

KAUNAS UNIVERSITY OF TECHNOLOGY

DARIUS AŠERISKIS

**MODELLING AND EVALUATION OF SOFTWARE  
SYSTEM GAMIFICATION ELEMENTS**

Doctoral Dissertation  
Technological Sciences, Informatics Engineering (07T)

2017 Kaunas

This dissertation was prepared in years 2013 – 2017 in Department of Software Engineering of Faculty of Informatics of Kaunas University of Technology.

**Scientific Supervisor:**

Prof. Dr. Robertas DAMAŠEVIČIUS (Kaunas University of Technology, Technological Sciences, Informatics Engineering – 07T).

Doctoral dissertation has been published in:

<http://ktu.edu>

**Editor:**

Bronius Broniauskas (Publishing Office “Technologija”)

© D. Ašeriškis, 2017

ISBN 978-609-02-1375-9

The bibliographic information about the publication is available in the National Bibliographic Data Bank (NBDB) of the Martynas Mažvydas National Library of Lithuania

KAUNO TECHNOLOGIJOS UNIVERSITETAS

DARIUS AŠERIŠKIS

**PROGRAMŲ SISTEMŲ ŽAIDYBINIMO ELEMENTŲ  
MODELIAVIMAS IR VERTINIMAS**

Daktaro disertacija  
Technologijos mokslai, Informatikos inžinerija (07T)

2017 Kaunas

Disertacija rengta 2013-2017 metais Kauno technologijos universitete, Informatikos fakultete, Programų sistemų katedroje.

**Mokslinis vadovas:**

Prof. dr. Robertas DAMAŠEVIČIUS (Kauno technologijos universitetas, technologijos mokslai, informatikos inžinerija – 07T)

Interneto svetainės, kurioje skelbiama disertacija, adresas:

<http://ktu.edu>

**Redagavo:**

Bronius Broniauskas (leidykla “Technologija”)

© D. Ašeriškis, 2017

ISBN 978-609-02-1375-9

Leidinio bibliografinė informacija pateikiama Lietuvos nacionalinės Martyno Mažvydo bibliotekos Nacionalinės bibliografijos duomenų banke (NBDB)

## ACKNOWLEDGMENT

The author would like to thank Prof. Dr. Robertas Damaševičius for traveling through this six-year journey together. The author is grateful to Robertas for the opportunity to work side by side and to reach the desired goals.

The author would also like to thank his wife Andžela and daughter Deimantė for support and understanding. The Author is grateful to his parents Laima and Antanas and sisters Dovilė and Daiva for believing in him. Also, the author thanks his wife's family for moral support.

## TABLE OF CONTENTS

TERMS AND ABBREVIATIONS .....	9
FIGURES .....	11
TABLES .....	13
1. INTRODUCTION .....	14
1.1. Motivation.....	14
1.2. Object and scope of the research .....	15
1.3. Problem statement and research questions .....	15
1.4. Aim and objectives .....	15
1.5. Defended propositions .....	16
1.6. Major contributions and novelty of the research .....	16
1.7. Practical significance .....	17
1.8. Scientific approval .....	17
1.9. Structure of the dissertation .....	17
2. THE ANALYSIS OF GAMIFICATION.....	18
2.1. Background of gamification .....	18
2.2. Psychological and social aspects of gamification.....	19
2.3. Modeling of Gamification .....	25
2.3.1. Overview of gamification analysis methods.....	25
2.3.2. Introduction to Machinations modeling framework .....	29
2.3.3. Formal models of game design and gamification .....	30
2.4. Agent-based simulation and social gaming .....	33
2.5. Software generation from models.....	34
2.6. Gamification architectural design.....	35
2.7. Related research by Lithuanian authors.....	35
2.8. Summary.....	36
3. SPECIFICATION OF GAMIFIED SYSTEMS.....	37
3.1. Methodology for the gamified system analysis .....	37
3.2. Formal model of the gamified system .....	38
3.3. Pattern description scheme .....	39

3.4. Gamification patterns .....	39
3.5. Example of pattern application.....	43
3.6. Abstract formal model.....	44
3.7. Graphical notation of the UAREI model .....	46
3.8. Summary.....	46
4. IMPLEMENTATION OF THE GAMIFIED SYSTEM ANALYSIS TOOL	47
4.1. The Method for Gamified System Development.....	47
4.2. Transformation of the UAREI to UAREI JSON .....	48
4.3. UAREI JSON transformation and simulation .....	53
4.4. GMOD UAREI modelling and simulation tool.....	55
4.5. Summary.....	57
5. CASE STUDIES .....	57
5.1. Trogon PMS .....	57
5.2. eLearning model for programming contest .....	59
5.3. Minority Game .....	61
5.4. OilTrader .....	65
6. EVALUATION OF THE GAMIFIED SYSTEMS .....	67
6.1. Gamified system evaluation .....	67
6.1.1. Visual evaluation of the gamified systems .....	67
6.1.2. Evaluating gamified systems using SUS .....	70
6.2. Modelling gamification of Trogon PMS .....	71
6.2.1. Modelled system description .....	71
6.2.2. Trogon UAREI model .....	72
6.2.3. UML model of Trogon .....	73
6.2.4. Trogon in Machinations.....	74
6.2.5. Model comparison .....	75
6.2.6. Evaluation.....	75
6.3. Gamification model of eLearning system.....	78
6.3.1. UAREI model of eLearning system.....	78
6.3.2. Simulation of a Hybrid Gamification Model.....	79
6.3.3. Experimental evaluation of gamification model of eLearning system ...	80

6.3.4.	Experiment results .....	81
6.4.	Modelling Minority Games in UAREI .....	83
6.4.1.	Extending UAREI for MG support.....	83
6.4.2.	Simulation and results.....	86
6.4.3.	Summary & the reinforcement model.....	89
6.5.	OilTrader Game experiment .....	90
6.5.1.	Experiment setup .....	90
6.5.2.	Purpose of the Game Experiment .....	91
6.5.3.	Experimental Subjects .....	91
6.5.4.	Research Tool .....	91
6.5.5.	Results.....	92
6.5.6.	Extending UAREI MG with motivation.....	97
6.5.7.	Modelling OilTrader .....	98
6.5.8.	Simulation and results.....	99
6.6.	Discussion of the Results & Conclusions .....	102
7.	CONCLUSIONS.....	106
	REFERENCES .....	107
	LIST OF PUBLICATIONS OF DARIUS AŠERISŠKIS ON DISSERTATION THEME .....	123
	APPENDIXES.....	125
	APPENDIX A .....	125



## TERMS AND ABBREVIATIONS

<b>Agent based model</b>	is one of a class of computational models for simulating the actions and interactions of autonomous agents (both individual or collective entities such as organizations or groups) with a view to assessing their effects on the system as a whole.
<b>Artificial intelligence (AI)</b>	is intelligence exhibited by machines. In computer science, an ideal "intelligent" machine is a flexible rational agent that perceives its environment and takes actions that maximize its chance of success at some goal.
<b>API</b>	a set of functions and procedures that allow the creation of applications which access the features or data of an operating system, application, or other service.
<b>ARCS</b>	attention, relevance, confidence, and satisfaction.
<b>CLG</b>	characteristics of a Learning Game.
<b>Cyclomatic Complexity</b>	is a software metric (measurement), used to indicate the complexity of a program. It is a quantitative measure of the number of linearly independent paths through a program's source code.
<b>Game Engine</b>	the basic software of a computer game or video game.
<b>Gamification</b>	the application of typical elements of game playing to other areas of activity, typically as an online marketing technique to encourage engagement with a product or service.
<b>GaML</b>	is an XML based format for storing and archiving data from a wide range of analytical instrumentation.
<b>GMOD</b>	gamification modeling and simulation tool.
<b>GWAP</b>	a human-based computation game or game with a purpose is a human-based computation technique of outsourcing steps within a computational process to humans in an entertaining way.
<b>HEXAD</b>	questionnaire evaluating Human Engagement, eXperience and Activity Design.
<b>HTTP(s)</b>	HTTPS is a protocol for secure communication over a computer network which is widely used on the Internet. HTTPS consists of communication over Hypertext.
<b>Hypergraph</b>	is a generalization of a graph in which an edge can connect any number of vertices. Formally, a hypergraph is a pair where is a set of elements called nodes or vertices, and is a set of non-empty subsets of called hyperedges or edges.
<b>Information Technology (IT)</b>	the study or use of systems (especially computers and telecommunications) for storing, retrieving, and sending information.
<b>Information System (IS)</b>	is any organized system for the collection, organization, storage and communication of information.
<b>JPG</b>	is a commonly used method of lossy compression for digital images, particularly for those images produced by digital photography. The degree of compression can be adjusted, allowing a selectable tradeoff between storage size and image quality.
<b>JavaScript Object Notation (JSON)</b>	is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate.
<b>Machinations</b>	is a theoretical framework and an interactive, dynamic, graphical representation that describes games as dynamic systems and focuses on closed feedback loops within them.
<b>Mechanics-Dynamics-Aesthetics (MDA)</b>	in game design the framework is a tool used to analyze games. It formalizes the consumption of games by breaking them down into three components - Mechanics, Dynamics and Aesthetics.
<b>Meta-language</b>	a form of language or set of terms used for the description or analysis of another language.
<b>Minority Game (MG)</b>	a simple model for the collective behavior of agents in an idealized situation where they have to compete through adaptation for a finite resource.
<b>Pattern</b>	a regular and intelligible form or sequence discernible in the way in which something happens or is done.
<b>Petri Nets</b>	is one of several mathematical modeling languages for the description of distributed systems.
<b>SQL</b>	is a special-purpose domain-specific language used in programming and designed for managing data held in a relational database management system (RDBMS).
<b>Stack</b>	is an abstract data type that serves as a collection of elements, with two principal operations: push, which adds an element to the collection, and pop, which removes the most recently added element that was not yet removed.
<b>Serious game</b>	a serious game or applied game is a game designed for a primary purpose other than pure entertainment. The "serious" adjective is generally prepended to refer to video

<b>System Usability Scale (SUS)</b>	games used by industries like defense, education, scientific exploration, health care, emergency management, city planning, engineering, and politics. in systems engineering, is a simple, ten-item attitude Likert scale giving a global view of subjective assessments of usability.
<b>Systems Modeling Language (SysML)</b>	a general-purpose modeling language for systems engineering applications. It supports the specification, analysis, design, verification and validation of a broad range of systems and systems-of-systems.
<b>UAREI</b>	common formal model composed from Users, Actions, Rules, Entities and Interfaces.
<b>Unified modeling language (UML)</b>	a general-purpose, developmental, modeling language in the field of software engineering, that is intended to provide a standard way to visualize the design of a system.
<b>WCAG</b>	the WCAG technical documents are developed by the Web Content Accessibility Guidelines Working Group (WCAG WG), which is part of the World Wide Web Consortium (W3C) Web Accessibility Initiative (WAI).
<b>XML</b>	a metalanguage which allows users to define their own customized markup languages, especially in order to display documents on the Internet.

## FIGURES

Figure 1. Screenshots of Trogon PMS.....	38
Figure 2. (a) Infinite quantity source and (b) limited quantity source.....	40
Figure 3. (a) Time limit and (b) dynamic limit patterns.....	40
Figure 4. (a) Random result pattern and (b) drain pattern.....	41
Figure 5. (a) Constrain pattern and (b) extension pattern.....	42
Figure 6. (a) Property and chance pattern; and (b) solver pattern.....	42
Figure 7. Trogon PMS rule model.....	44
Figure 8. UAREI activity diagram.....	47
Figure 9. UAREI metamodel represented in UML Class Diagram.....	48
Figure 10. UAREI JSON format UML Class Diagram.....	49
Figure 11. CodeBlock UML Class Diagram.....	50
Figure 12. Query UML Class Diagram.....	50
Figure 13. Entity Scheme UML Class Diagram.....	51
Figure 14. UAREI to UAREI JSON transformation rules.....	52
Figure 15. Application action execution.....	54
Figure 16. Simulation execution UML Activity diagram.....	54
Figure 17. GMOD UAREI modelling and simulation tool.....	55
Figure 18. Game badges and badge levels.....	58
Figure 19. Elements of project forest.....	58
Figure 20. Project forest.....	58
Figure 21. Gamification solution class diagram.....	58
Figure 22. Informik Environment.....	60
Figure 23. The levels of game-based education.....	60
Figure 24. The gamified eLearning hybrid model activity diagram.....	61
Figure 25. Algorithm of the coalition-based Minority Game.....	64
Figure 26. Schematic diagram of the game.....	65
Figure 27. Flow diagram for game steps.....	66
Figure 28. Trogon PMS monthly badge board page.....	68
Figure 29. Trogon PMS monthly leaderboard page.....	68
Figure 30. Trogon PMS employee task page.....	68
Figure 31. Trogon PMS project forest page.....	69
Figure 32. Trogon PMS dashboard page.....	69
Figure 33. Trogon PMS task page.....	69
Figure 36. Visual model of Trogon PMS gamification.....	72
Figure 37. Gamification model of Trogon PMS specified using UML activity diagram.....	73
Figure 38. Gamification model of Trogon PMS specified using Machinations.....	74
Figure 39. The gamified eLearning hybrid model for increasing participant's engagement.....	78
Figure 40. Simulation results.....	79

Figure 41. Probabilities assuming normal distribution.....	80
Figure 42. Box plot of hybrid eLearning model contestant results. ....	81
Figure 43. Probability distribution of gamified and control groups .....	82
Figure 44. Learning results of enjoyment groups.....	82
Figure 45. Probability distribution engagement groups .....	83
Figure 46. Minority Game model in UAREI.....	85
Figure 47. Model of coalition (a) and ternary voting (b) variants of Minority Game in UAREI.....	85
Figure 48. Histogram of wins in simulated classic Minority Game .....	86
Figure 49. Histogram of wins in simulated variable payoff Minority Game .....	87
Figure 50. Histogram of wins in simulated coalition-based Minority Game .....	87
Figure 51. Histogram of wins in ternary voting Minority Game.....	88
Figure 52. OilTrader leaderboard of (a) control group, and (b) experiment group (with streak, win and loss incentives).....	90
Figure 53. Distribution of players between player types .....	93
Figure 54. Median gameplay duration for main group and control group .....	94
Figure 55. Probability of playing longer (permutation test) .....	94
Figure 56. Duration of gameplay (in rounds) .....	95
Figure 57. Analysis of questions from HeXAD questionnaire .....	96
Figure 58. Duration of gameplay for each player type.....	96
Figure 59. Results of permutation test for different player types .....	97
Figure 60. OilTrader UAREI model.....	99
Figure 61. XP and Control group simulation results .....	101
Figure 67. Differences between groups by player types.....	102

## TABLES

Table 1. Machinations modelling framework (J Dormans, 2013).....	30
Table 2. Gamification modelling framework comparison criteria. ....	31
Table 3. Comparison of formal description, timed automata, Petri nets modelling frameworks.....	32
Table 4. Comparison of GaML, UML and Machinations modelling frameworks ...	32
Table 5. Description of the limited quantity source and infinite quantity source patterns. ....	40
Table 6. Description of time limit and dynamic limit patterns.....	40
Table 7. Description of random result pattern and drain pattern. ....	41
Table 8. Description of constrain pattern and extension pattern. ....	42
Table 9. Description of the property and chance pattern and solver pattern. ....	43
Table 10. Graphical notation of the UAREI modelling language .....	46
Table 11. UAREI to UAREI JSON model transformation .....	51
Table 12. Variants of Minority Game .....	63
Table 13. Visual complexity of Trogon PMS models. ....	75
Table 14. Graphical notation of UAREI modelling language .....	76
Table 15. Cognitive dimensions of UAREI and Machinations. ....	77
Table 16. Modelling variants of Minority Game in UAREI .....	86
Table 17. Statistical evaluation of variants of Minority Game.....	88
Table 18. The HeXAD Questionnaire (L. Diamond G. F. Tondello & Tscheligi, 2015).....	92
Table 19. User classification by motivation weights.....	101

# 1. INTRODUCTION

## 1.1. Motivation

Recently, gamification has gained popularity in the development of enterprise information and e-commerce systems (McGonigal, 2011). Gamification is the use of elements of game design (game rules, game techniques, gamified interfaces) in non-game contexts (Deterding, Dixon, Khaled, & Nacke, 2011), such as marketing, employee performance and training, and innovation management.

Game domain is the closest domain to the Gamification domain. Gamified systems can be viewed as simpler versions of games. According to the Webster Dictionary, a game is a physical or mental competition conducted according to rules. In many cases, gamification can be easily applied to games and vice versa.

In a survey by Pew Research Center, 53% of people surveyed said that, by 2020, the use of gamification will be widespread (Anderson & Rainie, 2012). A well-known study of Gartner claimed that by 2015, more than 50% of organizations that manage innovation processes will gamify those processes (Gartner Research, 2012). Over 70% of Forbes Global 2000 companies plan to use at least some elements of gamification for product marketing and customer retention (Van Grove, 2011). While some of the expectations of the spread of gamification may be overhyped, there are several examples of successful gamification, which include Idea Street (Burke & Mesaglio, 2010), a social collaboration platform that uses game mechanics, Badgeville (Sims, n.d.), a platform that enables businesses to apply gamification across their web and mobile experiences; and RedCritic Tracker (RedCritic Corp, 2011), an Agile Project Management service with badges, rewards, leaderboards, and real-time Twitter-style feeds. These gamified systems have some aspects in common: an attractive graphical user interface, strong emphasis on social competition, and an engaging award system.

According to Gartner Inc. (Gartner Research, 2012), the widespread interest that gamification has been attracting recently lies in its potential to strengthen engagement, change user behaviors and support innovation. Game theory-based models are being widely adopted now in different contexts and used as a driver for solving problems in a wide variety of domains, including disaster management (Vásquez, Sepulveda, Alfaro, & Osorio-Valenzuela, 2013), education (Botra, Rerselman, & Ford, 2014; Caponetto, Earp, & Ott, 2014), e-learning (Gené, Núñez, & Blanco, 2014), workplace improvement (Sammur, Seychell, & Attard, 2014), marketing (Freudmann & Bakamitsos, 2014), healthcare management (Wilson & McDonagh, 2014; Wortley, 2014), IT service management (da Conceicao, da Silva, de Oliveira Filho, & Silva Filho, 2014), social policy (Hall, Kimbrough, Haas, Weinhardt, & Caton, 2012), sports and fitness (Stålnacke Larsson, 2013), tourism business (Wells et al., 2014), customer engagement (Harwood & Garry, 2015), social missions (Hamari & Koivisto, 2013), fostering creativity (Barata, Gama, Fonseca, & Gonçalves, 2013), employee engagement and training (Narayanan, 2014), etc.

The modelling of gamification is important for the design of systems based on the principles of serious games, in order to quantify and validate the impact of gamification and to better understand why and how gamification works. Existing evaluations of gamification usually focus on the application of user questionnaires and other methods of qualitative evaluation. There is still a lack of modelling methods and

tools to aid the design and development of gamification in serious systems (Herzig, Jugel, Momm, Ameling, & Schill, 2013; Mora, Riera, Gonzalez, & Arnedo-Moreno, 2015).

This dissertation aims to introduce modeling and simulation methods which would allow us to build a bridge between the formal modeling of gamification and quantitative simulation of games, analysis and evaluation of game rules and processes.

## **1.2. Object and scope of the research**

The object of the research is methods and tools for simulation, analysis, and evaluation of gamified software systems. These methods and tools are necessary if we want to employ more powerful game patterns, elements and mechanics into our systems at the same time as understanding how the system will impact user behavior.

The scope of the research involves:

- Methods for game modeling, analysis, prototyping.
- Formal mathematical modelling of gamified systems.
- Multi-user system user behavior modeling and model behavior analysis.
- Gamified system analysis and evaluation.

## **1.3. Problem statement and research questions**

The problem of this work focuses on the lack of methods and tools for quantitative analysis and understanding of gamification patterns, elements and mechanics. The same problem is observed in the related game domain. Currently, there is no single integrated process which would lead game designers from the idea of gamification to the final implementation of a gamified system.

This dissertation gives answers the following research questions:

- What is state-of-the-art in the domain of gamification modelling?
- How can gamified systems be evaluated?
- How can gamified systems and elements of gamification be modeled abstractly?
- How can user behavior in gamified systems be predicted?

## **1.4. Aim and objectives**

The aim of the research is providing the gamification domain with a tool and methods for modeling, analyzing, simulating and generating gamified systems, isolating patterns, and understanding gamification pattern impact on user behavior.

For the aim of the thesis to be achieved, the following objectives have been set out:

1. To conduct static and dynamic analyses of gamified systems to identify methods for evaluation of the system gamification.
2. To consider gamified systems for patterns and identify commonalities in gamified systems, and create a gamification modelling method.
3. To examine known solutions for evaluation of usability and efficiency of gamification solutions, and create a method for analyzing and computationally modeling the impact of gamification on the behavior of users with respect to gamified systems.

### **1.5. Defended propositions**

1. The proposed gamification evaluation methods can be used for quantitative and qualitative evaluations of gamified systems.
2. The proposed visual gamification modelling method allows for creating gamified system models, extracting gamification patterns, simulating user behavior, analyzing simulated user behavior, comparing gamified systems and generating gamified applications.
3. The UAREI model simulation in the GMOD tool reproduces similar behavior of other tools and enables to predict how an implemented prototype gamified system will act in a real-world environment. The efficiency of the gamification solutions can be modelled by analyzing the behavior of users using a game player type evaluation based on the HEXAD questionnaire.

### **1.6. Major contributions and novelty of the research**

Major contributions of this work:

- Formal abstract gamification modeling method as a common method for analyzing gamified systems has been created. This method allows us to model, examine and evaluate gamified systems has been developed, which has similar or better capabilities versus current industry methods.
- A new method for evaluating gamified system attraction was created by comparing player winning distribution to normal distribution. Method tested by analysis of minority game variation.
- A new method for evaluating gamification reinforcement models by psychological player types was proposed. Method tested by analysis of OilTrader game experiment.
- Proposed methods for gamified system user interface evaluations using Web Content Accessibility Guidelines and System Usability Scale, tested with Trogon project management system.
- The proposed methods and gamified system modelling method allows game designers and scientists to develop gamification models, simulate gamified systems, improve models to achieve desired outcomes and generate systems from models has been created. This increases the development speed of gamified systems.

The novelty of the method:

- UAREI is a new formal modeling method dedicated for gamified system modeling with visualization and simulation capabilities.
- The created method allows different ways to simulate gamified applications. For simulation UAREI utilizes custom selection functions, agent based modelling and Minority Game engine.
- Simulating gamified systems allows to evaluate gamified system performance in new ways like: increase of motivation, results or game interestingness.
- Created gamified system evaluation methods based on usability, visual attractiveness (contrast ratios), and player motivation by player types.



## **1.7. Practical significance**

- The proposed formal abstract model enables scientists to formally model gamified systems. Using the same formal notation extraction of common gamification patterns and pattern composition into new systems (UAREI is mathematically expressed through sets which allows easy mathematical manipulation using algebra of sets) becomes easier.
- Simulation of the gamified system model gives valuable insights on how the gamified system affects user behavior to game designer and scientists. Simulation provides faster feedback and predicts what kind of behavior might be expected from users interacting with the gamified systems.
- Software generation from UAREI creates the shortest path from gamified system models to a prototype system ensuring fast iteration over the gamified system solution which allows for better results.
- The analysis of the gamified systems using the UAREI modeling method helps to better understand how gamified system patterns effect end-user behavior, how patterns interact with each other and how to achieve best gamified systems performance, before deploying ready-to-use gamified software systems.

## **1.8. Scientific approval**

The results of the research have been presented in two international conferences, and two articles have been published in journals indexed in Web of Science Journal List. One article is still in review in the journal included in Web of Science Journal List. One paper is published in a peer-reviewed journal. Three topic-related papers have been presented in Lithuanian conferences. Two papers have been presented in international conferences in Spain and France. The full list of publications can be found in chapter titled “LIST OF PUBLICATIONS OF DARIUS AŠERISKIS ON DISSERTATION THEME”. A list of conferences:

- The 22<sup>th</sup> International Master and PHD Students Conference “Information Society and University Studies”, 2017 April 28, Kaunas;
- The Seventh International Conference on Advances in Computer-Human Interactions ACHI 2014 March 23 - 27, Barcelona, Spain;
- The Sixth International Conference on Intelligent Human Computer Interaction IHCI 2014 December 8-10, Envy, France;
- The 18<sup>th</sup> International Master and PHD Students Conference “Information Society and University Studies”, 2013 April 25, Kaunas;
- The 17<sup>th</sup> Master and PHD Students Conference “Information Society and University Studies”, 2012, Kaunas.

## **1.9. Structure of the dissertation**

In Chapter 2, related works from various areas of gamification like motivation and psychological foundation, modeling, software architecture and agent-based simulation have been analyzed. When analysis of gamification is conducted, patterns from gamified system models are extracted. Also, the UAREI abstract formal model is identified, which is the basis for further research. GMOD is presented as an UAREI

modeling tool which is used for system model analysis and simulation. In Chapter 5, case studies used for research are described.

Chapter 6 covers methods for gamified system quantitative and qualitative evaluation. Furthermore, the UAREI modeling method is evaluated against UML and Machinations. Furthermore, hybrid eLearning UAREI model simulation is evaluated against real system user behavior. UAREI is extended to support a market-based simulation adapted from Minority Game engine and evaluation of gamified system by player types, which is presented thusly. In Chapter 7, conclusions of the work are presented. Finally, references and a list of publications by the thesis author is given.

## **2. THE ANALYSIS OF GAMIFICATION**

### **2.1. Background of gamification**

According to Eric Zimmerman, a game is an activity with some rules engaged in for an outcome (Salen & Zimmerman, 2004). Chris Crawford defines games in this way (Crawford, 2003):

1. Creative expression is art if made for its own beauty and entertainment if made for money;
2. A piece of entertainment is a plaything if it is interactive.
3. If no goals are associated with a plaything, it is a toy. If it has goals, a plaything is a challenge.
4. If a challenge has no active agent against whom you compete, it is a puzzle; if there is one, it is a conflict.
5. Finally, if the player can only outperform the opponent, but not attack them to interfere with their performance, the conflict is a competition. However, if attacks are allowed, then the conflict qualifies as a game.

Games and game-like experiences can be split by design attempts. A game combines game thinking, game elements, virtual world, game play and non-purposefulness (A. C. Marczewski, 2015).

Gamification (Deterding, Dixon, et al., 2011) has been employed to enable attitude change and increase user motivation. It refers to adding ‘gamefulness’ to existing systems in non-game contexts, usually aiming to increase the value of a service or business product beyond its face value, as well as to boost user engagement, loyalty, and satisfaction or otherwise affect user behavior (Huotari & Hamari, 2012). Concepts related to gamification are “gameful design” or “gameful work”.

Gamification is applied to enhance the value of a service or business product beyond its face value, as well as to boost user engagement, loyalty, and satisfaction. Gamification is usually implemented using game elements, such as badges and scoreboards, combined with meaningful game rules (or game mechanics) that encourage competition between game players trying to reach some objectives or quantifiable outcome (Deterding, 2012). Gamification that encourages competition between game players may help to achieve positive outcomes such as higher sales of a product, drive marketing or increase job performance.

However, gamification still poses great challenges to software designers: 1) how to design meaningful and engaging game rules as well as integrate them with business rules, 2) how to create an attractive game interface, which integrates smoothly with a

user interface of a serious system, 3) how to evaluate success of gamification both in terms of its usability (aesthetic aspect) and customer retainment (pragmatic aspect).

Several cases of gamification application are described in literature in the context of enterprise information systems (IS) such as a generic platform for enterprise gamification (Herzig, Ameling, & Schill, 2012), implemented using service oriented and event-driven principles and best practices; authentication games (Kroeze & Olivier, 2012) for improving user behavior regarding security; and the demand dispatch system (Gnauk, Dannecker, & Hahmann, 2012) with a special scoring system, leaderboards and social competition aspects embedded into user interface.

Gamification is widely adapted in the field of eLearning technology. Various approaches are often used to improve the virtual learning environment. One of the main goals is to improve motivation of teachers and students via innovative learning tools and gamification techniques. The practice shows that the technological solutions for gamification implementation bring more innovations to the educational process. Game-based activities improve users' logical deductive thinking (Yuizono, Xing, & Furukawa, 2014) reaction even by applying existing technologies and creating new knowledge management. Gamification items can serve as efficient catalysts determining the ideas of fluency, flexibility and originality, while the use of game mechanics can contribute to promoting involvement (Witt, Scheiner, & Robra-Bissantz, 2011). Six main elements of gaming (Barata, Gama, Jorge, & Gonçalves, 2013; O'Donovan, Gain, & Marais, 2013) have been identified as being particularly effective in education (Meece, Anderman, & Anderman, 2006; Smith-Robbins, 2011): Choice (Freedom to Fail), Rapid Feedback, Collaborative Processes, Evidence of Progression and Competition, as well as Evidence of Storytelling in some of the studies (Kapp, 2016). The greatest noticeable difference between a typical game model and gamification model (especially in eLearning context) lies in the sustainability (often perceived as knowledge) after "Engagement" and "Reward" activities (Salen & Zimmerman, 2004).

Gamification is defined as a process which shapes the world (achieves goals/objectives) by influencing the actions, behaviors, characteristics and state of entities within the world through the use of games strategies and enabling technologies (Wortley, 2014). The concept is relatively new, but it has gained considerable interest in the software development and user interface design community over the last few years. The roots of gamification are in game design, with some elements from psychology, so there is still little academic research on how to design and develop software systems with and for gamification.

## **2.2. Psychological and social aspects of gamification**

Playing can be a powerful motivating factor, facilitating learning and supporting the physical and intellectual development of a person (Deci & Ryan, 2000). In 2012, there were more than one billion computer game players (Kuss, 2013) leading to a boom in the online gaming market. There have been many efforts to exploit games for more serious use such as gamification. Gamification is the use of game thinking and game mechanics in non-game contexts (Werbach & Hunter, 2012) in order to engage users and solve serious problems (Zichermann & Cunningham, 2011) such as to promote or assess sustainability of complex intelligent physical environments (Silva, Analide, Rosa, Felgueiras, & Pimenta, 2013).

Using gamified systems and applications, the engagement, interaction, collaboration, awareness, participation, productivity and learning motivation of users can be increased in various domains such as team organization (J. T. Kim & Lee, 2015), project management (Ašeriškis & Damaševičius, 2014a), e-learning (Luo, Yang, & Meinel, n.d.), healthy lifestyles (Berger & Schrader, 2016), tourism applications (Negrușă, Toader, Sofică, Tutunea, & Rus, 2015), etc. Such mechanisms can be applied to reinforce player motivation to play as they contribute to initiation, development, and maintenance of gaming behavior (King, Delfabbro, & Griffiths, 2010). Games can evoke a lot of different affective states, and can to some extent be utilized to keep the player involved in the game (Chanel, Rebetez, Bétrancourt, & Pun, 2008). The aim of the gamification designer should be to increase and retain a number of game players as well as to prolong game lifetime by maximizing user involvement and satisfaction, while minimizing negative emotional episodes caused by frustration, for example, which can lead the player to stop playing the game.

The primary motivation for gamification is a psychology-based one, namely to enhance user or customer motivation to do a job or to increase and retain addiction to a service or product using a game as a tool.

Gamification can be explained by Fogg Behavior Model (FBM) (Fogg, 2009), which claims that both, motivation to perform and ability to perform, must converge at the same moment for behavior to occur. Motivation must be supported by positive feedback from game mechanics that continuously triggers a user to perform specific actions and keeps him interested in the game.

Psychological foundation of gamification has been elaborated further by Wu (Wu, 2011), who analyzes why and how gamification is able to drive actions, and by Gnauk et al. (Gnauk et al., 2012), who studied extrinsic and intrinsic motivation and analyzed its relationship with external incentives and rewards.

Another motivation for gamification is social competition. Here, gamification is driven by the need to interact with other players and compare one's results. Thus, gamification requires the introduction of real-time multi-user games with complex rules of a game that have some similarity to social networking platforms.

The underlying concept of gamification is motivation. Gamification is driven primarily by external motivation, i.e., the users strive to compete against other playing users and to get recognized by the game community (Stålnacke Larsson, 2013). As motivation tends to decay over time, it, however, must be supported by the increasing complexity and evolving dynamics of game mechanics (Bauckhage et al., 2012). Meaningful gamification (otherwise known as "serious game") is the use of game design elements to help users find meaning in a non-game context. Rather than just using game mechanics to give points or badges to users as external rewards, meaningful gamification focuses on the playing process (aka game mechanics) itself to engage players to perform meaningful tasks in the real world.

Developing motivation enhancement and reinforcement models and methods is important for many areas where active and sustainable participation of agents is key for the success of the entire process, e.g., in digital game-based learning (S. Kim, 2015; Rico, Agudo, & Sánchez, 2015), to foster entrepreneurship education (Fonseca et al., 2012), or to facilitate management of software development processes (Herranz, Palacios, de Amescua Seco, & Yilmaz, 2014).

Many gamified systems encourage user participation using virtual forms of incentive like points, badges, leaderboards (Nah, Zeng, Telaprolu, Ayyappa, &

Eschenbrenner, 2014), progress bars, performance graphs or avatars (Sailer, Hense, Mandl, & Klevers, 2013). These incentives translate a player's time and effort investments into a form that is quantifiable, comparable and communicable to his peer (Gou, 2006). As such, they indicate player status and in-game progress, as well as motivate them to continue engaging in gameplay. For example, several of the most popular user-contribution based sites such as StackOverflow, TripAdvisor and Quora, provide some form of recognition to their users for their overall contributions to the site such as "Highest scoring answer that outscored an accepted answer with score of more than 10 by more than 2x" (Populist Badge) (<http://stackoverflow.com/help/badges>). Such badges are meaningful incentives for their users contributing to the success of an entire community as well.

Incentives usually reflect various site-level accomplishments based on players' performance. Badges primarily have a social-psychological meaning, and usually have only a symbolic value within a virtual community (Immorlica, Stoddard, & Syrgkanis, 2015). Different players may value winning a badge differently. The value of incentives depends upon the number of incentives already given to the player and other players, and tends to decay over time (Easley & Ghosh, 2013). Therefore, badges have a diminishing utility, where the value of each badge decreases over time as the number of players who have earned that badge, increases.

Playing games is not always enjoyable. If the challenges presented in a game repeatedly exceed player's skills, they can cause frustration (Breuer, Scharkow, & Quandt, 2015). In zero-sum games, the success of one competitor leads to the failure of another, which is likely to cause negative emotional reactions. While competition in itself can also be fun and rewarding, the possibility of losing to a competitor introduces the risk of adverse emotional experiences. An unfavorable outcome (i.e., losing) can increase negative emotions such as aggression. Players that get frustrated have a higher chance of quitting the game (Canossa, Drachen, & Sørensen, 2011). Therefore, the game (or gamification) designer should design (or adopt) a player reinforcement model that can help to alleviate player frustration by providing awards and recognizing player effort aiming to sustain long-term users' motivation.

On the other hand, if there is a player that is significantly better in playing the game and is constantly (and predictably) winning, it introduces the elements of boredom in the game both for the constant winner as well as to other players and game spectators. Boredom encourages the pursuit of alternative goals outside of the game (Bench & Lench, 2013), thus reducing the number of players staying in the game.

As the emotional impact of the game is mainly based on success and failure, the properly constructed reinforcement model must assure and increase positive emotions of players by incentives, which provide immediate recognition of players' success, or keep encouraging players when they fail, but still show good results. Incentives can be awarded for meeting absolute targets or relative targets. However, if the reinforcement model is connected only to the absolute achievements, the model may work against itself as the lesser performing players are likely to be disincentivized and may give up and leave the game ('discouragement effect') (Minor, 2013). Special incentives should be made for successful comebacks after failures to reinforce such behavior rather than game quitting.

To avoid that, the motivation reinforcement model should be carefully designed to fit differences in player skills and promote continuation of the game. If the motivation reinforcement model is properly balanced, it can drive the players to a

highly motivating emotional flow-state (Csikszentmihalyi & Bose, n.d.). Deeper knowledge in this area can help researchers to understand the behavior of a gamer better, while game designers can promote serious games better.

The following section provides an overview of different psychological theories on gamification and models of reinforcement as well as factors affecting the player during the game.

Until now the concept of reinforcement models has been mainly studied in the fields of artificial intelligence, machine learning and control theory (for a review, see (Kaelbling, Littman, & Moore, 1996)). A wide variety of physical notions are employed in the models of socio-technical systems involving elements of human behavior as cooperation, willingness, and morale (Meyers, 2009). In the domain of game design, the game designer creates various player emotions in a game to generate player enjoyment. Various affective states of players such as engagement, anxiety, frustration, and distress have been studied before (Kokil, 2013; Sharek & Wiebe, 2014).

Several researchers have been motivated to identify the reasons of people's engagement in computer games. Psychological approaches include Malone's principles of intrinsic qualitative factors (challenge, curiosity and fantasy) for engaging game play (Malone, 1981).

Modern psychological theories of emotion such as Flow Theory (Csikszentmihalyi & Bose, n.d.) are based on the concept of flow. It is argued that an individual becomes strongly involved in a task when his skills match the challenge of a task. Too difficult a challenge raises anxiety but prevents boredom. The state of the player can change because of the player's progression through the game levels leading to increased complexity of the game and potentially giving rise to anxiety, or because of the increased competence of the player while the game stays at the same level of difficulty, which potentially can lead to boredom. In both cases, the game designer should develop the scenario of the game to maintain a player's state of pleasure and involvement, while keep gradually increasing difficulty of the game in relation to the competence and emotions of the player.

Under the Festinger's (Festinger, 1957) theory of dissonance, the state of the consumer depends upon the perceived performance of a product as compared to his/her expectations regarding the product. Discrepancy between expectations and perceived performance is likely to cause the dissonance. In the case of a game, the player raises expectations based on his/her own performance results and projects these expectations to the future. The assimilation-contrast theory claims that satisfaction is a function of the magnitude of the discrepancy between expected and perceived performance (Hovland, Harvey, & Sherif, 1957). For example, if the player had expected to perform poorly, he/she would not be as upset about losing as a player who originally had expected to perform well.

The self-determination theory (Ryan, Rigby, & Przybylski, 2006) addresses both intrinsic and extrinsic motives for action. The player has psychological needs for autonomy, competence and relatedness, which can be addressed by introducing changes to the game scenario (Przybylski, Rigby, & Ryan, 2010). Competence can be fostered by feedback and rewards for tasks. Self-efficacy can be positively stimulated by recognizing player accomplishments (Reeves & Read, 2013). Players with high self-efficacy can be kept at a task by rewarding their competence as well as ensuring their autonomy to maintain or enhance intrinsic motivation (Berger & Schrader, 2016).

The Oliver's Expectancy disconfirmation theory (Oliver, 1980) claims that user satisfaction is caused by the difference between the expected and perceived product performance. A product or a process (in our case, the results of a game) satisfying higher initial expectations are predicted to produce greater satisfaction than the one that meets low expectations. Expectations originate from user beliefs about the level of performance that he/she will achieve. Satisfaction arises when a product or service is better than expected. When performance is worse than the expected, it causes dissatisfaction.

The Yield Shift Theory of Satisfaction (Briggs, Reinig, & de Vreede, 2014) defines satisfaction as an emotional (affective) response with respect to some object that has reference to some state or outcome desired by an individual. Satisfaction can manifest itself through many phenomena (Briggs (Briggs et al., 2014) provides a list of 10 phenomena for the IT domain), some of which are also relevant for the gaming domain. These are: goal attainment effect, when users feel satisfied on attainment of a desired state or outcome; confirmation effect, when users feel satisfied when outcomes match expectations or desires, and feel dissatisfied when outcomes are less than expectations or desires; nostalgia effect, when users feel satisfied or dissatisfied when thinking about past achievements or failures; attenuation effect, when users' satisfaction responses diminish over time.

Based on Flow Theory (Csikszentmihalyi & Bose, n.d.), Chanel et al. (Chanel et al., 2008) defines three different emotional states: boredom (negative-calm), engagement (positive-excited) and anxiety (negative-excited). Flow includes many elements such as engagement, immersion, enjoyment, interestingness, impressiveness and surprise. Enjoyment appears at the boundary between boredom and anxiety, when the challenges are just balanced with the person's capacity to act in a game (Csikszentmihalyi & Bose, n.d.). Engagement and immersion have been defined mainly in terms of how cognitive and psychological states such as participation, presence, and arousal contribute to engagement (Martey et al., 2014). Immersion makes the player to focus his/her attention into the game world resulting in a lack of awareness of time and the real world (Nylund & Landfors, 2015). The immersion can be maintained by keeping proper complexity and interestingness of gameplay and its results.

Boredom arises due to unchanging environment, or monotonous, predictable or repetitive changes. Boredom is also related to the sense of the lack of novelty and interestingness. Hill and Perkins (Hill & Perkins, 1985) states that "boredom occurs when stimuli is construed as subjectively monotonous". Boredom can be defined as low entropy of the game results. The experience of boredom is negative and aversive, creating desire to change from the current state and avoid future states of boredom. Boredom can be recognized by the lower cognitive load of a player during the game (Sharek & Wiebe, 2014) as well as by the change of the physiological parameters of the player registered using facial electromyography (EMG) or electrodermal activity (EDA) (Kivikangas et al., 2011).

Boredom is also related to fatigue. Fatigue has been defined as "decline in ability" or "decline in performance" in the presence of cyclical load (in materials science) or intense load (sports medicine). As with boredom, fatigue also leads to negative emotional responses such as a decrease of interest and reduced performance. The fatigue models proposed in the domain of sports medicine such as the Banister model (Banister, Calvert, Savage, & Bach, 1975) and its various elaborations (Busso,

Benoit, Bonnefoy, Feasson, & Lacour, 2002; Calvert, Banister, Savage, & Bach, 1976; Morton, Fitz-Clarke, & Banister, 1990) are based on the exponential decay function that is widely used to describe natural phenomena such as heat transfer between the object and its medium, rate of enzyme-catalysed chemical reactions, fluid dynamics, metabolization of drugs in patients. These examples provide a logical foundation for application of exponential decay functions to boredom modeling. The models of boredom have also been developed in the domain of the intelligent controllers design (Yamamoto & Ishikawa, 2010) and human learning process (Zgonnikov & Lubashevsky, 2012).

From a psychological point of view, frustration is the feeling that occurs when a person is stopped in his or her progress while pursuing a goal (Nylund & Landfors, 2015). In multi-agent systems, frustration has been defined as the failure to achieve an optimal state of the system, in which all agents would win. The need to compete between players inevitably leads to wins and losses, which create local minima in the energy landscape of the system (Burgos, Ceva, & Perazzo, 2004). The frustration of the system can be reduced by minimizing the number of losers in the game as much as possible. While frustration in some cases can serve for motivating players to overcome the presented challenge, frustration still should be avoided. While engagement tends to decay over time, boredom and frustration tend to increase with time if the conditions remain the same. Furthermore, frustration can worsen if the game or the reward system is considered by the player as not fair or transparent.

Flow also can be defined as the state of user satisfaction. Models of user satisfaction models have been proposed mainly in the domain of economics and marketing, but also in the IT domain. Satisfaction can be seen both as an outcome of some activity or experience and a process. Parker and Mathews (Parker & Mathews, 2001) define satisfaction as a process of evaluation between received and expected outcomes.

Feedback is the central functional subsystem of human communication (Allwood, Nivre, & Ahlsén, 1992). It consists of methods that allow providing, without interrupting the dialog, information about quality of communication such as ability and willingness to have contact, the ability to understand communicated information as well as the emotions and attitudes triggered by the information in the recipient. According to Kotzé (Kotzé, Renaud, & Van Dyk, 2002), feedback has three main elements: 1) response, which serves to confirm that the recipient has received information, 2) modification of behavior, which ensures the user that his input is relevant and has the power to change, and 3) intelligence (or “wisdom of crowds” (Surowiecki, 2005)) that the opinion or understanding of the community can lead to improved quality of work or a product.

In gamification, feedback can be used to engage individuals in performing serious activities, and implemented by designing a proper reinforcement system that provides immediate feedback on player performance (Richter, Raban, & Rafaeli, 2015). However, feedback does not have a direct positive effect on performance (Kluger & DeNisi, 1996). The implementation of feedback (e.g., the level of detail, the timing of feedback) directly influences the results of feedback (Weiser, Bucher, Cellina, & De Luca, 2015). Positive feedback (agreement) reinforces the change in the same direction; while negative feedback (disagreement) causes a change in the opposite direction, and homeostatic feedback maintains equilibrium (Spink & Saracevic, 1998).



The role of feedback is especially important in social networks and other collaboration-based practices that underline the importance of effective communication in virtual communities. The strength of relationships that bind a member to a community can be influenced by the impact a member can make as well as feedback that a member can receive from a community. The success of a virtual community relies on the voluntary contribution of valuable intellectual property of individuals to a community without explicit compensation (Roberts, Hann, & Slaughter, 2006). Even if an individual does not receive any explicit reward for his/her contribution, he/she often wants his/her contribution to make an impact or at least be seen. Capturing and understanding feedback received from users is also critical for understanding user motivation and engagement.

According to (Heller, Lichtschlag, Wittenhagen, Karrer, & Borchers, 2011) and (Muñoz, Mendoza, Álavarez, Martin, & Ochoa, 2007), in order to be effective, feedback must be 1) persuasive (i.e. influencing future state of the community and behavior of the community members), 2) contextual (i.e. include context information by default), and 3) informative (i.e. convey useful information), 4) contributive (i.e. contribute towards benefit of the community as a whole), 5) continual (i.e. to support conversation as narrative of the community), 6) expressive (i.e. demonstrate polarity using affective means such as emotions), and 7) effortless (easy to use).

The concept of interestingness has been mainly analyzed before in the data mining domain. In association rule mining, interestingness is used as an objective criterion to select certain patterns or rules over others (Geng & Hamilton, 2006). In knowledge discovery algorithms, interestingness is used as a measure of unexpectedness (Hidi & Baird, 1986; Padmanabhan & Tuzhilin, 1999). In machine learning, interestingness is one of the criteria used to rank media content such as photographs on the content sharing websites. Impressiveness (which is synonymous with interestingness) has been defined as a rarity (Lehman & Stanley, 2012). Impressiveness can be estimated as the difficulty to re-create an observed property (state or results) of a game, i.e. if the future state of the game is uncertain and is not replicating the past states, the game is considered as interesting.

## **2.3. Modeling of Gamification**

### **2.3.1. Overview of gamification analysis methods**

Several efforts exist at classifying and codifying recurring gamification practices and common techniques such as 1) Mechanics-Dynamics-Aesthetics (MDA) framework (Hunicke, LeBlanc, & Zubek, 2004), a conceptual model of game elements; 2) game design atoms (Brathwaite & Schreiber, 2008); 3) game design patterns (Adams & Dormans, 2012), commonly reoccurring parts of game design; 4) game mechanics (Neeli, 2012); and 5) game interface design patterns, common successful game design components and solutions such as badges, levels, or leader boards (Deterding, Dixon, et al., 2011). Each game element can be described using the Frang (Kristoffer & Robin, 2012) scheme: summary (visualization of an element with a proper description), purpose, ability, motivation, Radoff's type(s) of fun (such as competition or exploration) (Hunicke et al., 2004), dependencies with other game elements, and importance.

According to Salen and Zimmerman (Salen & Zimmerman, 2004), a game must have 1) rules, 2) players, 3) struggle (artificial conflict), and 4) goals (quantifiable outcomes). While the general goal of each game is a win, where can be multiple ways or elements of a game to reflect the player's path towards victory such as badges, which represent player achievements; leader-boards, which allow comparing one's achievements among multiple players; and levels, which reflect the growth of player's skill.

Most games have certain common aspects. Defining and formalizing structural solutions to commonly recurring problems is the main idea behind patterns. A pattern usually consists of a name, definition, general description, description on how the pattern can be used, description of consequences of using the pattern, and relations to other patterns (Gamma, 1995). Solutions to these aspects may vary system to system but have many commonalities. The concept of design patterns (Gamma, 1995), which so far have proven successful in object-oriented design and software engineering, seeks to communicate these solutions in an easy to understand manner. Similar concepts exist in the games domain too, e.g., gameplay design patterns (Bjork & Holopainen, 2004), game patterns (Kelle, Klemke, & Specht, 2011), game design patterns (Kiili, 2010), viral and collaborative patterns (Wendeus, 2013), etc. For standardization of serious game design patterns, a serious game design pattern canvas which combines business model canvas has been proposed by Žavcer et al. (Žavcer, Mayr, & Petta, 2014).

Kreimeier (Kreimeier, 2002) suggests using game design patterns as a way to formalize and codify knowledge about game design. Bjork and Holopainen (Bjork & Holopainen, 2004) propose gameplay design patterns as semiformal interdependent descriptions of commonly reoccurring parts of the design of a game that concerns gameplay. Game-patterns encapsulate common design problems and solutions for those and game designers typically combine several patterns for good gameplay (Kelle et al., 2011).

The design of serious games is a complex process. Two opposing principles must be united: achievement of serious objectives and meaningful gameplay. This can be achieved by detailed technical modeling and implementation (Kelle et al., 2011). However, the only way to really understand gamification is to identify its basic elements and model structural relationships between them. Adams and Rollings discern four basic economic functions for games: sources, drains, converters and traders (Rollings & Adams, 2006). Sources create resources, drains destroy resources. Converters replace one type of resource for another, whereas traders allow the exchange of resources between players or game elements. These economic functions set up a network of economic transactions that determine the flow of a game. A game also can be modelled as a flow of resources, and abstract aspects of games, such as player skill level and strategic position, can be modelled through the use of resources; as well as a state machine: an initial state or condition and actions of the player can bring about new states until the end state is reached (Grünvogel, 2005).

Gamification can be specified and modelled in many ways, e.g., with formal description (Bista, Nepal, Colineau, & Paris, 2012), using textual descriptions and modelling methods, e.g., with UML diagrams (Joris Dormans, 2008; Taylor, Gresty, & Baskett, 2006), Petri Nets (M. Araújo & Roque, 2009), or other standard or custom tools (Grünvogel, 2005; R Koster, 2005). MDA (Hunicke et al., 2004) is a formal approach, which attempts to bridge the gap between game design and development, game criticism, and technical game research. Mechanics describes the particular

components of the game, at the level of data representation and algorithms. Dynamics describes the run-time behavior of the mechanics acting on player inputs and each other's outputs over time. Aesthetics describes the desirable emotional responses evoked in the player, when he interacts with the game system.

Gamification models define game design elements which should be used in non-game contexts, leaving a lot of wiggle space about how these game design elements should behave and look like (Groh, 2012). The majority of tried and tested design principles have already been established by multiple researchers, e.g. (Iosup & Epema, 2014; Lee & Hammer, 2011; Morrison & DiSalvo, 2014; Nah et al., 2014; Pirker, Riffnaller-Schiefer, & Gütl, 2014; Salen & Zimmerman, 2004; Simões, Redondo, & Vilas, 2013) specifying goals, challenges, progress, feedback and other components.

Gamification systems can be classified into these categories as suggested by (Werbach & Hunter, 2012):

- Internal Gamification, aiming to improve productivity and reduce resource costs internally within the organization.
- External Gamification, aiming to involve external people (students) to produce increased engagement, identification and results.
- Behavior-changing gamification, aims to encourage people to make better choices thus increasing motivation.

Most modern gamification systems for education often combine all three categories, especially focusing on the behavior-changing capabilities (Charles, Bustard, & Black, 2011; Deterding, Dixon, et al., 2011). On these assumptions, dynamic models (J. T. Kim & Lee, 2013) capable of simulating some main factors on effective learning, can be established. Practical Implementations of educational gamification (Gené et al., 2014; Kapp, 2016) can be classified into 4 main model categories, none of which is optimal only by itself, often factoring and explaining resulting learning behaviors:

- Non-systemic model (Pedreira, García, Brisaboa, & Piattini, 2015). The model designers add mostly elements of the game design, but not additional system features allowing dynamic interaction of users *with all system* components.
- reward-oriented model (Dichev, Dicheva, Angelova, & Agre, 2014). This model mostly focuses on motivating, often though some perceivable rewards, instead of the intrinsic motivations characteristic to games.
- Non-user-centric model (Zuckerman & Gal-Oz, 2014). This type emphasizes the ideas of the running organization or the owner of educational resources, often neglecting or even being detrimental to users' goals.
- pattern-bound model (Zuckerman & Gal-Oz, 2014). This model is a feedback feature-based implementation, often gathering results of some surveys and design (points, badges, leader boards, etc.), rather than focusing structural qualities of games that inspire gamified experiences.

A gamification process itself can be defined upon 4 theories (J. T. Kim & Lee, 2013):

- A theory of Game Design Features (Raph Koster, 2013) (GDF)
- A theory of Key Characteristics of a Learning Game (Van Eck, 2006) (CLG)
- A theory of attention, relevance, confidence, and satisfaction (J. T. Kim & Lee, 2013) (ARCS)

- A theory of game mechanics, dynamics and aesthetics (Salen & Zimmerman, 2004) (MDA).

It is difficult to compare GDF to others (in the sense of e-learning) as it was mostly created for building entertaining computer games. Nonetheless, it does closely correlate with CLG, thus introducing fun, motivation and attractive challenges to e-learning based on better gameplay type experience (Carron, Marty, & Heraud, 2008; ChanLin, 2009). ARCS applies to CLG as well, directly benefiting from the included characteristics, higher curiosity, attention span, confidence and finally satisfaction in the learning process by achieving goals (Park, 2012). Classic gaming MDA, however, applies only for a few aspects of CLG, such as challenge in dynamics and level of difficulty, and curiosity (Kapp, 2016).

In game research, there is a strong separation between design methodologies and usability evaluation tools, which are rarely employed in the early stages of the design process. Although in the majority of cases, the game developers use heuristically designed tools to assist the design, there are still few existing methods employed to connect design practices with gamification and game design (Rao & Pandas, 2014). Currently game and gamification development is strongly related to the qualifications and skills of game designers. This limitation drives the need for better and faster game building. Recently, several new tools have been developed or adapted to help game designers to model, build and analyze games.

Unified Modelling Language (UML) is a *de-facto* standard modelling language used in multiple domains. Tenzer (Tenzer, 2004) argues that UML modelling tools could also be used to build games and proposes a framework for building games using UML. The advantage of UML is that it is well known in the software engineering community. SysML is a general-purpose modeling language for system engineering applications that supports specification, analysis, design and verification of various systems. SysML has been used for building a training game (Hetherinton, 2014).

The most notable examples of domain-specific game description languages are GaML (Herzig et al., 2013; Matallaoui, Herzig, & Zarnekow, 2015) and ATTAC-L (Janssens, Samyny, Van de Walle, & Van Hoecke, 2014). GaML is a formalized language for specifying and automatically generating gamification solutions. This allows to free the IT expert from the development of gamification solutions. ATTAC-L is a domain-specific language which allows the user to specify the game scenario in XML and to build a game using a code generator.

Serious Game Logic and Structure Modeling Language (GLiSMo) (Thillainathan, 2013) proposed by Thillainathan offers a modeling framework consisting of two models: structure and logic. The proposed language is targeted towards non-technical educators who would be empowered to build serious games. The modeling framework uses model-driven development techniques enclosing modeling language, visual editor, transformation engine and generator.

Another approach to gamification modeling is based on using formal (or mathematical) models (Nummenmaa, Berki, & Mikkonen, 2009). Kim and Lee (J. T. Kim & Lee, 2013) model the effectiveness of gamification effectiveness using a mathematical model based on a sigmoidal equation. They argue that gamification effectiveness can be represented applying curiosity, challenge, fantasy and control factors. Bista *et al.* (Bista et al., 2012) have proposed the first formal gamification model. Chan *et al.* (Chan, King, & Yuen, 2009) offer a similar approach on social game modeling, which also allows for verification of the built model. Oliveira *et al.* (de

Oliveira, Julia, & Passos, 2011) model games using Petri nets. The disadvantage of this approach is the lack of domain specificity which is preventing its adoption by game designers.












More abstractly, game elements can be specified using a XML-based Gamification Modelling Language (GaML) (Herzig et al., 2013), which provides a mechanism for a precise definition of gamification concepts that is suitable for exchange on game mechanics. Finally, game rules connect game elements into a game layer. Such game rules can be modeled using a Petri Net based Machinations visual modeling notation (Joris Dormans, 2012).

The third category of gamification modeling approaches is visual languages for fast prototyping in gamification domain. The best known examples are Sketch-It-Up (Agustin et al., 2007), Ludocore (Smith, Nelson, & Mateas, 2010), and Machinations (Joris Dormans, 2009). Sketch-It-Up is a tool for creating sketches of possible games. Ludocore is a logical “game engine”, which employs formal logic used by automated reasoning tools in the AI domain to enable automated design and prototyping of game systems and providing fast feedback to the designer. Machinations is a conceptual framework and diagram tool that focusses on structural qualities of game mechanics. Graphical diagrams of machinations are an abstraction of Petri nets for modeling and simulating games and game-like systems on a varying level of abstraction. Micro-Machinations (Van Rozen & Dormans, 2014) have been proposed for reusing Machinations models in software development.

### **2.3.2. Introduction to Machinations modeling framework**

To represent the patterns graphically, we use Machinations, a visual modeling framework for game mechanics (J Dormans, 2013) that facilitates the design, simulation and testing of the internal economy of a game at various levels of abstraction. At the heart of the framework is a graphical notation designed to capture the dynamics of games. Machinations diagrams are a class of Petri Nets, wrapped in a formalism that makes them more palatable to game designers. The logic behind Machinations is what gameplay is ultimately determined by the flow of resources. Resource flows allow to visualize how the system is constructed and what feedback structures exist within the game structure. The Machinations diagram has four parts: nodes, connections, other elements and other concepts. There are several different types of the nodes: *Sources* provide the flow of resources, *Drains* remove resources from the system, *Pools* allow to store resources, and *Converters* destroy resources to create new resources. *Trader* allows the exchange of resources between players or game elements. *Gates* control (randomly or deterministically) resource flow. *Delays* delay the resource flow. *Resource connections* determine how the resource flows between the nodes. *State connections* determine how the node state changes affect other elements. *Label types* are a part of the state and resource connections passing specific control information. The full description of Machinations is given in Table 1.

Table 1. Machinations modelling framework (J Dormans, 2013)

Name	Visualization	Description
Pool		A node which collects resources
Gate		A Gate is a node that controls resource flow. Gates can be used to create random or deterministic distribution, or limit the number of resources passing through it.
Source		A node which creates resources
Drain		A node which destroys resources.
Converter		A Converter is a node that destroys resources to produce new resources.
Trader		A Trader is a node that controls the exchange of resources between two players (or rather, locations in the diagram).
Flow connection		Flow Connections determine how resources flow through a diagram. The Label of a Resource Connection determines the number of resources that are produced, exchanged or consumed by various nodes.
State connection		State Connections represent how the state changes of a node affect another node or the Label. A node's state is a numeric value that is equal to the number of resources currently on the element.
Trigger		Triggers are State Connections that connect two nodes or connects a node to the label of a resource connection.
Activator		An activator is a state connection that has a condition assists label. This condition can be written down as a simple expression.
Resource		The main data which flows through the model.

### 2.3.3. Formal models of game design and gamification

Games are kinds of systems and the design of games is the creation of models for games (Grünvogel, 2005). In computer science, games can be considered as certain information systems consisting of modelled use of objects (or entities, concepts), attributes (properties), their relationships and the environment (or context) (Salen & Zimmerman, 2004). A similar approach has been adopted by ontology engineering (Devedzić, 2002) for building ontologies, i.e., formal representations of concepts within a domain and the relationships between those concepts.

Formally, games can be modelled as abstract control systems (Tabuada, Pappas, & Lima, 2004) consisting of a set of states and a definition of the evolution of the state of a game under different actions of a player. The game can be represented by a set of states, for which transition functions define when to move from one state to another.

Formally, gamified systems can be described using a theory of multi-games. Multi-Games is a class of games when each player can allocate its resources in varying proportions to play in a number of different environments, each representing a basic game in its own right (Edalat, Ghoroghi, & Sakellariou, 2012). Each player can have different sets of strategies for different basic games. The actors are permitted to play multiple games simultaneously. This multiplicity means that the actor must take interactions among relevant games and other players into account (Sallach, North, &

Tatara, 2010). Gamified IS can be interpreted as a multi-game, i.e. a system of two games, where one game is a serious game (i.e., target IS) and another game is an entertainment game (i.e., gamification layer in target IS), where an action in the serious game leads to a reward in the entertainment game.

Following Grunvogel (Grünvogel, 2005), each game  $G$  is a triple  $(S, M, F)$ , where  $S$  is a set that represents the states of the different game objects,  $M$  is a monoid that represents the input of the players, and  $F$  is the action of the monoid  $M$  on set  $S$  as follows:  $F : S \times M \rightarrow S$ .

Then gamification can be described as a product of two games  $G_1$  and  $G_2$  as follows:  $G_1 \times G_2 = (S_1 \times S_2, M_1 \times M_2, F_1 \times F_2)$ , where  $G_1$  is a serious (economical) game with tangible external actions and rewards, and  $G_2$  is a non-serious game based on top of  $G_1$  with virtual actions and rewards.

Another game modeling framework presented in (Narayanasamy, Wong, Rai, & Chiou, 2010) incorporates structural, temporal and boundary frameworks (subsystems). The structural subsystem consists of Game Elements, Game Time, Players, Interface and the Facilitator, the arbitrating entity between the players and the game system, which takes care of setting up the game, synchronizes the game state and maintains the game time. The temporal subsystem represents the flow and causality of the game by defining the actions that are provided and the actions that can be taken at the particular states in the game. The boundary subsystem defines the constraints in the game that limit the activities performed in a game by establishing social contracts between the players which must be satisfied through a set of limitations while playing.

In (Salazar, 2004), another kind of formal model (Petri Nets and Hypergraphs) is investigated and methods and tools for the integration of formal modelling into the game design and production process are proposed.

Martin Mazanec and Ondrej Macek identify criteria for the evaluation of modeling languages (Mazanec & Macek, 2012): design whole software, describe various levels of abstraction, readability and simplicity, unambiguity, supportability and integrability. Lethbridge (Lethbridge, 2013) distinguishes four key properties for identifying modeling languages: usability, completeness and scalability. In the article describing how to gamify applications (Morschheuser, Hamari, Werder, & Abe, 2017) the requirements for gamification project are described. Seungkeun Song and Joohyeon Lee identify key factors of heuristic evaluation for game design (Song & Lee, 2007) and Eric Sanchez (Sanchez, 2011) single out key criteria for game design. Using these sources criteria for gamification modelling frameworks are formulated and summarized in Table 2.

Table 2. Gamification modelling framework comparison criteria.

Index	Criteria	Description	Literature
C1	Modelling features	Modelling features supported for the whole software design.	(Lethbridge, 2013; Mazanec & Macek, 2012)
C2	Levels of abstraction	Describes how framework supports multiple levels of abstraction allowing users to decide the level of the details exposed.	(Lethbridge, 2013; Mazanec & Macek, 2012)
C3	Readability and simplicity	System models modelled with the framework readability and simplicity.	(Lethbridge, 2013; Mazanec & Macek, 2012)

C4	Unambiguity	System model dependency on user specification.	(Mazanec & Macek, 2012)
C5	Supportability and integrability	Modeling framework tools which allow working with the modeling framework and tooling integrability with other tools and processes.	(Lethbridge, 2013; Mazanec & Macek, 2012)
C6	Iterative gamification development (design, creation, analysis, simulation, transformation and optimization)	Framework capabilities of quick iterations over build models, painfulness of changes to the system model in case of adding, removing and updating functionality. Modeling framework support for simulation and transformation into executable applications.	(Morschheuser et al., 2017; Sanchez, 2011; Song & Lee, 2007)
C7	User centric feedback support	Modeling framework has a concept of users and incorporates user feedback loops.	(Morschheuser et al., 2017; Sanchez, 2011; Song & Lee, 2007)
C8	User motivation and behavior evaluation	Modeling framework gives insights into user behavior and motivation. System accounts for user psychologig factors.	(Morschheuser et al., 2017; Sanchez, 2011; Song & Lee, 2007)

Industry provides a lot of different modeling frameworks which serve different purposes. Table 3 and Table 4 compare different modeling frameworks under criteria described in Table 2.

**Table 3.** Comparison of formal description, timed automata, Petri nets modelling frameworks

	Custom formal description	Timed automata	Petri nets
C1	Supports mathematical model specifications and verification. Primary focus lies on system modeling.	Models described in mathematical model, supports simulation, verification and visualization.	Supports mathematical modeling, simulation, verification, visualization and basic model transformation.
C2	The syntax depends on domain and application. Generally, supports a necessary level of abstraction.	Gives highly verbose models of the timed system. Lacks flexibility of controlling system abstraction levels.	Produces a highly verbose system models, unable to control abstraction.
C3	Mathematical models tend to be readable until a certain level of complexity. It is difficult to explain complex logical operations and data manipulations.	At its core language is very readable, complexity comes from application complexity.	At its core language is very readable. The more nodes and edges the graph has, the more complex the model is.
C4	Specification by different users is very likely to be different.	Language is well defined which makes models similar.	Language is well defined which makes models similar.
C5	There are no tools designed specifically for formal descriptions.	Has multiple tools, i.e. UPPAAL, TART, Synthia, etc.	Multiple tools by multiple researchers ("Petri net Java applets," 2017)
C6	Not suitable for quick development and iteration.	Tools enable quick design, simulation and iteration.	Tools enable quick design, simulation, and iteration.
C7	Takes system centric approach.	Takes system centric approach from time perspective.	Takes system centric approach from the resource flow perspective.
C8	The approach does not focus on simulation from user perspective.	Approach does not focus on simulation from user perspective	Approach does not focus on simulation from user perspective

**Table 4.** Comparison of GaML, UML and Machinations modeling frameworks

	GaML	UML	Machinations
C1	Supports textual description of gamification models and model transformation.	Supports visual description, model validation and transformation. Focuses on the whole application from different aspects.	Supports visual models of games through four economic functions. Supports simulation and software transformation.



C2	Highly verbose because requires the whole system description. The model is targeting real systems.	Allows the whole system modeling from different levels and aspects of abstraction.	Allows single level of abstraction, controlled by the user.
C3	Very readable, simple and is very well-structured modeling language, but verbose.	UML allows higher and lower level views which ensure model readability and simplicity.	Simple and readable models at the right level of abstraction depending on system complexity.
C4	Models are similar.	Models are different due to their visual nature.	Models are different due to their visual nature.
C5	MatLab and other mathematical languages.	StarUML, Draw.io, UMLet, Magic Draw and many more.	Machinations tool has been developed by Joris Dormans.
C6	Theoretically allows software generation and simulation.	Quick and iterative design, no features for simulation and software generation.	Supports visual design, simulation and iteration.
C7	Takes software as a service approach.	Covers user centric and system centric approaches	Takes user centric approach.
C8	Does not support simulation.	No support of simulation from user perspective.	Models can be built in user behavior.

Using any of the modeling frameworks, users can model gamified systems. Modeling frameworks help building consistent models that can be analyzed for commonalities and patterns.

## 2.4. Agent-based simulation and social gaming

A complex system is a system which is made up of many interrelated agents. In such systems, the individual agents and the complex interactions between them often lead to behaviors which are not easily predicted from knowledge of the behavior of individual agents. The concepts of complex systems such as self-organization, emergence and level hierarchies (Mayer, Bekebrede, & van Bilsen, 2010), and methodologies such as multi-agent modeling and simulation gaming, are applicable to a wide range of natural and social phenomena such as ecosystems (Balbi & Giupponi, 2010), social interactions (Shapiro et al., 2015), the economy and financial markets (Z. Zhang, Wang, & Gao, 2008), road traffic (Sur, Sharma, & Shukla, 2012), cloud computing (Wozniak, n.d.), the Internet (H.-F. Zhang, Yang, Wu, Wang, & Zhou, 2013), disease epidemic modelling, cybersecurity (Casey et al., 2014) and even entire human societies (Kohler & Gumerman, 2000).

Agent-based models are computational models, which simulate interactions among agents in order to understand the emerging behavior of the overall system based on the microscopic behavior dynamics of each agent (Marsan, 2009). Agent-based modeling and simulation enables the researcher to create, analyze, and experiment with models composed of agents that interact within an artificial environment (Gilbert, 2008). This approach combines elements of game theory, multi-agent systems and stochastic methods.

Game theory recently has become widely used in social sciences and economics (Roth, 2002). A game can be described as any social situation involving two or more players. A game is a system in which players are drawn in an artificially made-up conflict, which is defined by rules, and the outcome can be measured (Salen & Zimmerman, 2004). Game theory aims to find and describe players' behavior, which provides best responses to other players' individual decision choices. The rules governing interaction between the two players are defined as a part of the description of the game. In a game, the rewards as points, badges, etc. are defined by the rules of

the game. The player is free to make a move or to make an action as defined by the rules of the game aiming to increase his outcome of the game (reward). A social game is a game defined over the elements of social state, social motivations, and social moves (Shapiro et al., 2015). Social gaming is directly related to games with a purpose (GWAP). GWAP are games, in which some useful computation is performed by humans as an element of a game (Von Ahn & Dabbish, 2008). GWAP have been applied in areas of computer vision (Galli, Fraternali, Martinenghi, Tagliasacchi, & Novak, 2012), content management (Giouvanakis, Kotropoulos, Theodoridis, & Pitas, 2013), semantic search (Lux, Guggenberger, & Riegler, 2014), and education (Muratet, Torguet, Jessel, & Viallet, 2009). Humans, however, require some incentive (reward or engagement) to become and remain a part of a social game of GWAP, which is defined as the reinforcement model.

Designing social games or GWAP requires gamification, i.e. turning humans' everyday interactions or work into games that allow to enhance productivity and engagement of a user for business purposes or achieving other meaningful results. Gamification (Deterding, Sicart, Nacke, O'Hara, & Dixon, 2011) involves the use of game mechanics to non-game activities to influence people's behavior, engage audiences and solve problems. Gamification and serious games are related, because their common aim is to achieve some value beyond plain entertainment. Serious games offer an enjoyable way to solve real-world problems (Richter et al., 2015). However, the design of engaging games that can keep their players interested in continuing playing games for a long time is still a major problem in gamification research. To understand gamification and its effects the use of effective game modeling and simulation methods and tools are required. Recently, game mechanics of GWAPs have begun to be modelled formally (Chan et al., 2009), aiming to standardize the design of GWAP.

## **2.5. Software generation from models**

The idea of code generation from models is not new and has been discussed for years. Model driven engineering is an approach where tools enable developers to generate software codes automatically and achieve very high productivity (Klein, 2015). It is very natural for us to expect any models representing a software system to allow some level of code generation. Code generation is researched in many domains like embedded systems (Kwon, Yi, Kim, & Ha, 2005; Yu, Dömer, & Gajski, 2004), test generation (Vock, Schmid, & Von Staudt, 2006), robot control software (Bruccoleri, D'onofrio, & La Commare, 2007), antivirus software (Koike, Nakaya, & Koi, 2007) and many more.

There are attempts to allow code generation from models built with UML, SysML and other methods. González and his team offered a description method for SysML that allows making a better design than using UML standard tools by using XSD-Schema inferences (Alonso, Fuente, & Brugos, 2009).

Some attempts were made to propose software generation methods from UML models using automatic mapping finite state machines or other models (Brisolara, Oliveira, Redin, Lamb, & Wagner, 2008). Another approach of adding meta information into UML sequence and activity models was explored by Viswanathan and Samuel (Viswanathan & Samuel, 2016). Alternative use case for UML code generation is automating test case generation (Yongfeng, Bin, Minyan, & Zhen, 2009).

Finite state machines are part of a formal model class. Formal models are often chosen for software generation for their well-defined format. In many cases, formal models are used for critical software generation (Oz, Sener, Kaymakci, Ustoglu, & Cansever, 2015) or test generation (Rayadurgam, 2001).

The holy grail of software generation form models is converting natural language into executable programs (Eder, Filieri, Kurz, Heistracher, & Pezzuto, 2008). Furthermore, scientists attempted natural language transformation into other models like UML (Gulia & Choudhury, 2016). To summarize, the idea behind model transformation into executable code is to take a higher abstraction language and transform it to lower abstraction language.

## **2.6. Gamification architectural design**

Gamification can be implemented using several architectural design methods such as:

1. service: a separate gamification system is developed in a way which provides elements of gamification to other systems as a service (e.g., Mozilla Foundation OpenBadges (Jovanovic & Devedzic, 2014));
2. module: a separate gamification module is developed in a way which is integrated into a target system at a later stage of design (e.g., EcoDriving (Barkenbus, 2010));
3. plugin: full implementation of gamification is developed in a way which is later added to a target system without any additional effort (e.g., Jira (Hoarau, 2012));
4. separate system: a gamification system and a target system are implemented separately and communicate with each other via messages (e.g., TaskVille (Nikkila, Linn, Sundaram, & Kelliher, 2011));
5. integrated system: an integrated system is developed in a way which combines both target functionality as well as game behavior/mechanics (e.g., RedCritic Tracker (RedCritic Corp, 2011)).

According to Neeli (Neeli, 2012), gamification of a business IS can be performed at different levels with respect to business activities: 1) at a superficial level, the game mechanics are used independently of business activity being performed, 2) at integrated level, the game mechanics are integrated into the business activity being performed, and 3) at embedded level, the business activity is designed based on game mechanics.

## **2.7. Related research by Lithuanian authors**

Lithuanian researchers have adapted gamification to various applications. To name a few domains: employee motivation (Gatautis, Vitkauskaitė, Gadeikiene, & Piligrimiene, 2016), education (Dagienė, Pelikis, & Stupurienė, 2015; Dagiene & Stupuriene, 2016; Stupurienė, Vinikienė, & Dagienė, 2016), social problems (Bieliūnaitė-Jankauskienė & Auruškevičienė, 2016; Pitrenaitė-Zileniene & Skarzauskiene, 2013), and others. Multiple authors argue about the necessity of applying gamification for ensuring user engagement and motivation (Barisas, Duracz, & Taha, 2014; Gatautis & Vitkauskaitė, 2014).

Kalinauskas focuses on theoretical foundation behind player types increasing user creativity and how to apply gamification to creativity flow (Kalinauskas, 2014a, 2014b). Skaržauskienė and Kalinauskas analyze how gamification can be applied to increase collective creativity, they have formulated the main premises on which gamification should work (Skaržauskienė & Kalinauskas, 2014). Dagienė examines gamification application in Lithuanian schools and focuses gamification application in their custom educational software solution (Dagiene & Stupuriene, 2016). In addition, Stupurienė and her team summarize a 6-year study of observed pupils in the Bebras computing challenge, which has shown the importance of long term participation in such contests (Stupurienė et al., 2016).

Piligrimienė and her team discuss the value that consumer engagement brings to the company and how can gamification help (Piligrimiene, Dovaliene, & Virvilaite, 2015). Bieliūnaitė-Jankauskienė in her theses (Bieliūnaitė-Jankauskienė & Auruškevičienė, 2016) analyses how gamification can help financing social causes. She has found that carefully tailored gamification elements have a direct, positive impact on individual donation intentions.

Kostecka and Davidavičienė analyze gamification effects of employee motivation in gamified information systems (Kostecka & Davidavičienė, 2015). They describe their gamification model to improve accountant monotonic activities.

Rimantas Gatautis and his team study impact of gamification on consumer brand engagement, and they have found that gamification is positively related with brand engagement, but the relation is quite weak (Gatautis, Banyte, Piligrimiene, Vitkauskaite, & Tarute, 2016). Furthermore, Gatautis et al. analyze online consumer behavior from a perspective of stimulus-organism-reaction models and propose a topology of game components and consumer interpretation (Gatautis, Vitkauskaite, et al., 2016).

## **2.8. Summary**

Gamification is a methodology that is applied to improve software systems via game mechanics and game elements. Gamification is becoming adapted worldwide and in different industries. Literature suggests that the number of gamified systems will continue to increase. The main gamification solutions take points, leaderboards and badges, which indicates that evaluation and understanding of gamified systems are still in their early stages and prevent more complex gamification solutions to evolve.

The greatest motivation behind the adaption of gamification is altering user behavior. Gamification is used for different goals, for example, increasing user engagement, employee motivation, reinforcing desired user behavior, driving social change, retaining users, increasing brand loyalty, and many more. Psychological models and theories have been defined to explain why and how gamification alters user behavior in desired ways. Gamification effects tend to decay and require reinforcement models to slow down decay. The main emotional factors and states which need to be incorporated into reinforcement models are: boredom, fatigue, frustration, satisfaction, feedback and interestingness.

Modeling gamification can be done in different ways, for example, UML, MDA, GaML, Machinations, Petri nets, etc. These approaches have their pros and cons. The criteria for gamification modeling framework comparison such as: modeling features for software design, levels of abstraction, readability and simplicity, unambiguity,

supportability and integrability, iterative development, user centric feedback, user motivation and behavior evaluation have been identified. None of the modeling frameworks combine simulation, user centric feedback, code generation and formal modeling into one modeling method, but multiple methods include several of these criteria.

Gamification can be expressed as a multi-game where one part is the IT system and the other part is the game, and this allows gamification to be analyzed separately. Agent-based simulation is perfect for simulating user behavior in gamified systems. Gamification can be implemented as a service, plugin, module, separate or integrated system. There are many Lithuanian researchers working on gamification in various areas. Lithuanian researchers' findings show positive impact of gamification which aligns with international findings.

The gamification domain lacks domain-oriented approaches for solving domain specific problems such as:

- the analysis of game and gamification patterns, extraction and understanding how patterns effect different applications;
- abstract methods to model gamification of the systems and to perform simulation of gamification;
- quantitative evaluation of gamification solutions;
- tools to model, simulate, analyze and generate gamified solutions for further deeper understanding of the gamified system;
- simulation of gamification from a user-centric and behavior-focused perspective based on a solid psychological foundation.

### **3. SPECIFICATION OF GAMIFIED SYSTEMS**

#### **3.1. Methodology for the gamified system analysis**

For the analysis and identification of gamification patterns, seven different gamified applications have been selected (Emo-bin (Berengueres, Alsuwairi, Zaki, & Ng, 2013), Meeco (Vara, Macias, Gracia, Torrents, & Lee, 2011), Teamfeed (Singer & Schneider, 2012), CAPTCHINO (Saha, Manna, & Geetha, 2012), Taskville (Nikkila et al., 2011), Power House (Reeves, Cummings, Scarborough, Flora, & Anderson, 2012), Trogon (Ašeriškis & Damaševičius, 2014a)). All analyzed applications have common attributes: user-centric, which means that all of them have the underlying concept of player; user interaction with the system which triggers the basic gameplay; game rules in one or other form; game-oriented interface elements such as badges and leaderboards. For each application we have created two types of models using the Machinations game modeling framework (Adams & Dormans, 2012): 1) Simple model – the highest level of abstraction of the system. This view shows the core system concepts. 2) The advanced model is made up of two parts: a) static model which models as many details of the system as possible, and b) dynamic model, which is modeling interaction between players. All Machinations models are given in Appendix A. In addition, for each model formal model description and textual description have been built. Based on the result of model comparison and analysis we have identified common patterns of gamified systems.

In this chapter, gamification patterns are presented and their textual and visual descriptions are provided. The novelty of the proposed gamification patterns lies in

visual specification of patterns using domain-specific Machinations modelling language and framework (Adams & Dormans, 2012).

### 3.2. Formal model of the gamified system

For each system, formal models have been built to analyze what kind of common elements each of them has. Let us analyze gamification of Project Management System (PMS) Trogon (see Figure 1), as an example of a business IS (Information System). Following Bista *et al.* (Bista et al., 2012), gamification of a Project Management System is a tuple:

$$G = \langle J, B, R, F, P, W, T, I, D \rangle \quad (1)$$

here  $J$  – jobs which were entered into the PMS;  $B$  – badges defined in the PMS;  $R$  – ratings based on the number of finished jobs;  $W$  – registered workers;  $F$  – trees which represent jobs in the project forest;  $P$  – worker points received;  $I$  – month or week time interval;  $T$  – time represented in 15-minute time intervals; and  $D$  – a function to determine difficulty of jobs.



Figure 1. Screenshots of Trogon PMS

The value of points that a worker receives is a function:

$$P(j) = \sum_{n=0}^{count(j)} \left( \frac{T_r(j_n) \cdot b(j_n) \cdot y(j_n, j) - (0.1 \cdot (T_r(j_n) - T_p(j_n)))}{1} \right) \cdot D(j_n) \quad (2)$$

Here,  $P(j)$  is the number of points a worker receives in a time interval,  $b(j_n)$  is a badge that a worker receives,  $y(j_n, j)$  is a function that maps badges to points;  $T_r(j_n)$  - time to complete the job  $j_n$ ,  $T_p(j_n)$  - planned time to complete the job, and  $D(j_n)$  – difficulty of a the job. 0.1 constant represents 10%.

The game rules are as follows: (1) Every job can have badge  $b$  and planned work time  $T_p(j_n)$ . (2) Every worker has real work time  $T_r(j_n)$ . (3) Every job has its difficulty  $D(j_n)$ . (4) Badge  $b$  is awarded if it is not withdrawn until the job status

becomes “done”. The badge can be withdrawn by a project manager if the job quality is low or it has taken too long to finish. Quality assurance team members can remove the badge if there are many quality defects. The player ratings are computed as follows:

- A set of points is computed including all employee points for a considered time interval.

$$P = \langle p_1, p_2, \dots, p_n \rangle \quad (3)$$

- Set  $P$  is sorted by descending point count.

$$R = \text{sort\_desc}(P) \quad (4)$$

- Badge board sort order is computed as:

$$\text{sort\_desc}(\text{count}(b)) \quad (5)$$

Project forests are sorted by total forest size, which represents the time it has taken to complete all jobs.

### 3.3. Pattern description scheme

The following pattern description scheme adopted from UML pattern description (Gamma, 1995) is used:

- **Intent:** a short statement that describes what the pattern does, and what problem it addresses.
- **Motivation:** a more detailed discussion of the pattern and how it works.
- **Applicability:** the situations where the pattern can be applied to.
- **Structure:** graphical representation of the pattern using visual modelling language.
- **Participants:** the elements, mechanics and compound structures that are identifiable parts of the pattern.
- **Collaborations:** how participants collaborate.
- **Consequences:** the results of using the pattern, including trade-offs and possible risks.
- **Implementation:** a more detailed discussion on different techniques to implement the pattern.
- **Examples:** at least two existing examples of the pattern in games are discussed. Preferably, the examples of all patterns from a large variety of different games are drawn.
- **Related Patterns:** patterns are related to this pattern. Opportunities for pattern combination.
- **Discussion:** any discussion about the pattern itself, its viability, suggestions, alternative constructions, etc.

### 3.4. Gamification patterns

Every gamified system model should have a source to drive the whole system. We have discovered several main source patterns for modelling gamified systems as follows:



**Figure 2.** (a) Infinite quantity source and (b) limited quantity source.

**Infinite quantity source** (see **Figure 2.a** & Table 5)– in this case it is chosen to believe that maximum number of points is never reachable, for example, it is impossible to determine the number of user actions.

**Limited quantity source** (see **Figure 2.b** & **Table 5**) – it imposes a system constrain that the maximum number of points received is limited at every moment of the gameplay. The limit can be either physical or virtual. For example, in Emo-bin there is a limited number of bottles which is limited by local vending machine.

**Table 5.** Description of the limited quantity source and infinite quantity source patterns.

Property	Limited quantity source	Infinite quantity source
Intent	Enforce limit on a resource	Models unlimited resource economy
Motivation	This allows us to model limited economies	Sometimes resources can be viewed as unlimited. This allows us to model unlimited economies
Applicability	Modeling an economy with a limited number of resources	Modeling economy or part of economy with no economic restriction
Structure	Uses a pool with automatic push	Source node
Participants	Pool node	Source node
Consequences	Limits economic growth	Allows unlimited growth
Examples	Trogon, TaskVille, Emo-bin, Captchino	Teamfeed, Meeco, PowerHouse
Related Patterns	All	All

In addition to these two qualities we can add additional limitation or more realistic conditions:

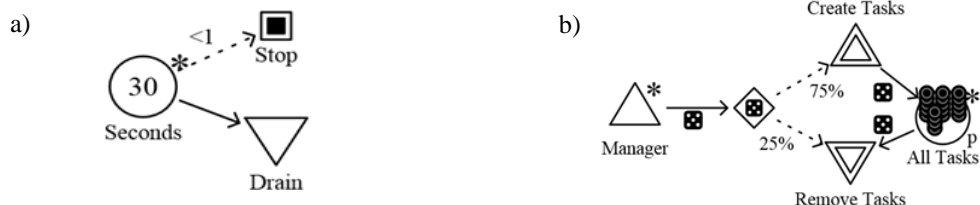


Figure 3. (a) Time limit and (b) dynamic limit patterns.

**Time limit** (see Figure 3.a & Table 6) – adds time limit to the system. Such limit imposed over infinite quantity source will make it bounded by time limit.

**Dynamic limit** (see Figure 3.b & Table 6) – it is limit which is imposed by model implication. For example, in a software company we have a project manager checking all tickets before development and there is a chance that a ticket might not be added to the pool of tickets.

Table 6. Description of time limit and dynamic limit patterns.

Property	Time limit	Dynamic limit
Intent	Stop the game after some time has passed	Control source growth
Motivation	Using such pattern allows to limit a game in time.	Such a pattern allows to add dynamic qualities to resources



Applicability	To impose time restrictions or rounds, for example, in Trogon there is limit for each round.	Normally the growth of resources is not linear and depends on different properties
Structure	A pushing pool of limited quantity connected to a drain and end condition node. End condition is connected with the pool labeled "<1".	A composition of a random gate with drain and source node connected with a pool from a limited source.
Participants	Pool, end condition, drain	Pool, gate, drain and source
Collaborations	The pool acts as a counter and is connected to a drain for decreasing the counter value. When the counter value is equal to zero, the end condition is triggered.	The pool connects with a gate. Then follows multiple connections to drains or pools which creates the desired logic model
Consequences	Changes the economy by setting up limitations to resources	
Examples	Trogon, Emo-bin	Trogon, TaskVille
Related Patterns	Limited quantity, property and chance pattern	-



Figure 4. (a) Random result pattern and (b) drain pattern.

**Random result** (see Figure 4.a & Table 7) – a connection with dice label is used. This type of pattern models an abstract connection. For example, “An executed action is worth X points”. This allows to change a part of the gamified system model with a high level of abstraction.

**Drain pattern** (see Figure 4.b & Table 7) – allows to decrease the score or counter under certain conditions. Drain pattern is useful to model penalty rules in the gamification systems.

Table 7. Description of random result pattern and drain pattern.

Property	Random result	Drain pattern
Intent	Aggregate logic	Invert logic
Motivation	Sometimes rules are too complex to model so it is easier to aggregate the whole logic into a single path	Economy grows and falls over time. This is a pattern to simulate economic falls
Applicability	Any case when a rule can be replaced by random number	Convert or model negative aspects of a game
Structure	Two nodes connected with random connections	Manual drain, gate and pool.
Participants	Connection and any two nodes	Drain, gate and pool
Collaborations	Connection passes random amount of points	When the gate triggers the drain the pool loses elements
Consequences	Aggregates the logic into one abstraction	Allows to destroy resources
Examples	All cases	Emo-bin, Captchino, TaskVille, PowerHouse
Related Patterns	-	Solver pattern

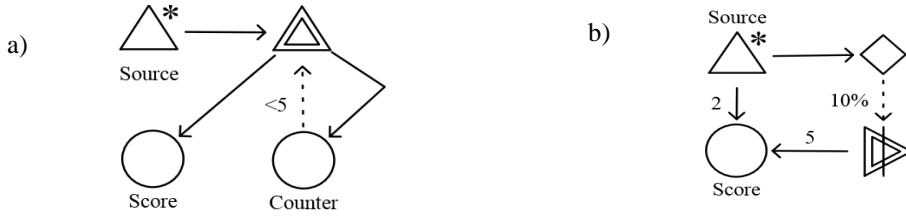


Figure 5. (a) Constrain pattern and (b) extension pattern.

**Constrain pattern** (see Figure 5.a & Table 8) allows to block certain paths in the model based on certain conditions.

**Extension pattern** (see Figure 5.b & Table 8) is a pattern of adding an additional random path under certain conditions. This allows to extend normal behavior with additional random bonuses.

Table 8. Description of constrain pattern and extension pattern.

Property	Constrain pattern	Extension pattern
Intent	Control flow on certain conditions	Introduce concurrent paths
Motivation	Considering the system state, it is useful to limit or open a path in relation to the state.	Sometimes we need to create an extension to default behavior.
Applicability	Any system which contains multiple paths under certain conditions	Any case when the default path is extended with a concurrent path.
Structure	Manual source and pool connected with a normal and state connection	Node having at least two paths and ending with one node.
Participants	Manual source, pool	Source, gate, converter, and pool
Collaborations	The path is turned off then counter reaches its target	Once a source is triggered multiple paths activate simultaneously
Consequences	Paths can be open or closed	Extend a path with additional concurrent path
Examples	Captchino, Trogon, Meeco	Trogon, Captchino.
Related Patterns	-	Property and chance pattern

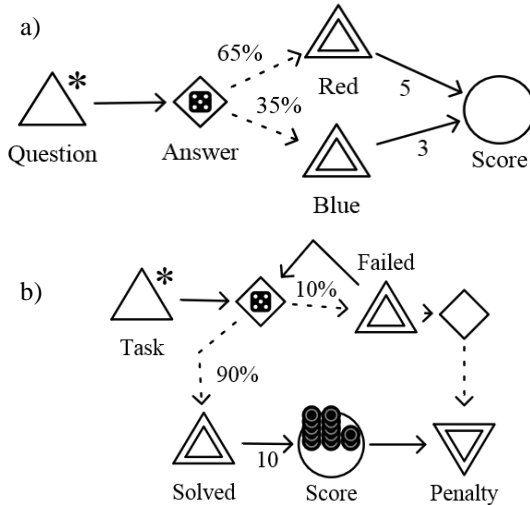


Figure 6. (a) Property and chance pattern; and (b) solver pattern.

**Property and chance pattern** (Figure 6.a & Table 9) is a pattern for creating multiple paths or modeling a certain user property. For example, we need to model

multiple actions in a single model, like “Buy” and “Attack”. In this case, we leverage the economic and aggressive user properties; the higher “Attack” percentage, the more aggressive the user’s strategy is and vice versa.

**Solver pattern** (see Figure 6.b & Table 9) allows to model user solving a problem. Solver pattern enables to create a delay in the system.

Table 9. Description of the property and chance pattern and solver pattern.

Property	Property and chance pattern	Solver pattern
Intent	To model a property or random chance	Models problem solving
Motivation	To model simple user or entity behavior.	In real world, actions do not occur instantly. Normally it takes time for the problem to be solved
Applicability	Any place we want to model a chance of occurring action or user behavior	When we want to randomize the amount of time it takes to accomplish a task
Structure	Random gate and multiple manual sources	This combines the pattern of the drain and chance pattern
Participants	Random gate and manual sources	Random gate, sources, gate and drain
Collaborations	Gate triggers a source randomly	This pattern combines property chance pattern with a source which models a problem-solving skill. The source is also connected with a drain pattern to model negative consequences of incorrect solutions. This is optional for this pattern
Consequences	One of multiple paths is chosen	Random time is spent to solve a problem
Examples	All	Captchino, TaskVille, Trogon
Related Patterns	Solver pattern	Drain pattern, property and chance pattern

### 3.5. Example of pattern application

Trogon Project Management System (PMS) (Ašeriškis & Damaševičius, 2014a) is an example of enterprise Information System. The gamified PMS has a leaderboard, badge board and project forest as the main elements of gamification. Every element has its purpose. 1) The leaderboard creates competition between individual employees and allows to determine a game winner, who should be awarded additionally. 2) The badge board enables observing the skills of employees. In the badge board, the employees are sorted by the total number of badges collected. Each badge represents a skill and has its personal level. Progress between levels is displayed as a progress bar. 3) The project forest provides the element of scalability to represent the size of different projects.

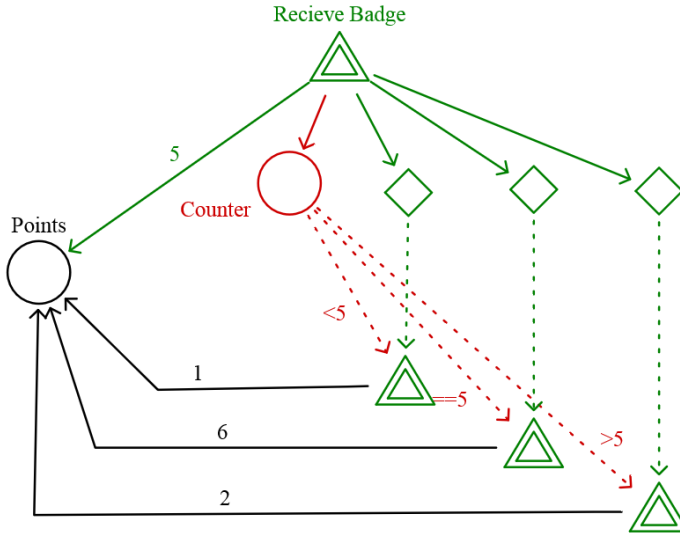


Figure 7. Trogon PMS rule model

The gamification model (Figure 7) simulates a Trogon game rule: “For every task solved a user gets X points. If a badge is earned for the solved task then a user gets Y points bonus. A user gets 2Y bonus for each task if he receives more than four badges”. To simplify real live computations for finishing tasks, a user receives five points. A bonus adds a single additional point.

This model has two pattern applications: 1) Constrain pattern (red) helps to control flow depending on how many badges are received. 2) Extension pattern (green) provides the necessary paths to model earning. Every time a badge is received, a source is triggered and user points increase by five points. In parallel, the counter is increased by one. For the received badge, a bonus point is rewarded. As you can see, there are three sources connected to the counter. The first counter to source connection has label “<5” which means while the user has less than five badges he gets only one point. After five badges are received new “=5” path opens and the user gets a reward of 6 points. Also, previous path “<5” is closed. When the next badge is received, path “>5” opens and the user is awarded with two points. All other paths are closed. This workflow models the behavior of the rules described in a previous paragraph.

### 3.6. Abstract formal model

Using the analysis of gamified systems, it has been found that all analyzed gamified systems share a common structure. Every gamified system is a collision of users, rules and data. Users execute rules through actions. Rules interact with data generating content which is stored in entities. Interface allows for data display to the user. It can be claimed that gamified systems can be modeled using the proposed abstract gamification model, which we call the UAREI (User-Action-Rule-Entities-Interface) model. We can use UAREI model for formal specification of gamification, and the UAREI visual modeling language for graphical representation of game mechanics. The whole UAREI system can be used for full gamification development process. Using the analysis of patterns

The gamified systems can be described as a tuple

$$G = \{ U, A, R, E, I \} \quad (6)$$

here:  $U$  – users interacting with the system;  $A$  – actions, which trigger system behavior;  $R$  – rules, which encapsulate logic in the system;  $E$  – data entities; and  $I$  – interfaces which define data format.

The users are defined as tuple  $U = \{ L_U, S_U \}$ , here:  $L_U$  – a set of all outgoing links to other elements in the model; and  $S_U$  – a selection function which defines how a user is selected from a collection in a simulation mode.

Actions are denoted as collection  $A = \{ A_1, A_2, \dots, A_i, \dots, A_n \}$ , here  $A_i$  is a single action,  $n$  is the total number of actions. A single action is defined as  $A_i = \{ L_A, S_A \}$ , here:  $L_A$  – a set of all outgoing links to other elements in the model, and  $S_A$  – a selection function, which defines the way an action related data entity is selected from the collection.

Rules are noted as collection  $R = \{ R_1, R_2, \dots, R_i, \dots, R_m \}$ , where  $R_i$  is a single rule,  $m$  is the total number of rules. A single rule is defined as  $R_i = \{ L_R, r_i(C, M) \}$ , where:  $L_R$  – a set of all outgoing links to other elements in the model, and  $r_i(C, M)$  is a rule function defined as:

$$r_i(C, M) = \begin{cases} NULL & \text{if no value is computed} \\ y & \text{if value is computed by rule} \end{cases} \quad (7)$$

where:  $C$  – the context of a current execution path;  $M$  – a system model;  $y$  is a computed result value, and  $NULL$  is returned if rule doesn't apply.

After rule execution, the returned value is stored inside  $C$ . Also, a rule can manipulate the context. Rules are used to control context flow in the system. If a rule execution evaluates to an empty result, a current execution path is continued. We can define “else” path by using inversion “ $\neg R_i$ ”. No data will be stored in storage and no other rules will execute if a previous rule failed or an empty value is returned, but the system flow will continue giving feedback to the user node. Rules can update the context in anyway needed for the application.

Entity collection is a collection of all data entities in the system  $E = \{ E_1, E_2, \dots, E_i, \dots, E_k \}$ , where  $E_i$  is a single storage entity and  $k$  is the total number of storage entities. A single entity is defined as  $E_i = \{ D, O, L_E \}$ , where:  $D$  – entity scheme definition,  $O$  – data objects, and  $L_E$  – a set of all outgoing links to other elements in the model. A triggered Entity can store any value from the execution context, which value to store is defined in entity scheme definition.

Interface is a collection  $I = \{ I_1, I_2, \dots, I_i, \dots, I_l \}$ , where  $I_i$  is a single interface and  $l$  is the total number of interfaces. A single interface is defined as  $I_i = \{ L_I, Q \}$ , where:  $L_I$  – a set of all outgoing links to other elements in the model,  $Q$  – data query, on which data for the interface is selected.





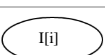

During simulation or program execution, before an action is triggered, a key-value map is created, which we call execution context  $C$ . This context is used to passed data between nodes through links. Also, the model flow works like this: a triggered action triggers all outbound linked nodes. Each other node will trigger all its outbound lined nodes. During triggering Rules execute their functions, Entities store values depending on scheme description, and Interfaces compute query results.

### 3.7. Graphical notation of the UAREI model

The UAREI model is visualized as a directed graph consisting of nodes (vertices) and links (edges) between nodes as follows:  $G = \{L, N\}$ , where  $N$  is a set all nodes  $N = \{N_1, N_2, \dots, N_i, \dots, N_m\} = U \cup A \cup R \cup E \cup I$ ;  $L$  is a set of links between nodes  $L = L_U \cup L_A \cup L_R \cup L_E \cup L_I$ , and  $L_U, L_A, L_R, L_E, L_I$  are collections of corresponding types of nodes  $L_X = \{L_{X_1}, L_{X_2}, \dots, L_{X_i}, \dots, L_{X_{n_X}}\}$ ,  $L_i$  is the list of links,  $L_i = (N_{out}; N_{in})$ , where  $N_{in}, N_{out} \in N$ ,  $L_{N_i}$  are links which start  $N_i$  node.

In Table 10 we present a list of graphical symbols (graphemes) used in the UAREI model diagrams.

Table 10. Graphical notation of the UAREI modelling language

Type	Grapheme	Description
User node	$U[i]$ 	Visualizes system user group. Normally a single action is triggered from this node.
Action node	$A[i]$ 	Visualizes an action. The action triggers its outgoing connections. Normally actions are connected to rules and other actions
Rule node	$A[i]$ 	Visualizes a rule node. Rule encloses all logic of a model. Rule triggers other rules, entities and interfaces.
Entity node	$E[i]$ 	Visualizes data entity. On triggering the node stores the data is received with the current context.
Interface node	 $I[i]$	Visualizes user interfaces, triggers user nodes finishing the feedback loop.
Connection		Visualizes relationships in the model. The direction of the arrow points from the outgoing node to the incoming node.

### 3.8. Summary

In this chapter, we have analyzed seven gamified systems and have identified gamification patterns common for two or more gamified applications. Ten gamification patterns: infinite source, limited source, time limit, dynamic limit, random result, drain patterns, constrain, extension, property and change, have been identified and solved. Each pattern has its own motivation, structure, applicability and consequences. The patterns are modelled using the Machinations framework (Adams & Dormans, 2012; J Dormans, 2013). This modeling tool allows prototyping ideas rapidly and testing them before implementation. A case study on applications of a gamification pattern combination has been demonstrated using Machinations in the context of a gamified project management system.

An abstract formal model gives an advantage of extracting gamification patterns from multiple formal definitions written for different gamification applications. The abstract model is constructed from users, actions, rules, data and interfaces which are common to analyzed systems. This model connected with a graph-based modeling language allows simple yet powerful visualization. Learnings from gamification modeling methods analysis were incorporated into UAREI.

## 4. IMPLEMENTATION OF THE GAMIFIED SYSTEM ANALYSIS TOOL

### 4.1. The Method for Gamified System Development

Now let us see how the UAREI model integrates into the whole UAREI development method for gamified systems.

Figure 8 presents the UAREI activity diagram which represents how development with UAREI formal method works. We start from building a formal UAREI model and decide if we want to analyze the model. If the answer is “no”, we are done; if “yes”, we transform the model into the UAREI model. If we do not want to improve the model we are done; if we want to improve the model we choose between generating the model or running simulations. If we decide to generate the model, we export a working application and evaluate it. If we are happy with system, we are done; if we want to improve the system, we transform it back to the UAREI model. If we do not want to generate the application, we run simulation on the model. After evaluating data, we decide whether we want to update the model or not. If we want to update the model, we make changes and rerun simulations. If we do not wish to update the model, we check whether it satisfies our requirements, and if so, we transform the model back to the UAREI JSON format; further on, we rerun the simulations.

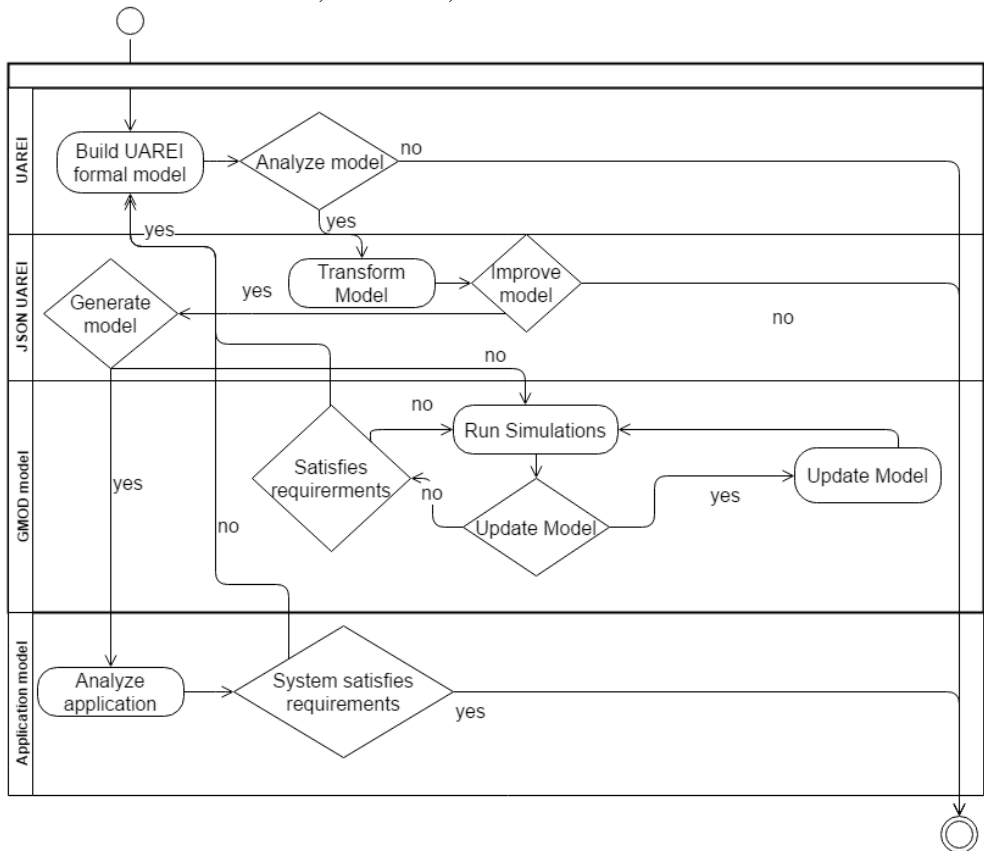


Figure 8. UAREI activity diagram

A formal UAREI model forms the basis of our system. Normally we should start by describing a new gamification solution in this formal model. Using JSON as a meta-language is not new (Giurca & Pascalau, 2008), a different JSON meta-language has been developed for UAREI purposes. Internally, the JSON model is a bridge between software, design and simulation tools, and a formal model. It is possible to transform formal model definition to the JSON notation, but this transformation could be semi-automated with limitations fully automated. The biggest problem for such a transformation is the mathematical language or textual description transcription to the JSON format. This can be solved by using structured languages. The JSON format can be applied in other systems for simulating and designing gamification solution.

Finally, JSON model can be converted into a working application. The generator discards unnecessary parts of the model and builds a working application. Running the real application can produce data which can be used in further refining the gamification solution.

#### 4.2. Transformation of the UAREI to UAREI JSON

Mathematical and natural language computational processing is a complex problem, and to simplify it, a simpler UAREI JSON model has been adapted. In MDA Meta Object Facility (MOF) enables the definition of modeling languages and transformation rules and due to this MDA Processing Process (MDAPP), which allows meta model transformation from one model to another model (Kriouile, Addamssiri, & Gadi, 2015), can be defined. Using customized version of MDAPP the UAREI to UAREI JSON transformation is defined.

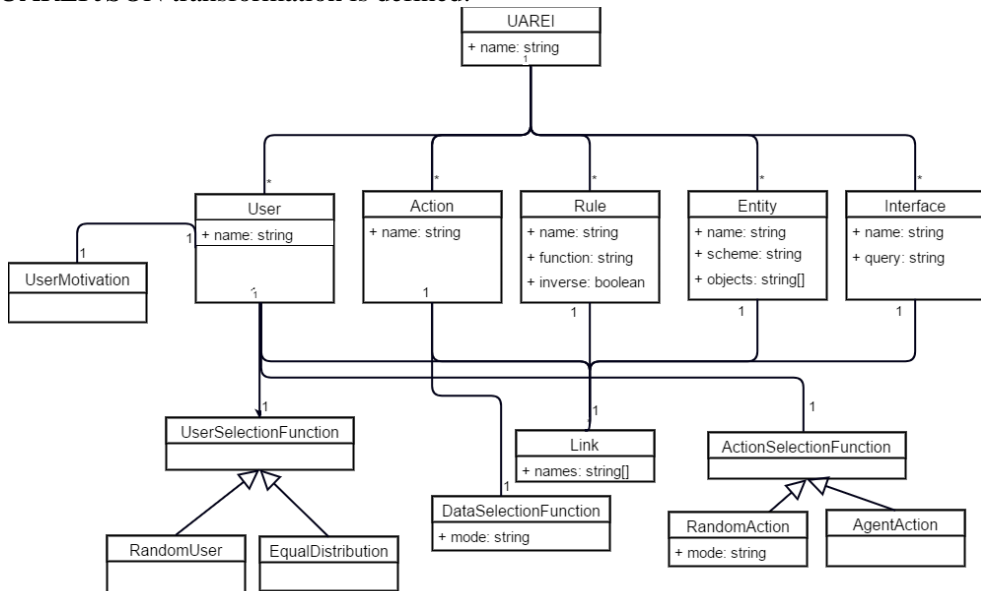


Figure 9. UAREI metamodel represented in UML Class Diagram

Figure 9 shows how the UAREI model can be represented by UML Class diagram. A mathematical definition of UAREI model matches the UML diagram. Each instance of User, Action, Rule, Entity and Interface nodes has one-to-one relation with



Link class. User motivation and user selection function classes can be defined for user nodes. Each User has action selection function class which defines how an action is picked. Each action has a data selection function which resolves data entity required for processing the action.

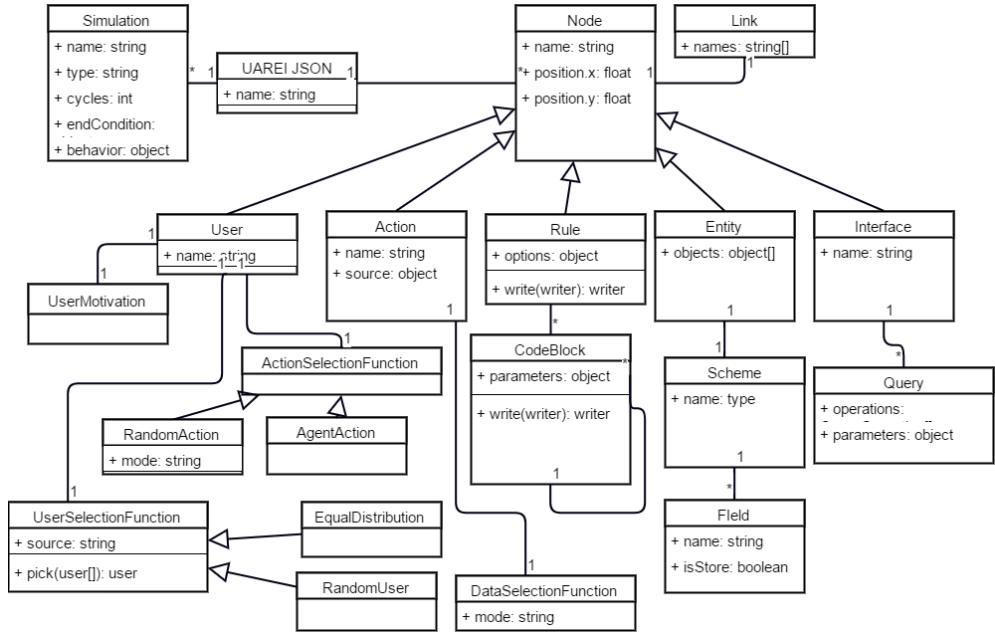


Figure 10. UAREI JSON format UML Class Diagram

Certain user selection functions and action selection functions are built into the model. They are used for model simulation. Any required selection function can be defined. It is worth noting that rule function, entity scheme and interface query are defined using string data format (textual description). We will assume mathematical formulas and text are in string data format which is easily transformed into computer analyzable form.

Javascript Object Notation (JSON) is a universal data scheme language used widely in a computer science domain. Figure 10 shows the UAREI JSON format as an UML class diagram. The major change is that all User, Action, Rule, Entity, and Interface nodes are joined into a single collection of nodes. UAREI model has a name, a list of nodes and a list of simulations. Simulations are used to run simulations on a defined model. Simulations are included only in the design phase and otherwise dropped from the model during transformations.

The three most important model changes refer to Rule, Entity and Interface nodes. Rules may have CodeBlock instances which can compose many CodeBlocks. CodeBlocks help us to define rule function in pseudo language constructs which can be transformed into executable functions. Entity has an array of objects which contains initial data set. Entity has a Scheme with Fields which define data entity structure. Interface node integrates Query which is used to select data on transform into necessary format.

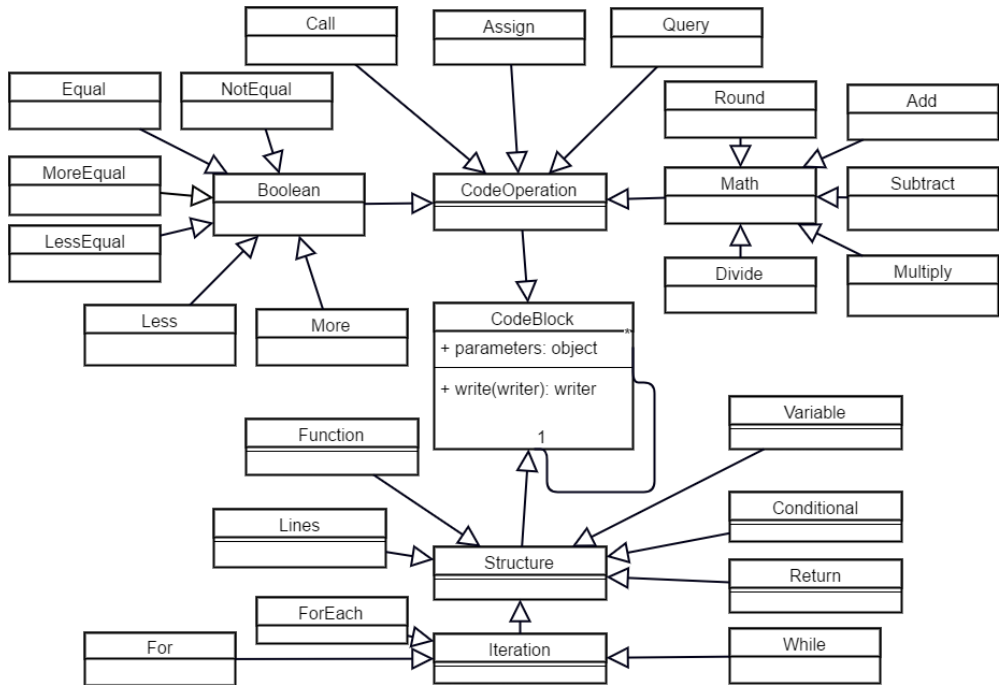


Figure 11. CodeBlock UML Class Diagram

In Figure 11, CodeBlock class relation to concrete programming language elements can be seen. It is worth noting that the model does not define any concrete programming language and its syntax can be extended to match specific needs, current composition matches the needs of the experiments analyzed in this thesis. CodeBlock has two important part operations and structures. CodeOperations is the abstract entity for all operations used in our pseudo language. The structure defines the main programming language structures such as variables, functions, etc. These classes can be implemented to generate code for any programming language.

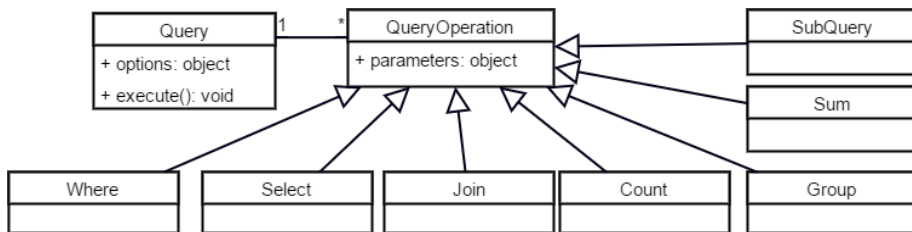


Figure 12. Query UML Class Diagram

Each interface has a Query which defines what data need to be selected and how manipulated before returning it to users (Figure 12). Query has options which are used during simulation for defining data visualization mode. Also, Query implements an execution function which generates the data query, runs query and returns the data. Query internally generates the required data query from a pseudo query language. Language could be expressed in any industry standard language, for example, SQL, MongoDB, etc.

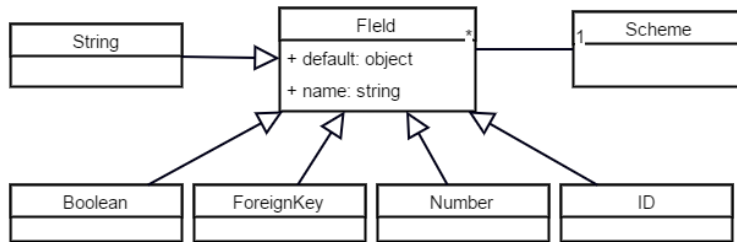


Figure 13. Entity Scheme UML Class Diagram

Figure 13 provides with Entity scheme structure. A Scheme is made up from a list of Fields. Each Field has a concrete type, optional default value, isStore flag and a name. If a field has a isStore flag, it will be populated during runtime with Rule generated result if it is not null.

Table 11. UAREI to UAREI JSON model transformation

Input - UAREI		Output – UAREI JSON		Rule
Class	Attribute	Class	Attribute	
UAREI	name	UAREI JSON	name	Rule 1
User	name	Node	name	Rule 1
UserMotivation	-	UserMotivation	-	Rule 1
UserSelectionFunction	-	UserSelectionFunction	-	Rule 1
ActionSelectionFunction	-	ActionSelectionFunction	-	Rule 1
Link	names	Link	Names	Rule 1
Action	name	Action	name	Rule 1
DataSelectionFunction	-	DataSelectionFunction	-	Rule 1
Rule	name	Node	name	Rule 1
Rule	function	CodeBlock	-	Rule 2
Rule	inverse	Rule	inverse	Rule 1
Entity	name	Node	name	Rule 1
Entity	scheme	Scheme	-	Rule 3
Entity	objects	Entity	objects	Rule 4
Interface	name	Node	name	Rule 1
Interface	query	Query	-	Rule 5
-	-	Node	position.x	Rule 6
-	-	Node	position.y	Rule 6
-	-	Simulation	*	Rule 6
-	-	Interface	options	Rule 6

Table 11 describes how UAREI and UAREI JSON both way transformation works. The six rules for this transformation:

- Rule 1: copy value from target to source without any changes.
- Rule 2: rule logic transformation. If the source model is UAREI, then transform a rule to pseudo code, else transcribe code into textual format. This transformation does not always match, because the input can be of any format and backward transformation always generates result in a textual format, which means UAREI -> UAREI JSON -> UAREI will not match unless textual format matches output formatting of the transcription.
- Rule 3: textual or mathematical text is transformed into Scheme class. If the source model is UAREI, then transform a text to Scheme class, else transcribe Scheme into a textual format. This rule has the same limitation as Rule 2.

- Rule 4: if the source model is UAREI, then transform a text to Object class, else transcribe Object text into a textual format. This rule has the same limitation as Rule 2.
- Rule 5: if the source model is UAREI, then transform a text to Query class, else transcribe Query text into a textual format. This rule has the same limitation as Rule 2.
- Rule 6: if the source mode is UAREI JSON, then remove an attribute or a class, else do nothing.

In the rules, transcription and transformation operations have been applied. Both these operations are conducted manually so the transformation limitation is not relevant. The rules described can be expressed in a pseudo code in Figure 14. In the rules textToCode, textToScheme, textToObjects and textToQuery methods describe how the transformation from mathematical expression and free text form are transcribed to the UAREI JSON structures. Currently this part is done manually. In the rules codeToText, schemeToText, objectsToText and queryToText there are methods which transcribe the UAREI JSON structures to free text or mathematical expressions. In the spoken situation, this part is done manually.

```
function RULE1(source, target) {
    target = source;
}
function Rule2(source, target) {
    if(source typeof UAREI) {
        target = textToCode(source)
    } else {
        target = codeToText(source)
    }
}
function Rule3(source, target) {
    if(source typeof UAREI) {
        target = textToScheme(source)
    } else {
        target = schemeToText(source)
    }
}
function Rule4(source, target) {
    if(source typeof UAREI) {
        target = textToObjects(source)
    } else {
        target = objectsToText(source)
    }
}
function Rule5(source, target) {
    if(source typeof UAREI) {
        target = textToQuery(source)
    } else {
        target = queryToText(source)
    }
}
function Rule6(source, target) {
    if(source typeof UAREI-JSON) {
        target = null
    }
}
```

Figure 14. UAREI to UAREI JSON transformation rules

### 4.3. UAREI JSON transformation and simulation

A process for UAREI JSON transformation to an executable application can be defined in such abstract steps:

1. Transformation setup – define settings for the target application.
2. Model registration with ModelExecutor
3. For each node do appropriate transformation:
  - a. entity transformation
  - b. rule transformation
  - c. interface transformation
  - d. action transformation
4. Export application – the process of building a final runnable application.

Let's start from step one. We need to choose the initial parameters for our application:

- target language: we need to load all CodeBlock instances implemented in a target programming language, in the analyzed situation the system supports only Javascript.
- target database: we need to define what kind of the database we will use and load the appropriate database manager. Currently the simulator supports only virtual database.
- target environment: we need to define in which environment the application will work in and to generate the appropriate scripts.

Once we have the set-up ready, we start model path extraction. Path extraction is a very simple process: we should get a list of actions for each user and build an array of paths for the action.

Now iterating over all the nodes, we can take necessary actions for each node based on its type. First, we have entity transformation. Entity transformation performs two tasks: it generates database creation scripts (for example, SQL) and registers each entity with the Database manager runtime.

Furthermore, we create a writer (buffer) which will hold our code, next we go through each block and call its write method, presumably block call renders ourselves and children blocks. Then we define rule execute method to be a new function having context, model and runtime parameters, function body (generated code), new the rule can be carried out calling execute.

Interface transformation registers each interface with the Database manager runtime. As these steps are completed, the output server implements executable file, database scripts and execution scripts.

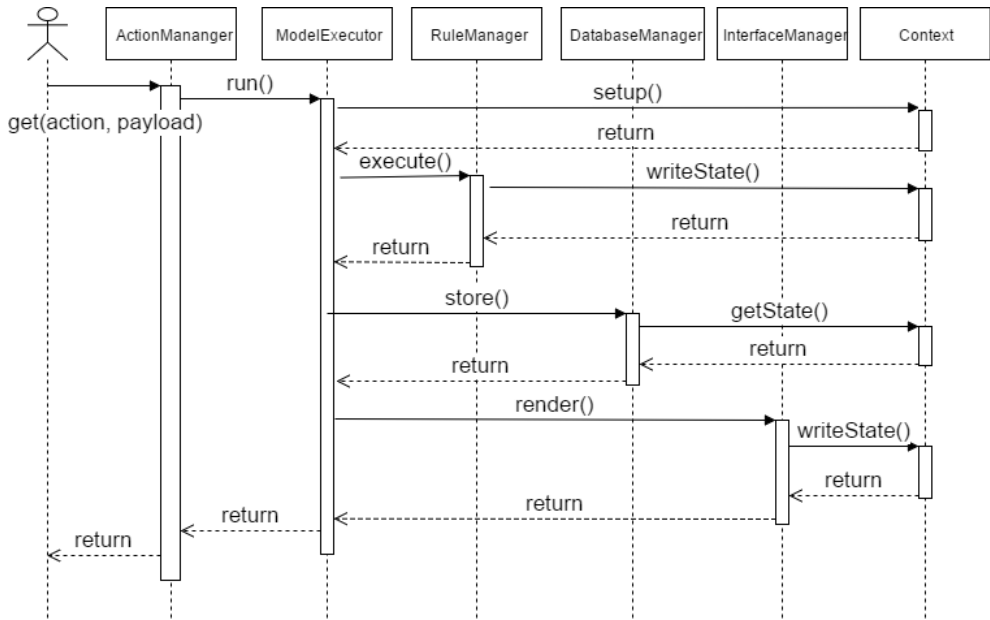


Figure 15. Application action execution

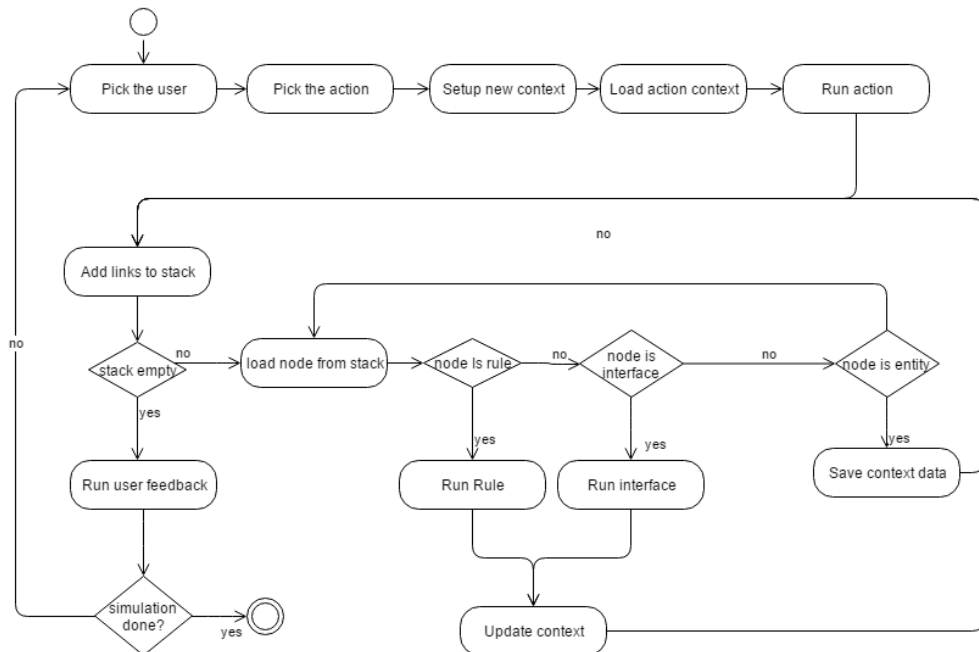


Figure 16. Simulation execution UML Activity diagram

In Figure 15 we have a sequence diagram of one simple action chain implemented as an API. Action links to one rule, the rule is combined with one entity, and one entity joins the interface. A user calls an action endpoint where the action manager passes a request to a model executor. Model executor sets up action context and executes a call to a rule manager that computes a rule result, updates the context

and sends a response to the model executor. Furthermore, the model executor sends a store request to database manager. The database manager takes data from the context and saves it. Next, the model executor calls the interface manager to render interface response which is added to a context. The model executor returns the context to the action manager that returns serialized version back to user.

The simulation activity diagram is illustrated in Figure 16. Each simulation cycle starts with selecting a user, applying user specific behavior to pick an action. Afterwards we set up the execution context which will be shared, and load action context (data entities needed for the action execution). Next, we run action and add all links to the execution stack. If the stack is not empty, we load a node from the stack and on the basis of that we type execute activities. If a node is a rule, we run the rule and update the context. If a node is an interface, we render the interface and update the context. If a node is an entity, we save context data to the entity. If the node is none of those types, we load the next node from the stack. After running node specific activities, we add all node links to the stack. If the stack is empty, we run user feedback and check if the simulation is performed; if simulation is not executed, we pick the next user and run the simulation cycle. If the simulation is done, we stop the simulation and exit.

#### 4.4. GMOD UAREI modeling and simulation tool

For scientific research, a tool has been built to formally model and simulate gamified systems described using the UAREI modeling method. System screenshot is shown in Figure 17. The system will be referred to as GMOD tool or **G**amification **M**odeling tool or simply GMOD.

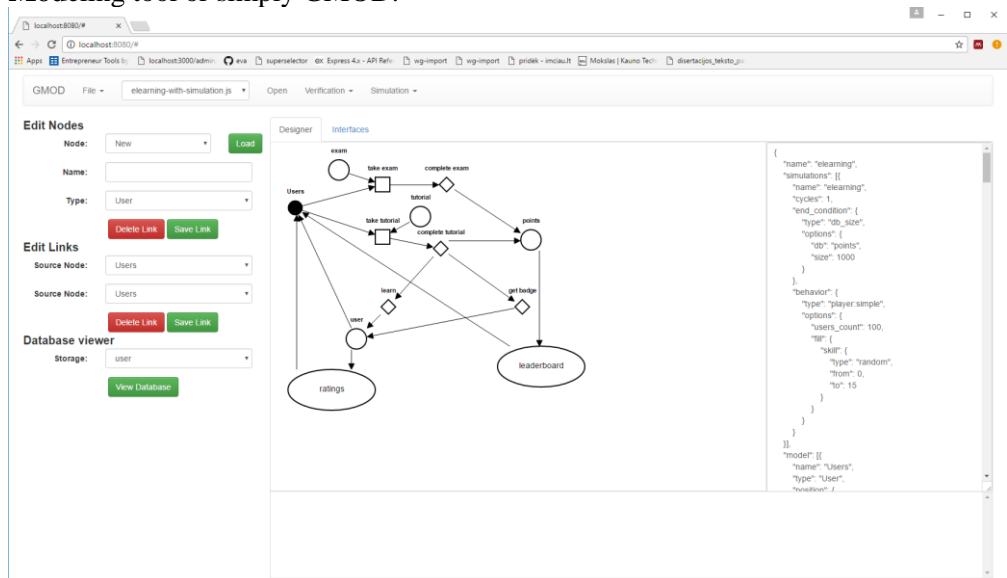


Figure 17. GMOD UAREI modelling and simulation tool

The system has five main parts. Firstly, the main menu which is responsible for main application operations like opening files, saving files, graph and simulation controls. The left rail controls model editing and viewing data entities. In the center, there is a graph visualizer which gives visual representation of the model. It also has

a function to move nodes and connections into a more visually attractive form. The right rail has the JSON UAREI model editor. It also allows to edit the model in a textual form. In the bottom, there is a console, which prints out data about the system state.

The editor has a second tab called interfaces. In this tab, there is user feedback information rendered into different graphs or other data representable formats. The data are updated in real time during the simulation. Interface and data entity values can be exported into csv format.

During the simulation, GMOD builds an executable application of the system, generates users and runs their actions through the system returning feedback information back to them. The JSON UAREI model is extended with additional meta information necessary for configuration of simulations, storing graph visual information. Also, the model can include initial data. The JSON notation also contains additional meta information necessary for building an executable application.

Possible user actions in the GMOD tool and their descriptions:

- Pick a file – pick which file you want to open in the simulator.
- Load a file – load a model from the file.
- Save a file – save a current model to a JSON file.
- View a graph – view visual model representation as a graph.
- Start the simulation – build an executable program and start the simulation.
- Stop the simulation – stop the simulation execution.
- View entities – view current data structures stored in memory.
- View interfaces – inspect the data returned from the application interfaces in an appropriate format.
- Manage nodes – add, edit and remove model nodes.
- Manage links – add, edit and remove model links.

The gamification modeling tool is a web application. NodeJS is used for model management and application resource loading. The server side is responsible for data management processes, everything else is running in the web browser. The application is implemented in Javascript programming language. A model is executed by building an in-memory runtime and dynamically evaluating a generated code.

To support the UAREI models, a framework has been built. The framework has three layers of abstraction of each type of the nodes (user, action, rule, entity and interface). The abstractions are:

- Basic – represents the core aspects of a node. Each node has its own concrete fields.
- Editor – the entity is extended by editor specific information about like node position and other.
- Simulation – simulation or execution abstraction level is the base for a real application.

A common runtime is used to provide necessary services to a model application. Runtime provides the application executor, global context, database management, memory management and other API's to the application. Exporting the application to a working system requires two steps.

First, the final particular entities of the UAREI model nodes should be exported. This step requires rule and data query transformation into real, executable code. The transformation is performed by using a writer framework which writes JSON notation



to an executable code into a buffer step by step. The buffer is written during executable node generation.

Further, a NodeJS server application with declared action endpoints should be generated. For each action, an HTTP(s) endpoint is generated. Any external application can call these endpoints and get system results back. The calling application is responsible for generating user interfaces to communicate the feedback information to the user.

#### **4.5. Summary**

The UAREI modeling framework and its graphical visualization were described based on abstract formal model. UAREI gamification development method was defined. Formal UAREI model can be transformed into UAREI JSON format, which is a more suitable format for machine analysis. The UAREI JSON can be transformed into an executable application.

The gamification Modeling tool (GMOD), which is an application for the gamified system design, modeling and simulation using models expressed in formal UAREI modeling framework, has been presented. GMOD is using the UAREI JSON format. UAREI models can be transformed into executable Javascript applications. The tool allows for gamified system modeling, simulation and generation.

### **5. CASE STUDIES**

#### **5.1. Trogon PMS**

For this case study, a simple Project Management System has been created with the system gamification in mind. A gamification layer of Trogon PMS has been encapsulated into a module. This gamification solution has been chosen for several reasons: integration into an existing Project Management System is too complex a problem and can affect the quality of gamification; a full system implementation is necessary for the gamification module to be practically useful.

Gamification of the system has been done like this: 1) Defined game rules. 2) Allowed players to see all employee ratings. 3) Introduced badge system, which consists of several types of badges and a badge board. 4) The badge system is coupled with a level system. Every badge defines a skill and the more badges of the same type are collected, the higher the skill level received. 5) Special awards and bonuses are presented to the most skilled employees as defined by the game rules.

The gamified Trogon PMS has a leaderboard, badge board and the project forest as the main elements of gamification. Every element has its purpose. 1) The leaderboard creates competition between individual employees and allows to determine a game winner, which should be additionally awarded. 2) The badge board allows observing the skills of employees. In the badge board, the employees are ordered by the total number of badges collected. Each badge (see Figure 18) represents a skill and has its own level. The progress between skill levels is displayed as a progress bar. 3) The project forest provides the element of scalability to represent the size of different projects.

The project forest (see Figure 19) is a visualization of teamwork, which has three distinct areas: an unoccupied plot means unfinished tasks, and areas with trees

represent finished jobs. Every tree shows a different time interval it took to finish the job, while different type and complexity of a tree (Figure 20) shows that the job required more time to complete it. This creates a forest view, which a project manager can use to visually evaluate and compare the complexity of jobs performed as well as the skill of the employee.



Figure 18. Game badges and badge levels.

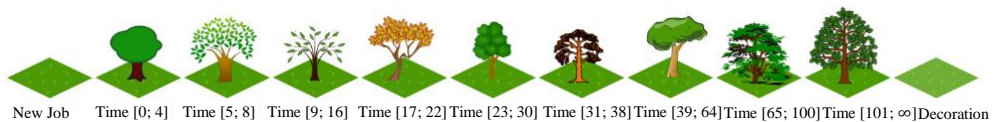


Figure 19. Elements of project forest.



Figure 20. Project forest

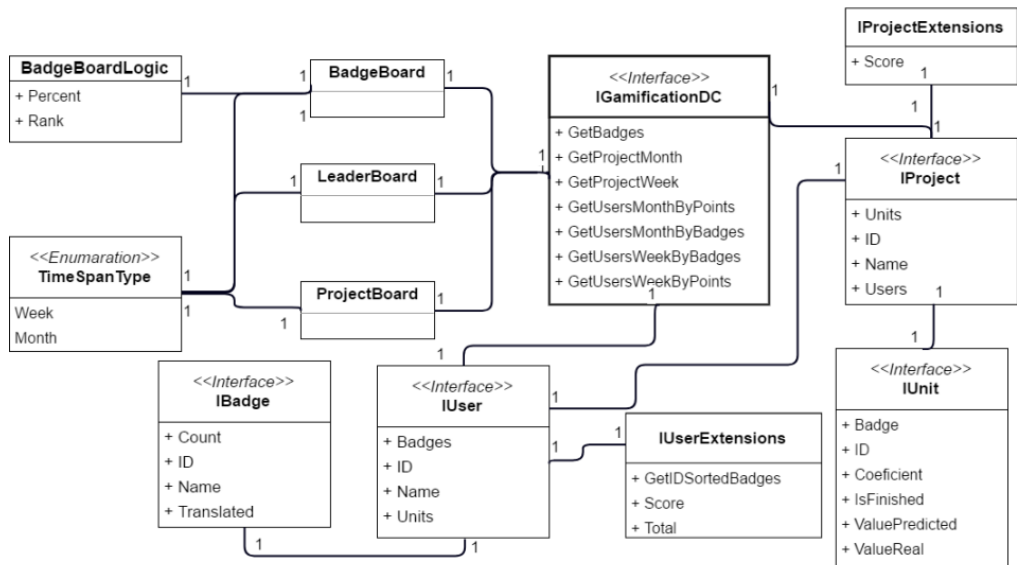


Figure 21. Gamification solution class diagram

The abstract architecture of Trogon PMS has three layers: 1) Website layer – this layer combines all visual elements into a single system. Every website is composed of

one or more solutions from the solution layer. 2) Solution layer – includes business level logic consolidated into specific solutions. Every solution targets a specific problem. 3) Database layer – this layer is shared by all solutions. Database maps data objects to specific tables in the relational database schema.

Gamification is one specific solution in the solution layer, and elements of gamification are applied in multiple website pages. The class diagram (Figure 21) shows the division of the solution into visual and logical parts. Visual and logical parts are connected by *IGamificationDC* (gamification data contract) and data object interfaces (*IUser*, *IProject*, *IUnit* and *IBadge*). Gamification data contract allows us to map any system, which implements gamification data contract. In implementation: *IProject* defines all project descriptive data; *IUser* describes all user descriptive data; *IUnit* denotes a unit of work, which connects the project, user and badge into a single system; and *IBadge* defines all badge descriptive data. *IProjectExtensions* and *IUserExtensions* introduce computational logic to *IProject* and *IUser* data objects. Computational logic is implemented as described in a formal gamification description. *BadgeBoard*, *LeaderBoard* and *ProjectBoard* are visual elements, which generate a graphical user interface for the end user to interact.

Game rules are formulated as follows. Tasks are registered and rewards for the task fulfillment are assigned. Tasks are split into atomic jobs for which a project manager can easily assign planned work time. Every job can hold a special skill badge. Employees enter information about their work results. A quality engineer/project manager checks completed jobs for defects, and awards badges. Employee points and badges become visible to all other employees. Every week the best employee is selected to be awarded.

The game flow is as follows: a software company employee receives a random stream of tasks appointed by the project manager. There are two main types of tasks: normal tasks and tasks with badges. There are nine distinct types of the badges rewarded regarding ticket specificity. Everything is translated into points. A certain number of points is awarded per task done. Based on the number of badges of the same type, a bonus is awarded. For every task completed with a badge, a user gets a 20% bonus. When five or more badges of the same type are collected, the user is awarded with an additional 20% bonus. There is a quality element for the tasks completed. If the task fails to pass Quality Assurance, a badge can be removed.

## **5.2. eLearning model for programming contest**

Like it or not, traditional methods of teaching are out of favor. People are bored of lectures, textbooks, and the things called eLearning are passive electronic versions of more or less the same typology. Those of us who teach must provide goal-oriented and engaging tools.

Lithuanian pupils are invited to participate in an online programming learning contest and thus, an eLearning environment set-up. Environment offered students access to tutorials and exams. Tutorials are not mandatory if the user wanted to participate in the contest. Exams were required to be done for students to get a certificate issued by a university.



Figure 22. Informik Environment

Finishing exams and tutorials gives user points (0-50 for tutorial and 0-100 for exam). The top 10 are displayed on the leaderboard and ratings, which show user points and badges, are seen for everyone. Students have been awarded with multiple types of badges as ladders to the next group.




First class	Second class	Third class
		
Basic level with totally 1000 points	Intermediate level with totally 10 000 points	Advances level with totally more than 20 000 points
Leaders 18	Leaders 0	Leaders 0

Figure 23. The levels of game-based education.

This approach in using gamification for eLearning has introduced a hybrid model by taking a non-user-centric model of the learning environment and adding a reward-oriented and pattern-bound model features to improve learning engagement, sustainability, intrinsic motivation and, as a result, academic performance. To test the effectiveness of this approach, the eLearning environment has been set up, and pupils from Lithuanian schools have been invited to participate in an online programming learning contest.

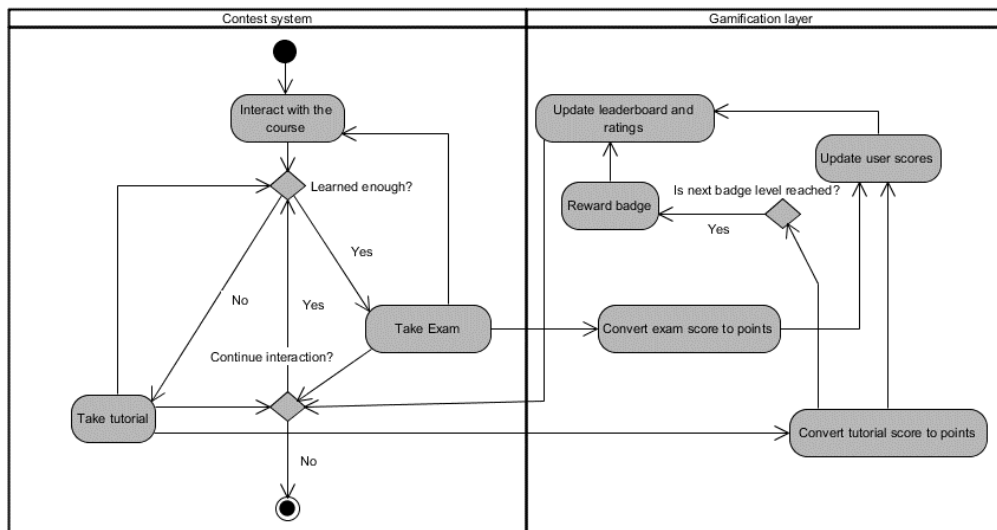


Figure 24. The gamified eLearning hybrid model activity diagram

Our hybrid model takes a non-user-centric programming contest, and adds a points and badges flow aiming at changing user motivation and academic performance.

Figure 24 explains how users interact with our system. The academic system of the contest is divided into two parts: contest and gamification layer. As mentioned before, the gamification layer is not mandatory for student participation in the organized contest. Every interaction with a course starts a flow where a user decides if he has learnt enough to take the exam or he needs to study more. Passing an exam or taking a tutorial leads to a decision about whether he is done interacting with the system or not. The described flow shows how the system normally works. To improve the system, a gamification layer has been added. Points received from taking exam or tutorial are converted to overall points. An updated score is announced in the leaderboard and ratings board, which help pupils about deciding whether to continue interaction with the system or not. The change of tutorial points triggers verification if a badge should be rewarded; if so, the badge is awarded.

The proposed gamified eLearning system will be tested with real users as well as using UAREI the system will be modelled and simulation will be run by GMOD tool.

### 5.3. Minority Game

The minority Game (MG) is a kind of social game with active coordination and competition mechanisms (Linde, Sonnemans, & Tuinstra, 2014). The MG has become a paradigm to study social phenomena with many competing agents. In the case of limited resources, agents taking the minority strategy are the guaranteed winners. MG has been extensively studied in the domain of statistical physics (Sherrington, 2006) and in various social and economic systems (Wawrzyniak, 2011). To analyze games, mathematical models are developed to predict and understand players in a game as well as for understanding and selecting strategies that will lead them towards a better pay-off in the future (Mazur, 2006).

The Minority Game has been studied previously as a model of market behavior (Ma, Li, Dong, & Qin, 2010). Another very popular application is in gambling theory.

Minority Game (MG) serves as a class of simple models which are able to produce some macroscopic features being observed in real financial markets (Ma et al., 2010). The MG is based on the idea that the decision of the majority is always wrong. Minority-like games occur frequently in everyday life, when an action taken by more people becomes less attractive. This occurs, e.g., in the selection of candidates in the university admission system, or when selecting a route in urban traffic systems. The MG is also related to congestion games, which can model diverse phenomena such as processor scheduling, routing, and network design (Nudelman, Wortman, Shoham, & Leyton-Brown, 2004). In these games, each agent can choose a subset from a set of resources, and agents' costs depend on the number of the other agents using the same resources. A congestion problem arises whenever there is a competition for a limited resource and the lack of coordination among users how to exploit it (Bottazzi, Devetag, & Dosi, 2002). The solution of many problems provided by the MG model is important to the sustainable development of many aspects of the society such as sustainable exploitation of resources, sustainable development of infrastructure and transportation (Ancel & Gheorghe, 2015), environmental efficiency and sustainable development of financial markets (Tanaka-Yamawaki & Tokuoka, 2006; Yuizono et al., 2014).

The classical MG is defined as follows (Challet & Zhang, 1997). The MG is played with an odd number of agents  $N$ . Each agent  $i$  can choose between two possible actions: to use the resource – represented by  $1$  – or not to use it – represented by  $0$ . The payoff is  $+1$  if the agent is in the minority and  $-1$  if it is in the majority. To succeed in the game, the players must consider the behavior of other players when taking decisions.

Consider a population of  $N$  agents playing game  $G$ . Game  $G$  consists of many game rounds  $g_j$ . Each agent has state  $S$  assigned to it. An agent is assumed to repeatedly choose between a finite number of alternatives (or actions, or options)  $x_i$ ,  $i = 1, \dots, N$ . Each alternative is associated with the result of a game round described by the win function and reward  $r_i > 0$ .

The principles of MG have been formulated in (Challet & Zhang, 1997) as follows: 1) Competition for limited resources: not all agents can win at the same time. 2) Behavior is good only with respect to other agents' behavior. 3) Good behavior may become bad when other agents change their behavior. 4) Agents try to predict the next winning choice, which is defined only by their own choices.

We begin by first introducing the notation and the terminology used:

- Agent: A player of the game that makes decisions based on its strategy. The number of agents that participate in the MG is  $N$ . The agent is indexed by integer  $i$ .
- Choice: An action of the agent. Choice  $C$  has two possible values:  $-1$  or  $+1$ . The total number of choices are  $N$ . In the game, the choices can be seen as a sequence of choices where  $C_n$  is the choice of  $n$ -th agent.
- Game: Every run of the MG is a “game”. The total number of games is specified as  $G$ .
- Minority Choice: The winning outcome of the game in the MG. Formally, the minority choice in a game is defined by:

$$o = \begin{cases} 1, & \sum C_i < \frac{n}{2} \\ 0, & \text{otherwise} \end{cases} \quad (8)$$

- Strategy: A set of rules of a player, which take the previous minority choices as inputs, and governs the choice of future individual actions of a player (Walsh, Das, Tesauro, & Kephart, 2002). Strategy  $S$  maps each possible combination of previous winning actions to action  $a_i$  to be taken next by agent  $i$ .

The following extensions (variants) of the MG have been defined (see Table 12).

Table 12. Variants of Minority Game

Variant of MG	Description	Ref.
Variable payoff	The payoff is $N-k$ if the agent is in the minority and $k-N$ if it is in the majority	(Li, VanDeemen, & Savit, 2000)
Extended memory	The agents can store the last $m$ actions of all their opponents	(R. M. Araújo & Lamb, 2005)
Coalition-based	The agents can organize coalitions (cartels) to share their payoffs. For example, two agents can guess oppositely and share they payoff +0.5 each.	(Sysi-Aho, Saramäki, & Kaski, 2005)
Limited resources	Each agent has only limited resources to make a bet. If it loses all resources, it loses life. If $bet = B$ , then payoff is $+B$ in a minority group, and $-B$ if in a majority group	(Xie, Wang, Hu, & Zhou, 2005)
Ternary voting	Each can choose between three possible actions: either to use the resource – represented by 1 – or not to use it – represented by -1, or abstain, represented by 0. In the case of abstention, the payoff is 0 disregarding the results of voting	(Chakraborti et al., 2015)
Traitor	Agents can choose to sell their votes or to buy the votes from another agent prior to each voting step	(Greenwood, 2009)
Local Minority Game	Each agent plays the MG only with his immediate neighbors	(Moelbert & De Los Rios, 2002)
Super agents	There is a small number of agents above the rules, who always win	(de Almeida & Menche, 2003)
Three game	Only three agents participate in the MG	(Chmura & Güth, 2011)
Mix Game	There are two groups of agents: one group plays the minority game and the other plays the MG	(Gou, 2006)
Grand Canonical Minority Game	A variable number of active traders at each time step	(Johnson, Jefferies, & Hui, 2003)

Hereinafter, we focus on the variable pay-off, coalition-based, and ternary variants of the MG. In multi-agent systems, coalitions allow to promote cooperation of agents aiming to improve their performance, or increase their benefits, with applications in e-business (He & Ioerger, 2006).

Variable payoff MG (VPMG) is important in studying emergent behavior in complex systems in real-world social and biological systems, which depend upon resources which increase or decrease in various ways as the size of the minority group changes (Li et al., 2000). In general cases, there may be various kinds of rewards and the pay-off may depend on the size of the minority group.

In ternary voting MG (TVMG), a third option is added for the decision of each agent: abstention. TVMG is important in decision theory with applications in political science (Felsenthal & Machover, 1997). Choosing a third option prevents a player from winning, but also from losing the game. Ternary voting introduces more options for bargaining (therefore, cooperation) in search of common agreement over a set of feasible alternatives.

In the coalition-based MG (CBMG), when a group of players agree to cooperate, they gain an advantage over other players. We consider the advantage obtained in CBMG by a coalition sharing their state. There can be two types of coalition: equal and unequal. In the case of equal coalition, the prize is shared into equal parts by the

members of the coalition. In the case of unequal coalition, the prize is shared into unequal parts using the ration defined by the coalition agreement. The winning strategy is to enter the most advantageous coalition agreement that guarantees the most generous pay-off. The game is transformed to the auction game, where players bid to each other for the most advantageous offer.

```

ALGORITHM: CoalitionGameRound
BEGIN
  FOREACH player from players
    IF player is in coalition
      Select best offer
      IF offer is better than current coalition
        Leave current coalition
        Join player with best offer
      ELSE if player is alone
        Bid other players for coalition
        Accept best offer and enter the coalition
      ENDIF
    ELSE
      IF player wants to join a coalition
        Join player with best offer
      ENDIF
    ENDIF
  ENDFOREACH
  Play canonical Minority Game
  Share rewards
  Generate leaderboard
END

```

Figure 25. Algorithm of the coalition-based Minority Game

The coalition game algorithm is defined in Figure 25.

What is common to the analyzed extensions of the MG is the influence of cooperation factors on the results of the game. To succeed in the game, the players must cooperate with other players or at least to consider the behavior of other players when taking the decisions. So the game moves to the meta-strategy level (Kiekintveld & Wellman, 2008).

The reinforcement model in the MG is defined as follows. For each player, satisfaction points are awarded in each step for:

- Winning (the player has won in the previous round of the game);
- Leadership (the player has been listed in one of the top positions of the leaderboard);
- Advancement (the player overtook competitors in the previous move);
- Achievement (the players has achieved the best result in some record, e.g., was in the smallest minority group);
- Power (the player, whose decision more often has decided the outcome of the game round).

Hereinafter, consider winning as the simplest variant of the reinforcement. The aim is to define a reinforcement system such that the agents would continue playing the game for a longer period. We assume that an agent takes the decision to continue playing or to exit the game based on his inner state. To be precise, let's assume that the agent can be in either the positive state (engagement) or negative state (frustration). If the agent is engaged after the previous round of the game, he takes the decision to



continue playing the game. Assume that an agent is in the state of engagement if it is not in the state of frustration, i.e., the states are mutually exclusive. An agent is in a state of frustration if he feels that he is not rewarded enough for his efforts. Being in the state of frustration increases the chance of leaving the game.

### 5.4. OilTrader

OilTrader is a game developed to model the influence of the reinforcement model on player’s decision to continue or leave the game. OilTrader is a market simulation game which allows to trade shares of oil for money or to buy oil shares. The game serves as an example how real-world markets would behave if there were no external influences. The interface of the game is presented in Figure 26.

OilTrader is a simulator which allows for users to experience simplified market conditions while trading the digital shares of the fantasy company OilFund. It involves seeing historical game outcomes and trying to predict outcome of the next round. The game consists of rounds, each thereof takes 15 seconds. Each player starts with 500 shares and 500 dollars. In each round, a player decides to sell or buy the OilFund shares, or not to do any trades in that round. Only a single trade can be done in a single round. The user sees four sections in the game. At the top, he sees his money and the OilFund shares. In the left column, the user sees trading controls and a round timer. Below that, he sees trade history data and the impact on his money or shares the trade had.

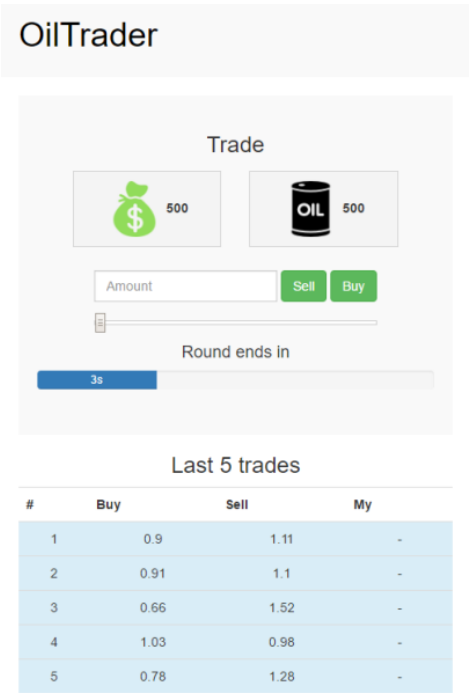


Figure 26. Schematic diagram of the game.

The physical aspects of the game (Figure 26) are comprised of tokens. Tokens are divided into the following three types:

- Oil tokens: cylindrical markers representing player's ownership of oil.
- Money tokens: sack-shaped markers representing player's ownership of money.

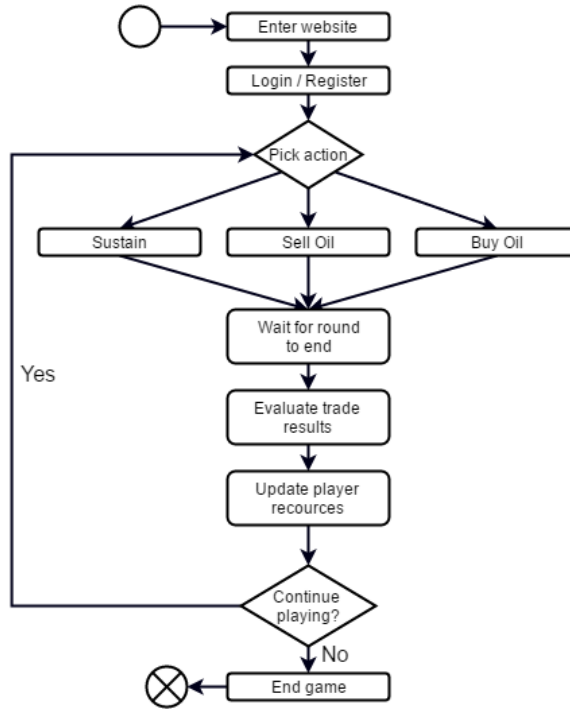


Figure 27. Flow diagram for game steps.

Each player starts by entering the game website. Next, he registers/logs in to the game. From the beginning, the player needs to pick an action for the current round. He can sustain, sell or buy oil. The player chooses an action and how many oil shares he wants to sell or how much money he is willing to spend to buy oil shares. After deciding, he waits for the round to end. The trade is evaluated determining the seller to buyer ratio. Using this ratio, player resources are redistributed based on the Minority Game logic. Finally, the player can decide to leave the game or continue to play the next round. Figure 27 shows the steps of the game as follows:

- Pick an action;
- See the results of the game round;
- Take a decision to play or not to play the next round.

## 6. EVALUATION OF THE GAMIFIED SYSTEMS

### 6.1. Gamified system evaluation

#### 6.1.1. Visual evaluation of the gamified systems

WCAG 2.0 (Reid & Snow-Weaver, 2008) is a standard method for determining accessibility of a web interface. There are two ratings described in WCAG 2.0: the AA rating is assigned when contrast is  $>4.5$ , and AAA is assigned when contrast is  $>7$ . Usually the WCAG 2.0 requirements are used for text only, but in this case, most of the information is presented in images, therefore, we extend these rules on graphical images. WCAG 2.0 evaluation scheme:

1. If the number of colors conforming to WCAG 2.0 contrast requirements is larger than the number of non-conforming colors, the interface is WCAG 2.0 compliant.
2. Else if the number of colors conforming to WCAG 2.0 contrast requirements is smaller than the number of non-conforming colors, but not by more than 50%, then the interface has small problems, which, if resolved, would make the interface WCAG 2.0 compliant.
3. Else the interface is non-compliant with WCAG 2.0.

If interface is compliant with WCAG 2.0 then:

1. If the AAA rating colors dominate, then interface is WCAG 2.0 compliant.
2. If the AA rating colors dominate, then interface is WCAG 2.0 compliant.

The following notation can be used to describe the interface compliance:

$$\text{WCAG 2.0 } \langle X\% \text{ AAA, } Y\% \text{ AA-}, Z\% \text{ AA} \rangle \quad (9)$$

where, X, Y and Z are percentage value of the AAA, AA-, and AA rating complying colors.

Let's take the previously discussed Trogon PMS System and illustrate gamified user interface evaluation using WCAG 2.0. In the part of the study on color analysis, six images of the Trogon PMS interface have been analyzed:

- Dashboard page, which shows all unfinished tasks, system events and inner office communications.
- Tasks page, which displays all tasks registered in the system.
- Employee task page, which displays all tasks assigned to the employee in a Gantt graph.
- Monthly ratings page, which displays the employee's ratings for the current month.
- Monthly badge page, which displays a sorted list of all employees and their badges with skill levels.
- Monthly project forest page, which displays all project forests, which had activity under this month.

Screenshots (JPG images) of the game layer interfaces have been analyzed. For this analysis, ImageMagick has been used to manipulate images, Lea Verou color contrast tool (Verou, n.d.) to compute color contrast and define WCAG 2.0 rating, and custom script to automate the experiment.

The experiment consists of such steps: 1) We register the image of interface. 2) Using ImageMagick we generate image color histogram. 3) Using Lea Verou tool we check the contrast of all colors against the background color. The tool returns one possible ratings:

- None is received when a color pair is not WCAG 2.0 compatible.
- AA- is received when a color pair is WCAG 2.0 AA compatible only for large elements.
- AA is received when a color pair is WCAG 2.0 AA compatible.
- AAA is received when a color pair is WCAG 2.0 AAA compatible.



Figure 28. Trogon PMS monthly badge board page



Figure 29. Trogon PMS monthly leaderboard page

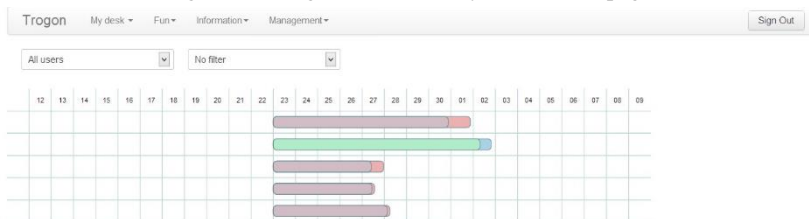


Figure 30. Trogon PMS employee task page

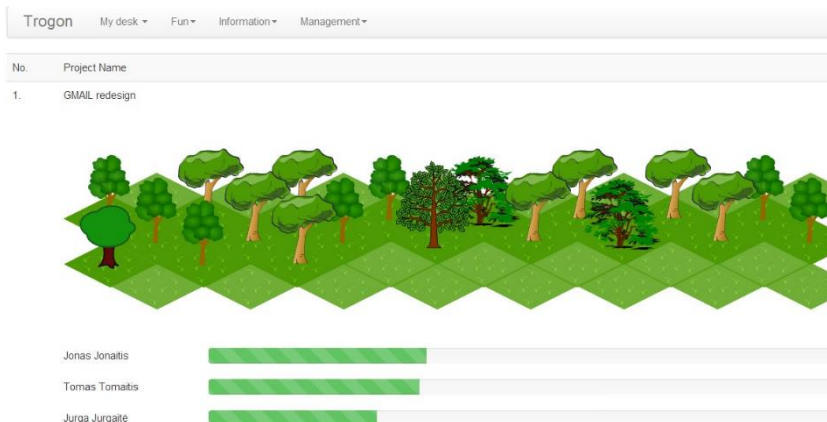


Figure 31. Trogon PMS project forest page

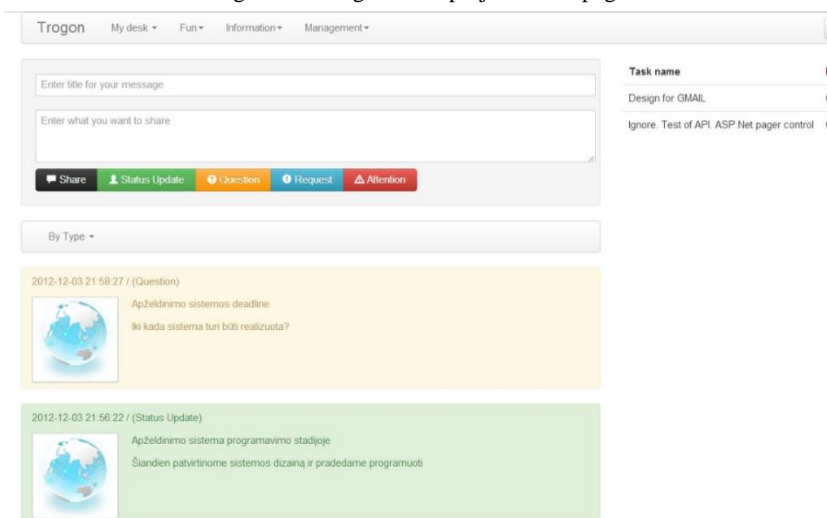


Figure 32. Trogon PMS dashboard page

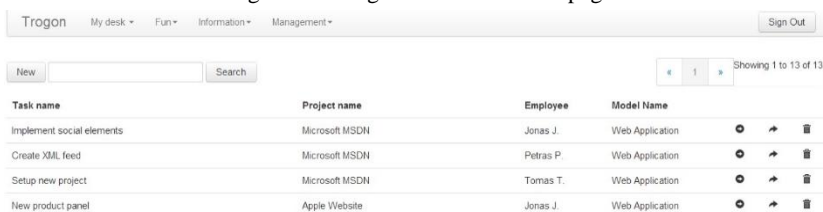


Figure 33. Trogon PMS task page

The results of WCAG 2.0 evaluations are as follows.

- Monthly badge board (Figure 28) is WCAG 2.0 compliant.  
WCAG 2.0 <AAA(48%), AA(23%), AA-(29%)> (10)
- Monthly leaderboard (Figure 29) is not WCAG 2.0 compliant but with small changes compliance could be achieved.
- Employee's task page (Figure 30) is not WCAG 2.0.
- Monthly project forest (Figure 31) is WCAG 2.0 compliant.

WCAG 2.0 <AAA(20%), AA(36%), AA-(44%)> (11)

- Dashboard page (Figure 32) is not WCAG 2.0.
- Task page (Figure 33) is WCAG 2.0 compliant.

WCAG 2.0 <AAA(69%), AA(13%), AA-(19%)> (12)

### 6.1.2. Evaluating gamified systems using SUS

To rate the usability of gamification System Usability Scale (SUS) (Brooke, 1996) methodology can be used. SUS already could be considered an industry standard for rating system or product usability. The main benefits of applying SUS are as follows. 1) A very small number of respondents. Even if the number of respondents is low, accurate results can be obtained. 2) A small number of questions allows a fast and efficient way to gather opinions. 3) A questionnaire can be used for the system, product or module usability assessment. A drawback of using SUS is that it focuses on pragmatic quality.

Normally SUS consists of ten questions (statements), which are divided into five question (statement) pairs. In a pair, both questions ask the same question, but one from a positive side and the other from a negative side. The SUS score is computed using such a methodology: every answer scores from 0 to 4 points. Point scale is from 1 to 5. Every question points are computed by subtracting 1 from the chosen scale value. Score scale of odd questions 1, 3, 5, 7 and 9 is from 0 to 4. Score scale of even questions 2, 4, 6, 8 and 10 is from 4 to 0. The final score is obtained by multiplying score by 2.5. The total SUS score is from 0 to 100.

SUS can be used to evaluate usability of Trogon PMS, a questionnaire consists of the evaluation of game elements in Project Management System; data tables; first and second round views (ratings, badges and project forest).

The respondents are asked to answer these statements:

1. I think what most people easily would learn game rules.
2. For me the game rules looked too difficult.
3. For me gameplay elements looked easy to understand.
4. I think what I would need some expert help to fully understand gameplay elements.
5. I would like to have a possibility to always view the leaderboard.
6. The leaderboard looks too complex for me.
7. I can easily understand the role of a badge board in this system.
8. I would need a lot of learning before I fully understand the badge board role in this system.
9. I think that project forest is easy to understand.
10. I think that project forest is highly imprecise.

Every pair of questions evaluates a part of system gamification and the whole questionnaire evaluates usability of the entire system. Every pair of questions has evaluated different parts of game elements: statements 1-2 ask in relation to usability evaluation for game rules. Statements 3-4 ask for evaluating gameplay elements. Statements 5-6 ask for evaluating leaderboards. Statements 7-8 ask for evaluating badge board. Statements 9-10 ask for evaluating project forest.

In the questionnaire, additional questions have been asked to provide information about the respondent for better data analysis:

- Your gender.
- Your age.
- Do you specialize in IT sector?
- Comments.

Let's look at the sample analysis of Trogon PMS with SUS questionnaire. 60 participants were asked to participate in the survey, and 30 participants filled the questionnaire form. The main group of the respondents were aged from 18 to 35 years. This age interval is the best suited for a gamification questionnaire, because this age group constitutes the largest player group. 22 men and 8 women participated in the survey. Work of 23 out of 30 participants is directly related with Information Technology (IT) systems.

Every SUS question is evaluated from 0 to 10 points and every element is covered by two questions. Therefore, every gamification element can receive from 0 to 20 points. The gamification of the entire system can receive from 0 to 100 points. To evaluate gamification qualitatively, we introduce the following intervals:

- 0-30 points – gamification is unusable.
- 31-50 points – gamification usability is poor.
- 51-70 points – gamification usability is average.
- 71-90 points – gamification usability is good.
- 91-100 points – gamification usability is excellent.

The results of SUS evaluation by gender shows that usability score average for men is  $72.27 \pm 20.97$  and  $69.06 \pm 28.53$  for women. Gamification usability by gender only has a small difference between male and female. The average difference is 3.5 points. This indicates that sex has almost no effect on gamification usability. Therefore, gamification of Trogon PMS is understood and evaluated pretty much without any differences between women and men. There is not enough data to claim statistical significance.

The difference between the evaluation of gamification usability based on the experience of users working with IT systems shows that IT group average  $75.33 \pm 19.92$  versus  $58.57 \pm 28.17$  for non-IT group. The difference in this case is equal to 17 points. The IT professionals rated the gamification of Trogon PMS at 75 points, which is 3.9 % higher than the average rating. Students test shows that the result is not statistically significant.

The entire gamified Trogon PMS was rated at 71 out of 100 points. Therefore, gamification usability is evaluated as "good". When analyzing usability evaluations of specific elements, leaderboards were evaluated as the easiest to understand, while game elements were found as the most difficult to understand.

## **6.2. Modelling gamification of Trogon PMS**

### **6.2.1. Modelled system description**

To illustrate gamification modeling, Trogon Project Management System (PMS), which has already been discussed in our previous work (Ašeriškis & Damaševičius, 2014a, 2014b), was selected. Here we demonstrate how gamification rules can be described and modelled using the proposed UAREI model as well as depicted graphically using the proposed graphical notation.

### 6.2.2. Trogon UAREI model

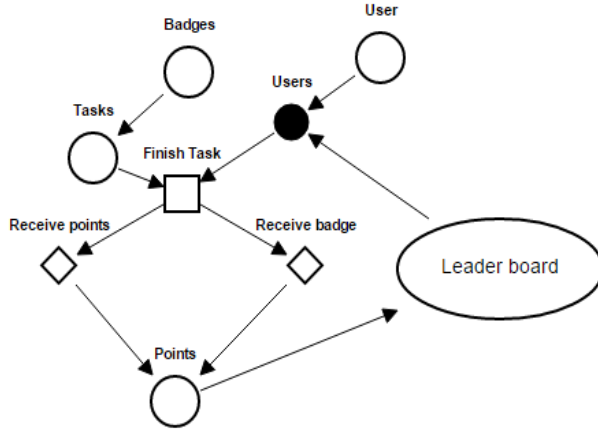


Figure 34. Visual model of Trogon PMS gamification.

The Trogon PMS gamification is defined using the UAREI model as follows:

$$G_{TROGON} = \{\{U_{employee}\}, \{A_{finish\ task}\}, \{R_{recieve\ points}, R_{recieve\ badge}\}, \\ \{E_{user}, E_{badges}, E_{tasks}, E_{points}\}, \{I_{leaderboard}\}\}$$

here:

- $U_{employee} = \{\{L_{finish\ task}\}, S_{random}\}$
- $A_{finish\ task} = \{\{L_{recieve\ points}, L_{recieve\ badge}\}, S_{random}\}$
- $R_{recieve\ points} = \{\{L_{points}\}, r_{recieve\ points}(C, M) = 5\}$
- $R_{recieve\ badge} = \{\{L_{points}\}, r_{recieve\ badge}(C, M) =$ 

$$\left\{ \begin{array}{l} \sum_i^{E_{points_{B_i}}} (E_{points_{s_i}} \cdot 1.2) + r_{recieve\ points}(C, M) \cdot 1.4, \text{ if } E_{task_i} \xrightarrow{badge} B_i \text{ and count}(E_{points_{B_i}}) = 5 \\ r_{recieve\ points}(C, M) \cdot 1.4, \text{ if } E_{task_i} \xrightarrow{badge} B_i \text{ and count}(E_{points_{B_i}}) > 5 \\ r_{recieve\ points}(C, M) \cdot 1.2, \text{ if } E_{task_i} \xrightarrow{badge} B_i \text{ and count}(E_{points_{B_i}}) < 5 \\ 0, \text{ if } E_{task_i} \xrightarrow{badge} \emptyset \end{array} \right\}$$
- $E_{user} = \{S_{user}, \{D_{John}\}, \{L_{Users}\}\}$
- $E_{Badges} = \{S_{Badges}, \{D_1, \dots, D_9\}, \{L_{Tasks}\}\}$
- $E_{Tasks} = \{S_{Tasks}, \{D_{B_1}, \dots, D_{B_9}, D_1, \dots, D_4\}, \{L_{finish\ task}\}\}$
- $E_{Points} = \{S_{Points}, \{\emptyset\}, \{L_{leaderboard}\}\}$
- $I_{leaderboard} = \{\{L_{users}\}, Q_{leaderboard}\}$

Here  $S_{user}, S_{Badges}, S_{Tasks}, S_{Points}$  define data schema.

The model of gamification of Trogon PMS using the UAREI modelling language is given in Figure 34. The model contains:

- Entities:  $E_{user}$  – all system employee,  $E_{Badges}$  – types of badges,  $E_{Tasks}$  – the tasks which can be completed by employees,  $E_{Points}$  – points gained by users.
- Users ( $U_{employee}$ ) node which is a starting point for interaction with the system.



- System has only a single action ( $A_{finish\ task}$ ) which is triggered by system users when a task is completed.
- System has two main rules: Points rule ( $R_{recieve\ points}$ ) describes normal behavior how a user receives the points for a completed task, and Badge rule ( $R_{recieve\ badge}$ ) describes how a user gets points for finished tasks which have badges associated with them.
- User feedback loop is finished by leader board interface ( $I_{leaderboard}$ ), which gives relevant feedback to the user.

### 6.2.3. UML model of Trogon

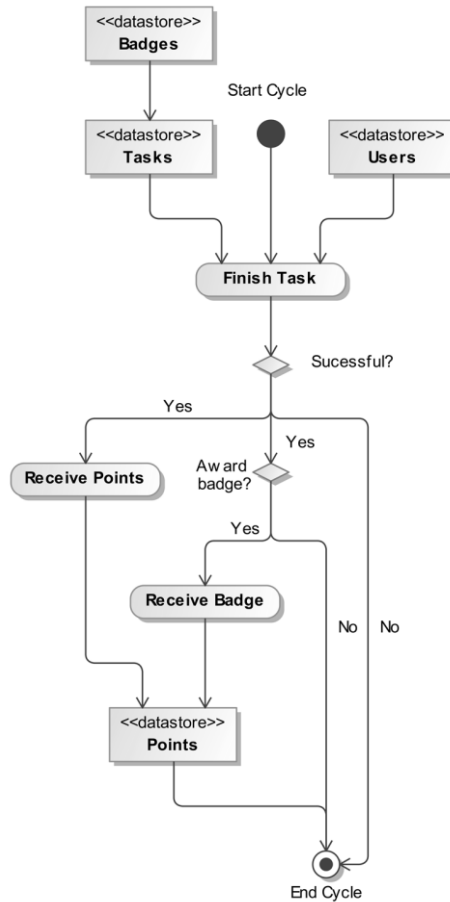


Figure 35. Gamification model of Trogon PMS specified using UML activity diagram

For comparison, the UML diagram, which represents the same logical flow, is given (see Figure 35)

UML model works as follows. Each round starts and the user is able to complete a task with an associated badge. If the employee has been successful in receiving points, he gets a badge awarded to the task. Results are stored in the database. If the

employee is not successful in finishing the task, he does not deserve a badge after finishing the cycle.

#### 6.2.4. Trogon in Machinations

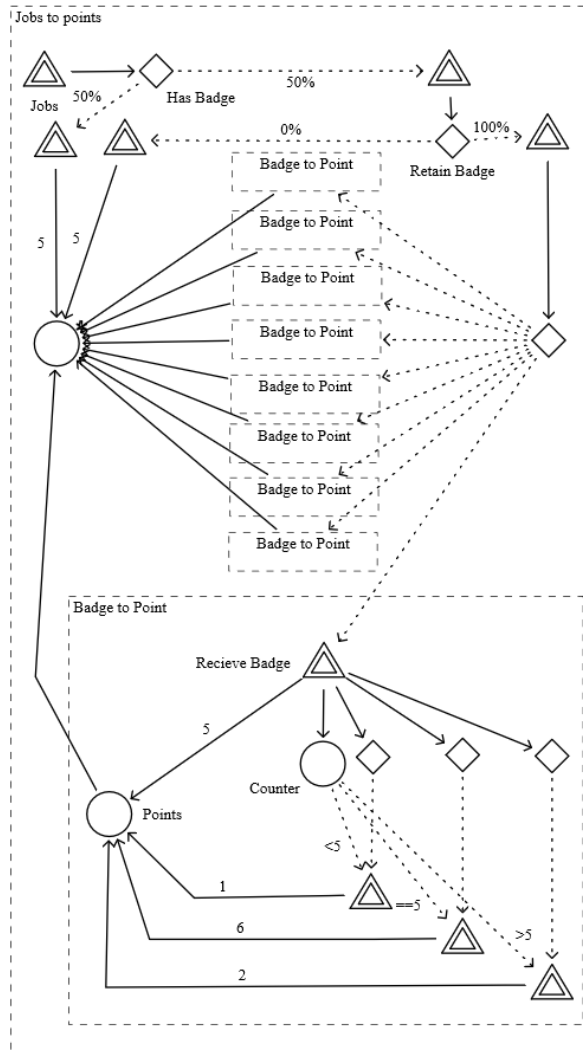


Figure 36. Gamification model of Trogon PMS specified using Machinations.

For comparison, the same model has been described by Machinations visual notation (Figure 36).

Machinations model consists of three parts: jobs and badge generators, badge counters and points pool. The most complex parts of the model are:

- Badge to point counter is implemented in a complex node configuration, which indicates it is difficult to generate rules with complex logic in machinations.

- The same badge to point counter must be repeated for each of the nine badge types which indicates that the model cannot use real-life data.

### 6.2.5. Model comparison

As the UAREI model is described using the elements of the graph theory, it is possible to use graph metrics to evaluate its visual complexity: number of nodes  $N$ , number of links  $E$ , and McCabe Cyclomatic Complexity defined as

$$M = E - N + 2P \quad (14)$$

where  $P$  is the number of independent paths in a graph.

Table 13. Visual complexity of Trogon PMS models.

Complexity metric	UAREI model	Machinations model	UML activity model
Number of nodes	9	90	11
Number of links	10	153	13
McCabe Cyclomatic complexity	3	65	4

The complexity of the UAREI and Machinations gamification models of Trogon PMS is summarized in Table 13. The comparison results show that the UAREI model is significantly less complex than its Machinations counterpart.

### 6.2.6. Evaluation

For comparative evaluation, we use the Machinations visual language (Joris Dormans, 2009). As comparison criteria, we use the most important problems / attributes in gamification modeling as criteria for the comparison.

The game rules are supported in both UAREI and Machinations. The main difference is that Machinations only allow to build a logical structure to imitate the “rule” concept. UAREI natively supports the rule concept. A rule in the model holds the logic inside it and does not disclose logic in model visualization. This is the main difference between these two modeling tools. The largest problem in Machinations is that model become too complex if one tries to model real-world systems. In UAREI, most of the game logic is encapsulated in rules which decreases model complexity.

Both modeling frameworks support user-centric modeling. However, in Machinations, every user behavior model has a separate copy of the model. UAREI supports multiple users working with the same model in parallel. Machinations currently support logical attributes which describe user behavior. UAREI as part of its UAREI JSON description contains user behavior descriptions for simulations.

Machinations is based on the economic functions and the resource concept. UAREI intrinsically focuses on the real data entities which carry more information. UAREI has the “context” concept which is carried through the model execution flow. From an abstraction point of view, the context concept of UAREI is like Machinations resource concept, only carrying more complex information.

UAREI supports real world data entities that allow mapping into software domain. UAREI separates actual data from the actual model. Usually in software engineering this is a common way to ensure data-program separation, the same concept is encapsulated into UAREI. Machinations does not have a concept of data.

Machinations does not have any model transformation capabilities and it was never designed for this aim. On the other hand, UAREI is designed for transformation

into an executable code. The rule logic is written in a meta-language which is processed into an executable Javascript code. Other parts of the model are executed using a simulator.

Both UAREI and Machinations have minimal analysis tools which allow to view model data. In Machinations, one can view “pool” changes over time. In UAREI one can see interface data change over time.

UAREI has a native feedback loop in the system. The modeling framework is designed to ensure feedback to model users. In Machinations, it is up to the designer to set up such a loop to model user behavior during simulation.

Both modeling frameworks do not support reusability. However, Machinations has support for importing parts of models from separate files. UAREI tools have not been developed yet.

UAREI has been designed for specifying gamification of the systems at a high level of abstraction. Machinations is more a tool to demonstrate game mechanics in action. In Machinations, the level of abstraction depends on designers’ choice. In UAREI abstraction of a visual model is high, but the formal model part provides the designer with flexibility.

We summarize the comparison of the UAREI and Machination modelling approaches in Table 14.

Table 14. Graphical notation of UAREI modelling language

Property	UAREI	Machinations	UML
Game rules	Native support	Logical support	Native and Logical support
Visual model complexity	Medium	High	Medium
User based simulation	Able to simulate any number of users	Every simulation is a copy of the model	No simulation
Real data support	Able to use real data entities	Resources are the only data used	Able to define real entities
Data-Model separation	Data are separated from the model, so it is possible to use any dataset	Data are directly encapsulated in the model	Data are not a part of the model
Model transformation	Future work	Model has no functionality to generate an executable code	Possible to convert to code
Feedback loop	Has native support feedback loops	It is possible to simulate feedback loops directly into the model	No feedback loops
Model reusability	Does not support yet	Importing is the only functions which allows incorporating other models.	Full support
Abstraction level	Higher	Designer-dependent	Designer-dependent

Cognitive Dimensions Framework (CDF) is a common approach for evaluating visual languages (Green & Petre, 1996). The evaluation of the analyzed languages under CDF is presented in Table 15.

Table 15. Cognitive dimensions of UAREI and Machinations.

Property	UAREI	Machinations	UML
Abstraction gradient	Model itself has a single level of abstraction, but a level of details needed to specify is chosen by user. Rules and interfaces encapsulate logic.	User chooses the level of abstraction. The more details are represented, the more complex model is build. One can build reusable parts of the model.	User customizes the level of abstraction by choosing which modeling tools to incorporate.
Closeness of mapping	Straightforward model. Problems appear while transcribing form formal to JSON model.	One needs to learn how to build complex logic. It works very well if you exchange parts of logic with simplifications. Also, one needs to understand four economic function paradigms.	Straightforward modeling language which allows different levels of abstraction.
Consistency	The whole language is built on top of 6 elements. After learning these constructs, you can build any system. The hardest part is query and rule logic function writing, which need to be learnt separately.	The language itself is quite extensive. It consists of 15 different elements and a lot of settings. The hardest part is implementing out complex logic, because the model lacks programmable logic nodes.	The language of UML activity diagram used in this case of study is composed of over 20 different types of elements that allow building many concepts into the model.
Diffuseness	Six graphic elements make up the language.	17 constructs allow building almost anything one needs for game modeling.	Over 20 elements and multiple types of connections.
Error-proneness	Errors originated from the rule and query specification.	We did not find error possibilities in small models. Problems would arise with big and complex models.	Low error-proneness. The model supports aggregation difficulty can be divided.
Difficult mental operations	Writing in JSON notations at some point would build too difficult structures to follow easily.	If a model has many asynchronous operations or high number of nodes, it can be difficult to follow.	Easy language with real natural meaning. Tracing the model requires hard mental operations.
Hidden dependencies	Dependencies are clearly visible because you see all incoming and outgoing connections.	Dependencies are clearly visible, but can be more difficult to understand due to specified logic on connections	Dependencies are clearly visible.
Premature commitment	No premature commitment	No premature commitment	Need to be committed to UML to optimize benefits.
Progressive evaluation	At any point, the model can be executed if is in a valid form.	At any point, the model can be evaluated.	The model has no automated evaluation.
Role expressiveness	The system dependencies are clearly visible.	The system dependencies are clearly visible, but can be difficult to interpret.	System dependencies can be difficult to deduct.
Secondary notation	Allows only label notation.	Allows label, color notations.	Allows labels and comments.
Viscosity	Any change is not more difficult to do as initially.	Can be more difficult to restructure complex rules.	Changes might be more difficult to introduce, depends on complexity.
Visibility	It is possible to view a model until it fits on the screen. Problems occur when the model is too big to fit on the screen. JSON notation of a complex rule can be difficult to follow.	Until the model is simple enough there are no problems. Problems arise with large models which don't fit in the screen and after some point zooming out doesn't help.	Complexity is decreased by decomposition into smaller parts. In large systems, it can be quite difficult to follow the whole system model.

### 6.3. Gamification model of eLearning system

#### 6.3.1. UAREI model of eLearning system

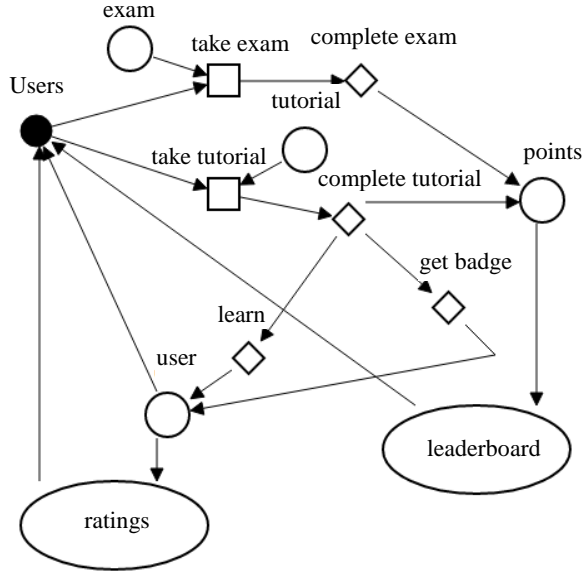


Figure 37. The gamified eLearning hybrid model for increasing participant's engagement.

Programing contest case study can be modelled using UAREI (Ašeriškis & Damaševičius, 2014b) (Users, Actions, Rules, Entities, and Interfaces) common modeling scheme for defining gamification, we can define the current study as follows:

$$G = \{U, A, R, E, I\} \quad (15)$$

Where:  $U = \{L_{users}, S_{strategy}\}$ ;  $L_{users} = \{A_{take\ tutorial}, A_{take\ exam}\}$ ,  $S_{strategy}$  - choose a random strategy.  $A = \{A_{take\ tutorial}, A_{take\ exam}\}$ ,  $A_{take\ tutorial} = \{L_{take\ tutorial}, S_{random}\}$   $A_{take\ exam} = \{L_{take\ exam}, S_{random}\}$ .  $L_{take\ tutorial} = \{R_{complete\ tutorial}\}$ ,  $L_{take\ exam} = \{R_{complete\ exam}\}$ .  $S_{random}$  - a random picking function.  $R = \{R_{complete\ tutorial}, R_{complete\ exam}, R_{get\ badge}, R_{learn}\}$ .  $R_{complete\ tutorial} = \{L_{complete\ tutorial}, r_{tutorial\ points}\}$ ,  $R_{complete\ exam} = \{L_{complete\ exam}, r_{exam\ points}\}$ .  $R_{get\ badge} = \{L_{get\ badge}, r_{get\ badge}\}$   $R_{learn} = \{L_{learn}, r_{learn}\}$   $L_{complete\ tutorial} = \{E_{points}, R_{learn}\}$ ,  $L_{complete\ exam} = \{E_{points}\}$ ,  $L_{get\ badge} = \{E_{user}\}$ ,  $L_{learn} = \{E_{user}\}$ ,  $r_{tutorial\ points}$  - random score from 0-50, accounts to user skill level.  $r_{exam\ points}$  - random score from 0 to 100, accounts to user skill level.  $r_{learn}$  - generates an increment for user skill level.  $r_{get\ badge}$  - updates user badge based on his point count.

$E = \{E_{user}, E_{tutorial}, E_{exam}, E_{points}\} = \{\{D_{user}, L_{user}\}, \{D_{tutorial}, O_{tutorial}, L_{tutorial}\}, \{D_{exam}, O_{exam}, L_{exam}\}, \{D_{points}, L_{points}\}\}$ ,  $E_{user}$  - user entity with skill level, badge,

and ID.  $E_{tutorial}$  – list of tutorials user can learn from.  $E_{exam}$  – list of exams the user can take.  $E_{points}$  – list of all user points.  $L_{user} = \{U, I_{ratings}\}$ ,  $L_{tutorial} = \{A_{take\ tutorial}\}$ ,  $L_{exam} = \{A_{take\ exam}\}$ ,  $L_{points} = \{I_{leaderboard}, I_{ratings}\}$ .  $I = \{I_{leaderboard}, I_{ratings}\} = \{\{L_{leaderboard}, Q_{leaderboard}\}, \{L_{ratings}, Q_{ratings}\}\}$ .  $I_{leaderboard}$  – display a leaderboard,  $I_{ratings}$  – display leaderboard with badges.  $Q_{leaderboard}$  – queries top ten students with highest scores.  $Q_{ratings}$  – queries all users sorted by scores including badge field.

Visually the following model is represented in Figure 37.

The formal description matches the model. Any user at random chooses to take the exam or take a tutorial randomly. More active users refer to more activity with the system. Learning is limited only to tutorials.

Therefore, user skill will increase over time. A feedback loop is closed by a user who receives feedback from the system via leaderboard and ratings.

### 6.3.2. Simulation of a Hybrid Gamification Model

**Assumptions.** Let us assume there are 100 students which have dispersed a set of programming knowledge from 0% to 15% (higher percentage means better knowledge of programming). If a user completes a tutorial, his knowledge will grow randomly by 1-5%. This skill value increases student's chance to receive a higher result from tutorials and exams. Student engagement with the system is represented by points, the higher number of points is, the more engaged the student is. It is assumed that gamification increases user engagement based on measured results in other systems (Hamari, Koivisto, & Sarsa, 2014). The higher a student engagement is, the higher impact gamification has on him. Each exam will give the user 0-100 points and tutorial respectively 0-50 points.

**Hypothesis.** More engaged students outperform less engaged students in the average result of exams.

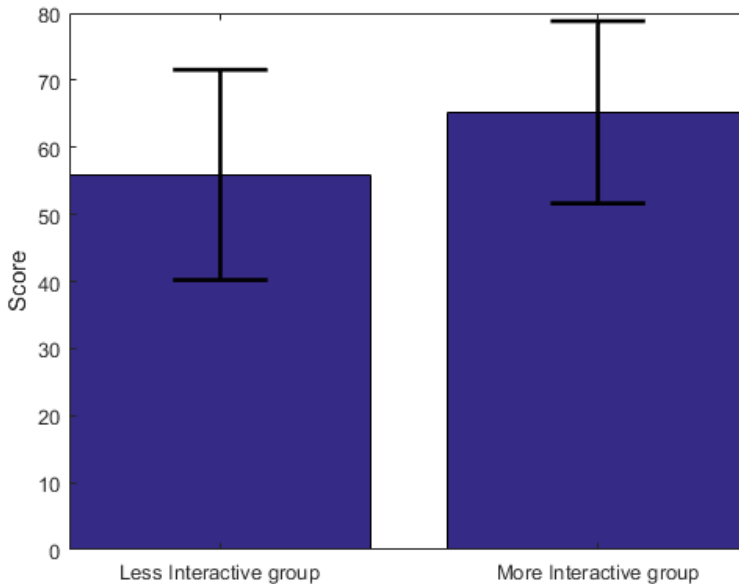


Figure 38. Simulation results.

Figure 38 illustrates one sample of simulation results. There are two groups: users who are more and less engaged with the gamification. Let us consider more engaged users who have gathered more than average points from tutorials. Simulation results are statistically significant and show the less engaged group has average exam scores  $55.9 \pm 15.7\%$  and more engaged group  $65.2 \pm 13.6\%$ . Permutation test shows what results as statistically significant ( $p = 0.727$ ). In Figure 39 we see that probabilities of the more interactive group are lower than less interactive groups, but shifted towards higher scores, which indicates that gamification does not work the same for everybody, but increases student performance.

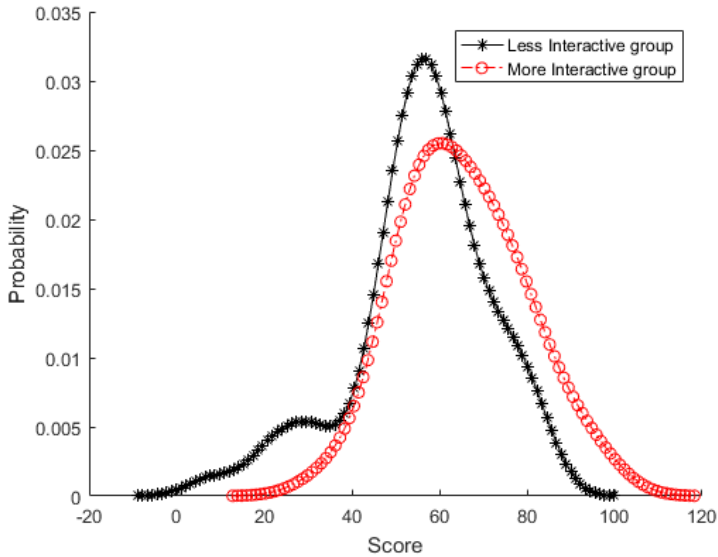


Figure 39. Probabilities assuming normal distribution

So, if a gamification of the described programming contest increases - student engagement success rate of active users will be higher. If a gamification creates enough competitiveness and increases student engagement - their results will be better. It's worth noting that different random seeding might lead to different results as it is known that different people chosen in any gamification system might produce different results.

### 6.3.3. Experimental evaluation of gamification model of eLearning system

The experiments were carried out online by delivering programming course system for HE students (Technology, n.d.). The system provides an online course on introduction in C++ programming language. The course is free online, but requires registration. Each month the system provides a set of problems, which must be solved by the following month, and solutions have to be uploaded to the system. Solutions are evaluated by real teachers, who are also registered users of the same programming course system. The evaluation range is from 0 to 100, where 0 is the lowest score and 100 is the highest score. Moreover, each student is assigned to a personal tutor, who guides the student through the whole learning process. Both tutor and student interact with each other remotely via web forum. Some of the students have an opportunity to solve additional problems. Students can use an integrated programming environment



with an online compiler and online test system. The solutions of additional problems are executed on online test system. The test system checks whether a solution passes all tests. If all tests are passed, the system considers that the problem is solved. Each solved problem is rewarded with 1 point. Points are summed up for each user. The scoreboard of the best students is announced on the homepage of the online course system. The system has a badge system to distinguish between the students with different amount of points. Students can earn up to 30 points for additional solutions.

Students were divided in two groups: the control and the experiment group, which has an opportunity to solve additional problems. 95 students were selected to participate in the experiment. A gamification group (students, who have solved additional problems) consisted of 48 students. A control group has had 47 students. Score averages of gamification and the control group were compared. Average points for an additional task and the average score of the gamification group were evaluated to determine the number of students, who were engaged by the gamification system.

#### 6.3.4. Experiment results

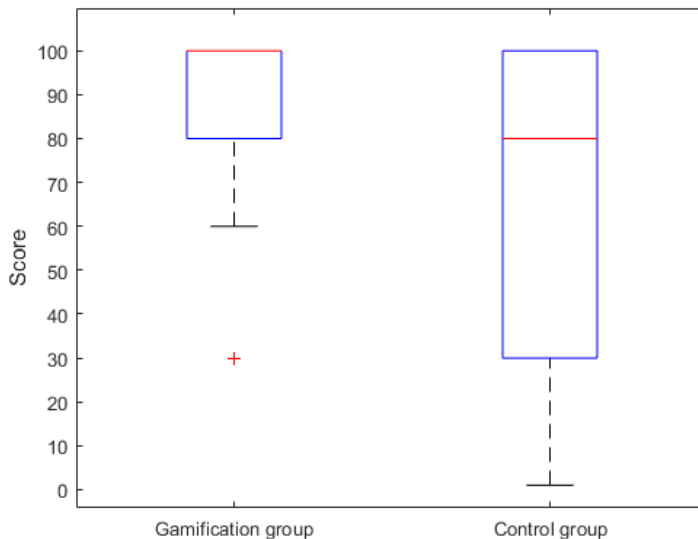


Figure 40. Box plot of hybrid eLearning model contestant results.

In Figure 40 we see the average score of the gamification group is  $83,13 \pm 23.26$ . The average score of the control group is  $66,83 \pm 29.89$ . The gamification group has shown the average score higher by 16,3. Random permutation probability is 56% which indicates that the results are statistically significant.

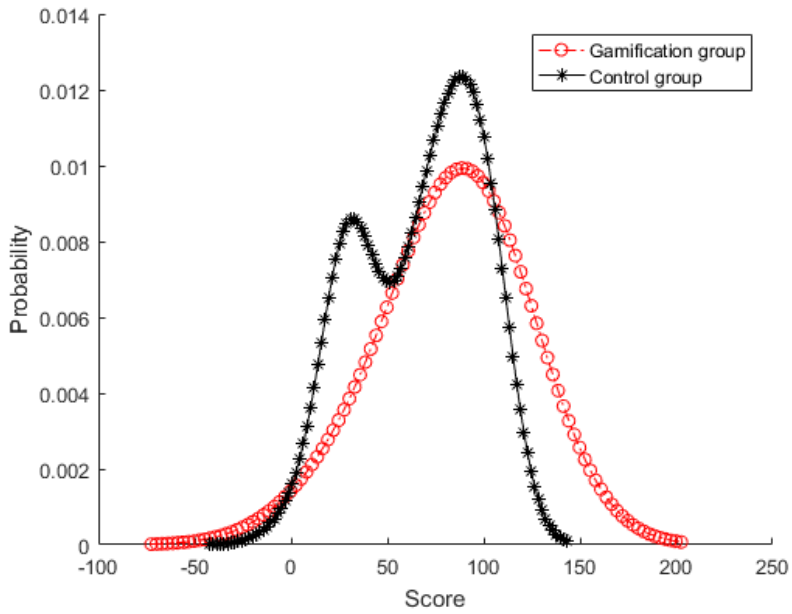


Figure 41. Probability distribution of gamified and control groups

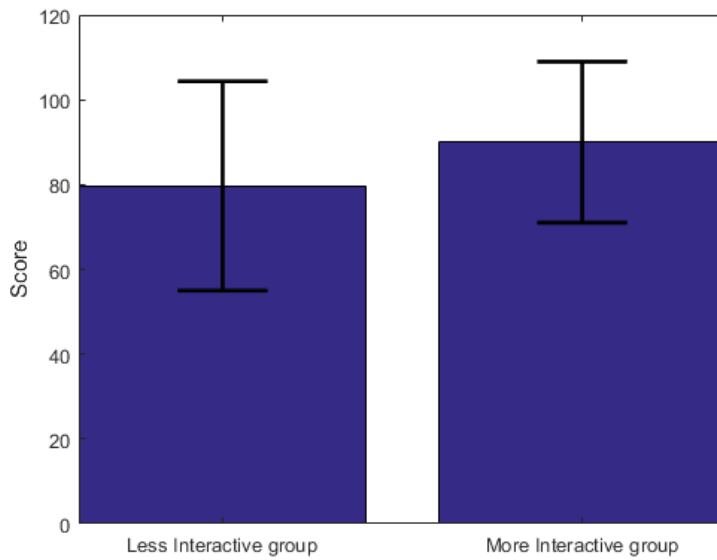


Figure 42. Learning results of enjoyment groups.

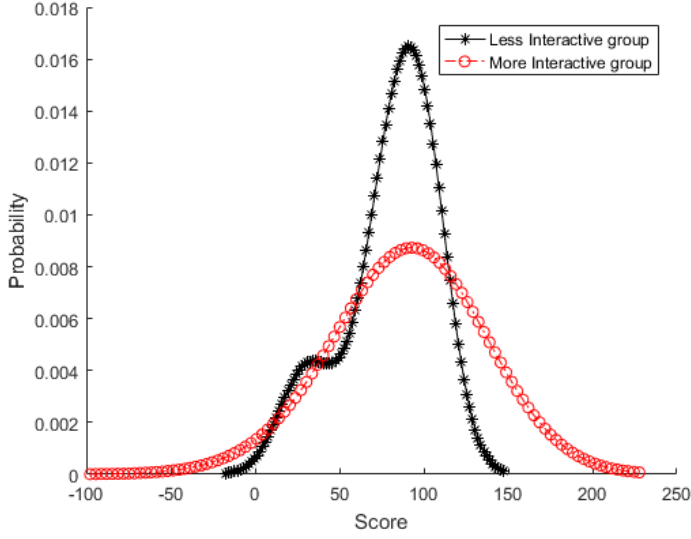


Figure 43. Probability distribution engagement groups

Probability distribution in Figure 41 indicates that gamification group probabilities are lower, which means that it does not work better for everyone. Note bimodal distribution of the control group.

In Figure 42 we see the results for less and more engaged groups. The less engaged group has an average exam result equal to  $79.7 \pm 24.7$  and the more engaged group has the average  $90 \pm 18.97$ . The results are not statistically significant based on permutation test results ( $p=0.455$ ). Even if the results are not statistically significant, they still indicate similar results observed during the simulation. Figure 43 on probability distribution indicates that higher engagement does not suggest better exam results.

## 6.4. Modelling Minority Games in UAREI

### 6.4.1. Extending UAREI for MG support

In general, user behavior can be supplemented by agent behavior. An agent first picks an action using  $a_{pick}$  function (in case of original MG definition as El Farol Bar problem (Arthur, 1994), the agent picks “go to bar” or “stay at home” based on his strategy). To map the strategy to the current model state we define a  $a_{key}$  function, which generates a memory key to reference the current situation. After the cycle ends the agent receives a call-back to  $a_{feedback}$  function to evaluate his choice.  $E_{agent\_state}$  entity stores all data relevant to agents.

In case of the classic MG, we can specify such agent description as:

$$\alpha_i = \{a_{pick}(model), a_{key}(model), a_{feedback}(model), E_{agent\_state}\} \quad (16)$$

here  $a_{pick}$ :

- On first call
  - Generate S random strategies for all possible keys.
  - Initialize strategy quality so one would be better.
- On all calls
  - Generate key using  $a_{key}(model)$  function for a current model state.
  - Return best quality strategy and take from it action for the generated key.

$a_{key}(model)$ : return M records from game win history entity;

$a_{feedback}(model)$ : if the action of the previously chosen strategy has won, then increase strategy quality by one. The function is executed in every round of the game.

$E_{agent\_state}$  has a collection of strategies and a vector of strategy quality. The user (player) behavior is defined as follows:

$$U = \{L_{user}, S_U, \alpha\} \quad (17)$$

The MG agent model has a problem because it is bound by  $N_{user} \times N_{actions}^{S^M}$  which causes performance issues when modeling large numbers of the agent in a model with large number actions where agents use large number of strategies and can remember large history. To avoid this problem by offering an alternative MG agent:

$$\alpha_{i,alternative\ minority\ game} = \{a_{pick}(model), a_{key}(model), a_{feedback}(model), E_{agent\_state}\} \quad (18)$$

here  $a_{pick}$  generates a key using  $a_{key}(model)$  function for a current model state, which returns a random action, if it is called for the first time, and the best action, if called subsequently.

Formally, MG is defined as  $G_{minority\ game} = \{U, A, R, E, I\}$ , here  $U = \{L_{users}, S_{order}, \alpha_{minority\ game}\}$ ;  $L_{users} = \{A_{Go\ to\ bar}, A_{Stay\ at\ home}\}$ ;  $S_{order}$  - pick user from order for each round;  $\alpha_{minority\ game}$  Minority Game agent list of N players with S strategies and M memory size;  $A = \{A_{Go\ to\ bar}, A_{Stay\ at\ home}\}$ ;  $A_{Go\ to\ bar} = \{S_{random}, L_{Go\ to\ bar}\}$ ;  $A_{Stay\ at\ home} = \{S_{random}, L_{Stay\ at\ home}\}$ ;  $S_{random}$  - randomly generated action data;  $L_{Go\ to\ bar} = \{R_{Record\ Option}\}$ ;  $L_{Stay\ at\ home} = \{R_{Record\ Option}\}$ .

$R_{Record\ Option} = \{r_{Record\ Option}, L_{Record\ Option}\}$   
 $r_{Record\ Option} = \begin{cases} "Go\ to\ bar", & \text{if chosen action was } A_{Go\ to\ bar} \\ "Stay\ at\ home", & \text{if chosen action was } A_{Stay\ at\ home} \end{cases};$   $L_{Record\ Option} = \{E_{Choices}\}$ ;  $E_{Choices} = \{L_{Choices}, D_{Choices}\}$  is the entity collecting all user choices, here  $L_{Choices} = \{I_A, R_{Win}\}$ ;  $D_{Choices}$  has three fields: user ID, chosen action, and game round;  $I_A$  - defines a view which groups users by choices;  $I_A = \{L_A, Q_A\} = \{\{U\}, Q_A\}$ ;  $Q_A$  - groups data from  $E_{Choices}$  by round and chosen action and counts all users in a group;  $R_{Win} = \{L_{Win}, r_{win}\} = \{\{E_{History}, U\}, r_{win}\}$ , here  $r_{win}$  - defines the winner for each round. The rule is executed once at the end of every round and returns the action which has been opted by the minority group  $E_{History} = \{L_{History}, D_{History}\} = \{\{U, I_{Leaderboard}\}, D_{History}\}$ ;  $D_{History}$  - has two fields: round number and winning action;  $I_{Leaderboard} = \{L_{Leaderboard}, Q_{Leaderboard}\} = \{\{U\}, Q_{Leaderboard}\}$ ; here  $Q_{Leaderboard}$  - computes user success rate.

In Figure 44, we can see the classic MG model represented visually in UAREI.

Using UAREI, we also can analyze different variants of the MG model. For variable pay-off MG, the visual representation of the model is the same as is given in Figure 44. The model for cooperation-based MG is presented in Figure 45.a. A new entity “Bank” has been introduced with a specialized interface to visualize the monetary situation of agents in the model. Figure 45.b represents the ternary-voting MG model. This model has all the attributes from cooperation-based MG model and action “Sustain” in addition.

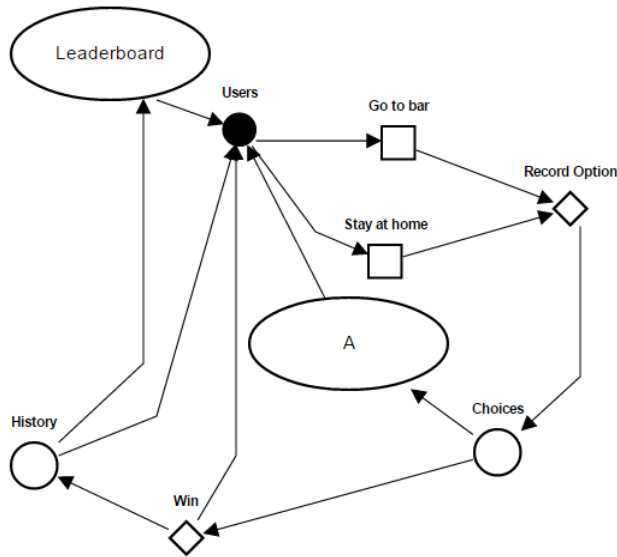


Figure 44. Minority Game model in UAREI

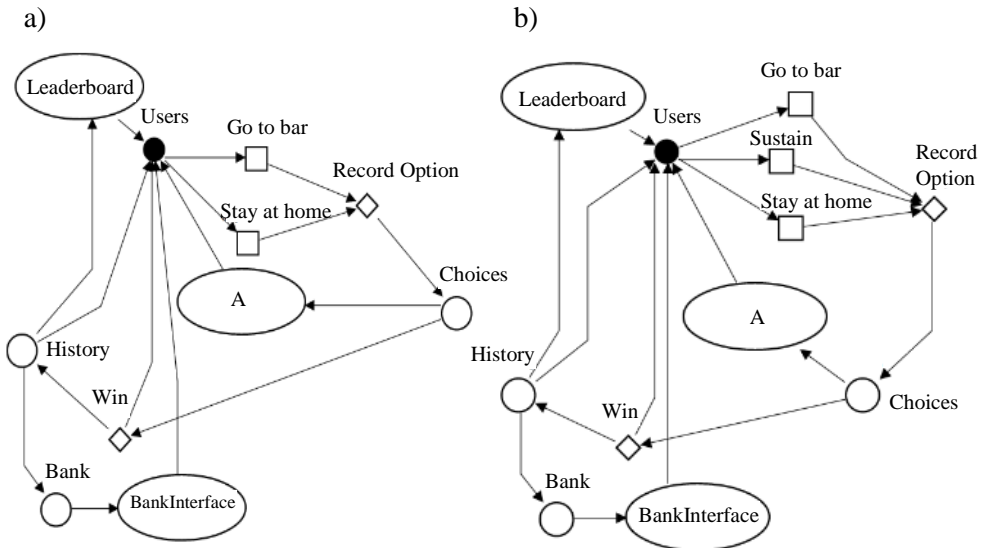


Figure 45. Model of coalition (a) and ternary voting (b) variants of Minority Game in UAREI  
The modifications of classic MG model are summarized in Table 16.

Table 16. Modelling variants of Minority Game in UAREI

Variant of MG	Change in classic MG model
Variable payoff	In this model $a_{feedback}(model)$ gives N-k if the user wins and k-N to the strategy.
Coalition-based	$\alpha = \{\alpha_{minority\ game}, \alpha_{cartel}\}$ $N_{minority\ game}$ and $N_{cartel}$
Ternary voting	$A = \{A_{Go\ to\ bar}, A_{Stay\ at\ home}, A_{Sustain}\}$

### 6.4.2. Simulation and results

The simulation uses  $N = 101$  agents with  $S = 2$  number strategies with  $M = 4$  memory size. Agent actions are represented as +1 and -1. The simulation results show the number of agents, who have taken a decision to “go to bar” or to “stay at home”. If the value of the sum of agents’ functions in a game round is equal to or below 50, then the group is the minority and has won the round.

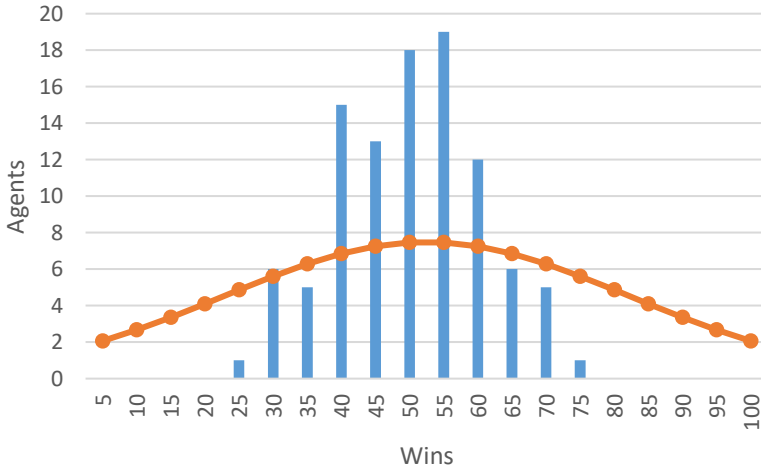


Figure 46. Histogram of wins in simulated classic Minority Game

The simulations were run with different variants of minority models defined in Table 16, including the classic MG. Observed model behavior is expressed as the winning function and defined as the ratio of wins to the number of played games in percentages. In Figure 46 we can see the histogram of winning functions after 100 game rounds in different simulations of the classic MG. We can see that the number of winning agents follows the Gaussian probability distribution (see the values of mean, std and Kolmogorov-Smirnov (KS) normality test in Table 17).

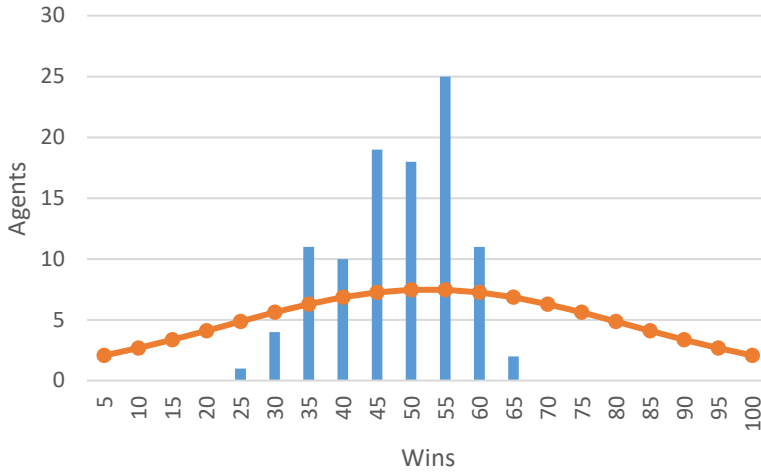


Figure 47. Histogram of wins in simulated variable payoff Minority Game

Distribution of wins for the variable payoff MG is presented in Figure 47. The size of reward is proportional to the size of a minority group, which favors the formation of small minority groups as well as allows for more rapid changes in the leaderboard of players during the game.

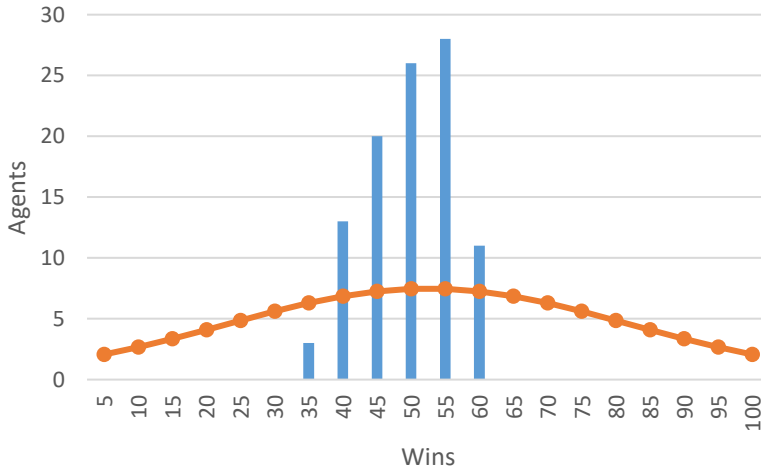


Figure 48. Histogram of wins in simulated coalition-based Minority Game

In the coalition-based MG, simulation introduces a 20-member coalition into the system. Members of the coalitions are divided into two equal groups which bid on different actions and split the reward between the members of the coalition. Each agent bids 1 point per round. Rewards are distributed equally to all players. The histogram of wins (Figure 48) shows a small shift over the random change success rate (see Table 17). Therefore, one can conclude that the introduction of coalition as a meta-game strategy into a classic MG model allows to improve the results of the game for some players.

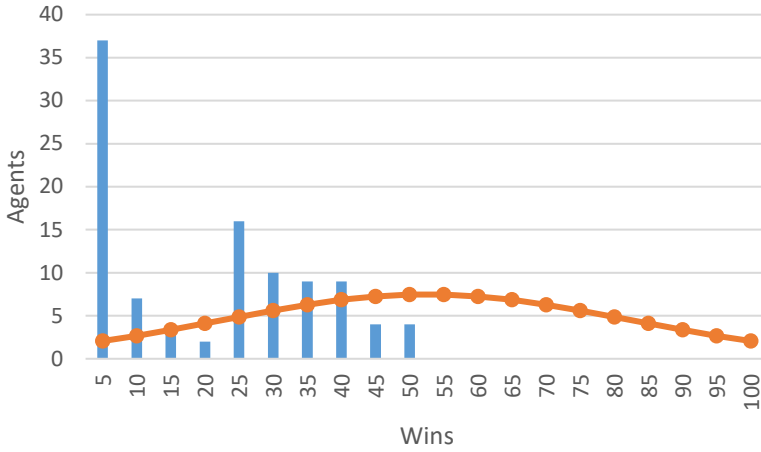


Figure 49. Histogram of wins in ternary voting Minority Game

In the simulation of the ternary voting MG (TVMG), the model introduces a third option for players to sustain from playing in their strategies. All players who choose to sustain do not participate in the current round of the game. All agents initially have 10 points each. In every round, each participating agent must bid 1 point. A player who has lost all his points must leave the game. In the histogram of wins in TVMG (Figure 49), we can see two peaks, which correspond to low performing agents and high performing agents, which is also confirmed by the results of the KS normality test (see Table 17).

To evaluate the interestingness of each variant of MG, we use the negentropy value of win function. In information theory and statistics, negentropy is used as a measure of distance to normality. The results of a coin tossing game, the simplest and the least interesting game without any strategy of playing, would have the Gaussian distribution. On the other hand, the games with uniform or constant probability of wins are equally uninteresting. Thus, the game, which differs more in terms of negentropy from the Gaussian distribution with the same mean and variance, can be considered more interesting. Such entropy-based measures have already been used for defining the concepts of interestingness and surprise of data, including that of algorithmic zero sum games (Schmidhuber, 2009). The results of the statistical analysis of win results in the analyzed variants of MG are presented in Table 17.

Table 17. Statistical evaluation of variants of Minority Game

Variant of MG	Mean	Std.	Skew-ness	KS test	Entropy	Negen-tropy
Classic	49.95	10.78	-0.04	1	0.932	0.02
Variable payoff	48.02	8.87	-0.41	1	0.838	0.03
Coalition-based	49.75	6.50	-0.29	1	0.710	0.03
Ternary voting	20.74	14.82	0.32	0	0.844	0.16

The win values for classic, variable payoff and coalition-based variants of MG are normally distributed and has acceptable asymmetry (skewness between -2 and 2). The ternary voting model departs from the normality due to the rules of the game, which throw players with poor performance out of the game. The value of negentropy, which is used to evaluate the interestingness of the game, shows that the classic MG, which has the simplest set of game rules, as the least interesting, whereas the ternary



voting variant of MG, which allows the users to abstain as well as to go bankrupt and leave the game, is the most interesting.

#### **6.4.3. Summary & the reinforcement model**

The main tools of keeping the player in the state of flow during the game are various types of rewards. The reward systems are usually multilevel systems, i.e., they are usually based on a hierarchy of different levels to attain (Dubina & Oskorbin, 2015). According to Wang et al. (Wang & Sun, 2011), there can be eight forms of reward in games:

1. Score systems such as leaderboards use numbers to mark player performance.
2. Experience point reward systems reflect player effort rather than skill.
3. Virtual item rewards have collecting and social comparison value.
4. Resources are virtual items that can be collected and used to affect gameplay.
5. Achievement systems encourage players to complete specific tasks of a game.
6. Feedback messages are used to create positive emotions and provide instant rewards in response to successful actions.
7. Animations and pictures are used as to provide a sense of fun and mark player achievement.
8. Unlocking mechanisms give players access to game content once some requirements are met.

The goals of the reinforcement systems have been summarized by the Corners of Reward model (James, Fletcher, & Wearn, 2013) as intrinsic (achieving own goals), extrinsic (succeeding in leaderboards) and social (competing with other players).

Hereinafter, the reinforcement model for games with the following elements is proposed:

- Winning: reward is provided if the player has won in the previous round of the game;
- Ranking: reward is provided if the player has excelled over his/her competitors over time and has been listed in one of the top positions of the leaderboard of winners;
- Advancement: reward is provided if the player has overtaken a significant number of his/her competitors in the previous round of the game;
- Achievement: reward is provided if the player has achieved the best result in some interesting nominations, e. g., has won over the largest number of his/her competitors;
- Luck: reward is provided on a random basis to some of the poor performing players just to increase persistence and total effort of players and incentivize them to keep playing.

These elements provide both static (momentous) and dynamic (continuous) views to the effort and contribution of players in time as well as introduce the element of randomness to allow a certain degree of uncertainty in the system. Each of these elements of reward is supported by a set of the desired characteristics as follows:

- Visibility: the rewards should be seen for other players to increase competition, gain social value and increase overall interestingness of a game.

- Fairness: the reward system is open to all players.
- Chance: the rewards should be awarded at random intervals to keep interest in the game.
- Scarcity: reward should not be a common thing in a game.
- Stability: there are agents awarded during each round.

## 6.5. OilTrader Game experiment

### 6.5.1. Experiment set-up

For simplification, it is assumed that a player can only be affected by the elements of the user interface of the game which he/she can see. A hypothesis of the experiment is that it is possible to evaluate the influence of the reward mechanism (visually represented as a leaderboard table) by using game play duration, which is different for each psychological player type.

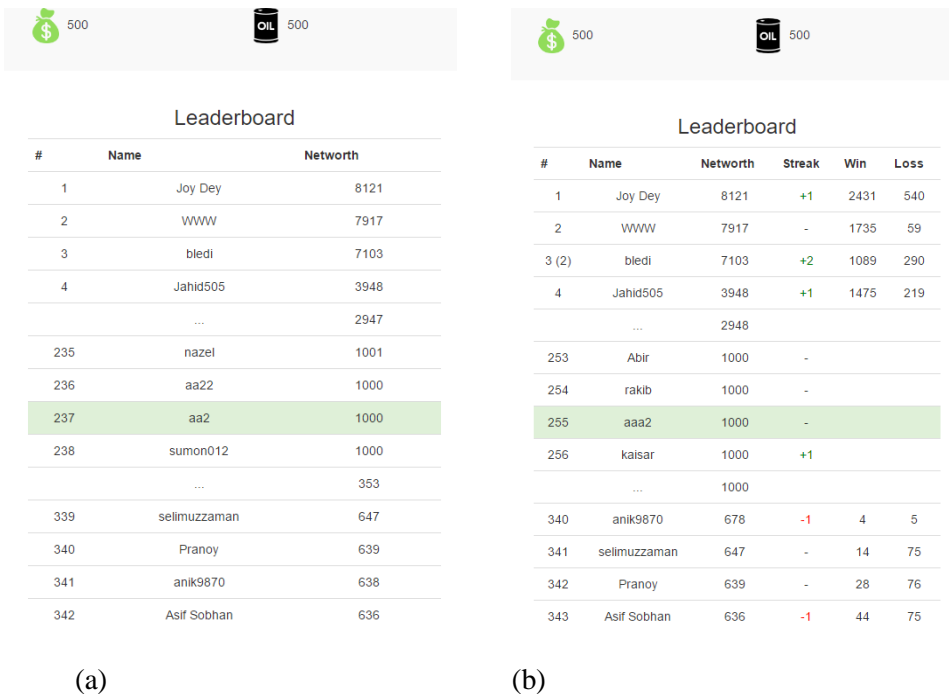


Figure 50. OilTrader leaderboard of (a) control group, and (b) experiment group (with streak, win and loss incentives)

Users will be divided randomly into two groups: the main (experiment) group and the control group. User interface of the game for the control group has the leaderboard which represents player achievement, and shows player position, net worth (shares + money) and win or lose state in the latest round of the game. User interface of the game for the experiment group has three additional metrics (streak, biggest win, and biggest loss), which represent player progress, and are aimed to incentivize the internal player reward (see Figure 50).

### 6.5.2. Purpose of the Game Experiment

This experiment has three main objectives:

- to validate the playing motivation of experimental subjects using the proposed motivation model and the HEXAD player typology (Tondello et al., 2016) & questionnaire (L. Diamond G. F. Tondello & Tscheligi, 2015);
- to identify any difference in the effectiveness of motivation-enhanced game interface between the experimental group (which was presented with used a motivation-enhancing leaderboard) and the control group (with used a basic leaderboard);
- to discuss the relationship between the HEXAD player types and player motivation to play the game longer.

### 6.5.3. Experimental Subjects

The experiment was carried out in June 2016. Using crowdsourced workers from microworkers.com (a web-based crowdsourcing platform to access the crowd which enables employers to submit individually designed tasks (Hirth, Hoßfeld, & Tran-Gia, 2011)), we set up a task to play the game and afterwards to fill in the player type questionnaire. We randomly assigned players to control and experiment groups once they created an account. In total, we enrolled 114 players who played the game. Participants in the study were mostly male (88.6%). 70.2% of the participants were between 20-30 years old. 17.5% of the participants were 30+ years old. 12.3% were younger than 20 years old. The majority of the participants play games up to 3 hours a day (67.5%) and 23.7% play more than 3 hours. Only 8.8 % of the participants do not play computer games regularly. 69.2% of the participants enjoyed the activity versus 31.8%, who said they took the task only for money.

All the participants received introductory information about the task they were asked to perform (to play a game). Then all players could start playing the game and exit from it at any time they wanted. After finishing the game, the participants were asked to complete the HEXAD questionnaire (L. Diamond G. F. Tondello & Tscheligi, 2015). The questionnaire was not mandatory, but only the results of players who completed the questionnaire voluntarily, were analyzed in this study (99 players, 86.8%).

### 6.5.4. Research Tool

To assess the motivation reinforcing aspects of the game interface, we use the HEXAD player type classification (L. Diamond G. F. Tondello & Tscheligi, 2015) which distinguishes 6 player types:

- *Socializers* are motivated by being closer to other people. They seek to create new social connections and relationships.
- *Free spirits* are motivated by autonomy and self-expression. They like to explore.
- *Achievers* are motivated by mastery and overcoming game challenges. They continuously need to improve themselves.
- *Philanthropists* are driven by altruism helping others without any reward for themselves.

- *Players* are motivated by extrinsic rewards. They are playing the game only if they expect to be rewarded.
- *Disruptors* are motivated by changes. They are willing to ‘disrupt’ the game rather by playing by its rules.

First, let us assign a player type to each player and then we evaluate the length of gameplay for each player type. To assess a player type, we employ the HEXAD questionnaire (see Table 18) (L. Diamond G. F. Tondello & Tscheligi, 2015; Tondello et al., 2016).

Table 18. The HeXAD Questionnaire (L. Diamond G. F. Tondello & Tscheligi, 2015)

Player type	No.	Items
Achiever	Q6	I am very ambitious.
	Q15	I like overcoming obstacles.
	Q20	It is important to me to always carry out my tasks completely.
	Q24	It is difficult for me to let go of a problem before I have found a solution.
	Q27	I like mastering difficult tasks.
Disruptor	Q5	I like to provoke.
	Q11	I like to question the status quo.
	Q18	I see myself as a rebel.
	Q22	I dislike following rules.
	Q29	I like to take changing things into my own hands.
Free Spirit	Q3	It is important to me to follow my own path.
	Q9	I often let my curiosity guide me.
	Q14	I like trying new things.
	Q21	I prefer setting my own goals.
	Q26	Being independent is important to me.
Philanthropist	Q2	It makes me happy if I am able to help others.
	Q10	I feel good taking on the role of a mentor.
	Q17	I like helping others to orient themselves in new situations.
	Q23	I like sharing my knowledge.
	Q28	The well-being of others is important to me.
Player	Q7	I like competitions where a prize can be won.
	Q13	Rewards are a great way to motivate me.
	Q16	I look out for my own interests.
	Q25	Return of investment is important to me.
	Q30	If the reward is sufficient I will put in the effort.
Socializer	Q1	Interacting with others is important to me.
	Q4	I like being a part of a team.
	Q8	It is important to me to feel like I am part of a community.
	Q12	It is more fun to be with others than by myself.
	Q19	I enjoy group activities.

### 6.5.5. Results

In this experiment, our hypothesis is that different player types are impacted differently by different reinforcement models of the OilTrader game. The motivation to keep playing is evaluated as the number of the game rounds played by the player. When analyzing the answers of the questionnaire, we noticed that some players filled it randomly. Following the recommendations presented in (Hoßfeld et al., 2014), we conducted the two-stage statistical analysis. The first stage tests the reliability of the players. This stage aims at creating a pseudo-reliable group of players, who are analyzed in the second stage. The unreliable player ratings are determined based on the results obtained from the HEXAD questionnaire. Only the results of the reliable players are used in further analysis. This approach is also known as pilot task and main

task (Soleymani & Larson, 2010). The number of players disqualified in the pilot stage can be as high as 60% (Soleymani & Larson, 2010).

The reliability of players' answers might be evaluated following certain steps. First, assuming that if some players belong to the same player types, their answers to questions defining this player type would be similar, while the answers to other questions would be scattered. Based on this assumption, players have been filtered out (10%, by rank), for which there was the smallest difference between standard deviations of answers to questions representing different player types. The second assumption was made that a player would choose the highest score for the answers which correspond to his player type, while for all other answers the scores would be scattered. In this case, we have removed players (10%, by rank), for whom there was the largest difference between the mean score of all answers and the largest mean score of answers for questions representing different player types.

We have assigned player types to the remaining players using the following rule: a player is assigned to a player group if the sum of answer scores to the player type questions exceeds the median of the sum of the answer scores for that player group by its Median Absolute Deviation (MAD) as follows:

$$\sum_{Q_i \in Q} S(Q_i) > \text{median} \left( \sum_{Q_i \in Q} S(Q_i) \right) + \text{mad} \left( \sum_{Q_i \in Q} S(Q_i) \right) \quad (19)$$

here  $Q_t$  – a subset of questions for a player type  $t$ , and  $S$  – score.

Similarity of the obtained player groups has been evaluated using the Jaccard similarity metric as follows:

$$J(G_1, G_2) = \frac{|G_1 \cap G_2|}{|G_1 \cup G_2|} \quad (20)$$

here  $G_1$  and  $G_2$  are player type groups.

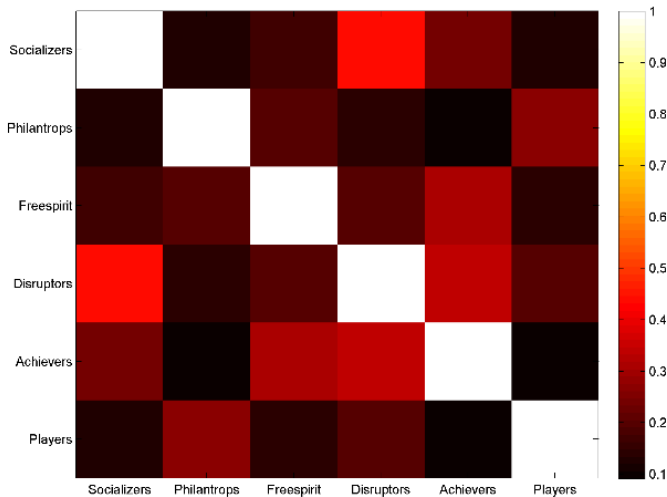


Figure 51. Distribution of players between player types

The results are presented in Figure 51. The number of unreliable players removed in the pilot stage of statistical analysis is 34 out of 99 (34.3%). It was allowed for the same player to be assigned to different player types. Mean overlap between player groups is 20.5%. In the best case, there is only 9.1% similarity (between Achievers and Philanthropists), and in the worst case there is ~44.4% similarity between two different player type groups (Disruptors and Socializers).

To evaluate the duration of gameplay, a median has been selected as a statistical measure that is more robust to outliers than an arithmetic mean. The median results show that the players of the experiment group played  $12.2 \pm 2.9$  rounds, while the players of the control group played  $10.3 \pm 2.4$  rounds (see Figure 52). The paired-sample t-test rejects the hypothesis that both data sets have equal means ( $p = 10^{-48}$ ).

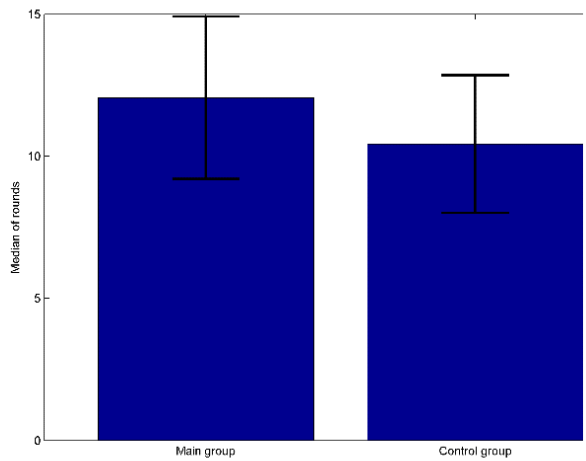


Figure 52. Median gameplay duration for main group and control group

The results of the permutation test (Figure 52) shows that players from the main group have higher probability ( $p = 0.671 \pm 0.008$ ) of playing longer than players from the control group ( $p = 0.329 \pm 0.008$ ).

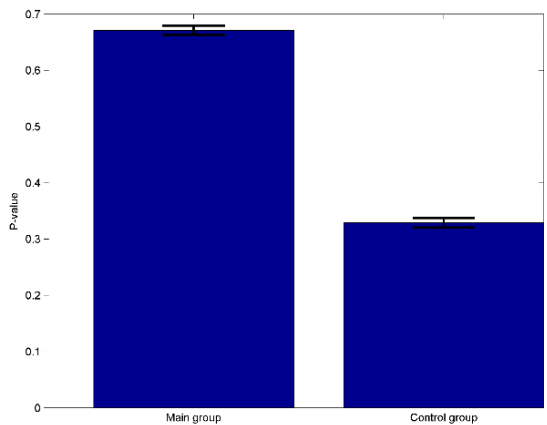


Figure 53. Probability of playing longer (permutation test)

The limits in which the experiment group outperformed the control group were evaluated. This is based on the assumption that the gameplay results (medians of rounds played) follow the Weibull probability distribution, which is often used to model time-to-failure in reliability engineering. Note that we can interpret the decision of a player to exit the game as game failure.

The number of players leaving at an early stage of the game is larger for the control group than for the main group. Two methods were used to evaluate the limits when there is a larger number of players from the main group exiting the game (see Figure 53).

First, using the bootstrapping method (bootstrap data sample is 1000) we calculated standard deviations for each round of the game and selected the limits where confidence intervals do not overlap. The results show that the players from the main group are more likely to exit the game starting from the 17th round up to the 25th round.

Second, the Students t-test was used to determine the limits where the test rejected the hypothesis that both datasets have the same mean. The results show that the players from the main group are more likely to exit the game starting from the 7th round up to the 44th round.

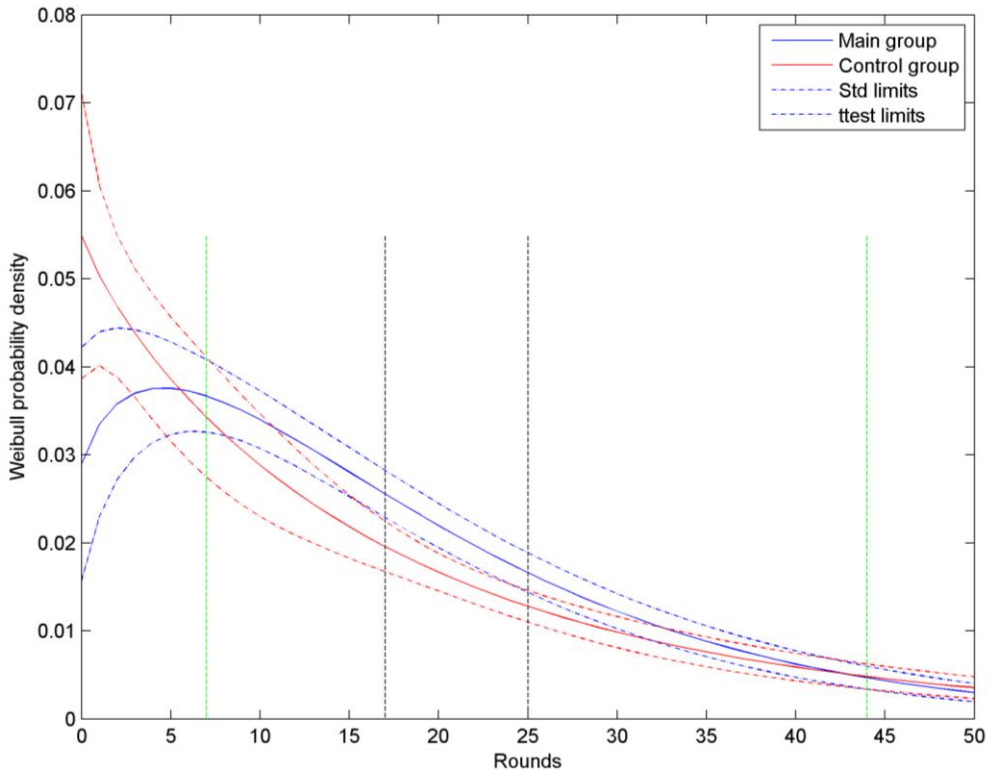


Figure 54. Duration of gameplay (in rounds)

The answers to questions of the HEXAD questionnaire which indicate the most important statistical differences between the main and the control groups in terms of median of played game rounds were identified (see Figure 54 & Table 18). For identification of the statistical importance of these questions, the bootstrapping method

and Student's t-test were applied. The results show that the most significant questions focus on competitiveness (Q7,  $p = 0.003$ ), curiosity (Q9,  $p = 0.003$ ), novelty (Q14,  $p = 0.004$ ), selfishness (Q16,  $p = 10^{-33}$ ; Q25,  $p = 10^{-44}$ ; Q30,  $p = 0.02$ ), autonomy (Q18,  $p = 10^{-94}$ ; Q22,  $p = 10^{-88}$ ), self-efficacy (Q20,  $p = 10^{-25}$ ), mastery (Q27,  $p = 10^{-79}$ ), empathy (Q28,  $p = 10^{-52}$ ). All these factors had a positive effect on the duration of gameplay. These results are consistent with the claims of the self-determination theory (Calvert et al., 1976), which emphasizes the role of autonomy and competence in the game play motivation.

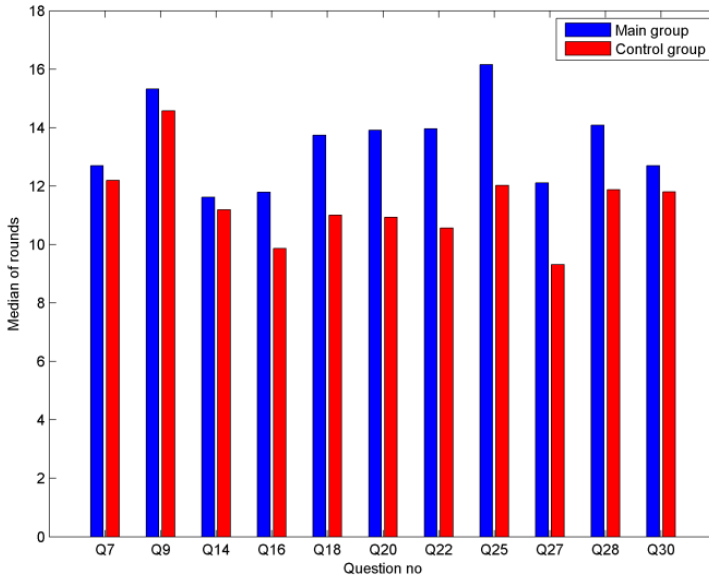


Figure 55. Analysis of questions from HeXAD questionnaire

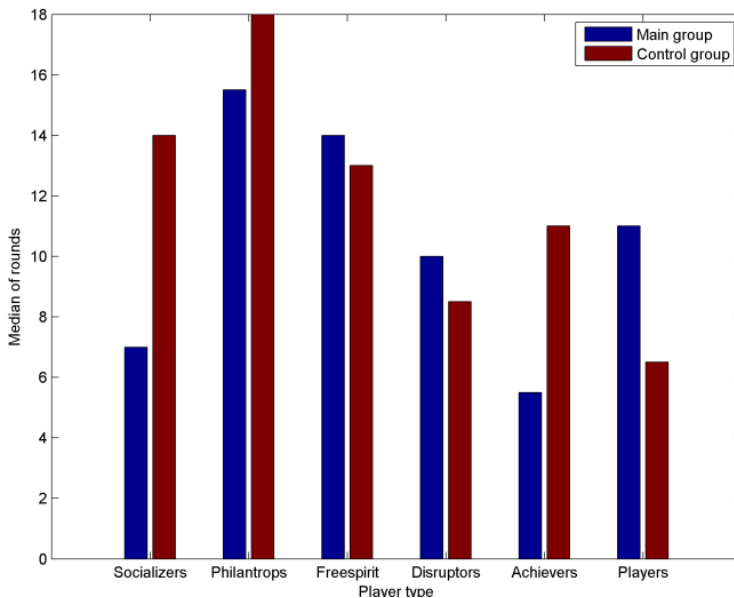


Figure 56. Duration of gameplay for each player type



The duration of gameplay (median number of game rounds play) for different player types is given in Figure 56. The game interface modified with additional incentives results in a longer gameplay for Free spirits, Disruptors and Players, while it is not effective for Socializers, Philanthropists, and Achievers.

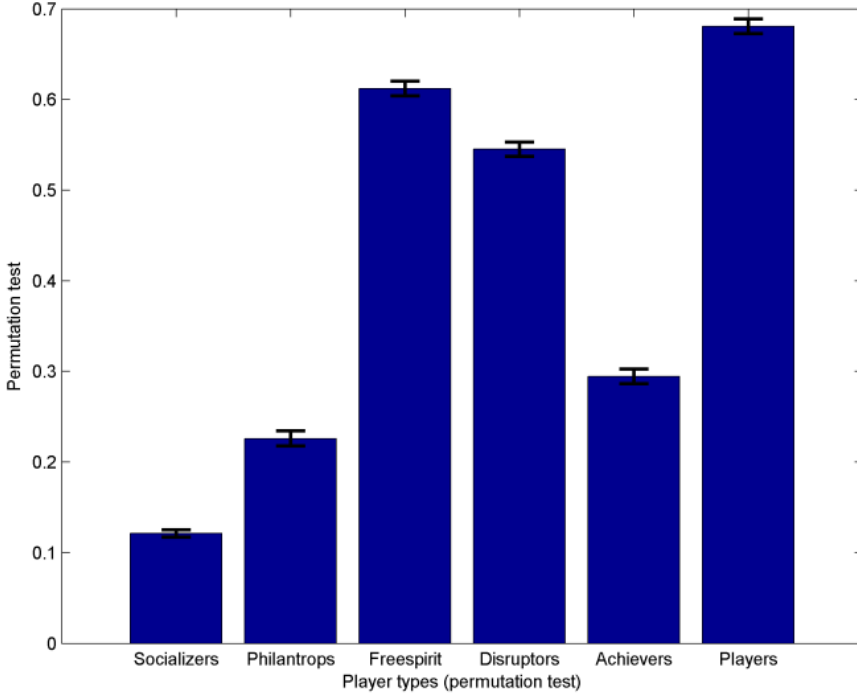


Figure 57. Results of permutation test for different player types

Figure 57 presents the results of a permutation test which shows the probability that the main group will have better result (longer gameplay) than the control group by player type. The most significant effects have been observed for Players ( $p = 0.6906 \pm 0.008$ ), Free spirits ( $p = 0.6267 \pm 0.008$ ), and Disruptors ( $p = 0.5688 \pm 0.008$ ).

#### 6.5.6. Extending UAREI MG with motivation

There is a need to incorporate user psychological decision-making process modelled behavior. To accomplish this, Minority Game decision making framework needs to integrate user motivation. We do this by extending Minority Game agent with an extra component which defines if a player wants to continue playing.

$$\alpha_i = \{a_{pick}(model), a_{key}(model), a_{playing}(model), a_{feedback}(model), E_{agent\_state}\} \quad (21)$$

The new member  $a_{playing}(model) = \begin{cases} true, & \text{if continue playing} \\ false, & \text{if done playing} \end{cases}$ , to abstract the decision making process we include a function  $m_i(model)$  and redefine  $a_{playing}(model) = \begin{cases} true, & \text{if } m_i(model) > 0 \\ false, & \text{if } m_i(model) \leq 0 \end{cases}$ . Now we have a numerical function

$m_i(model)$  which numerically represents user motivation.  $m(model)$  function can be chosen freely, but for our modeling we will use such form:

$$m_i(model) = m_{i-1}(model) + \sum_n^N O_{i,n} S_n W_n e^{-\frac{i-S_n}{\tau_n}} - S \quad (22)$$

Here each next motivation score depends on the previous motivation state.  $O_{i,n} \in \{-1,0,1\}$ - factor outcome based on model execution  $S_n$ - scalar value representing factor weight based on the game.  $W_n \in [-1,1]$ - scalar value representing factor weight based on the player.  $i$  is the round index.  $e^{-\frac{i-S_n}{\tau_n}}$  defines how each factor impact decreases over time,  $S_n$  and  $\tau_n$  values defining how fast the impact of the exponent factor decreases.  $S$  defines how fast a user loses interest in the game. Constant  $S$  can be chosen freely, but it is recommended to use the following formula:

$$S = \frac{\sum_n^N e^{-\frac{-S_n}{\tau_n}}}{K} \quad (23)$$

here the sum of maximum factors is divided by constant  $K$ .

### 6.5.7. Modelling OilTrader

$$G_{experiment} = \{U, A, R, E, I\} \quad (24)$$

$$G_{experiment} = \left\{ \begin{array}{l} \{U_{users}\}, \\ \{A_{sustain}, A_{buy}, A_{sell}\}, \\ \{R_{update stats}, R_{record choice}, R_{win}\}, \\ \{E_{history}, E_{trades}, E_{users}\}, \\ \{I_{HistoryBoard}, I_{Leaderboard}\} \end{array} \right\} \quad (25)$$

$U_{users} = \{\{A_{sell}, A_{buy}, A_{sustain}\}, S_{order}, \alpha_i\}$  – users are chosen one by one under  $S_{order}$ , users can pick one of three actions:  $A_{sell}, A_{buy}, A_{sustain}$  and

$$\alpha_i = \{a_{pick}(model), a_{key}(model), a_{playing}(model), a_{feedback}(model), E_{user,i}\} \quad (26)$$

here  $\alpha_i$  describes an agent who combines the Minority Game and user type motivation.  $a_{pick}(model)$  chooses which action to pick on the basis of game history.  $a_{key}(model)$  – it is a function which generates current state key; in this case players last 5 game outcomes.  $a_{playing}(model)$  defines if a player is still playing considering his player type motivation.  $a_{feedback}(model)$  updates agent state based on a current model state.  $E_{user,i}$  refers to current user entity.

$$\{A_{sustain}, A_{buy}, A_{sell}\} = \{\{R_{update stats}\}, S_{null}\}, \quad (27)$$

$$\{\{R_{update\ stats}\}, S_{null}\}, \{\{R_{update\ stats}\}, S_{null}\}\}$$

– here  $R_{update\ stats}$  is the rule which is triggered after the action.  $S_{null}$  – returns null, as no entity is associated with the action in this case.

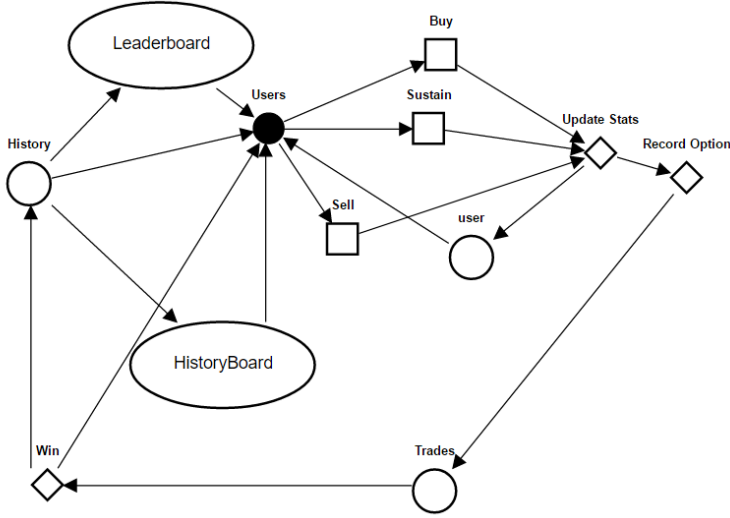


Figure 58. OilTrader UAREI model

$R_{update\ stats} = \{\{R_{record\ choice}, E_{user}\}, r_{update\ stats}\}$ ,  $R_{record\ choice}$  – records which action has been chosen.  $E_{user}$  means user entity.  $r_{update\ stats}$  records current game state for the current user. It captures user money, oil, biggest win or loss, network, last game outcome and streak.

$R_{record\ choice} = \{\{E_{trades}\}, r_{record\ choice}\}$ ,  $E_{trades}$  means entity which stores all trades,  $r_{record\ choice}$  saves which type of action has been chosen by the user and saves the amount which is traded, in case of sustain – 0, else random value from 0 to amount of money / oil is currently owned by user.

$E_{user} = \{\{U_{users}\}, D_{user}\}$ , here  $D_{user}$  is defined by such fields: Money, Oil, Network, Win (did the user win last round), Streak (how many times in a row a player has won), BWin and BLoss (biggest win and loss), Round, name, and motivation seed (m\_network, m\_position, m\_win, m\_streak, m\_bwin, m\_bloss).

$E_{trades} = \{\{R_{win}\}, D_{trades}\}$  here  $D_{trades}$  scheme is defined: UserID, Round, Action and Amount.

$R_{win} = \{\{U_{users}, E_{history}\}, r_{win}\}$  -  $r_{win}$  computes which group won and buy-to-sell and sell-to-buy ratios.

$E_{history} = \{\{U_{users}, I_{Leaderboard}, I_{HistoryBoard}\}, D_{history}\}$  -  $D_{history}$  has such fields: Round, Sell-to-buy, Buy-to-sell and outcome.

$I_{Leaderboard} = \{\{U_{users}\}, Q_{Leaderboard}\}$  -  $Q_{Leaderboard}$  selects all users and sorts by network.

$I_{HistoryBoard} = \{\{U_{users}\}, Q_{HistoryBoard}\}$  -  $Q_{HistoryBoard}$  - displays last 5 outcomes from user perspective.

### 6.5.8. Simulation and results

The hypothesis of this experiment is that it is possible to simulate synthetically random user behavior and analyze the results by player types based on classification. The result of such a simulation and analysis is that it was numerically evaluated how new factors introduced into the whole system affect user types.

The simulation was conducted in two groups: experimental and control. The experimental group sees additional UI elements during game play. We will assume that users are only affected by elements which they can see. Simulation with the following models were run on the basis of this configuration:

$$m_i(model) = m_{i-1}(model) + \sum_n^N O_{i,n} S_n W_n e^{\frac{i-S_n}{\tau_n}} - S \quad (28)$$

It is stated that there are 3 ( $N = 3$ ) factors in the control group and 6 ( $N = 6$ ) factors in the experimental group impacting how the user behavior will change. 3 shared factors are network, winning and position. Additional factors introduced in the experiment group are the biggest win, the biggest loss and streak.  $W_n$  defines that each player's factor is a random number between -1 and 1.  $O_{i,n}$  can be -1 if the impact of this factor is negative (losing money) and positive if 1 (gaining money).  $O_{i,n}$  is equal to zero if there is no change. In this model  $S_n$  and  $\tau_n$  are equal to 4 for all factors. For modeling, we assume the factor scale is  $S_n$  is equal to 1 for all factors.  $S$  will be chosen to be the same for control and experiment based on experiment group.

$$S = \frac{\sum_n^N e^{\frac{-S_n}{\tau_n}}}{K} = \frac{6e^{\frac{-4}{4}}}{3} = 2e \quad (29)$$

$m_0(model)$  is a random value between 30 and 60. The simulation is run until all players decide to stop playing. Using Marczewski's (A. Marczewski, 2015) player types, we will classify simulated players into the closest player type. Marczewski in his article identifies 6 player types:

- Socializers are motivated by being closer creating to other people.
- Free spirits are motivated by autonomy and self-expression.
- Achievers are motivated by mastery and overcoming challenges in the game.
- Philanthropists are driven by altruism helping others without any reward for themselves.
- Players are motivated by rewards, they are in the game for their own benefit.
- Disruptors are motivated by changes.

In this model, there are six elements in gamified version which effect player behavior. logical reasoning for picking each weight, which will be used for evaluating simulation of this system, is picked by reasoning logically. Three points are picked from -100 to 100 percent range, which are weight - 75, 0, 75. Being closer to 0 represents what factor has almost no impact on user behavior. The closer user factor weight is -75 more negative an outcome than the factor has to user motivation. The closer you are to 75 the more positive impact the factor has on the user's behavior. Table 19 shows how different factors should affect user motivation for different player types. It is worth noting that real experiment results should be used to justify the classification weights.

Using Table 19 we are going to classify player based motivation factor weights to the closest player category.

If we have player motivation factor weights as a vector  $W_{player} = (W_{networth}, W_{position}, W_{win}, W_{streak}, W_{biggest\ win}, W_{biggest\ loss})$ , and classification weight  $W_{player\ type} = (W_{networth}, W_{position}, W_{win}, W_{streak}, W_{biggest\ win}, W_{biggest\ loss})$ . Here

$$D_{player\ type} = \sqrt{\sum (W_{player\ type} - W_{player})^2}. \quad D_{player} =$$

$\{D_{desruptor}, D_{player}, D_{achiever}, D_{philantropist}, D_{freespirit}, D_{socializer}\}$  is a set of distances from each player type. The player's type is the player type which is closest ( $\min(D_{player})$ ) to the player type in Table 19.

Table 19. User classification by motivation weights.

Type \ Factor	Disruptors	Players	Achiever	Philanthropists	Free Spirit	Socializer
Net worth	-75	75	-75	75	75	-75
Position	75	75	75	-75	75	0
Win	0	75	-75	0	0	-75
Streak	0	75	-75	0	0	0
Biggest Win	75	75	75	-75	75	-75
Biggest Loss	75	0	75	75	75	0

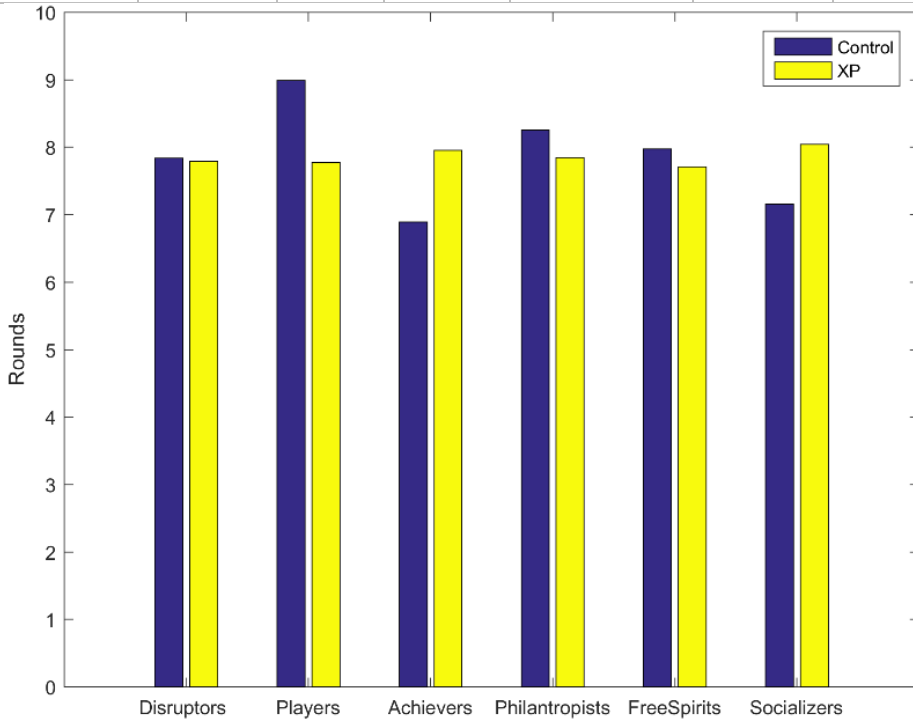


Figure 59. XP and Control group simulation results

In Figure 59 we see the simulation results. The experimental group and control group took part in the experiment. Looking at averages without classification between player types, there is no difference between round counts of control and experimental groups. Looking at the distinguished player types we see that there are differences for each player type behavior caused by the introduced changes to the experimental group.

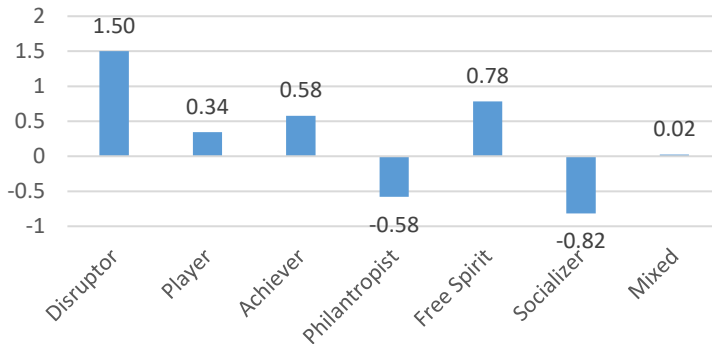


Figure 60. Differences between groups by player types

Figure 60 shows changes in the behavior of each player type. The experiment changes increased motivation for disruptors, players, achievers and free spirits, and decreased motivation for philanthropists and socializers. The results are not statistically significant per test, which indicates that the model is not calibrated correctly. To calibrate the model properly we need to adjust factor weights and S constant to achieve statistical significance. Also, a better classification could be used to achieve more accurate results.

## 6.6. Discussion of the Results & Conclusions

We have proposed two ways for evaluating gamified systems. The first method of evaluating gamified systems is using WCAG 2.0 color ratio analysis, which can indicate the visual attraction of gamified user interfaces. The second method of adapting the System Usability Scale (SUS) is through a questionnaire to gather user feedback and evaluate usability of the gamified system.

A case of study in modeling the Trogon PMS gamified application using UAREI has been demonstrated. The same gamified application has been modelled using the Machinations framework and UML activity diagrams. All modeling frameworks are proper tools for modeling gamification of software systems.

All analyzed models have been used to compare their visual complexity and it is found that UAREI has lower visual complexity score than Machinations model and UML activity diagram. A sample simulation of two players using the system under UAREI and Machinations has been done. The comparison shows that there are almost no differences between simulation results.

The advantages of the UAREI modeling method are: a high level of abstraction, native support for feedback loops, model transformation to executable code, explicit separation of data and code, user centric approach.

The analysis of the Hybrid Gamification Model has shown that despite minor differences between the theoretical model and real life model implementation, similar outcomes are observed. The case study results indicate that gamification increases

engagement, which leads to better results. The biggest difference is how many points are rewarded to the users, which has no impact on modeling results, which just creates larger user value distribution. In the real-life experiment points are attributed in discrete quantities. In the theoretical model, a user gets awarded with 0-50 points.

The hybrid eLearning model shows that gamification interaction improves students' exam results. Regarding engagement model simulation predicts better statistically significant results.

The model prediction of the increased academic performance is true. The same hypothesis is verified with real users. Gamification of a programming contest using the hybrid gamification model creates a positive impact on the contest results by increasing user engagement. Experiment results do not show statistically significant correlation between score and engagement.

Efficient and effective satisfaction of human needs is the key to success in many areas of activity. Gamification has been proposed as one of solutions aimed at increasing human motivation in various areas. However, it is not always clear how to design and implement gamification as there are many tools and mechanisms available for promoting motivation (such as points, badges, leaderboards), but their effectiveness with regards to different psychological types of players has not been studied before.

The proposed reinforcement model was developed for single player, turn-based games with infinite teleology according to the multi-dimensional typology of games (Aarseth, Smedstad, & Sunnanå, 2003), and targets the needs from beginners to intermediate players.

We have developed the visual modeling language and simulation framework UAREI, which is intended for visualizing and modeling game rules and game mechanics in the gamified systems. Four variants of Minority Game (MG) have been analyzed and computationally evaluated. The results of agents in each game are analyzed and compared using a simple win function, which registers the number of wins for each agent. The results of the classical MG model are like a random coin toss game, meaning that the game most likely would not be interesting for its players if played for a long time. The extensions of the classical MG introduce a layer of meta-game to the game, thus introducing new opportunities for the players to cooperate or compete among themselves. The variable pay-off MG provides an opportunity for an agent to earn more points in one game round and makes the game more interesting. The ternary voting MG model introduces a third option as well as bankruptcy of the player as one of the outcomes of the game. Such a model allows the analysis player behavior, which is more like real-world games. The coalition-based MG model enriches the rules of the game by an opportunity of bargaining between players, thus introducing a market-like behavior in the meta-game scenario.

The analysis and computation modeling of different MG models provide new insights into player behavior and allow to compare models based on their interestingness (evaluated in terms of negentropy of probability distribution of the win function). Such evaluations can help to develop the sustainable game mechanics, which can keep game players motivated in continuing playing the games on purpose.

The effectiveness of enhanced leaderboard has been evaluated with respect to standard leaderboard for different game player types using a simple game based on the game-theoretic framework of Minority Game set in the context of market trading. The results of the experiment show that there is a statistically significant difference between various types of players in accepting the motivation-enhancing mechanisms of

gamification. The analyzed user interface solution (progress leaderboard) has been effective in prolonging the time of gameplay for several types of players, i.e., FreeSpirits, Disruptors and Players (according to HEXAD typology (Tondello et al., 2016)), whereas for Socializers, Philanthropists and Achievers the motivation enhancing effect has not been achieved.

In this case, Players are known to be motivated by rewards, so presenting them more different types of rewards through the enhanced leaderboard has allowed them to keep more interested in the game. FreeSpirits want to explore the game and find different and new ways to gain rewards. Disruptors, on the other hand, are interested in breaking the system, so they keep playing longer just to observe the other players failing. The unexpected result is that Achievers, who are motivated by challenge and mastery, have been found to be not interested in the introduced motivational incentives. Perhaps the game itself has been too simplistic for them to keep them playing longer.

The achieved results can be explained by the inherent psychological differences of attitude towards playing: some types of players play because they like to compete with other players, therefore, different leaderboard-based solutions demonstrating different views on many aspects of competition are perfect for them, whereas other types of players play because of an opportunity to socialize without the need to compete, or just because there are fully immersed and enjoy the process of gameplay itself without considering the scores, or only for their own personal scores without the need to compare them with other players' results.

The HEXAD player questionnaire (L. Diamond G. F. Tondello & Tscheligi, 2015) method for determining psychological player types lacks protection from people entering random answers. Following the recommendations presented in (Höbßfeld et al., 2014), it has been possible to apply the two-stage statistical analysis to filter out unreliable players and to minimize the risk of error.

These results underscore the need for game and gamification designers to perform surveys and in-field studies of the user interface solutions to evaluate their effectiveness. These results also set the limits of gamification as for some player types motivation-based gamification mechanisms are not likely to be working due to their psychological attitude towards game playing.

The method described uses psychological HEXAD questionnaire results as basis for player type classification, which makes the evaluation method highly dependent of HEXAD reproducibility. Recent studies have shown problems with the reproducibility of psychological studies (Collaboration, 2015). It indicates the need for a HEXAD reproducibility study.

A method for simulating motivation of the user of the gamified systems has been offered. The method has been built on top of the UAREI modeling framework by introducing agent motivation. The results have been analyzed on the basis of the suggested player type classification. It has been found that motivation of gamified systems might be predictable if each system gamification element had a known impact on each player type.

Proposed evaluation methods target these gamified system problems: (a) system usability scale and Web Content Accessibility Guidelines target measuring usability of the gamified system; (b) player winning distribution analysis allows to evaluate the interestingness of the gamified solution and (c) gamified system analysis method by psychological player types allows to understand how the system effects different player types motivation over time and build a predictable models for computational analysis.



Deeper research is needed for analyzing different psychological player types as the result of an experiment performed here show that it is difficult to clearly assign player types to real subject, as the qualities of different psychological types maybe mixed for the same person. Rather than defining crisp player types, a fuzzy-like approach to player typology is required. These conclusions may spur the development of novel player classification taxonomies and motivation enhancing gamification mechanisms in the future.

## 7. CONCLUSIONS

Main results of dissertation:

1. It is important to evaluate gamified systems both qualitatively and quantitatively. Visual interfaces of gamification solutions can be evaluated quantitatively using the proposed method based on the Web Accessibility Guidelines (WCAG 2.0), and qualitatively using the modified SUS questionnaire (both validated in the Trogon PMS system). Statistical methods should be applied to validate any observed change in user behavior due to the effects of gamification (such as the improvement of students' exam results for gamification solutions applied in the programming contest).
2. Ten gamification patterns have been identified: infinite source, limited source, time limit, dynamic limit, random result, drain patterns, constrain, extension, property and change, and solver. The components of the abstract gamification model (User-Action-Rule-Entity-Interface formal model (UAREI)) have been identified: users, actions, rules, entities and interfaces. The model can be used for the visual specification of the gamified systems, abstract description of gamification patterns, executable modeling of gamification solutions, and generation of gamification applications. For executable modeling, the Gamification modeling (GMOD) tool, which allows for the transformation method from UAREI model to executable application, has been developed.
3. UAREI has similar or better capabilities than other industry solutions for simulating gamified systems and providing similar simulation feedback versus other known solutions. UAREI can be used to specify, model and predict randomized user behavior with respect to a real application of the gamified system and evaluate the effects of gamification. A method for simulating and evaluating gamified system impact on user motivation by psychological player types using motivation reinforcement model has been developed and demonstrated. The proposed models and methods have been tested by simulating behavior of market agents applying the Minority Game model (OilTrader game).

## REFERENCES

- Aarseth, E., Smedstad, S. M., & Sunnanå, L. (2003). A multidimensional typology of games. In *DIGRA Conf.*
- Adams, E., & Dormans, J. (2012). *Game mechanics: advanced game design*. New Riders.
- Agustin, M., Chuang, G., Delgado, A., Ortega, A., Seaver, J., & Buchanan, J. W. (2007). Game sketching. In *Proceedings of the 2nd international conference on Digital interactive media in entertainment and arts* (pp. 36–43). ACM.
- Allwood, J., Nivre, J., & Ahlsén, E. (1992). On the semantics and pragmatics of linguistic feedback. *Journal of Semantics*, 9(1), 1–26.
- Alonso, I. G., Fuente, M. P. A. G., & Brugos, J. A. L. (2009). Using sysml to describe a new methodology for semiautomatic software generation from inferred behavioral and data models. In *Systems, 2009. ICONS'09. Fourth International Conference on* (pp. 210–215). IEEE.
- Ancel, E., & Gheorghe, A. (2015). A Simulation Game Application for Improving the United States' Next Generation Air Transportation System NextGen. In *Game Theoretic Analysis of Congestion, Safety and Security* (pp. 219–253). Springer.
- Anderson, J., & Rainie, L. (2012). The future of Gamification. Pew Research Center. Washington, DC Retrieved from [Http://www. Pewinternet.org/2012/05/18/the-Future-of-Gamification](http://www.pewinternet.org/2012/05/18/the-Future-of-Gamification).
- Araújo, M., & Roque, L. (2009). Modeling games with petri nets. *Breaking New Ground: Innovation in Games, Play, Practice and Theory. DIGRA2009. Londres, Royaume Uni*.
- Araújo, R. M., & Lamb, L. C. (2005). On the evolution of memory size in the minority game. In *INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE* (Vol. 19, p. 1651). Citeseer.
- Arthur, W. B. (1994). Inductive reasoning and bounded rationality. *The American Economic Review*, 84(2), 406–411.
- Ašeriškis, D., & Damaševičius, R. (2014a). Gamification of a Project Management System. *The Seventh International Conference on Advances in Computer-Human Interactions Gamification*, (c), 200–207.
- Ašeriškis, D., & Damaševičius, R. (2014b). Gamification Pattern for Gamification Applications. *Procedia Computer Science*, 39, 83–90. <https://doi.org/10.1016/j.procs.2014.11.013>
- Balbi, S., & Giupponi, C. (2010). Agent-based modelling of socio-ecosystems: a methodology for the analysis of adaptation to climate change. *International Journal of Agent Technologies and Systems*, 2(4), 17–38.
- Banister, E. W., Calvert, T. W., Savage, M. V., & Bach, T. (1975). A systems model of training for athletic performance. *Aust J Sports Med*, 7(3), 57–61.
- Barata, G., Gama, S., Fonseca, M. J., & Gonçalves, D. (2013). Improving student creativity with gamification and virtual worlds. In *Proceedings of the First International Conference on Gameful Design, Research, and Applications* (pp. 95–98). ACM.

- Barata, G., Gama, S., Jorge, J., & Gonçalves, D. (2013). Engaging engineering students with gamification. In *Games and Virtual Worlds for Serious Applications (VS-GAMES), 2013 5th International Conference on* (pp. 1–8). IEEE.
- Barisas, D., Duracz, A., & Taha, W. (2014). DSLs Should be Online Applications. In *2014 Joint International Conference on Engineering Education & International Conference on Information Technology, 2-6 June 2014, Riga, Latvia* (pp. 314–319).
- Barkenbus, J. N. (2010). Eco-driving: An overlooked climate change initiative. *Energy Policy*, 38(2), 762–769.
- Bauchhage, C., Kersting, K., Sifa, R., Thureau, C., Drachen, A., & Canossa, A. (2012). How players lose interest in playing a game: An empirical study based on distributions of total playing times. In *2012 IEEE Conference on Computational Intelligence and Games (CIG)* (pp. 139–146). IEEE.
- Bench, S. W., & Lench, H. C. (2013). On the function of boredom. *Behavioral Sciences*, 3(3), 459–472.
- Berenguères, J., Alsuwairi, F., Zaki, N., & Ng, T. (2013). Emo-bin: How to recycle more by using emoticons. In *Proceedings of the 8th ACM/IEEE international conference on Human-robot interaction* (pp. 397–398). IEEE Press.
- Berger, V., & Schrader, U. (2016). Fostering Sustainable Nutrition Behavior through Gamification. *Sustainability*, 8(1), 67.
- Bieliūnaitė-Jankauskienė, I., & Auruškevičienė, V. (2016). A study of the application of gamification elements for individual donations. ISM University of Management and Economics.
- Bista, S. K., Nepal, S., Colineau, N., & Paris, C. (2012). Using gamification in an online community. In *Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), 2012 8th International Conference on* (pp. 611–618). IEEE.
- Bjork, S., & Holopainen, J. (2004). Patterns in game design (game development series).
- Botra, A., Rerselman, M., & Ford, M. (2014). Gamification beyond badges. In *IST-Africa Conference Proceedings, 2014* (pp. 1–10). IEEE.
- Bottazzi, G., Devetag, G., & Dosi, G. (2002). Adaptive learning and emergent coordination in minority games. *Simulation Modelling Practice and Theory*, 10(5), 321–347.
- Brathwaite, B., & Schreiber, I. (2008). Challenges for Game Designers, Charles River Media. Inc., Rockland, MA.
- Breuer, J., Scharkow, M., & Quandt, T. (2015). Sore losers? A reexamination of the frustration–aggression hypothesis for colocated video game play. *Psychology of Popular Media Culture*, 4(2), 126.
- Briggs, R. O., Reinig, B. A., & de Vreede, G.-J. (2014). An Empirical Field Study of the Yield Shift Theory of Satisfaction. In *2014 47th Hawaii International Conference on System Sciences* (pp. 492–499). IEEE.
- Brisolara, L. B., Oliveira, M. F. da S., Redin, R., Lamb, L. C., & Wagner, F. (2008). Using UML as front-end for heterogeneous software code generation

- strategies. In *Design, Automation and Test in Europe, 2008. DATE'08* (pp. 504–509). IEEE.
- Brooke, J. (1996). SUS-A quick and dirty usability scale. *Usability Evaluation in Industry, 189*(194), 4–7.
- Bruccoleri, M., D'onofrio, C., & La Commare, U. (2007). Off-line programming and simulation for automatic robot control software generation. In *Industrial Informatics, 2007 5th IEEE International Conference on* (Vol. 1, pp. 491–496). IEEE.
- Burgos, E., Ceva, H., & Perazzo, R. P. J. (2004). The evolutionary minority game with local coordination. *Physica A: Statistical Mechanics and Its Applications, 337*(3), 635–644.
- Burke, B., & Mesaglio, M. (2010). Case study: Innovation squared: The department for work and pensions turns innovation into a game. Gartner Research.
- Busso, T., Benoit, H., Bonnefoy, R., Feasson, L., & Lacour, J.-R. (2002). Effects of training frequency on the dynamics of performance response to a single training bout. *Journal of Applied Physiology, 92*(2), 572–580.
- Calvert, T. W., Banister, E. W., Savage, M. V., & Bach, T. (1976). A systems model of the effects of training on physical performance. *IEEE Transactions on Systems, Man, and Cybernetics, (2)*, 94–102.
- Canossa, A., Drachen, A., & Sørensen, J. R. M. (2011). Arrrgghh!!!: blending quantitative and qualitative methods to detect player frustration. In *Proceedings of the 6th international conference on foundations of digital games* (pp. 61–68). ACM.
- Caponetto, I., Earp, J., & Ott, M. (2014). Gamification and education: A literature review. In *ECGBL 2014: Eighth European Conference on Games Based Learning* (pp. 50–57).
- Carron, T., Marty, J.-C., & Heraud, J.-M. (2008). Teaching with game-based learning management systems: Exploring a pedagogical dungeon. *Simulation & Gaming, 39*(3), 353–378.
- Casey, W., Weaver, R., Metcalf, L., Morales, J. A., Wright, E., & Mishra, B. (2014). Cyber Security via Minority Games with Epistatic Signaling. In *BICT*.
- Chakraborti, A., Challet, D., Chatterjee, A., Marsili, M., Zhang, Y.-C., & Chakrabarti, B. K. (2015). Statistical mechanics of competitive resource allocation using agent-based models. *Physics Reports, 552*, 1–25.
- Challet, D., & Zhang, Y.-C. (1997). Emergence of cooperation and organization in an evolutionary game. *arXiv Preprint Adap-org/9708006*.
- Chan, K. T., King, I., & Yuen, M.-C. (2009). Mathematical modeling of social games. In *Computational Science and Engineering, 2009. CSE'09. International Conference on* (Vol. 4, pp. 1205–1210). IEEE.
- Chanel, G., Rebetez, C., Bétrancourt, M., & Pun, T. (2008). Boredom, engagement and anxiety as indicators for adaptation to difficulty in games. In *Proceedings of the 12th international conference on Entertainment and media in the ubiquitous era* (pp. 13–17). ACM.
- ChanLin, L. (2009). Applying motivational analysis in a Web-based course. *Innovations in Education and Teaching International, 46*(1), 91–103.

- Charles, T., Bustard, D., & Black, M. (2011). Experiences of promoting student engagement through game-enhanced learning. In *Serious games and edutainment applications* (pp. 425–445). Springer.
- Chmura, T., & Güth, W. (2011). The minority of three-game: An experimental and theoretical analysis. *Games*, 2(3), 333–354.
- Collaboration, O. S. (2015). Estimating the reproducibility of psychological science. *Science*, 349(6251), aac4716.
- Crawford, C. (2003). *Chris Crawford on game design*. New Riders.
- Csikszentmihalyi, M., & Bose, D. K. (n.d.). Flow: e Psychology of Optimal Experience.
- da Conceicao, F. S., da Silva, A. P., de Oliveira Filho, A. Q., & Silva Filho, R. C. (2014). Toward a gamification model to improve IT service management quality on service desk. In *Quality of Information and Communications Technology (QUATIC), 2014 9th International Conference on the* (pp. 255–260). IEEE.
- Dagienė, V., Pelikis, E., & Stupuriene, G. (2015). Introducing Computational Thinking through a Contest on Informatics: Problem-solving and Gender Issues. *Informacijos Mokslai/Information Sciences*, 73.
- Dagiene, V., & Stupuriene, G. (2016). Informatics Concepts and Computational Thinking in K-12 Education: A Lithuanian Perspective. *Journal of Information Processing*, 24(4), 732–739.
- de Almeida, J. R. L., & Menche, J. (2003). Quantifying a certain “advantage law”: minority game with above the rules agents. *Brazilian Journal of Physics*, 33(4), 895–898.
- de Oliveira, G. W., Julia, S., & Passos, L. M. S. (2011). Game modeling using workflow nets. In *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on* (pp. 838–843). IEEE.
- Deci, E. L., & Ryan, R. M. (2000). The “what” and “why” of goal pursuits: Human needs and the self-determination of behavior. *Psychological Inquiry*, 11(4), 227–268.
- Deterding, S. (2012). Gamification: Designing for Motivation. *interactions*, 19 (4), 14–17. July.
- Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011). From game design elements to gamefulness: defining gamification. In *Proceedings of the 15th international academic MindTrek conference: Envisioning future media environments* (pp. 9–15). ACM.
- Deterding, S., Sicart, M., Nacke, L., O’Hara, K., & Dixon, D. (2011). Gamification. using game-design elements in non-gaming contexts. In *CHI’11 Extended Abstracts on Human Factors in Computing Systems* (pp. 2425–2428). ACM.
- Devedzić, V. (2002). Understanding ontological engineering. *Communications of the ACM*, 45(4), 136–144.
- Dichev, C., Dicheva, D., Angelova, G., & Agre, G. (2014). From gamification to gameful design and gameful experience in learning. *Cybernetics and Information Technologies*, 14(4), 80–100.
- Dormans, J. (2008). Visualizing Game Dynamics and Emergent Gameplay. In

- Proceedings of the Meaningful Play conference.*
- Dormans, J. (2009). Machinations: Elemental feedback structures for game design. In *Proceedings of the GAMEON-NA Conference* (pp. 33–40).
- Dormans, J. (2012). *Engineering emergence: applied theory for game design*. Creative Commons.
- Dormans, J. (2013). Machinations Diagram Tutorial. *Portfolio of Joris Dormans*.
- Dubina, I. N., & Oskorbin, N. M. (2015). Game-Theoretic Models of Incentive and Control Strategies in Social and Economic Systems. *Cybernetics and Systems*, 46(5), 303–319.
- Easley, D., & Ghosh, A. (2013). Incentives, gamification, and game theory: an economic approach to badge design. In *Proceedings of the fourteenth ACM conference on Electronic commerce* (pp. 359–376). ACM.
- Edalat, A., Ghoroghi, A., & Sakellariou, G. (2012). Multi-games and a double game extension of the Prisoner's Dilemma. *arXiv Preprint arXiv:1205.4973*.
- Eder, R., Filieri, A., Kurz, T., Heistracher, T. J., & Pezzuto, M. (2008). Model-transformation-based Software Generation utilizing Natural language notations. In *Digital Ecosystems and Technologies, 2008. DEST 2008. 2nd IEEE International Conference on* (pp. 306–312). IEEE.
- Felsenthal, D. S., & Machover, M. (1997). Ternary voting games. *International Journal of Game Theory*, 26(3), 335–351.
- Festinger, L. (1957). A Theory of Cognitive Dissonance: Stanford University.
- Fogg, B. J. (2009). A behavior model for persuasive design. In *Proceedings of the 4th international Conference on Persuasive Technology* (p. 40). ACM.
- Fonseca, B., Pereira, Â., Sanders, R., Barracho, V., Lapajne, U., Rus, M., ... Bojovic, V. (2012). PLAYER-a European Project and a Game to Foster Entrepreneurship Education for Young People. *J. UCS*, 18(1), 86–105.
- Freudmann, E. A., & Bakamitsos, Y. (2014). The Role of Gamification in Non-profit Marketing: An Information Processing Account. *Procedia-Social and Behavioral Sciences*, 148, 567–572.
- Galli, L., Fraternali, P., Martinenghi, D., Tagliasacchi, M., & Novak, J. (2012). A draw-and-guess game to segment images. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)* (pp. 914–917). IEEE.
- Gamma, E. (1995). *Design patterns: elements of reusable object-oriented software*. Pearson Education India.
- Gartner Research. (2012). Gartner Says By 2015, More Than 50 Percent of Organizations That Manage Innovation Processes Will Gamify Those Processes. *Gartner Inc*, 2015. Retrieved from <http://www.gartner.com/newsroom/id/1629214>
- Gatautis, R., Banyte, J., Piligrimiene, Z., Vitkauskaite, E., & Tarute, A. (2016). THE IMPACT OF GAMIFICATION ON CONSUMER BRAND ENGAGEMENT. *Transformation in Business & Economics*, 15(1).
- Gatautis, R., & Vitkauskaite, E. (2014). Crowdsourcing application in marketing activities. *Procedia-Social and Behavioral Sciences*, 110, 1243–1250.
- Gatautis, R., Vitkauskaite, E., Gadeikiene, A., & Piligrimiene, Z. (2016).

- Gamification as a Mean of Driving Online Consumer Behaviour: SOR Model Perspective. *Engineering Economics*, 27(1), 90–97.
- Gené, O. B., Núñez, M. M., & Blanco, Á. F. (2014). Gamification in MOOC: challenges, opportunities and proposals for advancing MOOC model. In *Proceedings of the Second International Conference on Technological Ecosystems for Enhancing Multiculturality* (pp. 215–220). ACM.
- Geng, L., & Hamilton, H. J. (2006). Interestingness measures for data mining: A survey. *ACM Computing Surveys (CSUR)*, 38(3), 9.
- Gilbert, G. N. (2008). *Agent-based models*. Sage.
- Giouvanakis, E., Kotropoulos, C., Theodoridis, A., & Pitas, I. (2013). A game with a purpose for annotating Greek folk music in a web content management system. In *Digital Signal Processing (DSP), 2013 18th International Conference on* (pp. 1–6). IEEE.
- Giurca, A., & Pascalau, E. (2008). JSON Rules. In *KESE*.
- Gnauk, B., Dannecker, L., & Hahmann, M. (2012). Leveraging gamification in demand dispatch systems. In *Proceedings of the 2012 Joint EDBT/ICDT Workshops* (pp. 103–110). ACM.
- Gou, C. (2006). Agents play mix-game. In *Econophysics of Stock and other Markets* (pp. 123–132). Springer.
- Green, T. R. G., & Petre, M. (1996). Usability analysis of visual programming environments: a “cognitive dimensions” framework. *Journal of Visual Languages & Computing*, 7(2), 131–174.
- Greenwood, G. W. (2009). Deceptive strategies for the evolutionary minority game. In *2009 IEEE Symposium on Computational Intelligence and Games* (pp. 25–31). IEEE.
- Groh, F. (2012). Gamification: State of the art definition and utilization. *Institute of Media Informatics Ulm University*, 39.
- Grünvogel, S. M. (2005). Formal models and game design. *Game Studies*, 5(1), 1–9.
- Gulia, S., & Choudhury, T. (2016). An efficient automated design to generate UML diagram from Natural Language Specifications. In *Cloud System and Big Data Engineering (Confluence), 2016 6th International Conference* (pp. 641–648). IEEE.
- Hall, M., Kimbrough, S. O., Haas, C., Weinhardt, C., & Caton, S. (2012). Towards the gamification of well-being measures. In *E-Science (e-Science), 2012 IEEE 8th International Conference on* (pp. 1–8). IEEE.
- Hamari, J., & Koivisto, J. (2013). Social Motivations To Use Gamification: An Empirical Study Of Gamifying Exercise. In *ECIS* (p. 105).
- Hamari, J., Koivisto, J., & Sarsa, H. (2014). Does gamification work?--a literature review of empirical studies on gamification. In *2014 47th Hawaii International Conference on System Sciences* (pp. 3025–3034). IEEE.
- Harwood, T., & Garry, T. (2015). An investigation into gamification as a customer engagement experience environment. *Journal of Services Marketing*, 29(6/7), 533–546.
- He, L., & Ioerger, T. R. (2006). Combining bundle search with buyer coalition formation in electronic markets: A distributed approach through explicit



- negotiation. *Electronic Commerce Research and Applications*, 4(4), 329–344.
- Heller, F., Lichtschlag, L., Wittenhagen, M., Karrer, T., & Borchers, J. (2011). Me hates this: exploring different levels of user feedback for (usability) bug reporting. In *CHI'11 Extended Abstracts on Human Factors in Computing Systems* (pp. 1357–1362). ACM.
- Herranz, E., Palacios, R. C., de Amescua Seco, A., & Yilmaz, M. (2014). Gamification as a Disruptive Factor in Software Process Improvement Initiatives. *J. UCS*, 20(6), 885–906.
- Herzig, P., Ameling, M., & Schill, A. (2012). A generic platform for enterprise gamification. In *Software Architecture (WICSA) and European Conference on Software Architecture (ECSA), 2012 Joint Working IEEE/IFIP Conference on* (pp. 219–223). IEEE.
- Herzig, P., Jugel, K., Momm, C., Ameling, M., & Schill, A. (2013). GaML-A modeling language for gamification. In *Proceedings of the 2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing* (pp. 494–499). IEEE Computer Society.
- Hetherinton, D. (2014). SysML requirements for training game design. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)* (pp. 162–167). IEEE.
- Hidi, S., & Baird, W. (1986). Interestingness—A neglected variable in discourse processing. *Cognitive Science*, 10(2), 179–194.
- Hill, A. B., & Perkins, R. E. (1985). Towards a model of boredom. *British Journal of Psychology*, 76(2), 235–240.
- Hirth, M., Hoßfeld, T., & Tran-Gia, P. (2011). Anatomy of a crowdsourcing platform—using the example of microworkers. com. In *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2011 Fifth International Conference on* (pp. 322–329). IEEE.
- Hoarau, J. (2012). JIRA Hero. Retrieved January 11, 2017, from <https://marketplace.atlassian.com/archive/com.madgnome.jira.plugins.jirachievements>
- Hoßfeld, T., Keimel, C., Hirth, M., Gardlo, B., Habigt, J., Diepold, K., & Tran-Gia, P. (2014). Best practices for QoE crowdtesting: QoE assessment with crowdsourcing. *IEEE Transactions on Multimedia*, 16(2), 541–558.
- Hovland, C. I., Harvey, O. J., & Sherif, M. (1957). Assimilation and contrast effects in reactions to communication and attitude change. *The Journal of Abnormal and Social Psychology*, 55(2), 244.
- Hunicke, R., LeBlanc, M., & Zubek, R. (2004). MDA: A formal approach to game design and game research. In *Proceedings of the AAAI Workshop on Challenges in Game AI* (Vol. 4, p. 1).
- Huotari, K., & Hamari, J. (2012). Defining gamification: a service marketing perspective. In *Proceeding of the 16th International Academic MindTrek Conference* (pp. 17–22). ACM.
- Immorlica, N., Stoddard, G., & Syrgkanis, V. (2015). Social status and badge design. In *Proceedings of the 24th International Conference on World Wide Web* (pp. 473–483). ACM.

- Iosup, A., & Epema, D. (2014). An experience report on using gamification in technical higher education. In *Proceedings of the 45th ACM technical symposium on Computer science education* (pp. 27–32). ACM.
- James, B., Fletcher, B., & Wearn, N. (2013). Three corners of reward in computer games.
- Janssens, O., Samyny, K., Van de Walle, R., & Van Hoecke, S. (2014). Educational virtual game scenario generation for serious games. In *Serious Games and Applications for Health (SeGAH), 2014 IEEE 3rd International Conference on* (pp. 1–8). IEEE.
- Johnson, N. F., Jefferies, P., & Hui, P. M. (2003). Financial market complexity. *OUP Catalogue*.
- Jovanovic, J., & Devedzic, V. (2014). Open badges: Challenges and opportunities. In *International Conference on Web-Based Learning* (pp. 56–65). Springer.
- Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4, 237–285.
- Kalinauskas, M. (2014a). Gamification in fostering creativity. *Socialnės Technologijos*, (1), 62–75.
- Kalinauskas, M. (2014b). Gamification in Fostering Creativity: Player Type Approach. *Socialines Technologijos*, 4(2).
- Kapp, K. M. (2016). Gamification Designs for Instruction. *Instructional-Design Theories and Models, Volume IV: The Learner-Centered Paradigm of Education*, 351.
- Kelle, S., Klemke, R., & Specht, M. (2011). Design patterns for learning games. *International Journal of Technology Enhanced Learning*, 3(6), 555–569.
- Kiekintveld, C., & Wellman, M. P. (2008). Selecting strategies using empirical game models: an experimental analysis of meta-strategies. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 2* (pp. 1095–1101). International Foundation for Autonomous Agents and Multiagent Systems.
- Kiili, K. (2010). Call for learning-game design patterns. *Educational Games: Design, Learning, and Applications*. Nova Publishers.
- Kim, J. T., & Lee, W.-H. (2013). Dynamical Model and Simulations for Gamification of Learning. *International Journal of Multimedia and Ubiquitous Engineering*, 8(4), 179–190.
- Kim, J. T., & Lee, W.-H. (2015). Dynamical model for gamification of learning (DMGL). *Multimedia Tools and Applications*, 74(19), 8483–8493.
- Kim, S. (2015). Team Organization Method Using Salary Auction Game for Sustainable Motivation. *Sustainability*, 7(10), 14358–14370.
- King, D., Delfabbro, P., & Griffiths, M. (2010). Video game structural characteristics: A new psychological taxonomy. *International Journal of Mental Health and Addiction*, 8(1), 90–106.
- Kivikangas, J. M., Chanel, G., Cowley, B., Ekman, I., Salminen, M., Järvelä, S., & Ravaja, N. (2011). A review of the use of psychophysiological methods in game research. *Journal of Gaming & Virtual Worlds*, 3(3), 181–199.
- Klein, J. (2015). Model Driven Engineering: Automatic Code Generation and

- Beyond. Retrieved from [https://insights.sei.cmu.edu/sei\\_blog/2015/05/model-driven-engineering-automatic-code-generation-and-beyond.html](https://insights.sei.cmu.edu/sei_blog/2015/05/model-driven-engineering-automatic-code-generation-and-beyond.html)
- Kluger, A. N., & DeNisi, A. (1996). The effects of feedback interventions on performance: a historical review, a meta-analysis, and a preliminary feedback intervention theory. *Psychological Bulletin*, 119(2), 254.
- Kohler, T. A., & Gumerman, G. G. (2000). *Dynamics in human and primate societies: Agent-based modeling of social and spatial processes*. Oxford University Press.
- Koike, R., Nakaya, N., & Koi, Y. (2007). Development of system for the automatic generation of unknown virus extermination software. In *Applications and the Internet, 2007. SAINT 2007. International Symposium on* (p. 8). IEEE.
- Kokil, U. (2013). Investigating Players' Affective States in an Interactive Environment. *ThinkMind//ACHI*.
- Kostecka, J., & Davidavičienė, V. (2015). Darbuotojų motyvavimo žaidybinimo priemonėmis informacinėje sistemoje modelis. *Science: Future of Lithuania*, 7(2).
- Koster, R. (2005). A grammar of gameplay: game atoms: can games be diagrammed. In *Presentation at the Game Developers conference*.
- Koster, R. (2013). *Theory of fun for game design*. "O'Reilly Media, Inc."
- Kotzé, P., Renaud, K., & Van Dyk, T. (2002). Feedback And Task Analysis For E-Commerce Sites. In *ISSA* (pp. 1–17). Citeseer.
- Kreimeier, B. (2002). The case for game design patterns.
- Kriouile, A., Addamssiri, N., & Gadi, T. (2015). An MDA method for automatic transformation of models from CIM to PIM. *American Journal of Software Engineering and Applications*, 4(1), 1–14.
- Kristoffer, F., & Robin, M. (2012). Enterprise gamification of the employee development process at an infocom consultancy company.
- Kroeze, C., & Olivier, M. S. (2012). Gamifying authentication. In *2012 Information Security for South Africa* (pp. 1–8). IEEE.
- Kuss, D. J. (2013). Internet gaming addiction: current perspectives. *Psychol Res Behav Manag*, 6, 125–137.
- Kwon, K., Yi, Y., Kim, D., & Ha, S. (2005). Embedded software generation from system level specification for multi-tasking embedded systems. In *Proceedings of the 2005 Asia and South Pacific Design Automation Conference* (pp. 145–150). ACM.
- L. Diamond G. F. Tondello, A. M. L. E. N., & Tscheligi, M. (2015). The HEXAD Gamification User Types Questionnaire : Background and Development Process. In *Workshop on Personalization in Serious and Persuasive Games and Gamified Interactions*. London, UK. Retrieved from <https://hcigames.com/download/the-hexad-gamification-user-types-questionnaire-background-and-development-process>
- Lee, J. J., & Hammer, J. (2011). Gamification in education: What, how, why bother? *Academic Exchange Quarterly*, 15(2), 146.
- Lehman, J., & Stanley, K. O. (2012). Beyond open-endedness: Quantifying impressiveness. *Artificial Life*, 13, 75–82.

- Lethbridge, T. C. (2013). Key Properties for Comparing Modeling Languages and Tools: Usability, Completeness and Scalability.
- Li, Y., VanDeemen, A., & Savit, R. (2000). The minority game with variable payoffs. *Physica A: Statistical Mechanics and Its Applications*, 284(1), 461–477.
- Linde, J., Sonnemans, J., & Tuinstra, J. (2014). Strategies and evolution in the minority game: A multi-round strategy experiment. *Games and Economic Behavior*, 86, 77–95.
- Luo, S., Yang, H., & Meinel, C. (n.d.). Reward-based Intermittent Reinforcement in Gamification for E-learning.
- Lux, M., Guggenberger, M., & Riegler, M. (2014). PictureSort: gamification of image ranking. In *Proceedings of the First International Workshop on Gamification for Information Retrieval* (pp. 57–60). ACM.
- Ma, Y., Li, G., Dong, Y., & Qin, Z. (2010). Minority game data mining for stock market predictions. In *International Workshop on Agents and Data Mining Interaction* (pp. 178–189). Springer.
- Malone, T. (1981). *What makes computer games fun?* (Vol. 13). ACM.
- Marczewski, A. (2015). User types. *Even Ninja Monkeys Like to Play: Gamification, Game Thinking and Motivational Design*, 1, 65–80.
- Marczewski, A. C. (2015). *Even Ninja Monkeys Like to Play: Gamification, Game Thinking and Motivational Design*. CreateSpace Independent Publishing Platform.
- Marsan, G. A. (2009). New paradigms towards the modelling of complex systems in behavioral economics. *Mathematical and Computer Modelling*, 50(3), 584–597.
- Martey, R. M., Kenski, K., Folkestad, J., Feldman, L., Gordis, E., Shaw, A., ... Kaufman, N. (2014). Measuring game engagement multiple methods and construct complexity. *Simulation & Gaming*, 1046878114553575.
- Matallaoui, A., Herzig, P., & Zarnekow, R. (2015). Model-Driven Serious Game Development Integration of the Gamification Modeling Language GaML with Unity. In *System Sciences (HICSS), 2015 48th Hawaii International Conference on* (pp. 643–651). IEEE.
- Mayer, I., Bekebrede, G., & van Bilsen, A. (2010). Understanding complex adaptive systems by playing games. *Informatics in Education-An International Journal*, (Vol 9\_1), 1–18.
- Mazanec, M., & Macek, O. (2012). On General-purpose Textual Modeling Languages. In *DATESO* (Vol. 12, pp. 1–12). Citeseer.
- Mazur, J. E. (2006). Mathematical models and the experimental analysis of behavior. *Journal of the Experimental Analysis of Behavior*, 85(2), 275–291.
- McGonigal, J. (2011). *Reality is broken: Why games make us better and how they can change the world*. Penguin.
- Meece, J. L., Anderman, E. M., & Anderman, L. H. (2006). Classroom goal structure, student motivation, and academic achievement. *Annu. Rev. Psychol.*, 57, 487–503.
- Meyers, R. A. (2009). *Encyclopedia of complexity and systems science*. Springer.

- Minor, D. B. (2013). Increasing revenue through rewarding the best less (or not at all). *Northwestern University Typescript*.
- Moelbert, S., & De Los Rios, P. (2002). The local minority game. *Physica A: Statistical Mechanics and Its Applications*, 303(1), 217–225.
- Mora, A., Riera, D., Gonzalez, C., & Arnedo-Moreno, J. (2015). A literature review of gamification design frameworks. In *Games and Virtual Worlds for Serious Applications (VS-Games), 2015 7th International Conference on* (pp. 1–8). IEEE.
- Morrison, B. B., & DiSalvo, B. (2014). Khan academy gamifies computer science. In *Proceedings of the 45th ACM technical symposium on Computer science education* (pp. 39–44). ACM.
- Morschheuser, B., Hamari, J., Werder, K., & Abe, J. (2017). How to gamify? A method for designing gamification. In *Proceedings of the 50th Hawaii International Conference on System Sciences*.
- Morton, R. H., Fitz-Clarke, J. R., & Banister, E. W. (1990). Modeling human performance in running. *Journal of Applied Physiology*, 69(3), 1171–1177.
- Muñoz, J., Mendoza, R., Álvarez, F., Martín, M. V., & Ochoa, A. (2007). Integration of Auditive and Visual Feedback in the Design of Interfaces for Security Applications. *CLHC 2007*.
- Muratet, M., Torguet, P., Jessel, J.-P., & Viallet, F. (2009). Towards a serious game to help students learn computer programming. *International Journal of Computer Games Technology*, 2009, 3.
- Nah, F. F.-H., Zeng, Q., Telaprolu, V. R., Ayyappa, A. P., & Eschenbrenner, B. (2014). Gamification of education: a review of literature. In *International Conference on HCI in Business* (pp. 401–409). Springer.
- Narayanan, A. (2014). *Gamification for Employee Engagement*. Packt Publishing Ltd.
- Narayanasamy, V., Wong, K. W., Rai, S., & Chiou, A. (2010). Complex game design modeling. In *Cultural Computing* (pp. 65–74). Springer.
- Neeli, B. K. (2012). A method to engage employees using gamification in BPO industry. In *Services in Emerging Markets (ICSEM), 2012 Third International Conference On* (pp. 142–146). IEEE.
- Negruşa, A. L., Toader, V., Sofică, A., Tutunea, M. F., & Rus, R. V. (2015). Exploring gamification techniques and applications for sustainable tourism. *Sustainability*, 7(8), 11160–11189.
- Nikkila, S., Linn, S., Sundaram, H., & Kelliher, A. (2011). Playing in taskville: Designing a social game for the workplace. In *Workshop on Gamification: Using Game Design Elements in Non-Gaming Contexts* (pp. 1–4).
- Nudelman, E., Wortman, J., Shoham, Y., & Leyton-Brown, K. (2004). Run the GAMUT: A comprehensive approach to evaluating game-theoretic algorithms. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 2* (pp. 880–887). IEEE Computer Society.
- Nummenmaa, T., Berki, E., & Mikkonen, T. (2009). Exploring games as formal models. In *Formal Methods (SEEFM), 2009 Fourth South-East European*

- Workshop on* (pp. 60–65). IEEE.
- Nylund, A., & Landfors, O. (2015). Frustration and its effect on immersion in games: A developer viewpoint on the good and bad aspects of frustration.
- O'Donovan, S., Gain, J., & Marais, P. (2013). A case study in the gamification of a university-level games development course. In *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference* (pp. 242–251). ACM.
- Oliver, R. L. (1980). A cognitive model of the antecedents and consequences of satisfaction decisions. *Journal of Marketing Research*, 460–469.
- Oz, M. A. N., Sener, I., Kaymakci, O. T., Ustoglu, I., & Cansever, G. (2015). A tool for automatic formal modeling of railway interlocking systems. In *EUROCON 2015-International Conference on Computer as a Tool (EUROCON)*, IEEE (pp. 1–4). IEEE.
- Padmanabhan, B., & Tuzhilin, A. (1999). Unexpectedness as a measure of interestingness in knowledge discovery. *Decision Support Systems*, 27(3), 303–318.
- Park, H. (2012). Relationship between motivation and student's activity on educational game. *International Journal of Grid and Distributed Computing*, 5(1), 101–114.
- Parker, C., & Mathews, B. P. (2001). Customer satisfaction: contrasting academic and consumers' interpretations. *Marketing Intelligence & Planning*, 19(1), 38–44.
- Pedreira, O., García, F., Brisaboa, N., & Piattini, M. (2015). Gamification in software engineering—A systematic mapping. *Information and Software Technology*, 57, 157–168.
- Petri net Java applets. (2017). Retrieved from <https://www.informatik.uni-hamburg.de/TGI/PetriNets/tools/java/>
- Piligrimiene, Z., Dovaliene, A., & Virvilaite, R. (2015). Consumer Engagement in Value Co-Creation: what Kind of Value it creates for Company? *Engineering Economics*, 26(4), 452–460.
- Pirker, J., Riffnaller-Schiefer, M., & Gütl, C. (2014). Motivational active learning: Engaging university students in computer science education. In *Proceedings of the 2014 conference on Innovation & technology in computer science education* (pp. 297–302). ACM.
- Pitrenaitė-Zilėnienė, B., & Skarzauskienė, A. (2013). Potential and Challenges of Web-based Collective Intelligence to Tackle Societal Problems. *Socialines Technologijos*, 3(2).
- Przybylski, A. K., Rigby, C. S., & Ryan, R. M. (2010). A motivational model of video game engagement. *Review of General Psychology*, 14(2), 154.
- Rao, V., & Pandas, P. (2014). Heuristic Evaluation of Persuasive Game Systems in a Behavior Change Support Systems Perspective: Elements for Discussion. In *Proceedings of the Second International Workshop on Behavior Change Support Systems (BCSS2014)*, Padova, Italy.
- Rayadurgam, S. (2001). Automated test-data generation from formal models of software. In *Automated Software Engineering, 2001.(ASE 2001). Proceedings*.

- 16th Annual International Conference on (p. 438). IEEE.
- RedCritic Corp. (2011). RedCritic Tracker. Retrieved from <http://www.redcritictracker.com/>
- Reeves, B., Cummings, J. J., Scarborough, J. K., Flora, J., & Anderson, D. (2012). Leveraging the engagement of games to change energy behavior. In *Collaboration Technologies and Systems (CTS), 2012 International Conference on* (pp. 354–358). IEEE.
- Reeves, B., & Read, J. L. (2013). *Total engagement: How games and virtual worlds are changing the way people work and businesses compete*. Harvard Business Press.
- Reid, L. G., & Snow-Weaver, A. (2008). WCAG 2.0: a web accessibility standard for the evolving web. In *Proceedings of the 2008 international cross-disciplinary conference on Web accessibility (W4A)* (pp. 109–115). ACM.
- Richter, G., Raban, D. R., & Rafaeli, S. (2015). Studying gamification: the effect of rewards and incentives on motivation. In *Gamification in education and business* (pp. 21–46). Springer.
- Rico, M., Agudo, J. E., & Sánchez, H. (2015). Language Learning through Handheld Gaming: a Case Study of an English Course with Engineering Students. *Journal of Universal Computer Science*, 21(10), 1362–1378.
- Roberts, J. A., Hann, I.-H., & Slaughter, S. A. (2006). Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the Apache projects. *Management Science*, 52(7), 984–999.
- Rollings, A., & Adams, E. (2006). Fundamentals of game design. *New Challenges for Character-Based AI for Games. Chapter 20: Artificial Life and Puzzle Games. Prentice Hall*, 573–590.
- Roth, A. E. (2002). The economist as engineer: Game theory, experimentation, and computation as tools for design economics. *Econometrica*, 70(4), 1341–1378.
- Ryan, R. M., Rigby, C. S., & Przybylski, A. (2006). The motivational pull of video games: A self-determination theory approach. *Motivation and Emotion*, 30(4), 344–360.
- Saha, R., Manna, R., & Geetha, G. (2012). CAPTCHINO-A Gamification of Image-Based CAPTCHAs to Evaluate Usability Issues. In *Computing Sciences (ICCS), 2012 International Conference on* (pp. 95–99). IEEE.
- Sailer, M., Hense, J., Mandl, H., & Klevers, M. (2013). Psychological Perspectives on Motivation through Gamification. *IXD&A*, 19, 28–37.
- Salazar, L. (2004). Modélisation et analyse spatiale et temporelle des jeux vidéo basées sur les réseaux de Pétri. CNAM.
- Salen, K., & Zimmerman, E. (2004). *Rules of play: Game design fundamentals*. MIT press.
- Sallach, D. L., North, M. J., & Tatara, E. (2010). Multigame dynamics: Structures and strategies. In *International Workshop on Multi-Agent Systems and Agent-Based Simulation* (pp. 108–120). Springer.
- Sammut, R., Seychell, D., & Attard, N. (2014). Gamification of Project Management within a Corporate Environment: An Exploratory Study. In

- Games and Virtual Worlds for Serious Applications (VS-GAMES), 2014 6th International Conference on* (pp. 1–2). IEEE.
- Sanchez, E. (2011). Key criteria for game design. A framework.
- Schmidhuber, J. (2009). Driven by compression progress: A simple principle explains essential aspects of subjective beauty, novelty, surprise, interestingness, attention, curiosity, creativity, art, science, music, jokes. In *Anticipatory Behavior in Adaptive Learning Systems* (pp. 48–76). Springer.
- Shapiro, D., Tanenbaum, K., McCoy, J., LeBron, L., Reynolds, C., Stern, A., ... Moffitt, K. (2015). Composing Social Interactions via Social Games. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems* (pp. 573–580). International Foundation for Autonomous Agents and Multiagent Systems.
- Sharek, D., & Wiebe, E. (2014). Measuring video game engagement through the cognitive and affective dimensions. *Simulation & Gaming*, 45(4–5), 569–592.
- Sherrington, D. (2006). The minority game: A statistical physics perspective. *Physica A: Statistical Mechanics and Its Applications*, 370(1), 7–11.
- Silva, F., Analide, C., Rosa, L., Felgueiras, G., & Pimenta, C. (2013). Social networks gamification for sustainability recommendation systems. In *Distributed Computing and Artificial Intelligence* (pp. 307–315). Springer.
- Simões, J., Redondo, R. D., & Vilas, A. F. (2013). A social gamification framework for a K-6 learning platform. *Computers in Human Behavior*, 29(2), 345–353.
- Sims, S. (n.d.). Badgeville et al., “Intrinsic and Extrinsic Motivations.”
- Singer, L., & Schneider, K. (2012). It was a bit of a race: Gamification of version control. In *Games and Software Engineering (GAS), 2012 2nd International Workshop on* (pp. 5–8). IEEE.
- Skaržauskienė, A., & Kalinauskas, M. (2014). Fostering collective creativity through gamification. In *The proceedings of the ISPIM Americas Innovation Forum (October 2014): Montreal, Canada on 5-8 October 2014*.
- Smith-Robbins, S. (2011). This game sucks”: How to improve the gamification of education. *EDUCAUSE Review*, 46(1), 58–59.
- Smith, A. M., Nelson, M. J., & Mateas, M. (2010). Ludocore: A logical game engine for modeling videogames. In *Proceedings of the 2010 IEEE Conference on Computational Intelligence and Games* (pp. 91–98). IEEE.
- Soleymani, M., & Larson, M. (2010). Crowdsourcing for affective annotation of video: Development of a viewer-reported boredom corpus.
- Song, S., & Lee, J. (2007). RETRACTED: Key factors of heuristic evaluation for game design: Towards massively multi-player online role-playing game. Elsevier.
- Spink, A., & Saracevic, T. (1998). Human-computer interaction in information retrieval: nature and manifestations of feedback. *Interacting with Computers*, 10(3), 249–267.
- Stålnacke Larsson, R. (2013). Motivations in Sports and Fitness Gamification: A study to understand what motivates the users of sports and fitness gamification services.
- Stupurienė, G., Vinikienė, L., & Dagienė, V. (2016). Students’ Success in the



- Bebras Challenge in Lithuania: Focus on a Long-Term Participation. In *International Conference on Informatics in Schools: Situation, Evolution, and Perspectives* (pp. 78–89). Springer.
- Sur, C., Sharma, S., & Shukla, A. (2012). Analysis & modeling multi-breeded Mean-Minded ant colony optimization of agent based Road Vehicle Routing Management. In *Internet Technology and Secured Transactions, 2012 International Conference For* (pp. 634–641). IEEE.
- Surowiecki, J. (2005). *The wisdom of crowds*. Anchor.
- Sysi-Aho, M., Saramäki, J., & Kaski, K. (2005). Invisible hand effect in an evolutionary minority game model. *Physica A: Statistical Mechanics and Its Applications*, 347, 639–652.
- Tabuada, P., Pappas, G. J., & Lima, P. (2004). Compositional abstractions of hybrid control systems. *Discrete Event Dynamic Systems*, 14(2), 203–238.
- Tanaka-Yamawaki, M., & Tokuoka, S. (2006). Minority game as a model for the artificial financial markets. In *2006 IEEE International Conference on Evolutionary Computation* (pp. 2157–2162). IEEE.
- Taylor, M. J., Gresty, D., & Baskett, M. (2006). Computer game-flow design. *Computers in Entertainment (CIE)*, 4(1), 5.
- Technology, K. U. of. (n.d.). Informiko Akademija. Retrieved January 14, 2017, from <http://konkursai.if.ktu.lt/>
- Tenzer, J. (2004). Improving UML design tools by formal games. In *Software Engineering, 2004. ICSE 2004. Proceedings. 26th International Conference on* (pp. 75–77). IEEE.
- Thillainathan, N. (2013). A Model Driven Development Framework for Serious Games.
- Tondello, G. F., Wehbe, R. R., Diamond, L., Busch, M., Marczewski, A., & Nacke, L. E. (2016). The Gamification User Types Hexad Scale. In *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play* (pp. 229–243). ACM.
- Van Eck, R. (2006). Digital game-based learning: It's not just the digital natives who are restless. *EDUCAUSE Review*, 41(2), 16.
- Van Grove, J. (2011). Gamification: How competition is reinventing business, marketing & everyday life. *Электронный ресурс*.—2011. Режим Доступа: [Http://mashable.com/2011/07/28/gamification](http://mashable.com/2011/07/28/gamification).
- Van Rozen, R., & Dormans, J. (2014). Adapting game mechanics with micro-machinations. In *Foundations of Digital Games*. Society for the Advancement of the Science of Digital Games.
- Vara, D., Macias, E., Gracia, S., Torrents, A., & Lee, S. (2011). Meeco: Gamifying ecology through a social networking platform. In *2011 IEEE International Conference on Multimedia and Expo* (pp. 1–6). IEEE.
- Vásquez, Ó. C., Sepulveda, J. M., Alfaro, M. D., & Osorio-Valenzuela, L. (2013). Disaster response project scheduling problem: A resolution method based on a game-theoretical model. *International Journal of Computers Communications & Control*, 8(2), 334–345.
- Verau, L. (n.d.). Contrast Ratio tool. Retrieved January 1, 2017, from

- <http://leaverou.github.io/contrast-ratio/>
- Viswanathan, S. E., & Samuel, P. (2016). Automatic code generation using unified modeling language activity and sequence models. *IET Software*, 10(6), 164–172.
- Vock, S., Schmid, M., & Von Staudt, H. M. (2006). Test Software Generation Productivity and Code Quality Improvement by applying Software Engineering Techniques. In *Test Conference, 2006. ITC'06. IEEE International* (pp. 1–8). IEEE.
- Von Ahn, L., & Dabbish, L. (2008). Designing games with a purpose. *Communications of the ACM*, 51(8), 58–67.
- Walsh, W. E., Das, R., Tesauro, G., & Kephart, J. O. (2002). Analyzing complex strategic interactions in multi-agent systems. In *AAAI-02 Workshop on Game-Theoretic and Decision-Theoretic Agents* (pp. 109–118).
- Wang, H., & Sun, C.-T. (2011). Game reward systems: gaming experiences and social meanings. In *Proceedings of DiGRA 2011 Conference: Think Design Play* (pp. 1–12).
- Wawrzyniak, K. (2011). On Phenomenology, Dynamics and some Applications of the Minority Game. University of Warsaw.
- Weiser, P., Bucher, D., Cellina, F., & De Luca, V. (2015). A taxonomy of motivational affordances for meaningful gamified and persuasive technologies.
- Wells, S., Kotkanen, H., Schlafli, M., Gabrielli, S., Masthoff, J., Jylha, A., & Forbes, P. (2014). Towards an applied gamification model for tracking, managing, & encouraging sustainable travel behaviours.
- Wendeus, J. (2013). *Designing a Viral Collaborative Tool: Patterns and Guidelines for Virality-Driven Design. MSc Thesis*. Gothenburg, Sweden.
- Werbach, K., & Hunter, D. (2012). *For the win: How game thinking can revolutionize your business*. Wharton Digital Press.
- Wilson, A. S., & McDonagh, J. E. (2014). A Gamification Model to Encourage Positive Healthcare Behaviours in Young People with Long Term Conditions. *EAI Endorsed Trans. Serious Games*, 2, e3.
- Witt, M., Scheiner, C., & Robra-Bissantz, S. (2011). Gamification of online idea competitions: Insights from an explorative case. *Informatik Schafft Communities*, 192.
- Wortley, D. (2014). Gamification and geospatial health management. In *IOP Conference Series: Earth and Environmental Science* (Vol. 20, p. 12039). IOP Publishing.
- Wozniak, M. (n.d.). Recent Possibilities of Intelligent Agents in Distributed Systems.
- Wu, M. (2011). Gamification from a company of pro gamers. *Lithosphere Community*.
- Xie, Y.-B., Wang, B.-H., Hu, C.-K., & Zhou, T. (2005). Global optimization of minority game by intelligent agents. *The European Physical Journal B-Condensed Matter and Complex Systems*, 47(4), 587–593.
- Yamamoto, N., & Ishikawa, M. (2010). Curiosity and boredom based on prediction error as novel internal rewards. In *Brain-Inspired Information Technology* (pp.

- 51–55). Springer.
- Yongfeng, Y., Bin, L., Minyan, L., & Zhen, L. (2009). Test cases generation for embedded real-time software based on extended UML. In *Information Technology and Computer Science, 2009. ITCS 2009. International Conference on* (Vol. 1, pp. 69–74). IEEE.
- Yu, H., Dömer, R., & Gajski, D. (2004). Embedded software generation from system level design languages. In *Proceedings of the 2004 Asia and South Pacific Design Automation Conference* (pp. 463–468). IEEE Press.
- Yuizono, T., Xing, Q., & Furukawa, H. (2014). Effects of Gamification on Electronic Brainstorming Systems. In *International Conference on Collaboration Technologies* (pp. 54–61). Springer.
- Žavcer, G., Mayr, S., & Petta, P. (2014). Design pattern canvas: An introduction to unified serious game design patterns. *Interdisciplinary Description of Complex Systems*, 12(4), 280–292.
- Zgonnikov, A., & Lubashevsky, I. (2012). Choice oscillations caused by boredom effect in human learning model. In *2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)* (pp. 1785–1787). IEEE.
- Zhang, H.-F., Yang, Z., Wu, Z.-X., Wang, B.-H., & Zhou, T. (2013). Braess's paradox in epidemic game: Better condition results in less payoff. *arXiv Preprint arXiv:1305.0361*.
- Zhang, Z., Wang, R., & Gao, S. (2008). Modelling financial investment planning from agent perspectives. *International Journal of Modelling, Identification and Control*, 3(1), 41–49.
- Zichermann, G., & Cunningham, C. (2011). *Gamification by design: Implementing game mechanics in web and mobile apps*. “O'Reilly Media, Inc.”
- Zuckerman, O., & Gal-Oz, A. (2014). Deconstructing gamification: evaluating the effectiveness of continuous measurement, virtual rewards, and social comparison for promoting physical activity. *Personal and Ubiquitous Computing*, 18(7), 1705–1719.

## **LIST OF PUBLICATIONS OF DARIUS AŠERIŠKIS ON DISSERTATION THEME**

### **Articles in Journals referenced in Web of Science Journal**

1. Ašeriškis, D., & Damaševičius, R. (2017). Computational Evaluation of Effects of Motivation Reinforcement on Player Retention, *Journal for Universal Computer Science*. Graz: Technische Universität Graz. ISSN 0948-695X. 2017, vol. 23, iss. 5, p. 432-453.
2. Damaševičius, R., & Ašeriškis, D. (2017). Visual and Computational Modelling of Minority Games. *TEM JOURNAL*, 6(1), 108-116.

### **Articles in Journals referenced in other reviewed scientific publications [proceedings]**

1. Ašeriškis, D., Blažauskas T., & Damaševičius, R. (2017). UAREI: a model for formal description and visual representation of software gamification.

Journal of the Facultad de Minas, Universidad Nacional de Colombia-Medellin Campus, Vol. 84, Issue 200, 326-334.

2. Ašeriškis, D., & Damaševičius, R. (2014). Gamification patterns for gamification applications. *Procedia Computer Science*, 39, 83-90.
3. Ašeriškis, D., & Damaševičius, R. (2014). Gamification of a project management system. In *ACHI 2014: The Seventh International Conference on Advances in Computer-Human Interactions* (pp. 200-207).
4. Ašeriškis D., & Damaševičius, R. (2017). Player type simulation in gamified applications, *Proceedings of the IVUS International Conference on Information Technology. CEUR Workshop Proceedings*, Vol. 1856, 1-7.
5. Ašeriškis D., Tamošaitis J. *Kitokia PĮ kompanija, Mag&Doc IT2012: Kaunas*, 2012, p. 49-52.
6. Ašeriškis D., *Projektų valdymo sistemos žaidimizavimas, IVUS 2013: Kaunas*, 2013, p. 13-16.

#### **Articles published in other reviewed scientific publications**

1. Maskeliūnas, R., Gudonienė, D., Ašeriškis, D., Blažauskas, T., Vasiljevas, M., & Drėgvaitė, G. (2016). Hybrid eLearning Model for Increasing Learner's Engagement. *Transylvanian Review*, (1).

SL344. 2017-08-29, 21,25 leidyb. apsk. l. Tiražas 12 egz. Užsakymas 262 .

Išleido Kauno technologijos universitetas, K. Donelaičio g. 73, 44249 Kaunas

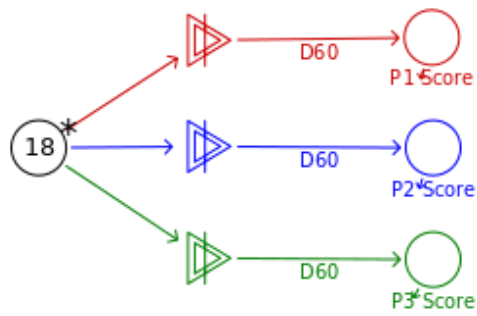
Spausdino leidyklos „Technologija“ spaustuvė, Studentų g. 54, 51424 Kaunas

APPENDIXES

APPENDIX A

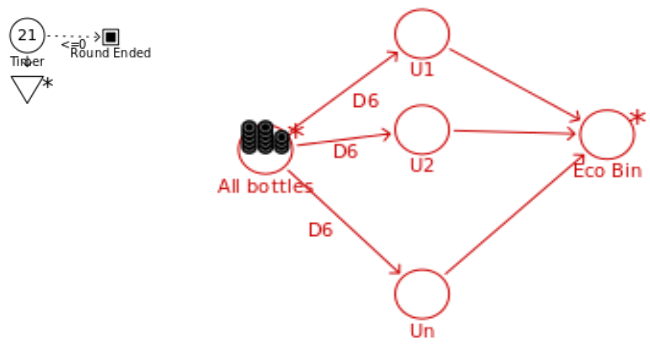
Simple models

Captchino:



Description	At its highest level Captchino is a creative approach to improve captcha user experience.
Model description	Model contains a pool with a fixed number of captchas each captcha is given one of three players, user gets a random amount of points for each captcha answered.
Patterns Identified	Limited quantity source, Random Result

Emo-bin:



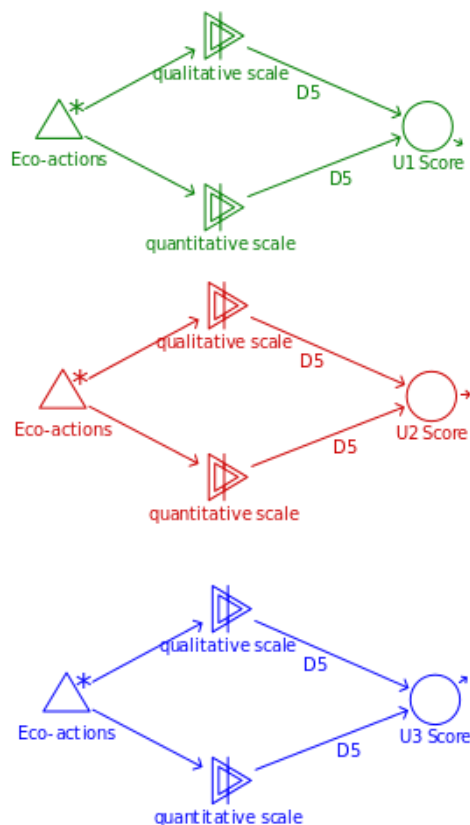
Description	Emo-bin represents a gamified recycling bin which thanks users for recycling. Emo-bin is bounded by timer which represents a day in which the results are counted.
Model description	Model contains a pool with a fixed number of bottles each bottle is given to one player. Player gets thanked for each recycled bottle.
Patterns Identified	Limited quantity source, Time limit

PowerHouse:



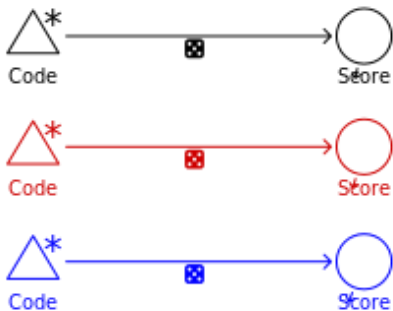
Description	Each player in PowerHouse consumes energy the one who consumes less has a better score.
Model description	Model contains a source and pool connected with random amount connection.
Patterns Identified	Infinite quantity source

Meeco:



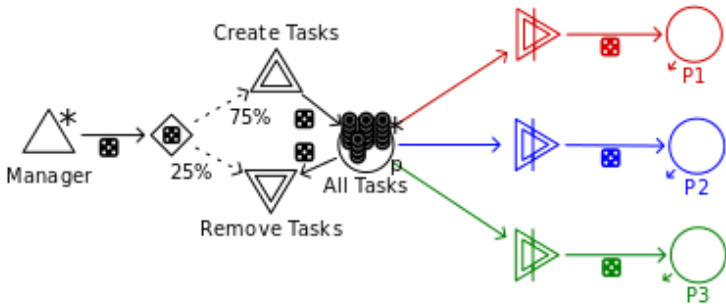
Description	Anybody in Meeco can post ecology tasks and players can solve them and other players can rate how the task was carried out.
Model description	Model contains a source connected to converters which are connected to a pool, where the connection is a random amount connection.
Patterns Identified	Infinite quantity source, Random Result

TeamFeed:



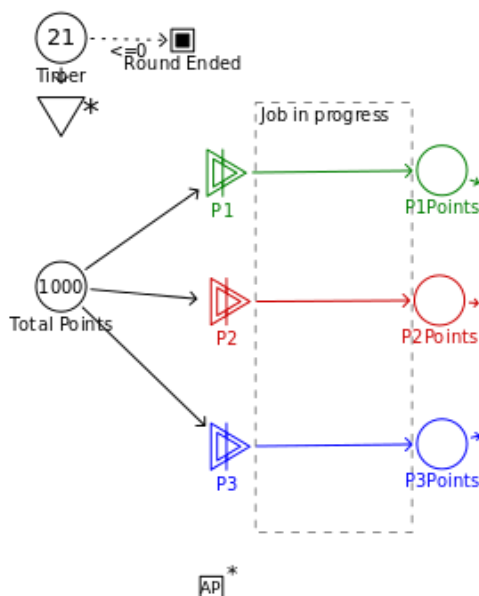
Description	In TeamFeed tracker user collects points for commits.
Model description	Model contains a infinite source connected with random amount connection to a pool.
Patterns Identified	Infinite quantity source, Random Result

TaskVille:



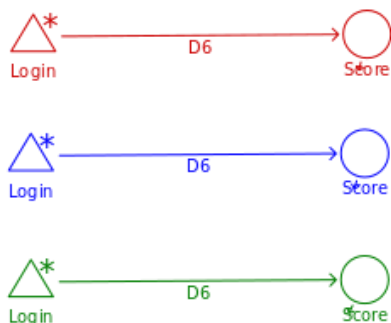
Description	In TaskVille user collects points for various solved task which are added and removed by manager.
Model description	Model contains a infinite source connected with random amount connection to a pool.
Patterns Identified	Infinite quantity source, Limited quantity source, Random Result, Dynamic limit, Drain pattern

Trogon:



Description	In Trogon user collects points for various solved task which can be represented in a pool of points.
Model description	Model contains a finite pool connected with converters. The points are randomly distributed to each player. Trogon has a time limit exposed. Total points can be replaced with dynamic limit pattern.
Patterns Identified	Limited quantity source, Time limit

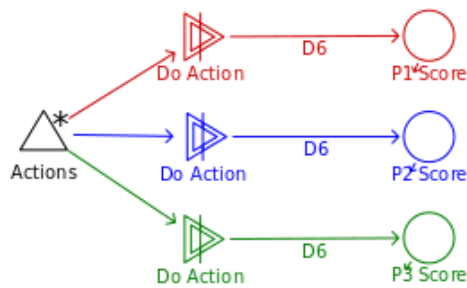
Gamified authentication:





Description	Gamified version control takes commits and represents them as points.
Model description	The model contains an infinite source of triggering converters which generate random points.
Patterns Identified	Infinite quantity source, Random Result

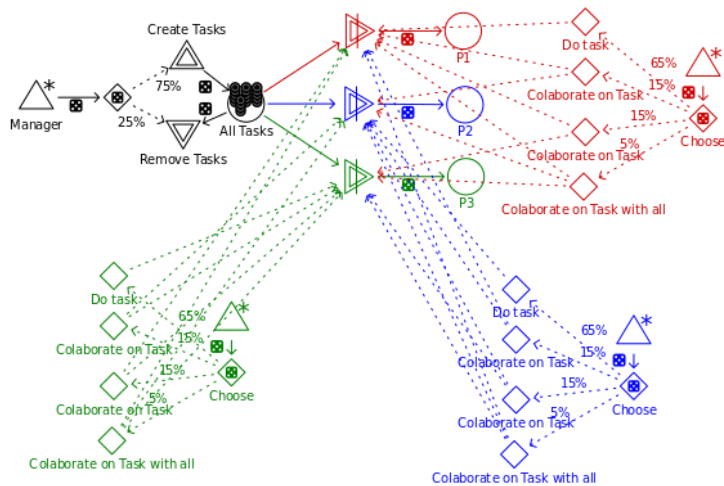
RedCritter:



Description	RedCritter is a gamified bug tracking service.
Model description	The model contains an infinite source of triggering converters which generate random points. Infinite source could be changed to limited source if the amount of issues is limited.
Patterns Identified	Infinite quantity source, Random Result, Finite quantity source

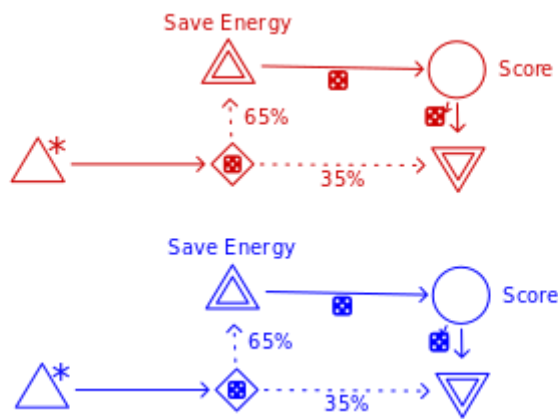
Advanced:

TaskVille:



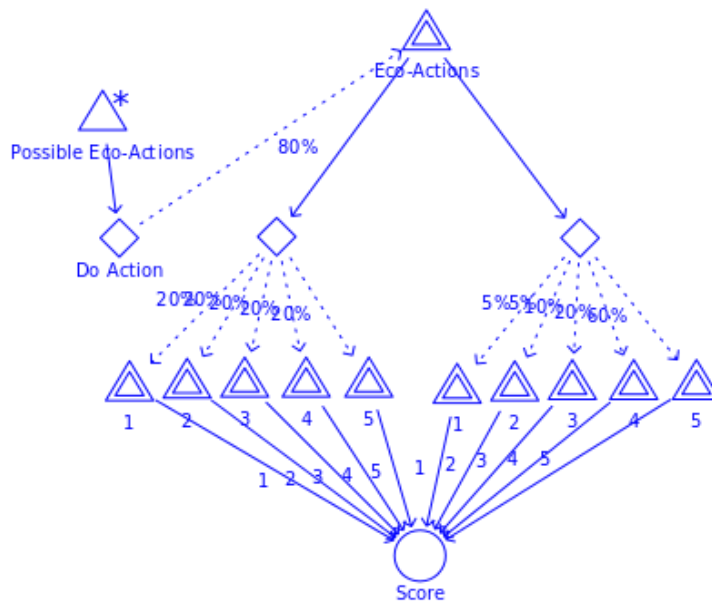
Description	In TaskVille user collects points for various solved task which are added and removed by manager. Model contains complex logic representing user specific behavior.
Model description	Model contains an infinite source connected with random amount connection to a pool.
Patterns Identified	Infinite quantity source, Limited quantity source, Random Result, Dynamic limit, Drain pattern, Property and Chance pattern

PowerHouse:



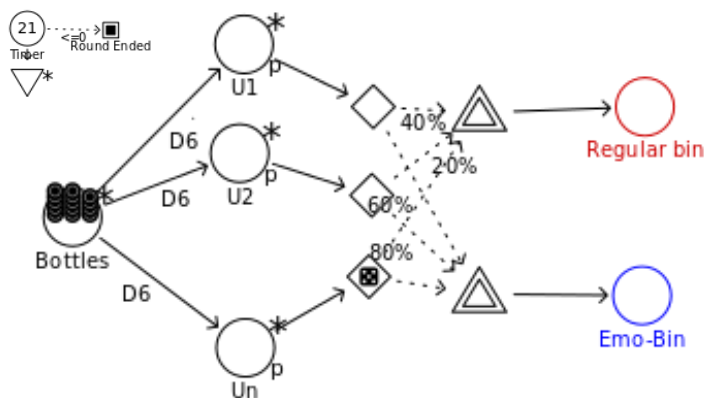
Description	Each player in PowerHouse consumes energy the one who consumes less has a better score.
Model description	Model contains a source and pool connected with random amount connection.
Patterns Identified	Infinite quantity source, Property and chance, Drain Pattern

Meeco:



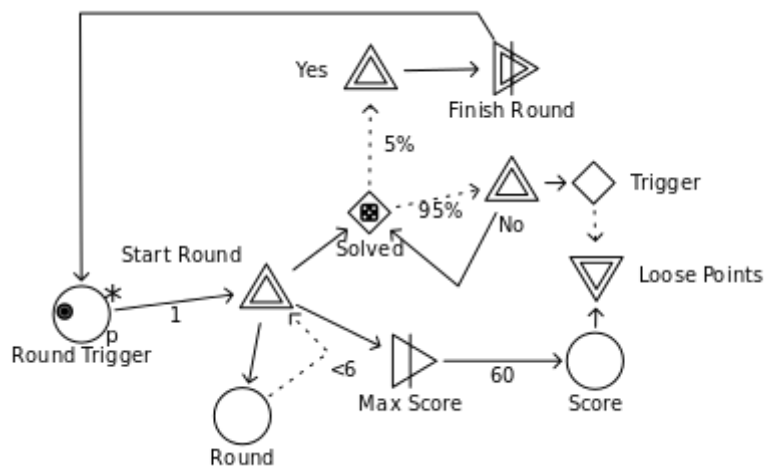
Description	Anybody in Meeeco can post ecology tasks and players can solve them and other players can rate how the task was carried out.
Model description	Model contains a source connected to converters which are connected to a pool, where the connection is a random amount connection.
Patterns Identified	Infinite quantity source, Constrain Pattern, Property and chance

## Emo-Bin:



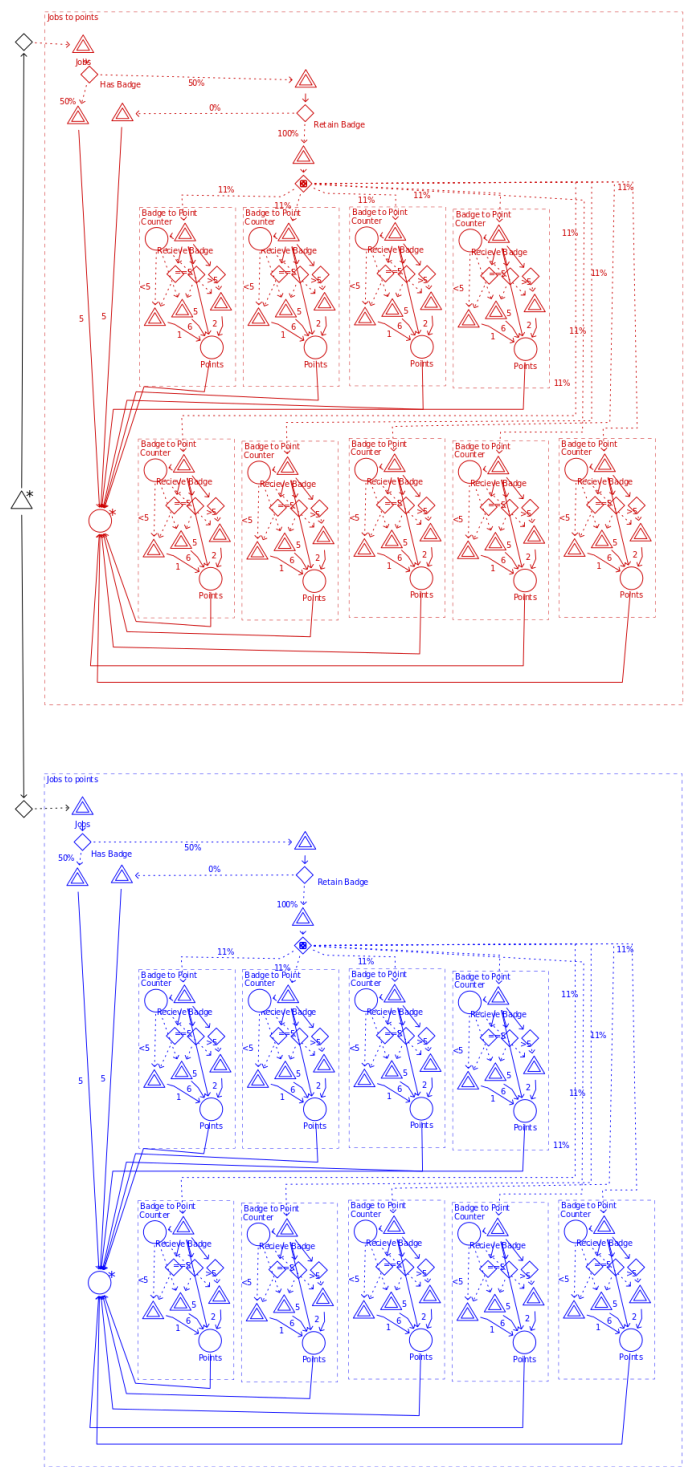
Description	Emo-bin represents a gamified recycling bin which thanks users for recycling. Emo-bin is bounded by timer which represents a day in which the results are counted.
Model description	Model contains a pool with a fixed number of bottles each bottle is given to one player. Player gets thanked for each recycled bottle.
Patterns Identified	Limited quantity source, Time limit, Random Result, Property and change

Capchino:



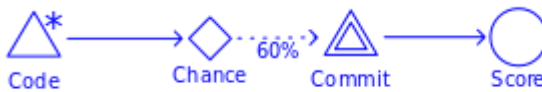
Description	At its highest level Captchino is a creative approach to improve captcha user experience.
Model description	Model contains a pool with a fixed number of captchas each captcha is given one of three players, user gets a random amount of points for each captcha answered.
Patterns Identified	Limited quantity source, Solver pattern, Property and chance, Drain Pattern, Extension pattern, Constrain pattern

Trogon PMS:



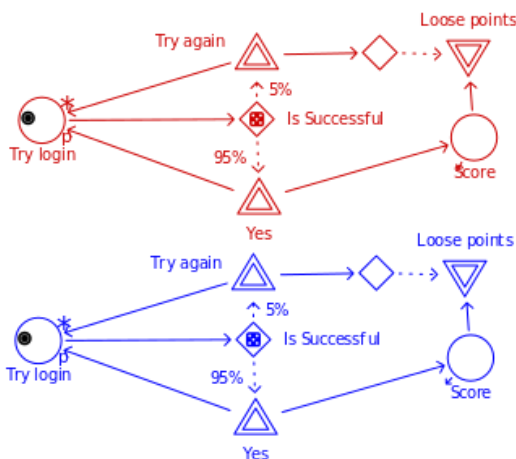
Description	In Trogon user collects points for various solved task which can be represented in a pool of points.
Model description	Model contains a finite pool connected with converters. The points are randomly distributed to each player. Trogon has a time limit exposed
Patterns Identified	Infinite quantity source, Property and chance, Extension pattern, Constrain pattern

TeamFeed:



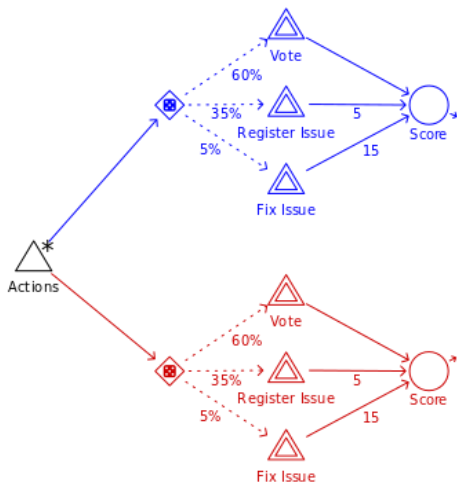
Description	In TeamFeed tracker user collects points for commits.
Model description	Model contains a infinite source connected with random amount connection to a pool.
Patterns Identified	Infinite quantity source, Property and chance pattern

Gamified Authentication:



Description	Gamified version control takes commits and represents them as points.
Model description	The model contains an infinite source triggering converters which generate random points.
Patterns Identified	Limited quantity source, Property and chance, Drain pattern

RedCriter:



Description	RedCrittter is a gamified bug tracking service.
Model description	The model contains an infinite source of triggering converters which generate random points. Infinite source could be changed to limited source if the amount of issues is limited.
Patterns Identified	Infinite quantity source, Property and chance