Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 39 (2014) 83 – 90

Procedia Computer Science

6th International conference on Intelligent Human Computer Interaction, IHCI 2014

Gamification Patterns for Gamification Applications

Darius Ašeriškis, Robertas Damaševičius*

Kaunas University of Technology, Studentų g. 50, Kaunas, LT- 51368, Lithuania

Abstract

Recently, gamification has gained popularity in the development of enterprise information systems. Gamification is usually implemented using game elements combined with game mechanics that encourage competition between game players trying to reach some objectives or quantifiable outcome. Most games contain certain common aspects that are frequently created or reinvented for each new game. Solutions to these aspects may vary system to system but they have many commonalities. The concept of design patterns, which so far have proven successful in object-oriented design and software engineering, seeks to communicate these solutions in an easy to understand manner. We extract gamification patterns from known gamified systems and describe them using the Machinations modelling tool and the pattern description scheme. A case study shows how patterns can be used in practice.

Peer-review under responsibility of the Scientific Committee of IHCI 2014

sKeywords: Gamification, modelling, game mechanics, patterns;

1. Introduction

Gamification [1] has been employed to enable attitude change and increase of 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 behaviour [2]. Concepts similar to gamification are "gameful design" or "gameful work" [3]. Recently, gamification also has gained popularity in the development of enterprise information and e-commerce systems [4].

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

* Corresponding author. Tel.: +370-670-58094; fax: +370-609-43772.
E-mail address: darius.aseriskis@ktu.edu, robertas.damasevicius@ktu.lt

Peer-review under responsibility of the Scientific Committee of IHCI 2014
doi:10.1016/j.procs.2014.11.013

objectives or quantifiable outcome [5].

Most games contain certain common aspects. Solutions to these aspects may vary system to system but have many commonalities. The concept of design patterns [6], 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 [7], game patterns [8], game design patterns [9], viral and collaborative patterns [10], etc. In this paper we argue for the *gamification* patterns and provide their textual and visual description. The novelty of the proposed gamification patterns is visual specification of patterns using domain-specific Machinations modelling language and framework [11].

The remaining parts of the paper are organized as follows. Section 2 discusses the related work. Section 3 describes gamification patterns. Section 4 presents the case study. Finally, section 5 presents conclusions.

2. Related work

The design of serious games is a complex process. Two opposing principles have to be united: achievement of serious objectives and meaningful gameplay. This can be achieved using detailed technical modelling and implementation [8]. However, the only way to really understand gamification is to identify its basic elements and model structural relationships between them. Adams and Rollings identify four basic economic functions for games: sources, drains, converters and traders [12]. Sources create resources, drains destroy resources. Converters replace one type of resource for another, where as 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 an end state is reached [13].

Gamification can be specified and modelled in many ways, e.g., with formal description [14], using textual descriptions and modelling methods, e.g., with UML diagrams [15, 16], Petri Nets [17], or other standard or custom tools [13, 18]. MDA [19] 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 behaviour 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.

Defining and formalizing structural solutions to commonly recurring problems is a 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 [6]. Kreimeier [20] proposed using game design patterns as a way to formalize and codify knowledge about game design. Bjork and Holopainen [7] 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 [8]. In this paper we introduce gamification patterns as semi-formal description of game rules for gamification of business information systems based on the analysis of commonalities in existing gamified applications.

3. Gamification patterns

3.1. Methodology

For analysis and identification of gamification patterns we have selected seven different gamified applications (Emo-bin [21], Meeco [22], Teamfeed [23], CAPTCHINO [24], Taskville [25], Power House [26], Trogon [28]). All analysed applications have common attributes: user-centric, that means what all of them have the concept of player in them; 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 leader boards. For each application we have created two types of models using the Machinations game modelling framework [11] as follows: 1) Simple model – a highest abstraction level of the system. This view shows the core system concepts. 2) Advanced model is made up of two parts: a) static model which models as many details as possible of the system, and b) dynamic model, which is

modelling interaction between players. Based on the result of model comparison and analysis we have identified common patterns of gamified systems.

3.2. Pattern description scheme

The following pattern description scheme adopted from UML pattern description [6] 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:** What are the situations the pattern can be applied?
- **Structure:** A 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 of different techniques to implement the pattern.
- **Examples:** At least two existing examples of the pattern in games. Preferably, the examples of all patterns draw from a large variety of different games.
- **Related Patterns**: What patterns are related to this pattern? Opportunities for pattern combination.
- **Discussion:** Any discussion about the pattern itself, its viability, suggestions, alternative constructions, etc.

3.3. Introduction into Machinations

To represent patterns graphically, we use Machinations, a visual modeling framework for game mechanics [27] 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 in the game structure. The Machinations diagram has four parts: nodes, connections, other elements and other concepts. There are a number of different types of 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 nodes. *State connections* determine how node state changes affect other elements. *Label types* are a part of state and resource connections passing specific control information.

3.4. Gamification patterns

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

a) b)

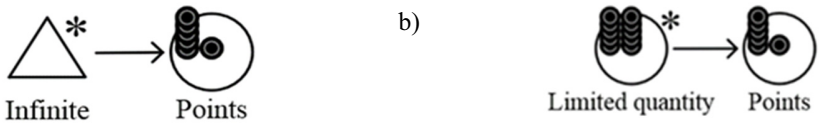


Fig. 1. (a) Infinite quantity source and (b) limited quantity source.

Infinite quantity source (see Fig. 1.a & Table 1)– it this case we choose to believe what maximum number of points is never reachable, for example the number of user actions is impossible to determine.

Limited quantity source (see Fig. 1.b & Table 1) – this imposes a system constrain that the maximum number of points received is limited at every moment of the gameplay. It can be physical or virtual limit. For example in

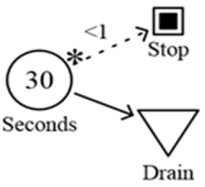
Emo-bin there are a limited number of bottles which is limited by local vending machine.

Table 1. Description of limited quantity source and infinite quantity source patterns.

Property	Limited quantity source	Infinite quantity source
Intent	Enforce a 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	Modelling an economy with limited number of resources	Modelling 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	Teamfeed, Meeco
Related Patterns	All	All

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

a)



b)

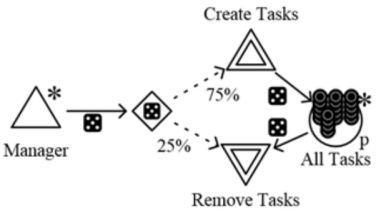


Fig. 2. (a) Time limit and (b) dynamic limit patterns.

Time limit (see Fig. 2.a & Table 2) – adds a time limit to the system. Such limit imposed over infinite quantity source will make it bounded by a time limit.

Dynamic limit (see Fig. 2.b & Table 2) – it is a limit which is imposed by model implication. For example we have a project manager in a software company checking all tickets before development and there is a chance what a ticket might not be added in to pool of tickets.

Table 2. 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 game in time.	Such pattern allows to add dynamic qualities to resources
Applicability	To impose a time restriction or rounds, for example in Trogon there is a 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 pool with label "<1".	A composition of a random gate with drain and source node connected with a pool from limited source
Participants	Pool, end condition, drain	Pool, gate, drain and source
Collaborations	Pool acts as a counter and is connected to a drain for decreasing the counter value. When counter value is equal to zero, the end condition is triggered.	Pool connects with a gate. Then follows a 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	-

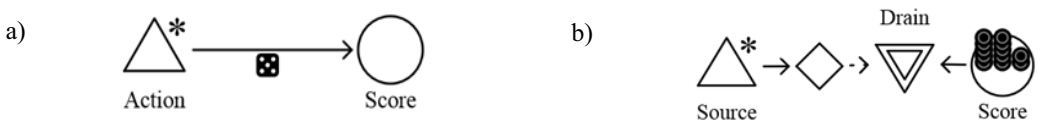


Fig. 3. (a) Random result pattern and (b) drain pattern.

Random result (see Fig. 3.a & Table 3) – 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 part of gamified system model with high level abstraction.

Drain pattern (see Fig. 3.b & Table 3) – allows decrease of score or counter under certain conditions. Drain pattern is useful to model penalty rules in the gamification systems.

Table 3. 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's 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
Related Patterns	-	Solver pattern

Fig. 4. (a) Constrain pattern and (b) extension pattern.

Constrain pattern (see Fig. 4.a & Table 4) allows to block certain paths in the model based on certain conditions.

Extension pattern (see Fig. 4.b & Table 4) is a pattern of adding an additional random path under certain conditions. This allows to extend normal behaviour with additional random bonuses.

Table 4. Description of constrain pattern and extension pattern.

Property	Constrain pattern	Extension pattern
Intent	Control flow on certain conditions	Introduce concurrent paths
Motivation	Based on system state is useful to limit or open a path according to state.	Sometimes we need to create an extension to default behaviour.
Applicability	Any system which contains multiple paths under certain conditions	Any case when default path is extended with a concurrent path.
Structure	Manual source and pool connected with 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

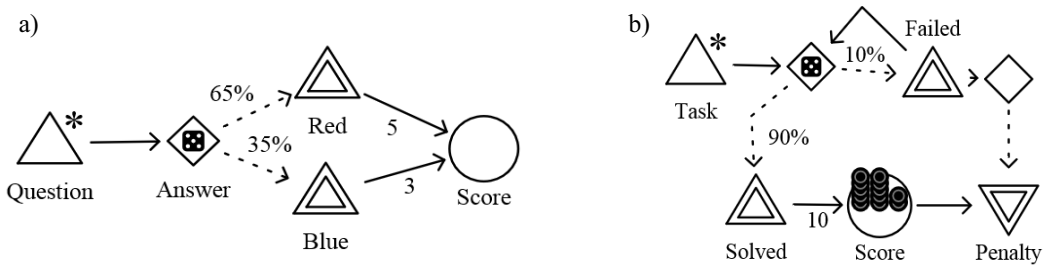


Fig. 5. (a) Property and chance pattern; and (b) solver pattern.

Property and chance pattern (see Fig. 5.a & Table 5) is a pattern for creating multiple paths or modelling 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 Fig. 5.b & Table 5) allows to model user solving a problem. Solver pattern allows to create a delay in the system.

Table 5. Description of property and chance pattern and solver pattern.

Property	Property and chance pattern	Solver pattern
Intent	Model a property or random chance	Models problem solving
Motivation	To model simple user or entity behaviour	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 behaviour	When we want to randomize the time it takes to accomplish a task
Structure	Random gate and multiple manual sources	This combines the pattern of 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

4. Case study

Trogon Project Management System (PMS) [28] is an example of enterprise Information System. The gamified PMS has a leader-board, badge board and project forest as 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 (see Fig. 6) allows observing the skills of employees. In the badge board the employees are ordered by the total number of badges collected. Each badge

represents a skill and has its own level. Progress between levels is displayed as a progress bar. 3) The project forest (see Fig. 6) provides the element of scalability to represent the size of different projects.

Fig. 6. Trogon elements: on the left game badges and badge levels on the right "project forest" page

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

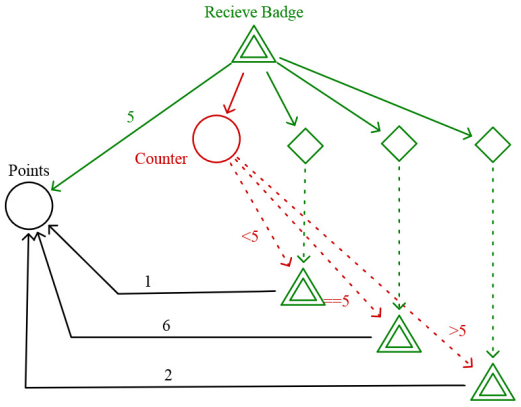


Fig. 7. Trogon PMS rule model

This model has two pattern usages: 1) Constrain pattern (red) allows to control flow depending on how many badges are received. 2) Extension pattern (green) provides the necessary paths for to model earning. Every time a badge is received a source is triggered and user points increase 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 counter. The first counter to source connection has label "<5" which means while user has bellow five badges he gets only single point. After five badges are received a new "==5" path opens and the user gets a reward of 6 points. Also the previous path "<5" is closed. When the next badge is received path ">5" opens which awards the user with two points. All other paths are closed. This workflow models the behaviour of the rules described in the previous paragraph.

5. Conclusions

In this paper, we have analyzed seven gamified systems and have identified gamification patterns common to two or more gamified applications. Each pattern has its own motivation, structure, applicability and consequences. The

patterns are modelled using the Machinations framework [11, 27]. This modelling tool allows rapid prototyping ideas and testing them before implementation.

The advantage of using gamification patterns is having an abstract formal model to unify multiple formal definitions written for different gamification applications. The abstract model is constructed from users, actions, rules, data and interfaces which are common to analysed systems. This model connected with a graph-based modelling language allows simple yet powerfull visualization. We also have demonstrated a case study of a gamification pattern combination in Machinations tool used in practice in a gamified project management system. Future work will consist of using abstract model to build a modelling tool allowing real world generation of game layers for gamified systems.

References

1. Deterding S, Dixon D, Khaled R, Nacke L. From Game Design Elements to Gamefulness: Defining 'Gamification'. MindTrek2011, 9–15.
2. Huotari K, Hamari J. Defining Gamification - A Service Marketing Perspective. Proc. of the 16th International Academic Mindtrek Conference, Tampere, Finland, October 3-5, 2012, 17-22.
3. McGonigal J. Reality is Broken. Why Games Make Us Better and How They Can Change the World. The Penguin Press, 2011.
4. Hamari J. Transforming homo economicus into homo ludens: A field experiment on gamification in a utilitarian peer-to-peer trading service. Electronic Commerce Research and Applications, 2013, 12(4): 236-245.
5. Deterding S. Gamification: designing for motivation. Interactions 2012; 19, 14-17.
6. Gamma E, Helm R, Johnson R, Vlissides J. Design patterns: elements of reusable object-oriented software. Pearson Education, 1994.
7. Björk S,Holopainen J. Patterns in Game Design. Charles River Media, 2004.
8. Kelle S.,Klemke R.,Specht M.Design patterns for learning games. International Journal of Technology Enhanced Learning, 3(6), 555-569, 2011.
9. Kiili K. Call for learning-game design patterns. A chapter in the book: Educational Games: Design, Learning, and Applications. Nova Publishers, 2010.
10. Wendeus JC. Designing a Viral Collaborative Tool: Patterns and Guidelines for Virality-Driven Design. MSc Thesis, Gothenburg, Sweden, 2013.
11. Adams E, Dormans J. Game Mechanics - Advanced Game Design. New Riders Games, 2012.
12. Adams E, Rollings A. Fundamentals of Game Design. Upper Saddle River: Pearson Education Inc., 2007.
13. Grunvogel, SM. Formal Models and Game Design. Game Studies, 2005. Online http://gamestudies.org/0501/gruenvogel/
14. Bista, Sanat K, Surya, NC, Paris C. Using gamification in an online community. In Proc. of 8th International Conference on,Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), 2012, 611-61.
15. Taylor MJ, Gresty D, Baskett M. Computer Game-Flow Design. ACM Computers in Entertainment, 2006; 4(1).
16. Dormans J. Visualizing Game Dynamics and Emergent Gameplay. Proceedings of the Meaningful Play Conference, 2008.
17. Araujo M, Roque L. Modeling Games with Petri Nets. In Breaking New Ground: Innovation in Games, Play, Practice and Theory. Proceedings of the 2009 Digital Games Research Association Conference, 2009.
18. Koster R. A Grammar of Gameplay: game atoms: can games be diagrammed? Presentation at the Game Developers Conference, 2015. Online http://www.raphkoster.com/gaming/atof/grammarofgameplay.pdf
19. Hunicke R, LeBlanc M, Zubek R. MDA: A formal approach to game design and game research. In Proc. of the AAAI Workshop on Challenges in Game, 2004.
20. Kreimeier B. The Case for Game Design Patterns. Gamasutra, 2002, Online www.gamasutra.com/features/20020313/kreimeier_01.htm.
21. Berengueres J, Alsuwairi F, Zak, N, Ng T. Emo-bin: How to Recycle more by using Emoticons. In Proc. of 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI), 2013, 397-397.
22. Vara D, Macias E, Gracia S, Torrents A, Lee S. Meeco: Gamifying ecology through a social networking platform. In Proc. of IEEE International Conference on Multimedia and Expo (ICME), 2011, 1-6.
23. Singer L, Schneider AK. It was a bit of a race: Gamification of version control. In 2nd International Workshop on Games and Software Engineering (GAS), 2012, 5-8.
24. Saha R, Manna R, Geetha G. CAPTCHINO - A Gamification of Image-Based CAPTCHAs to Evaluate Usability Issues. In Proc. of the 2012 Int. Conference on Computing Sciences (ICCS '12), 2012, 95-99.
25. Shawn N, Sundaram S, Linn H, Kelliher A. Playing in Taskville: Designing a Social Game for the Workplace. In CHI 2011 Workshop on Gamification: Using Game Design Elements in Non-game Contexts, 2011.
26. Reeves B, Cummings JJ, Scarborough JK, Flora J, Anderson D. Leveraging the engagement of games to change energy behavior. In Proc. of International Conference onCollaboration Technologies and Systems (CTS), 2012, 354-358.
27. Dormans J. Machinations Diagram Tutorial. Portfolio of Joris Dormans, 2013. Online http://www.jorisdormans.nl/machinations/wiki/index.pshp?title=Tutorial_1
28. Ašeriškis D, Damaševicius R. Gamification of a Project Management System. Proc. of Int. Conf. on Advances in Computer-Human Interactions (ACHI2014), 2014, 200-207.