Making interactive board games to learn: Reflections on AnyBoard

Simone Mora[*], Tomas Fagerbekk[*], Ines Di Loreto[**], Monica Divitini[*]

[*] Dept. of Information and Computer Science, NTNU, Trondheim, Norway
{simone.mora, divitini}@idi.ntnu.no
[**] TechCICO, ICD-Université de technologie de Troyes, France
ines.di_loreto@utt.fr

ABSTRACT. In this paper we discuss the making of interactive board games as a learning activity. We do this by presenting AnyBoard, a platform that we are currently developing to support the design and implementation of board games. In our approach we do not use a game board virtualised on an interactive surface, but rather achieve interactivity through technology-augmented game pieces. In this way, we aim at offering to game designers a broader design space and lower costs of the final product. In this paper we discuss the possible use of AnyBoard in the learning context.

1 Introduction

Making games, either analogic or computer-based, has long been used as a learning activity in different educational contexts. Making games has proved useful in areas as diverse as engaging students with cultural heritage [1] and teaching university students about software architectures [2]. Teachers have used making games as a way for teaching programming in high or middle schools for many years, for example using RPG maker[1] or Scratch[2].

In this context, we want to discuss the making of interactive board games to promote learning. With the term interactive we mean board games that use tangible computer-augmented objects. There are different benefits that could be achieved, mainly combining the learning strengths of creating games with the strengths of making tangible and interactive objects for educational purposes [3, 4]. In addition, board games are popular also among the elderly, offering a cross-generational form of entertainment. This is an aspect that could be exploited to create cross-generational maker activities. Finally, by making board games the learner does not get distracted by the complex graphics that is common to many video games. In this way, it is easier for the learner to concentrate on the game concept.

To ease the development of digital board game we present *AnyBoard*, a framework for supporting the making of interactive board games. *AnyBoard* supports the design of digital board games by providing theoretical constructs, software tools and a set of augmented game pieces (all currently under development). The platform was not originally

[1] RPG Makers - https://en.wikipedia.org/wiki/RPG_Maker
[2] The Scratch project – http://scratch.mit.edu

designed to be used in the educational context, but mainly targeting maker communities. However, it could potentially be useful in the context of educational maker activities. In this paper, after presenting the *AnyBoard* approach, we discuss the challenges connected to the use of this platform for learning building on our experience on the creation of an interactive board game.

2 The AnyBoard framework

The dominant paradigm for creating digital board games consists in designing games for interactive surfaces such as smartphones, tablets or tabletop computers. In some cases, artefacts that resemble game tokens, yet augmented with markers (e.g. barcodes, RFIDs), can facilitate interaction with the interactive surface [5, 6].

We propose a different approach: the game pieces are the means to bring interactivity, rather than the game board virtualised on an interactive surface. Distributing interactivity across multiple components opens for a wider space of possibility in designing game experiences. For example, game pieces can influence the state of a game not only when they sit on an interactive surface, but also when they are manipulated over and around it. In this way, the board is mainly used to stage the game and set a context for the use of the pieces, as in traditional board games. In addition, the interactive area of the board is less limited by size, which also determines the portability of the game and costs.

2.1 A new perspective on digital board games

In our approach to digital board games the role of technology is twofold. On one side it brings interactivity by augmenting, not virtualizing, pieces' material representations, for example we aim at providing developers with tangible game pieces augmented with visual, audio or haptic feedbacks (e.g. by means of LEDs or displays). On the other side sensor technology is used to capture players' physical interaction with pieces aiming at preserving their traditional physical affordances; for example, to sense the result of a dice throw, or the movements of pieces onto the board.

Game pieces still preserve their traditional aspect, having a tangible representation that complements an intangible one provided by technology. For example, in a revisited version of Monopoly tokens might preserve their physical semblances to identify players but might embed a graphical representation of the number of property owned by the player (e.g. in icons or symbols on a LCD display). The intangible representation is kept updated by a computer game engine during the playtime, as a consequence of players' interaction with game pieces and activation of game rules. The interaction with pieces is based on a double loop [7].

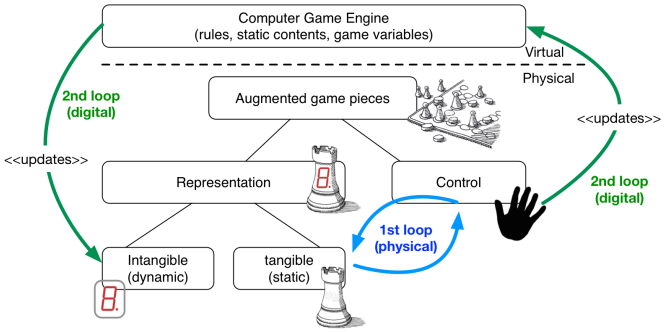

Fig. 1. Double interaction loop in interactive board games

A first interaction loop consists in the passive haptic and visual feedback the player perceives when manipulating pieces on the board, this loop is in common with traditional board games. A second loop adds interactivity by means of graphical and auditory feedbacks conveyed via the tokens' intangible representation (Figure 1). This loop requires technology for sensing tokens' manipulations as well as providing visual/audio feedbacks. The set of valid interactions with game pieces are defined by the affordances of pieces and by game rules. To formalize these rules we build on two theories: the T+C framework [8], providing a powerful descriptive language, and the MCRit model [9], addressing issues of representations and control in TUI.

2.2 Key design constructs

We define a game, which is composed by *game dynamics* (the sum of game logic and rules), as a sequence of player-initiated *interaction events* that modify *spatial configurations* of *tokens* with respect to board *constraints* and other tokens. In the following, building on the T+C framework, we describe key constructs required to develop an *Anyboard* game.

Tokens are technology-augmented artifacts capable of triggering digital operations that can activate game dynamics. They are an augmentation of traditional pawns, dice and cards. Tokens may be capable of sensing information (e.g. proximity with other tokens) and displaying computer graphic and sound.

Constraints are confining regions in the board space, for example checks in the Chess game and territories in Risk. The association or dissociation of a token within a constraint can be mapped to digital operations to activate game dynamic. Constrained regions are determined by a perimeter that could be visual, or physical.

Interaction events are player-triggered manipulations of tokens, that modify the (digital and physical) state of a game. We identified three types of events.

Solo-token event (T) - the manipulation of a single token over or on the board. For example, the action of rolling a dice or drawing a card.

Token-constraint event (T-C) - the operation of building transient token-constraint associations by adding or removing tokens to a constrained region of the board. T-C events can have different consequences depending on game rules.

Token-token (T-T) event - the operation of building transient token-token adjacency-relationships, achieved by moving tokens on the board. For example, approaching a token next to a different token to unlock special powers, or to exchange a resource between two players.

Sequence of *interaction events*, validated against game-specific rules, activate game dynamics and allow the game to evolve from a state to another. For example, we can model the act of capturing a piece in chess as a sequence of interaction events that modify proximity between two chess tokens within checkers constraints. For more details, see [10].

In the following section we describe how theoretical contracts have been implemented for the augmentation of an existing board game.

2.3 An example

Don't Panic, is a collaborative game inspired by Pandemic[3]. Four players start the game as member of a panic manager team that must work together to manage panicking crowds. A map representing a city map is displayed on the game board and the territory is divided in *sectors*. Each sector contains a number of people (PO) characterized by a panic level (PL). During the game randomly triggered panicking events (e.g. fires, explosions) increase PLs in determinate sectors. Each player is represented on the board space by a personal *pawn token* and gets a limited number of actions with the goal to lower the panic level in the city. Using the public *"Calm!" and "Move!" tokens* a player can either reduce the panic in a specific sector or move panicked people to an adjacent sector. Information *cards tokens* distributed in each turn can lower the panic in multiple sectors. Players collectively win the game when the PL in all sectors is zero. For a full description of game rules see [11].

Don't Panic is composed by a cardboard and a set of tokens:

The board (Figure 2-a) – is a cardboard that visualizes a map portraying a territory divided in nodes, sector and paths. Nodes feature physical *constraints* and no degree of freedoms for the hosted tokens; sector and paths provide visual *constraints* allowing tokens' translation and rotation, within the perimeter.

The card deck (Figure 2-c) – dynamically print information card *tokens*. Each card has a textual description of how it affects the game and a barcode that links the card to its digital representation. The top surface of the card deck can read the barcode on the card and trigger actions in the game (Figure 2-d).

Pawn tokens (Figure 2-b) – embody the players' presence on a node. Pawns can be moved from node to node and provide visual information via a LCD display. These include the role of the player, number of people present in sectors adjacent to each of the four pawn's sides; and their panic level.

The Calm! token – represent the field action of calming people talking to them, thus reducing the PL in a specific sector. This action is activated when Calm! is bumped towards a Pawn token (Figure 2-e).

[3] Pandemic board game - http://zmangames.com

The Move! token – simulates moving people across sectors, in this way people moved acquire the panic level of the recipient sector. This action is activated by dragging Move! across a border between two sectors (Figure 2-f).

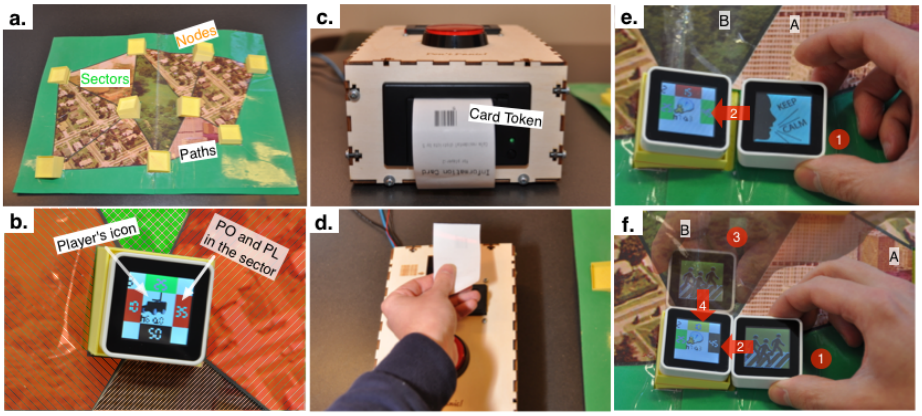

Fig. 2. Don´t Panic interactive tokens

3 The AnyBoard software library

We present here a library to bridge the gap between the theoretical constructs reported in section 2 and the making of interactive game pieces.

Game design communities, game developer environments and game engines already exist, and hence the main part that makes the AnyBoard framework unique is helping integrate the development interactive tokens interaction as a part of games (Figure 3). This role is performed by the *Token Manager* library.

The *token manager* provides a *Token API* to available game engines, so developers can listen to player-token events and send commands to the interactive tokens without the knowledge of the low level code or the tokens' hardware. The API provides primitives specifically suited for augmented board games, such as Token, Constraint and Interaction Events (Section 2.2). The API is generic enough to be used with popular game engines, both commercial and open source, such as Phaser[4] and Unity[5].

On the other end, the *token manager* is separated from any specific hardware implementation, and communicates with the physical tokens through *device-specific drivers*. Besides a set of tokens is provided as part of the framework, expert users can tinker them or build new tokens using popular toolkits such as Arduino and RaspberryPi. Drivers for the Arduino platform as well as a generic extendable driver will be provided to assist developers that wish to create their own tokens with specific technologies.

[4] Phaser game engine – http://phaser.io
[5] Unity game engine – http://unity3d.com

The AnyBoard library builds on the Apache Cordova[6] platform that enable games made for AnyBoard to compile to different operating systems, including mobile ones such as Android and iOS. We aim to use open source, free-to-use, modular and well documented tools, so that a developer can pick apart the AnyBoard system and add capabilities where need be.

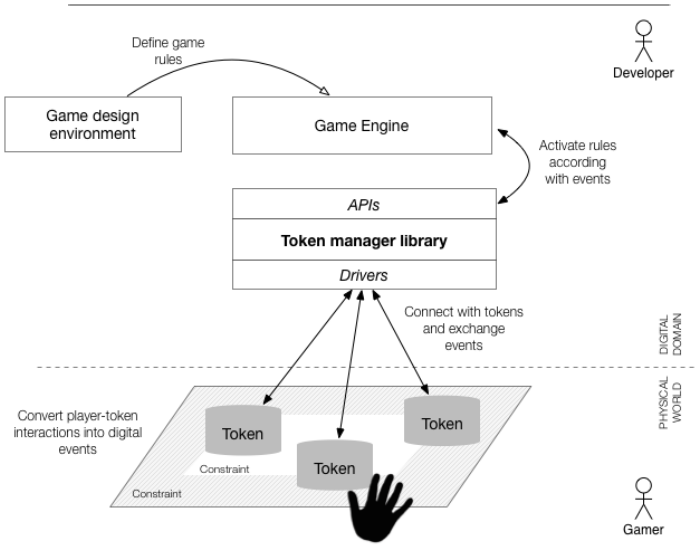

Fig. 3. High level components of AnyBoard

Standard example games, and templates implementing typical token capabilities, will be provided for developers that wish to create games with general token requirements.

Finally, a web-based community for AnyBoard is intended to grow a community and provide information for all roles involved with augmented board games. The AnyBoard platform will be available from there, and tokens sold from a third-party or made using prototyping techniques and open source schematics. The community will provide a knowledge base and tools for developers. Furthermore, it will feature a repository of ready to use *Anyboard games* and an assistive IDE for game.

4 AnyBoard for learning

Making an interactive board game with AnyBoard requires different competencies, varying from game design to software and hardware development skills, and it therefore opens for the design of learning activities with different and multiple learning objectives. In this section we reflect on how the platform, when fully developed, could be used for learning.

[6] Apache Cordova Platform - https://cordova.apache.org

The phases that are required for the full development of an AnyBoard game include:

- Game design, i.e. the definition of the game concept, logic and rules
- Interaction game design, i.e. the definition of the interactions of the players with the game tokens and the interaction among players, either directly or mediated by game elements
- Mapping of the game into the associated token+constraints system
- Implementation of hardware and software, this might range from implementation of the game engine to the development of the token interactions. This phase might also include the production of objects that are not computerized, like the board and cases to tailor the appearance of tokens, e.g. using 3D-printing.

The different phases allow to explore different learning goals through adequate activities. The design of the learning activity might focus on one or more of the following learning area:

- Specific subjects. If the game is designed as a serious game, i.e. by playing the game players are expected to learn X, it requires that students gain a knowledge of subject X to inform their design. In this perspective, the implementation part might be less relevant and the main focus is on the first phase of game design.

- Interaction design by designing the intended interactions among players and the interaction with the tokens. It should be noted that the actual design of the tokens appearance and interaction is strictly related to the game design. It can also become a way to learn about the game subject. For example, in developing a game for crisis management, one could work on tokens that resemble actual objects in the domain, mirroring their behavior in the real world.

- Abstract thinking/logic. To achieve this learning goal, in addition to the high level design, one should put focus on the translation of the high level rules into the framework constructs in terms of tokens and constraints.

- Coding. Learning to code can be achieved during the implementation phase. This might include both more traditional coding for the game engine and coding for embedded systems. In this way, different computational approaches, languages, and feedback systems might be explored.

- Tinkering. The design and implementation of the game requires to play around with different software and tangible components.

5 Conclusions: towards a revised framework

In this short paper we presented an innovative approach to design of interactive board games. The approach is based on the use of interactive tokens on an analogic surface, in alternative to current approaches that mainly rely on interactive surfaces. This approach, we claim, might be suitable to be used in the context of learning by making. The paper discusses the potential of the framework for learning.

To realize this vision there are a number of components that should be added to the framework, including:

- Graphical interface for coding, hiding if required by the design of the learning activity, the complexity of moving from high level rules to the token+constraint system.
- Templates for learning activities with different learning objectives. These templates should help the organizers of the learning activities to quick-start the design, choosing activities that reflect the intended learning objective.
- Scaffolding, possibly including support to hide complexity or irrelevant parts of the platform. This can be achieved in different ways at different level of complexity. It should be taken into account that board games might have a lot of objects and very complex rules that can be overwhelming for a non expert. At the same time, though learners are getting less distracted by developing graphics than in a traditional video games, still the development of the tangible parts might become very complex and distract the learner from other possible learning objectives.
- Community support oriented to education
- Analytics for reflection

As part of our future work we aim at developing these components following a learner-centered approach. This will require to identify more in detail the benefits of this approach compared to other types of game development for learning to focus the development of the component necessary for the more suitable learning activities.

REFERENCES

1. Yiannoutsou, N. & Avouris, N.: Game Design as a context for Learning in Cultural Institutions. In C. Karagiannidis, P. Politis, & I. Karasavvidis, eds. *Research on e-Learning and ICT in Education*. New York, NY: Springer New York, pp. 165–177, (2014).
2. Wang, A.I. & Wu, B.: Using Game Development to Teach Software Architecture. *Int. J. Comput. Games Technol.* (2011)
3. Horn, M.S., Crouser, R.J. & Bers, M.U.: Tangible interaction and learning: the case for a hybrid approach. *Personal and Ubiquitous Computing*, 16(4), pp.379–389 (2012).
4. Mellis, D.A. & Buechley, L.: Case studies in the personal fabrication of electronic products. In Proceedings of the Designing Interactive Systems Conference (DIS2012), ACM Press, pp. 268-277.
5. Haller, M., Forlines, C., Koeffel, C., Leitner, J., Shen, C.: Tabletop games: Platforms, experimental games and design recommendations. Art and Technology of Entertainment Computing and Communication. 271–297 (2010).
6. Bakker, S., Vorstenbosch, D., Van Den Hoven, E., Hollemans, G., Bergman, T.: Tangible interaction in tabletop games. In Proc. of ACE 2007. 163–170 (2007).
7. Ishii, H.: Tangible bits: beyond pixels. In Proc. of TEI 2008. (2008).
8. Ullmer, B., Ishii, H., Jacob, R.J.K.: Token+constraint systems for tangible interaction with digital information. ACM Transactions on Computer-Human Interaction (TOCHI). 12, (2005).
9. Ullmer, B., Ishii, H.: Emerging frameworks for tangible user interfaces. IBM systems journal. 39, 915–931 (2000).
10. Mora, S., Di Loreto, I., Divitini, M.: The interactive-token approach to board games. In Proceedings of AMI2015, LNCS, Springer (to appear).
11. Di Loreto, I., Mora, S., Divitini, M.: Don't Panic: Enhancing Soft Skills for Civil Protection Workers. In Proc of SGDA. 7528, 1–12 (2012).