin{bmatrix} 0.05 \\ 0.27 \\ 0.12 \\ 0.57 \end{bmatrix} \cong \begin{bmatrix} 0.15 \\ 0.36 \\ 0.32 \\ 0.17 \end{bmatrix}$$

Step 7 – Choice

Regarding the results of the AHP method the choice to select would be I4 followed by I5, in the method utilized however the choice would be I5 followed by I2, this result is certainly due to the exclusion of many criteria as well as the translation of the values utilized in the weighted evaluation matrix.

D. TOPSIS Utilization Example

Incredibly simple to apply to the Weighted Comparison Matrix utilized is the TOPSIS method which could have been used but was not due to preference of the evaluating party.

In the TOPSIS method there are some elements that should be considered:

Ideal Alternative and Negative Ideal Alternative are the basis for the selection process one being the perfect solution and the other the worse solution possible, the TOPSIS method selects the alternative closest to the ideal alternative and furthest for the negative ideal alternative.

For that matrix used in practice can be adapted. For the TOPSIS method the ideas have been renamed to belong in negative and positive categories and the values normalized for a scale of 1 to 9 in opposition to the 1 to 5 utilized. Then $\sum x_{ij}^2$ and $(\sum x_{ij}^2)^{1/2}$ are calculated.

Step 1

	I1-SE	I2-PT	I3-CP	I4-KD	I5-RM	I6-SS	I7-NP
Proximity of the Client with the Insurance Company promotion	1	5	1	1	9	5	5
New Customer Acquisition/Maintenance	1	7	1	1	5	7	3
Value Potential for the Insurance Company	5	9	3	5	9	5	7
Innovation Level for i2S	3	9	1	5	7	5	7
Ease of Integration with i2S solutions	5	1	1	1	5	1	3
i2S Core Independence	7	3	9	7	3	5	3
Market Potential	3	9	3	7	9	5	3
Insurance Company Marketing Potential	1	9	1	1	7	3	7
Insurance Sector Relevance	5	9	3	3	7	3	5
Previous Information Dependence	1	1	1	1	7	5	9
Development Difficulty	5	7	3	3	5	5	9
$\sum x_{ij}^2$	171	539	123	171	523	243	395
$(\sum x_{ij}^2)^{1/2}$	13.08	23.22	11.09	13.08	22.87	15.59	19.87

All columns are then divided by $(\sum x_{ij}^2)^{1/2}$

	I1-SE	I2-PT	I3-CP	I4-KD	I5-RM	I6-SS	I7-NP
Proximity of the Client with the Insurance Company promotion	0.08	0.22	0.09	0.08	0.39	0.32	0.25
New Customer Acquisition/Maintenance	0.08	0.30	0.09	0.08	0.22	0.45	0.15
Value Potential for the Insurance Company	0.38	0.39	0.27	0.38	0.39	0.32	0.35
Innovation Level for i2S	0.23	0.39	0.09	0.38	0.31	0.32	0.35
Ease of Integration with i2S solutions	0.38	0.04	0.09	0.08	0.22	0.06	0.15
i2S Core Independence	0.54	0.13	0.81	0.54	0.13	0.32	0.15
Market Potential	0.23	0.39	0.27	0.54	0.39	0.32	0.15
Insurance Company Marketing Potential	0.08	0.39	0.09	0.08	0.31	0.19	0.35
Insurance Sector Relevance	0.38	0.39	0.27	0.23	0.31	0.19	0.25
Previous Information Dependence	0.08	0.04	0.09	0.08	0.31	0.32	0.45
Development Difficulty	0.38	0.30	0.27	0.23	0.22	0.32	0.45

Step 2 Then Multiplied by the weight factor that is not present in this table but is on the one that was used (Refer to Table 9).

	I1-SE	I2-PT	I3-CP	I4-KD	I5-RM	I6-SS	I7-NP
Proximity of the Client with the Insurance Company promotion	0.00	0.01	0.00	0.00	0.02	0.02	0.01
New Customer Acquisition/Maintenance	0.00	0.02	0.00	0.00	0.01	0.02	0.01
Value Potential for the Insurance Company	0.02	0.02	0.01	0.02	0.02	0.02	0.02
Innovation Level for i2S	0.02	0.04	0.01	0.04	0.03	0.03	0.04
Ease of Integration with i2S solutions	0.06	0.01	0.01	0.01	0.03	0.01	0.02
i2S Core Independence	0.05	0.01	0.08	0.05	0.01	0.03	0.02
Market Potential	0.02	0.04	0.03	0.05	0.04	0.03	0.02
Insurance Company Marketing Potential	0.01	0.04	0.01	0.01	0.03	0.02	0.04
Insurance Sector Relevance	0.02	0.02	0.01	0.01	0.02	0.01	0.01
Previous Information Dependence	0.00	0.00	0.00	0.00	0.02	0.02	0.02
Development Difficulty	0.08	0.06	0.05	0.05	0.04	0.06	0.09

Step 3 Now the ideal and the negative ideal alternatives are calculated

$$A^{*} = \{0.02 \quad 0.02 \quad 0.02 \quad 0.04 \quad 0.06 \quad 0.08 \quad 0.05 \quad 0.04 \quad 0.02 \quad 0.00 \quad 0.04\}$$

$$A' = \{0.00 \quad 0.00 \quad 0.01 \quad 0.01 \quad 0.01 \quad 0.01 \quad 0.02 \quad 0.01 \quad 0.01 \quad 0.02 \quad 0.09\}$$

Step 4 Next each alternative's separation to the ideal is calculated by using the formulas presented.

$$S_i^{*} = \left[\sum(v_j^{*} - v_{ij})^2\right]^{1/2}$$

$$S^{*} = \{0.07 \quad 0.09 \quad 0.07 \quad 0.07 \quad 0.08 \quad 0.08 \quad 0.10\}$$

$$S_i' = \left[\sum(v_j' - v_{ij})^2\right]^{1/2}$$

$$S' = \{0.07 \quad 0.06 \quad 0.08 \quad 0.08 \quad 0.07 \quad 0.05 \quad 0.04\}$$

Step 5 The best possible solution by determining the closeness of the two parameters is calculated

$$C_i^{*} = \frac{S_i'}{S_i^{*} + S_i'}$$

$$C^{*} = \{0.51 \quad 0.42 \quad 0.53 \quad 0.54 \quad 0.48 \quad 0.39 \quad 0.30\}$$

According to this result the ideas can be sorted from best to worse as follows.

I4-KD > I3-CP > I1-SE > I5-RM > I2-PT > I6-SS > I7-NP

Weirdly enough the chosen idea, I5-RM, was in the middle of the classification, outmatched by ideas that were summarily dismissed during the selection process.

E. Technology Readiness Level Detail

All the information contained in this annex refers to a single reference by the European Association of Research and Technology Organizations(EARTO) and refers to the TRL definitions provided in the Horizon 2020 Programme (H2020) by the European Commission(EC). These levels cannot be directly translated to a software context as the EARTO focuses their definitions of physical products and not virtual ones (European Association of Research and Technology Organizations, 2014; European Comission, 2017).

Invention Cluster

TRL 1

H2020: Basic principles observed

EARTO: Basic principles observed

Definition and Description

Basic scientific research is translated into potential new basic principles that can be used in new technologies.

TRL 2

H2020: Technology concept formulated

EARTO: Technology concept formulated

Definition and Description

Potential of the basic principles are identified, including their technological concept. Also the first manufacturing principles are explored, as well as possible markets identified. A small research team is established to facilitate assessment of technological feasibility.

Concept Validation Cluster

TRL 3

H2020: Experimental proof of concept

EARTO: First assessment of feasibility of the concept and technologies

Definition and Description

Based on preliminary study, now actual research is conducted to assess technical and market feasibility of the concept. This includes active R&D on a laboratory scale and first discussions with potential clients. The research team is further expanded and early market feasibility assessed.

TRL 4

H2020: Technological validity in a lab

EARTO: Validation of an integrated prototype in a laboratory

Definition and Description

Basic technological components are integrated to assess early feasibility by testing in a laboratory environment. Manufacturing is actively research, identifying the main production principles. Lead Market are engaged to ensure connection with demand. Organisation is prepared to enter into scale up, possible services prepared and full market analysis conducted.

Prototyping and Incubation Cluster

TRL 5

H2020: Technology validated in relevant environment

EARTO: Testing of the prototype in a user environment

Definition and Description

The system is tested in a user environment, connected to the broader technological infrastructure. Actual use is tested and validated. Manufacturing is prepared and tested in a laboratory environment and lead markets can test pre-production products. First activities within the organization are established to further scale up to pilot production and marketing.

Pilot Production and Demonstration Cluster

TRL 6

H2020: Technology demonstrated in relevant environment

EARTO: Pre-production of the product, including testing in a user environment

Definition and Description

Product and manufacturing technologies are now fully integrated in a pilot line or pilot plant (low rate manufacturing). The interaction between the product and manufacturing technologies are assessed and fine-tuned, including additional R&D. Lead markets test the early products and manufacturing process and the organization of production is made operational (including marketing, logistics, production and others).

TRL 7

H2020: System prototype demonstration in an operational environment

EARTO: Low scale pilot production demonstrated

Definition and Description

Manufacturing of the product is now fully operational at low rate, producing actual commercial products. Lead markets test these final products and organisational implementation is finalized (full marketing established, as well as all other production activities fully organized). The product is formally launched into first early adopter markets.

Initial Market Introduction Cluster

TRL 8

H2020: System completed and qualified

EARTO: Manufacturing fully tested, validated and qualified

Definition and Description

Manufacturing of the product, as well as the product final version is now fully established, as well as the organisation of production and marketing. Full launch of the product is now established in national and generally early majority markets.

Market Expansion Cluster

TRL 9

H2020: Actual system proven in operational environment

EARTO: Production and product fully operational and competitive

Definition and Description

Full production is sustained, product expanded to larger markets and incremental changes in the product create new versions. Manufacturing and overall production is optimized by continuous incremental innovations to the process. Early majority markets are fully addressed.

F. Gamification Methods Evaluation Table

Method	Preparation	Analysis	Ideation	Design	Implementation	Evaluation	Monitoring	Method Evaluation
Towards a framework for gamification design on crowdsourcing systems: the G.A.M.E. approach	√	√	-	√	√	√	-	Case Study
Gamify: How Gamification Motivates People to Do Extraordinary Things	√	√	√		-	√	√	No evaluation
The lens of intrinsic skill atoms: a method for gameful design	√	√	√		√	√	-	Development workshops, case study
Game frame: Using games As a Strategy For Success	√	√	√		-	√	-	No evaluation
Achievement Unlocked: Investigating the Design of Effective Gamification Experiences For Mobile Applications and Devices	√		√		-	√		Case study
Gamification: Analysis and Application	√	√	√		-	√	-	No evaluation
A method for the design of gamified trainings	√		√		√	√		Case study
Enterprise Gamification: Engaging People by Letting Them Have Fun	√	√	√	√	√	√	√	No evaluation
Implementing Gamification: Requirements and Gamification Platforms	√	√	√		√	√		Two case studies
The Gamification of Learning and Instruction: Game-Based Methods and Strategies for Training and Education	√	√	√		√	√	-	No evaluation
Implementation model for the gamification of business processes a study from the field of material handling (removed from source)	√		√			√		Case study
Gamification At Work	√	√	√		-	√		No evaluation
Process for gamification: From the decision of gamification to its practical implementation	√	√	√		-	-	-	No evaluation
Game on: Energize Your Business With Social Media Games	√	√	√		-	√	-	No evaluation
Is it all a game? Understanding the principles of gamification	-	√	√		-	√	-	No evaluation
Ein Vorgehensmodell für angewandte Spielformen	√	√	√	√	√	√	-	No evaluation
For the Win: How game Thinking Can Revolutionize Your Business	√	√	√		√	√	-	No evaluation

Further information available in reference (Morschheuser *et al.*, 2018).

G. Concept/Rule JSON Evolution

G.1 Concept JSON

During the design of the concept system as well as early prototyping changes to its design were necessary as this does not directly pertain to the state of the final design it was left out of the main document, however the existence of an evolution in its design requires the description of why and how changes were made to the system to answer to the requirements detected.

On a first design the JSON for Concepts was the one displayed in

Code 37.

Code 37 – Stage 1 Concept JSON

```
{
    "bar":{
        "CK":"00002-0000000001",
        "display_text":"Progress Bar",
        "value": 3,
        "composed": [
            {
                "tag":"ranges",
                "concepts":[
                    "00001-0000000001",
                    "00001-0000000002",
                    "00001-0000000003",
                    "00001-0000000004"
                ]
            }
        ]
    }
}
```

Originally this design seemed appropriate, an ID is present in the form of CK composed by the concept type key and the concept instance key, we have field types like display_text and value and we have an array of elements that compose a concept. This would change greatly when MongoDB was considered for the database when concepts would be saved in the system.

It was quickly determined that the CK should be separated into two different keys, CTK and CIK, instead of the single string key we now have two numeric keys. Additional data was also deemed necessary for the proper utilization of concepts by the system. With this in mind a new object was created in the concept called “meta” which contained metadata relating to the object utilization as well as the CK object. Leading to the JSON structure seen in Code 38.

Code 38 – Stage 2 Concept JSON

```
{
    "user":{
        "meta":{
            "ck":{
                "ctk":0,
                "cid":0
            },
            "direct_access":false,
            "prebuilt":{
                "isprebuilt":false,
                "timesettings":"tbd"
            },
            "static":{
                "isStatic" :false,
                "populator":{

                }
            },
            "lazy":false,
            "trigger":[]
        },
        "username": "Sir Charles Hatford",
        "password": "FishNChips",
        "composed":[
            "00002-0000000004"
        ]
    }
}
```

This type of JSON still had some issues, namely on the composed part of the concept, it still used the old CK key, this had to be changed to allow proper searching. The CK object being inside the meta tag also, unnecessarily, increased search times due to its nesting, as such it was moved to the root object. Leading to the JSON presented in As can be seen this JSON still requires completion of certain details namely it is possible to see that the prebuilt, static and trigger objects are not yet completely determined. These still required some work to be complete.

Code 39.

As can be seen this JSON still requires completion of certain details namely it is possible to see that the prebuilt, static and trigger objects are not yet completely determined. These still required some work to be complete.

Code 39 – Stage 3 Concept JSON

```
{
    "user":{
        "ck":{
            "ctk":0,
            "cid":0
        },
        "meta":{

            "direct_access":false,
            "prebuilt":{
                "isprebuilt":false,
                "timesettings":"tbd"
            },
            "static":{
                "isStatic" :false,
                "populator":{

                }
            },
            "lazy":false,
            "trigger":[]
        },
        "username": "Sir Charles Hatford",
        "password": "FishNChips",
        "composed":[
            {
                "ctk":2,
                "cik":[
                    4
                ]
            }
        ]
    }
}
```

Stage 4 brought a significant overhaul to the structure of the fields of the concept as well as metadata, root element naming and trigger array. A new, more complete JSON requires the proper definition of fields and allows the passage of parameters to triggered rules, which are triggered in order. Most metadata was removed however as the systems to put it in place were not yet present and as such were adding useless data to the files, addition of these fields in future iterations will require small changes in the already implemented code but they should not be significant as the database is quickly updated utilizing the JPA implementation. Triggers can be utilized in this concept definition however they should be in the new Concept Template which means they will be triggered by all concepts of that Concept Template.

Code 40 - Stage 4 Concept JSON

```
{
    "trigger" : [
        {
            "rule" : 1,
            "parameters" : [
                "[nome]"
            ]
        }
    ],
    "ck" : {
        "ctk" : 1,
        "cik" : 1
    },
    "tag" : "pessoa",
    "directAccess" : true,
    "fields" : {
        "ref" : {
            "type" : "String",
            "value" : "efaa5d4604cd016660ba"
        },
        "nome" : {
            "type" : "String",
            "value" : "John Shepard"
        }
    },
    "composing" : [
        {
            "ctk" : 2,
            "cik" : [
                1
            ]
        }
    ]
}
```

After the stage 4 JSON a single evolution happened which created the JSON present in the report.

G.2 Template JSON

With the increase of detail regarding the Configuration module it was deemed that another type of JSON was necessary. The Concept Template JSON which would be used to define and validate the structure of individual Instances. Originally these files looked as seen in CODE.

Code 41 – Stage 1 Concept Template JSON

```
{
    "concept_template":{
        "meta":{
            "ctk":1,
            "tag":"user",
            "direct_access":false,
            "prebuilt":{
                "isprebuilt":false,
                "timesettings":"tbd"
            },
            "static":false,
            "lazy":false,
            "trigger":[]
        },
        "fields":[
            {
                "tag":"userid",
                "type":"String"
            },
            {
                "tag":"currentsession",
                "type":"Int32"
            }
        ],
        "composed":[]
    }
}
```

Although the utilization of JSON Schema was debated to replace this Concept Template it was deemed too volatile in its current state to utilize properly and as such this system was maintained, in any case the Concept Template did suffer some alterations after it was out of use for a time leading to the JSON file that can be seen in

Code 42.

Code 42 – Stage 2 ConceptTemplate JSON

```
{
    "ctk" : 1,
    "tag" : "pessoa",
    "directaccess" : true,
    "trigger" : [

    ],
    "fields" : {
        "ref" : {
            "type" : "String",
            "value" : ""
        },
        "nome" : {
            "type" : "String",
            "value" : ""
        }
    }
}
```

This new version follows mostly the same structure as the Concept JSON itself, including the fields tag as well as the deemed old “meta” tags such as direct access, it however possesses a ctk instead of a ck.

This new version also possesses two important capabilities, it can define default values for new concepts and allows the definition of triggers that can then be utilized to execute rules for all concepts of that type.

Another version exists after this and its described in the main body.

G.3 Rule JSON

When it was determined that the Rules had to be handled by the Gamification Module itself it was necessary to allow the configurator to determine rules for various areas. For this a Rule JSON was design. This Rule system had to handle multiple types of operations, namely mathematical, logical and function Type operations. Mathematical operations were simple as it referred to only sums, subtractions, multiplications and divisions, however logical operations, such as ifs, ands, ors and comparisons, required a more detailed design as did function operations, much as the concept instances Rule JSONs had to be validated utilizing a Draft 7 Schema; however, this had not been determined at this point. As such a Rule JSON for this stage of design can be seen in CODE.

Code 43 – Stage 1 Rule JSON

```
{
    "rule":{
        "meta":{
            "key":"1"
        },
        "expression":{
            "left":{
                "type":"Number",
                "value":"1"
            },
            "operation":"+",
            "right":{
                "type":"Number",
                "value":"2"
            },
            "return":"Number"
        }
    }
}
```

This JSON presented some problems from the start. First, meta information contained only one field, the key field, which has determined to not be necessary to be present in the meta object. As such this object had to be purged. The expression object also had some issues in its elements, left and right have the value type discriminated, however in some cases, like the presented one, this can be known by requesting the JSON element’s type. As such this was an unnecessary tag for this. The operation tag was also extremely simple, only containing a string value, this worked fine for mathematical and logical comparisons however it presented a problem when functions had to be called, a lack of information was obviously present. Finally the left and right objects did not have any concept of where to get the values to populate the, did they come from a concept or where they just “hardcoded” into the JSON itself, was a question not answered at this stage Finally the return type is only necessary when it comes to function operations as, when regarding any other type (mathematical or logical), the return type can be inferred by the operation itself.

Another issue was the complete impossibility to use this to do a larger set of actions in a row as well as having to define a complex rule structure to chain conditions. Utilizing an expression parser to deal with the expression liberated the Rule to deal with higher level operations. And as such the Rule JSON evolved to the one presented in Stage 2 CODE.

This stage 2 JSON still had some issues, for one all the variables utilized to calculate the rule were hard coded into the rule itself, as such it was necessary to alter this to allow the rules to be parameterized. Another issue that presents itself is that the Condition can only have one expression evaluation, and not multiple like the then(renamed from when) and else parts of the structure. This required then a small restructuring of the JSON into what is called Stage 3.

In stage 3 the rule JSON allows various things, the transfer of facts down along a list of actions, the possibility of facts being initialized by values sent by the triggerer. And an extended list of functions that allow rules to interact with concepts. Returning multiple fields from a rule is however impossible in this stage, and may be reviewed in a stage 4.

Stage 4's json suffered a major changes, not only to address the issues presented by the stage 3 Json but also to comply with the usage of JPA to persist the objects in MongoDB.

Code 44 - Stage 2 Rule JSON

```
{
    "_id":"5a9ff270ea9e3116acf99cf7",
    "rule":{
        "meta":{
            "key":"1",
            "priority":"AVERAGE",
            "autotrigger":false
        },
        "condition":{
            "facts":[
                {
                    "type":"String",
                    "key1":"potato"
                },
                {
                    "type":"String",
                    "key2":"potato"
                }
            ],
            "operation":"key1 = key2"
        },
        "when":{
            "actions":[
                {
                    "facts":[
                        {
                            "type":"String",
                            "key1":"potato"
                        }
                    ],
                    "action":{
                        "operation":"printString(key1)",
                        "returns":{
                            "type":"Boolean",
                            "key":"return1"
                        }
                    }
                }
            ]
        },
        "else":{
            "actions":[]
        }
    }
}
```

Code 45 – Stage 1 Uinfo Template JSON

G.4 UInfo JSON

Code 46 - UInfo JSON Example

```
{
    "ctk" : "1",
    "infofields" : [
        {
            "fields" : [
                "nome"
            ],
            "uinfo" : [
                {
                    "langs" : {
                        "de" : "Name",
                        "pt" : "Nome",
                        "en" : "Name",
                        "es" : "Nombre"
                    }
                }
            ]
        }
    ],
    "_id" : ObjectId("5b72af0001811844747f4ecf")
}
```

In Code 46 we can see an example of a UInfoTemplate which was used for testing in the early stages of the prototype implementation. It refers to the Concept Template with CTK 1 as it shares an index with it and UInfoTemplates were supposed to work on a 1 to 1 basis.

As can be seen the Template possessed an array element “infofields” that stored UInfoField objects. This object specified with set of fields the UInfo referred to and the UInfo to apply, the object in this file is an Append Type, more specifically an NfoDisplayText, determined by the “langs” tag, the content to append being inside the tag itself, this would then be added to a Concept’s field and result in the following JSON:

H. Game Design Lens Answers

Lens #1 The Lens of Essential Experience

The objective of this game is to make the user feel like they are aiding not only themselves but also the world around them while also bettering themselves while driving. Obfuscating the insurance data behind the game itself.

An idea to obfuscate the idea that you are paying an insurance is to classify the price as something else such as presenting the discount as the Agent's Salary, you go along the lines and your Agent Salary goes up, your premium going down.

Lens #2 The Lens of Surprise

Surprise is a good idea to implement in this system however the concept of surprise itself is rather odd in this situation since it's based on a deterministic basis, the Insurance Policy.

Due to this restriction the only possible application to this is the unknown of the classification for this week. Not knowing the statistics about the current week gives a feeling of expectation for the end of the week when these values are published.

The rules of the system are however as they don't allow for extreme flexibility.

Lens #3 The Lens of Fun

The completion of challenges needs to be more fun and rewarding to allow the player to feel engaged in the system as it is the only interactive portion.

The classification system can also be detrimental as it may make the player feel as he is being judged.

This needs further consideration.

(A Loot System may be an option) The user gets a set of prize boxes according to their ranking which give various rewards of similar value. A deterministic approach where a rank gets you something specific is also an option.

LOSING IS NOT FUN HERE

Lens #4 The Lens of Curiosity

What questions does in fact this system pose to the player. The player can question "How do I make my driving better?", a set of tips could be present for the player to know what to do in certain situations. A score can cause questions to the player of why exactly that score was warranted so this must be broken down.

Criteria must be presented and allowed to be analysed so they can think how they can change these criteria next week. As such a presentation of this data should be available on each cycle.

Lens #5 The Lens of Endogenous Value

In the context of this system the value the player saves from their insurance is the real value item in the game, any other rewards are tangential.

Therefor the Agent Salary or Agent Bonus.

Lens #6 The Lens of Problem Solving

Inherently the problem presented by the SDA game is "How do I get the biggest Agent Bonus", however this has sub problems such as "How do I complete these challenges"and moment to moment details also arise such as "How do I get to this location faster?", "Do I go faster though this high risk road or slower trough the low risk one?".

Lens #9 The Lens of Unification

The theme is an Aliance of Agents that are trying to improve the safety of the roads of their area, everything must be aligned with that.

The Agency logo is driving themed.

You have Operations instead of mission sets with Operational Objectives.

Lens #10 The Lens of Resonance (WIP)

What makes the game feel special? I don't know I need a theme that resonates with the audience but, when the audience is this general, how can the theme resonate?

Lens #11 The Lens of Infinite Inspiration

Need to find the essence of something to take and put into the project.

Lens #12 The Lens of the Problem Statement

I am trying to create a gamified approach to a Telematic insurance product that allows an increase of interaction between the player and insurance company. This problem is solved utilizing the criteria defined in the report.

Lens #14 The Lens of Risk Mitigation

People may straight up not like the game since it is based on an insurance product that must be purchased beforehand. People are, according to personal, informal interviews, reticent in accepting this kind of product into their insurance.

Lens #15 The Lens of the Toy

No, it wouldn't be fun without the goal as its inherently based on the goal itself, as the mechanics are bound to driving itself, all the fun is derived from the subjective enjoyment of driving of a person and driving with a set of goals.

Although people do find enjoyment with just playing with their car and this system would merely be something on top of the car itself.

Lens #16 The Lens of the Player

This product hits many players and as such has to take into account the scattered variables for people from 18 to 80 which is hard, even without taking into account the gender of the users, which require different approaches.

This is more targeted for the 18 to 35, male segment. Its very focused on master of driving and challenge completion and the possibilities within the insurance medium don't allow for any significant social aspect.

Lens #17 The Lens of Pleasure

I don't see this setting giving players any other kind of pleasure than the one caused by the completion of goals and milestones, only enabling the Achiever Bartle Type player and to some extent the explorer variety and I don't see how Killers and Socializers can be enabled.

Lens #18 The Lens of Flow

Challenges need to scale according to the rating of the user in the previous week, dips in this must not have a major impact on the challenge difficulty.

Lens #19 The Lens of Needs

This game works on the Self-Esteem level of Maslow's Hierarchy of Needs, it cannot possibly go any lower, unless we see it as the Safety level when regarding financial and safety and freedom from fear, however this is related to insurance itself and not the game.

Lens #20 The Lens of Judgement

The player is judged on their driving prowess and safety, this data is made according to only telematic data and is devoid of context making the judgement sometimes feel unfair. A measure of leeway may be necessary to properly adapt the judgement formula.

Lens #21 The Lens of Functional Space

As the game is technically the driving portion the game space is functionally the entire world. Separate from this however is the game screen, the SDA Agent Control Panel.

Lens #22 The Lens of Dynamic State

The objects in the game are the driver's stats only, this information is also not completely public but instead from treated raw data. None of this information is available to anyone but the player and the game. The game does however know this about all players.

Lens #24 The Lens of Action

The mechanics of this area are based on the car itself and as such the system cannot appropriate them, add to them or alter them to cause emergence.

Lens #25 The Lens of Goals

The goal of the game is to increase your Agent Bonus trough the completion of attributed goals, a set of 3 goals is generated and the player has to complete only 2 for a successfully week, but 3 will hold a greater reward.

Lens #26 The Lens of Rules

Most rules of the game are the laws of the road. Within the system some rules exist such as the number of goals that can be selected.

Goals have their own rules, and these are completely enforced after definition.

Lens #27 The Lens of Skill

Driving skill.

Lens #28 The Lens of Expected Value

This game doesn't have any chance-based events as it is entirely dependant on the users performance. Analysing the risk factors themselves may be taken into account.

Lens #29 The Lens of Chance

Nothing.

Lens #30 The Lens of Fairness

As this game isn't inherently competitive fairness isn't a great factor, players can be ranked in the leader boards utilizing their distance travelled ranks though, giving players who travel more a reasonable chance to achieve top places within the leader board.

Lens #31 The Lens of Challenge

Challenges are weekly and must be strived towards the difficulty of these challenges should scale with performance, increasing and decreasing in difficulty with the skill level of the player.

Lens #32 The Lens of Meaningful Choices

The player has no choices in the current situation that are extremely meaningful

Lens #33 The Lens of Triangularity

Longer distance travels with high classification yield better rewards for the players, a 10 in max distance is worth double in min distance.

Lens #34 The Lens of Skill vs. Chance

There is no chance.

Lens #35 The Lens of Head and Hands

The players will be looking for accomplishment and the game itself is techicly played outside .

Lens #36 The Lens of Competition

Players have a leaderboard according to their rank with semi-public info available, this is the only aspect of competition

Lens #39 The Lens of Time

Gameplay activates take their normal ride, utilizing the Agent Board should only take around 5 to 10 minutes a week when the update rolls out.

Lens #40 The Lens of Reward

The game gives out rewards in monetary discounts from their insurance premiums by default.

Lens #41 The Lens of Punishment

Punishment comes at a higher cost to their insurance policy each week than they would normally have, while they would pay less if they hit the average result.

Lens #44 The Lens of Character

The game is presented as missions that an agency gives out to the player, the agent, this uniqueness will possibly resonate with a few people.

Lens #45 The Lens of Imagination

The player must know how to drive and have a car, other than that most information will be supplied by the game, namely their personal statistics.

Lens #46 The Lens of Economy

The players can earn Reputation trough mission accomplishment and rankings, this reputation can be redeemed for certain rewards such as policy extensions or possibly products that the company would be open to distribute that, although cost little for the company to produce would make the user more engaged by having a tiny sticker to put on their car or a pin to display.

Lens #48 The Lens of Accessibility

All the solving of the challenges to the game should be pretty recognizable from the get go, if not some information can be displayed to the player in the screen.

Lens #49 The Lens of Visible Progress

Progress is shown by the player Agent Ranking bar that gets filled as they complete challenges and earn rankings. All this progress is visible they are also presented their challenge tier.

Lens #50 The Lens of Parallelism

The main bottleneck is the fact that the updates to the system can only happen once a day and the challenges can only be completed on a weekly basis, making the player be absent for some time.

The challenges themselves are weekly or monthly.

Lens #51 The Lens of the Pyramid

All challenges feed into the Yearly Mission which calculates the final premium of the insurance. It slowly grows over time until it is achieved at the end of the year.

Lens #53 The Lens of Control

The interface must be able to quickly show the following thins

User stats and information

Challenges

User Trophies and Accomplishments

Agent Trophies and Rankings

Rewards Store

Lens #62 The Lens of Inherent Interest

This game only appeals to the interest of mastery and money saving.

Lens #63 The Lens of Beauty

It’s a dummy use standard Bootstrap.

Lens #72 The Lens of Indirect Control

I want players to pick a set challenges I want them to come back to see the challenges and pick them.

This can be done by a weekly reminder.

Lens #73 The Lens of Collusion

I want the player to experience that they are moving forward in the ranks of an agency that, although there are no characters in the game.

(Make a character The AI)

Lens #82 The Lens of Inner Contradiction

The purpose of the game is to motivate people to have additional interaction with their insurance company and closely track their risk values, specifically regarding driving.

Lens #94 The Lens of the Client

The Client wants to make their products more innovating in to the consumer utilizing novel systems that increase the interaction with the customer.

Lens #97 The Lens of Transformation

This game can improve players by adding to their understanding of their driving prowess and advising them on methods to allow them to augment their skills, creating a better, safer, environment for everyone.

I. GM Configuration File

```
# ----------------------------------------
# CORE PROPERTIES
# ----------------------------------------

debug=false
trace=false

server.port = 30000

# DATABASE ACCESS
data.access.database=mongodb

# MONGO DB CONFIG
spring.data.mongodb.host=localhost
spring.data.mongodb.port=27017
spring.data.mongodb.database=gamification

# REF PROVIDER
ref.provider.url=http://localhost:30001
ref.provider.token=7e019643b73065bce6baed062e7a739b

# TASK CONFIGURATION
task.generalupdate.delay = 300000
task.generalupdate.rate = 300000

task.prebuild.delay = 86400000
task.prebuild.rate = 86400000

# JWT
jwt.header=Authorization
jwt.secret=mySecret
jwt.expiration=604800
jwt.route.authentication.path=/auth
jwt.route.authentication.refresh=/refresh
```