




Toward a Meta-design Method for Learning Games

Marne Bertrand¹^a, Muratet Mathieu^{2,3}^b and Sehaba Karim⁴^c

¹ICAR UMR 5191, Université Lumière Lyon 2, France

²Sorbonne Université, CNRS, LIP6, F-75005 Paris, France

³INS HEA, 92150 Suresnes, France

⁴Université de Lyon, CNRS. Université Lyon 2, LIRIS, UMR5205, F-69676, France

Keywords: Serious Games, Meta-design, Computational Thinking, Authoring Tools.

Abstract: To support the appropriation of serious learning games by teachers, we are studying design methods that focus on both design and use phases: meta-design methods. Our objective is to propose models and tools for designing levels and scenarios for Blockly Maze and to provide the teacher with monitoring indicators allowing him/her to appropriate and adapt the game according to the observed uses. In this article, we detail the first contributions of this study based on qualitative data: analysis of the game Blockly Maze and interviews with two teachers. First results draw three main needs: level design tool, scenario design tool and monitoring tool. Beyond these needs we introduce underlying models of each of them and future works.

1 INTRODUCTION

Learning games offer significant benefits over traditional teaching tools (Freitas, 2006; Prensky, 2004). Indeed, many authors consider learning games as promising, especially to increase learner engagement (Bouvier et al., 2014) and motivation (Garris et al., 2002; Malone and Lepper, 2005), others are more interested in modeling and evaluating player experience (Kiili, 2005; Sweetser and Wyeth, 2005) or in using learning games to promote a more constructivist learning (Bogost, 2007; Marne, 2019; Ryan et al., 2012). However, their adoption, especially by teachers, remains scarce (Li, 2018; Sardone, 2018). To foster the adoption of learning games by teachers, we work on the hypothesis that a participatory design method such as meta-design might be suitable.

Meta-design is an advanced participatory design method, in which the design process is centred on the end users (“owners of problems”). However, end users must continue to have the means to design during the artefact use phase (Fischer and Herrmann, 2011). This is made possible, among other things, by the *underdesign* that Fischer et al. (Fischer et al., 2004) define as:

“[...] underdesign aims to provide social and technical instruments for the owners of problems to create the solutions [of their problems] themselves at use time.”

Our work questions this approach in the context of the use of serious games by teachers as *owners of problems*.


The work presented in this paper reports the beginning of a project on this topic. We focused our study on the teaching of computational thinking, and the implementation of a meta-design approach for learning games. This project aims to exploit an existing learning game and to study to what extent it is possible for teachers to “appropriate” it (*underdesign*). We investigated how allowing them to modify its content according to their pedagogical needs and observed or intended uses. We chose Blockly Maze¹ (BM), among other things, because it is well-tested and free software. Indeed, inspired by Scratch (Resnick et al., 2009), BM has been reused in many derivative works. For instance, the *Hour of Code*², *Algorea*³, and more than 400 derivatives on the forge *GitHub*⁴. The fact that the source code is available,


¹Blockly Maze is a learning game developed by Google: <https://blockly.games/maze> consulted on 2021/02/03


²<https://code.org/> consulted on 2021/02/03

³<https://algorea.org/> consulted on 2021/02/03

⁴<https://github.com/google/blockly->

^a <https://orcid.org/0000-0002-4953-9360>

^b <https://orcid.org/0000-0001-6101-5132>

^c <https://orcid.org/0000-0002-6541-1877>

freely reusable, and widely tested has particularly interested us to design and evaluate methods and tools allowing meta-design and *underdesign* for this learning game.

The limits of the original Google version of Blockly Maze are the same limits as those found in many other learning games: (1) **limited number of levels**, Blockly Maze offers 10 levels covering a few programming skills; (2) **non-modifiable scenario**, the course of these 10 levels is a linear sequence that the teacher cannot modify; (3) **the players' activity is not recorded**, hence the teacher cannot retrieve any trace of the learners' performance. For these reasons, it is difficult for the teacher to understand the practices and difficulties of the learners and thus to appropriately adapt the learning game to the pedagogical context.

Our goal is to provide tools and methods enabling teachers, more or less comfortable with teaching computer science, to use Blockly Maze, to build their own sequences of levels corresponding to their needs, to build levels that complement those available, and to follow the learners' activity to better understand their use and thus adapt learning. Therefore, our ultimate goal is to build a meta-design context based on a generic model for the design and development of levels.

Our first step is the deconstruction of Blockly Maze, both on a conceptual perspective and on a software one. Thus, we intend to develop explicit models of the levels and their educational content. This paper describes this first part of our work.

In the first part of the paper, we present Blockly Maze and the scientific issues we are studying, as they emerge from the different obstacles arising from the literature and the practice of teaching computational thinking. In the second part, we present our methodological context and our choices. Finally, before concluding, we present our first results in the third part.

2 INTRODUCING BLOCKLY MAZE AND OUR SCIENTIFIC PROBLEMS

In order to work on meta-design, we chose an open source learning game (source code available, freely reusable, and modifiable). We chose Blockly Maze because of the popularity of the Blockly engine, but also because of the wide adoption of Blockly Maze (BM) by many organizations such as the *Hour of Code* or *Algoréa*. The concept of this learning

games/network/members consulted on 2021/02/03

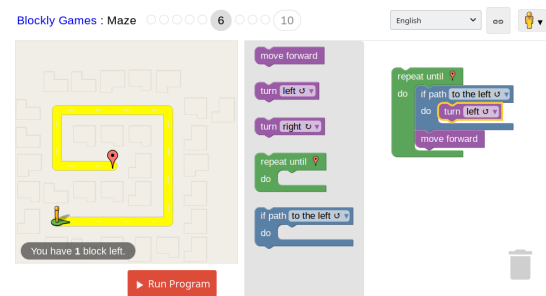


Figure 1: Screenshot of Blockly Maze showing block programming to guide the avatar toward the exit.

game developed by Google is to teach programming through the guidance of an automatic avatar in a maze (see figure 1). The guidance is done by programming with blocks of instructions, inspired by Scratch (Resnick et al., 2009).

However, the original BM from Google only offers 10 levels. They train to the use instructions sequences, conditions (*if/else*), loops (*repeat/until*), including nested ones, and basic algorithmic (*left-hand method*). This is rather limited considering the various needs of teaching computational thinking (Brunet et al., 2020; Vandeveldde and Fluckiger, 2020). Thus, many derived works (*forks*) and learning games similar to BM have emerged, offering numerous other levels. We mentioned the *Hour of Code*, which offers a large selection of additional levels dealing with multiple aspects of computational thinking distributed in several “courses”. The “Hour of Code” has a wide choice of scenarios (combining other types of learning games than BM), which means that, for a teacher, an informed choice of the resource or course to use requires a global vision of the available options, and therefore also requires a significant amount of time to grip the whole platform. We also mentioned *Algoréa* (Léonard, 2020), it offers learning games that are close to BM (although, apparently on a different code base). Like the *Hour of Code*, these learning games are provided with a variety of other educational resources, structured to prepare a French programming competition: *Algoréa*. The learning games provided are puzzles that can be solved with blocks like in Blockly (in addition to the Python code and Scratch blocks). Nevertheless, only a few of them use a maze as puzzle. Those maze games only focus on a few aspects of computational thinking: instruction sequences, function calls, simple or nested loops, conditional instructions. The learning games based on block programming which address more complex notions are not based on maze puzzles.

In any case, it seems difficult for a teacher, who is familiar with these three resources, to build a curriculum for his or her learners by picking out what is

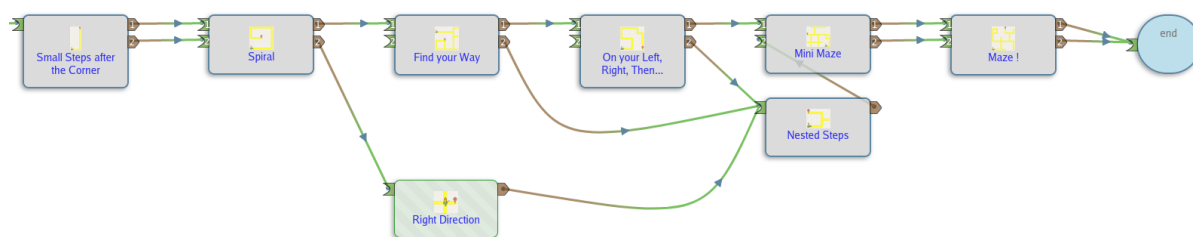


Figure 2: Screenshot of a sample Blockly Maze scenario provided by APPLiq.

needed for them. Moreover, the resources are not designed for this kind of flexibility, but rather to be used in the form of indivisible sequences of several levels.

Given these limitations, our problem is therefore to provide teachers with the proper tools and methods to be able to appropriate BM. On the one hand, by giving them the opportunity to understand how BM works and how it is structured (*instrumentation*), and, on the other hand, by enabling them to adapt, modify, remix BM and its levels (*instrumentalisation* and *meta-design*) (Rabardel, 2003).

To foster appropriation of BM by teachers through mastery and adaptation, we relied mainly on the meta-study of Dermeval et al. (Dermeval et al., 2018) to explore available authoring tools. Our goal was to find out which ones could be used to adapt the levels or the succession of levels of BM. However, BM is not intended to work with such tools. In our investigations, beyond this meta-study, we only found the authoring tool *APPLiq* which can be used to adapt a BM scenario, as it has already been tested with it by its author (Marne and Labat, 2014; Marne, 2014).

APPLiq enables teachers/users to prepare and provide learners/players with a succession of levels (called *activities*) in a learning game, taking into account the pre-requisite and worked on pedagogical objectives at each of these stages. This succession of levels is not necessarily linear (see figure 2), and may therefore depend on the actions and performance of the learner/player (objectives worked on or not worked on during the activities).

Nevertheless, *APPLiq* has several limitations. In the context of our study, the most important limitation is that this authoring tool only allows the adaptation of the levels *order*. Therefore, while it enables teachers/users to change the sequence and order of the levels, they cannot change the BM levels themselves. In *APPLiq*, one can add new levels in the description of the model of a specific learning game. However, it is the concern of the user who makes this modification to make the necessary adaptations to the learning game itself. Indeed, in *MOPPLiq* (Marne and Labat, 2014), *APPLiq*'s underlying scenario model, the activities (levels) are modelled as black boxes.

In addition, the author of *APPLiq* provides a modified version of BM working with his authoring tool. But the latter is derived from a version of BM released by Google in 2012. Since then, BM has changed a lot, and these very profound adaptations would have to be made again.

In our particular situation, choosing *APPLiq* as a tool enabling teachers to adapt BM implies, on the one hand, providing a major update of BM and, on the other hand, providing another authoring tool that allows adaptation of levels.

For this work of learning games modelling and development of authoring tools for teachers, our approach is a design-based collaborative research (Sanchez et al., 2017). We describe the three main axes of this research in the next section.

3 APPROACH AND METHODOLOGY

Our goal is to provide tools and methods inspired by meta-design that allow teachers to master Blockly Maze in order to design, use and adapt levels and scenarios that meet their pedagogical needs in the field of computer science teaching.

To achieve this goal, our approach is to involve teachers in order to discuss and conceptualize computational thinking for pedagogical purposes in the form of a concept map (see figure 3). To co-design this map⁵, we chose to conduct qualitative interviews with two teachers instead of quantitative survey based on questionnaire. The first teacher is an experienced mathematics teacher who teaches hospitalized secondary school children, and the other is the director of children's education at the same hospital.

The *co-design* of this map was, above all, the main discursive tool we used to address many of the main pedagogical issues of the teachers we worked with. The map is therefore not intended to model

⁵See the complete concept map (in French): <https://mycore.cloud.net/index.php/s/NsntsseDxGUILM4/download>.

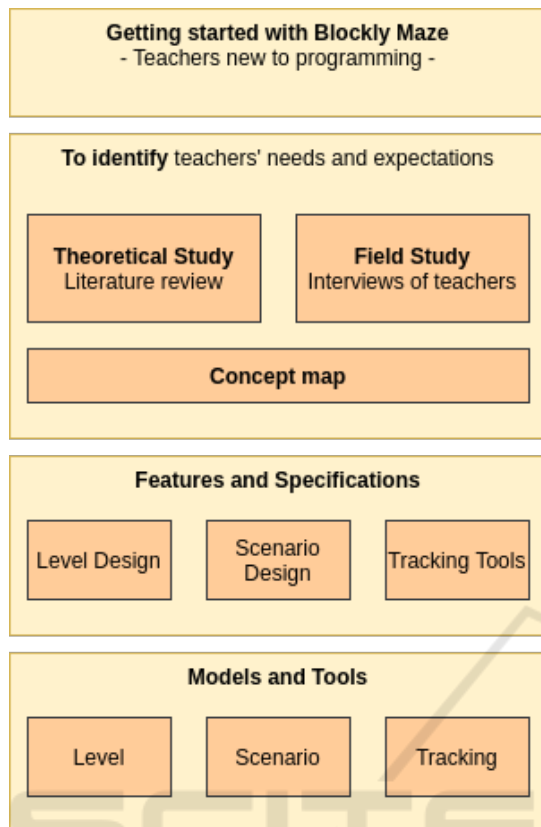


Figure 3: Overview of our approach.

computational thinking in an exhaustive or consensual way. Rather, it allows us to understand the approaches taken by the teachers in constructing levels/scenarios and to identify possible difficulties they might be experiencing. Thus, we were less interested in the actual map itself than in discussing its content.

The summary of the interviews carried out with each teacher regarding the conceptual map allowed us to pinpoint:

- The concepts taught and the teaching methodology adopted by them.
- The relevance of storytelling and games to foster learning.
- The need to adapt the BM levels to, among other things, adjust statements' wording, blocks used, dialogues, and add features (variables, timers, etc.).
- The need for pedagogical monitoring indicators enabling teachers to measure the progress of their learners and identify their appropriation of the concepts studied.
- The need for meta-design indicators that make it easier for teachers to adapt and update levels/scenarios according to the uses observed.

The first study resulted in specifying three main directions that bring together the main features of a set of authoring tools dedicated to meta-design: (1) **Level design tool:** this tool will allow the teacher to implement new levels or to reuse existing levels following a simple and intuitive model and methodology. Therefore, for each level, the aims are to be able to (re)define the mazes themselves, the list of blocks made available to the learner, the dialogues and the conditions for triggering them, as well as other constraints: number of blocks available, time limits for resolution, blocks already set, etc.; (2) **Scenario design tool:** this tool should enable the teacher to define the sequence of the levels and the conditions for triggering them. It should allow the teacher to define a succession of specific levels for each learner according to his or her performance; (3) **Monitoring tools:** this tool should allow the teacher to collect data related to interactions between the learner and the levels/scenarios and to calculate indicators on learning and the quality of teaching resources. These are high-level indicators such as success rates, levels completed or not, completion times, or more specifically the number of blocks used, the number of help messages displayed, etc.

In the next section, we present our work and its results in each of these three directions.

4 FIRST WORK AND RESULTS

Following this meta-design approach, we have conceptually deconstructed BM in order to develop three models presented in the subsections.

4.1 Level Model

Interviews with teachers about the design of the conceptual map showed us that the BM levels, as they currently exist, were not sufficient enough to meet the needs (theirs, those of their colleagues) for teaching computational thinking. We worked with them to discuss possible improvements and to establish a model of BM levels, both on a conceptual standpoint and within the source code itself.

Modelling BM levels was done in two steps: (1) isolation of all the components of the BM source code related to the description of levels, within a very monolithic program (a single file) where these components are spread over many functions and objects; (2) Conceptualization of the level source code in the form of a UML model (see figure 4).

The main features of this BM level model are, on the one hand, the maze, the starting conditions (ori-

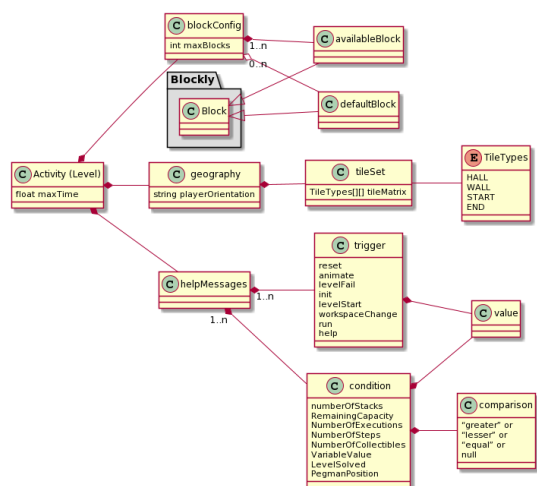


Figure 4: UML model for Blockly Maze levels.

entation, number of blocks available, etc.) which are modelled as simple variables. On the other hand, the message system (*helpMessages*) for players based on their actions. This system is much more complex and needs to be modelled with both a *conditions* system and an event (*trigger*) system.

To contribute to verify this model, we have successfully described again all the ten BM levels by instantiating them in JSON files. Furthermore, we have significantly reworked BM’s source code so it is now able to read these JSON files. In addition, we are preparing a *pull request* to be submitted to BM developers (Google), so that all potential users can easily make new levels written with JSON.

Our first experiments with the teachers show that we can work with them to write new levels with the JSON model. The weakness of this approach is the design of the help messages, which is still very complex. Therefore, the next step in our work is to build an authoring tool that generates these JSON files, and helps to set up help messages for players.

The tiniest possible scale of the changes that teachers might consider is the levels. We also intend to allow the teachers to modify learning games on a bigger scale: the sequence of levels (scenario).

4.2 Scenario Model

Our interviews confirm the teachers’ need to be able to organize a progression through the different levels according to their specific teaching context.

For this purpose, APPLiq seems relevant (Marne and Labat, 2014). It is based on an XML model of the scenario (*MoPPLiq*) in which the levels (*activities*) are considered as black boxes, meant to let the players work on some specific sets of pedagogical and

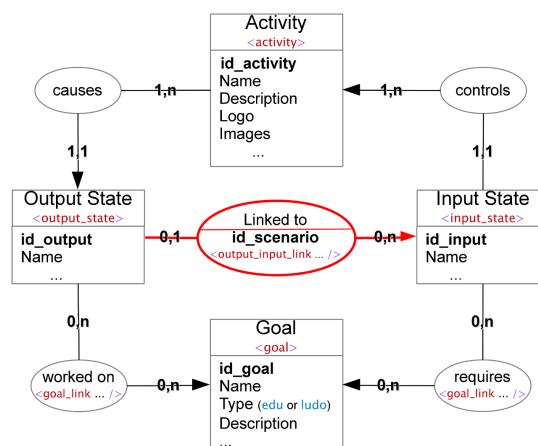


Figure 5: MoPPLiq Entity-Relationship Diagram (Marne and Labat, 2014).

playful objectives. Each activity can have several *output states* depending on the set of objectives effectively worked on. Each activity can also have several *input states* restricting the possibilities of connecting the activity to the rest of the scenario according to a set of prerequisite objectives. The *scenario* is therefore described as a sequence of all the desired output state/input state links (see figures 2 and 5).

When refactoring BM’s source code to be able to load levels described with JSON, we also used the API provided by APPLiq’s author (Marne, 2014) to embed the ability to load MoPPLiq models.

However, we are facing several issues: (1) The *new* levels designed with the JSON model must also be described (modelled) within APPLiq. Therefore, we think that we should allow our level authoring tool to communicate with APPLiq to prevent teachers from having to describe their levels twice and according to different approaches (BM JSON, on the one hand, and MoPPLiq on the other hand); (2) APPLiq enables teachers to build a scenario that adapts to the learner’s performance according to the “*worked on*” pedagogical objectives described with the output states. However, in our BM level model, the complex assessment of what is or is not worked on is not implemented. We consider adapting our model and its conditions and trigger system (currently related to the help messages) to measure whether the objectives are being worked on; (3) APPLiq provides a static scenario. Therefore, adaptation to the performances of learners is pre-designed. We are studying the possibility of modifying APPLiq to benefit from the results of a monitoring system such as the one we have started to model.

4.3 Monitoring Model

Based on discussions with the teachers, we also identified the need for a tracking and monitoring system. Its purpose is to identify whether the levels and scenarios designed meet teachers' needs after one or more play sessions by the learners.

We decided to use a version of *xAPI* adapted to serious games (Serrano-Laguna et al., 2017) as a basis for our tracking system. *xAPI* allows defining indicators (*statements* specified with triplets *actor verb object*) which are stored in a *Learning Record Store* (LRS).

We identified two different levels of monitoring: monitoring *within* the level, and monitoring *between* the levels. With the teachers help, we designed statements they felt relevant: the time taken to complete the levels (*Actor initialized level, Actor exited level*); the completion of the levels (*Actor completed level, Actor unlocked level*); the use of external support (*Actor unfocused game Windows, Actor focused game Windows*); the number of blocks used and the number of tests of their program ran by the learner (*Actor interacted blocks, Actor executed program*).

We are currently implementing this *xAPI* monitoring system in BM's source code.

To conclude, thanks to the work carried out with the teachers who accompanied us, we were able to model three significant aspects of BM: the levels, the scenario and the monitoring.

5 CONCLUSIONS AND FUTURE WORK

To foster adoption and appropriation of learning games we planned to rely on the meta-design approach (Fischer and Herrmann, 2011), and we focused on learning games for computational thinking such as Blockly Maze (BM). Meta-design goal is to enable teachers to act as designers of learning games both at the initial design stage and in the use stage.

In this paper, we have presented a description of our work in progress. The work is focused on the deconstruction of BM, both from a conceptual and a software standpoint. Our objective is to obtain, with the help of participating teachers, explicit models of the levels and the educational contents. This work was required to provide an *underdesigned* (Fischer et al., 2004) version of BM. This means a usable, "turnkey" BM, but above all a learning game that offers the main features required to provide an effective and facilitated instrumental genesis for teachers

(Rabardel, 2003).

In collaboration with the teachers involved as contributors, we experimented a research method based on the design of a concept map related to a teaching field (computational thinking). The map's design brought to light many issues related to their teaching as well as the needs around BM.

Based on these needs, and always in collaboration with the teachers, we developed three models describing BM. Then, we implemented them in a new version of BM, which we will submit to its authors (*pull request*). The **Level Model**, implemented in JSON, proved to be able to describe all the current BM levels, as well as to be used to design new ones with teachers. For the **Scenario Model**, we choose MoP-PLiq and APPLiq and plan to evolve them to answer our research questions (Marne and Labat, 2014). We have started work on a **Monitoring Model** based on *xAPI*.

The research reported in this paper was severely limited by the COVID-19 pandemic, which prevented us from working with as many teachers as we would have liked. Therefore, our current and future work consists of collaborating with more teachers in (1) developing levels to enrich and test our model of BM levels; (2) improving and experimenting with the scenario design and adaptation; and (3) implementing the monitoring of learners to allow teachers to validate their BM adaptations.

ACKNOWLEDGEMENTS

The authors would like to acknowledge The University Lumière Lyon 2 for the APPI 2020 Grant.

REFERENCES

- Bogost, I. (2007). *Persuasive Games: The Expressive Power of Videogames*. MIT Press, Cambridge, UK.
- Bouvier, P., Sehaba, K., and Elise, L. (2014). A trace-based approach to identifying users' engagement and qualifying their engaged-behaviours in interactive systems. Application to a social game. *User Modeling and User-Adapted Interaction (UMUAI'14)*, 45:413–451.
- Brunet, O., Yessad, A., Muratet, M., and Carron, T. (2020). Vers un modèle de scénarisation pour l'enseignement de la pensée informatique à l'école primaire. In *Didapros 8 – DidaSTIC*, Lille, France.
- Dermeval, D., Paiva, R., Bittencourt, I. I., Vassileva, J., and Borges, D. (2018). Authoring tools for designing intelligent tutoring systems: A systematic review of the literature. *International Journal of Artificial Intelligence in Education*, 28(3):336–384.

- Fischer, G., Giaccardi, E., Ye, Y., Sutcliffe, A. G., and Mehandjiev, N. (2004). Meta-design: A manifesto for end-user development. *Communications of the ACM*, 47(9):33–37.
- Fischer, G. and Herrmann, T. (2011). Socio-Technical Systems: A Meta-Design Perspective. *International Journal of Sociotechnology and Knowledge Development*, 3(1):33.
- Freitas, S. (2006). Learning in immersive worlds: A review of game-based learning.
- Garris, R., Ahlers, R., and Driskell, J. E. (2002). Games, motivation, and learning: A research and practice model. *Simulation & gaming*, 33(4):441–467.
- Kiili, K. (2005). Digital game-based learning: Towards an experiential gaming model. *The Internet and Higher Education*, 8(1):13–24.
- Léonard, M. (2020). Score et chronomètre sur l'interface: Quels effets sur les résultats et la perception d'un concours en ligne? In *8e Rencontres Jeunes Chercheurs En EIAH, 2020*, Poitiers.
- Li, Q. (2018). Enactivism and teacher instructional game building: An inquiry of theory adoption and design consideration. *Educational Technology Research and Development*, 66(6):1339–1358.
- Malone, T. and Lepper, M. (2005). Making learning fun: A taxonomy of intrinsic motivations for learning. *Making Learning Fun: A Taxonomy of Intrinsic Motivations for Learning*, 3.
- Marne, B. (2014). *Modèles et outils pour la conception de jeux sérieux : une approche meta-design*. Thèse de Doctorat en Informatique, Université Pierre et Marie Curie (UPMC), Paris.
- Marne, B. (2019). Éléments de réflexion autour de la conception du jouet pour un jeu sérieux. In *Actes de la 9ème Conférence sur les Environnements Informatiques pour l'Apprentissage Humain*, pages 1–12, Paris, France.
- Marne, B. and Labat, J.-M. (2014). Model and Authoring Tool to Help Teachers Adapt Serious Games to their Educational Contexts. *International Journal of Learning Technology*, 9(2):161–180.
- Prensky, M. (2004). *Digital Game-Based Learning*. McGraw-Hill.
- Rabardel, P. (2003). From artefact to instrument. *Interacting with Computers*, 15(5):641–645.
- Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., and Kafai, Y. (2009). Scratch: Programming for All. *Communications of the ACM*, 52(11):60–67.
- Ryan, M., Costello, B., and Stapleton, A. (2012). Deep Learning Games through the Lens of the Toy. In *Meaningful Play 2012*, pages 1–29, East Lansing, USA.
- Sanchez, E., Monod-Ansaldi, R., Vincent, C., and Safadi-Katouzian, S. (2017). A praxeological perspective for the design and implementation of a digital role-play game. *Education and Information technologies*, 22(6):2805–2824.
- Sardone, N. B. (2018). Attitudes Toward Game Adoption: Preservice Teachers Consider Game-Based Teaching and Learning. *International Journal of Game-Based Learning (IJGBL)*, 8(3):1–14.
- Serrano-Laguna, Á., Martínez-Ortiz, I., Haag, J., Regan, D., Johnson, A., and Fernández-Manjón, B. (2017). Applying standards to systematize learning analytics in serious games. *Computer Standards & Interfaces*, 50:116–123.
- Sweetser, P. and Wyeth, P. (2005). Gameflow: A model for evaluating player enjoyment in games. *Comput. Entertain.*, 3(3):3.
- Vandeveldel, I. and Fluckiger, C. (2020). L'informatique prescrite à l'école primaire. Analyse de programmes, ouvrages d'enseignement et discours institutionnels. *L'informatique, objets d'enseignements enjeux épistémologiques, didactiques et de formation*, page 23.