Toward a Meta-Design Method for Learning Games

About Our Project

Learning Games adoption remains scarce. To foster their adoption by **teachers** our hypothesis is that **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).

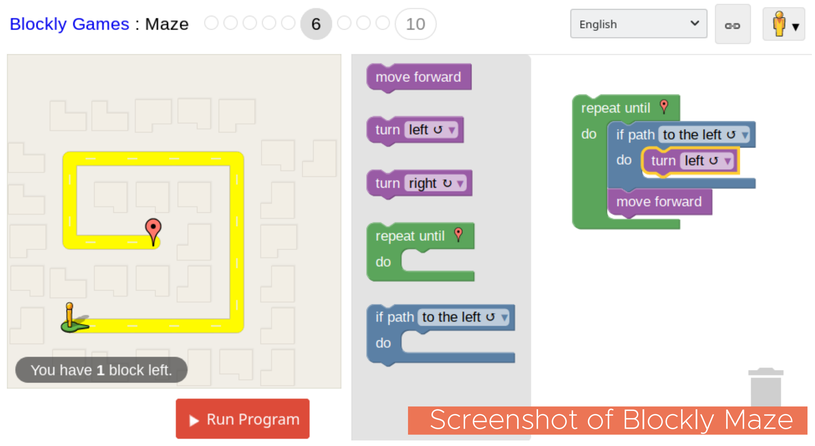
We focused our study on the teaching of **computational thinking**, and the implementation of a meta-design approach for learning games. We chose Blockly Maze because it is well-tested and free software.

Our goal is to provide tools and methods enabling teachers, more or less comfortable with teaching computer science, to use Blockly Maze.

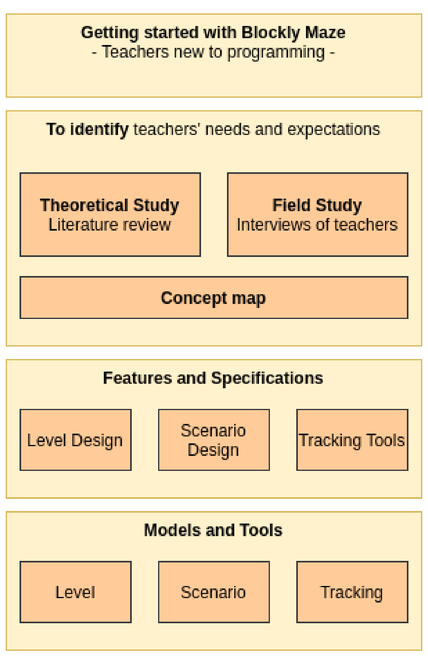
Our Methodology

Our approach, described on the left, is a Design Based Approach involving teachers.

We modelled three specific aspects of Blockly Maze in order to provide model-based authoring tools to enable them to meta-design.



Screenshot of Blockly Maze



Overview of our approach

Blockly Maze?

This learning game developed by Google is meant to teach programming through the guidance of an automatic avatar in a maze. It is done by programming with blocks of instructions.

However, BM has several obstacles to foster adoption by teachers:

1) limited number of levels. It offers 10 levels;

2) non-modifiable scenario. The course is a linear sequence that the teacher cannot modify;

3) the players' activity is not recorded. Teacher cannot retrieve any performance.

With the help of Meta-Design and Design Based Research, we wish to solve these issues.

Level Model

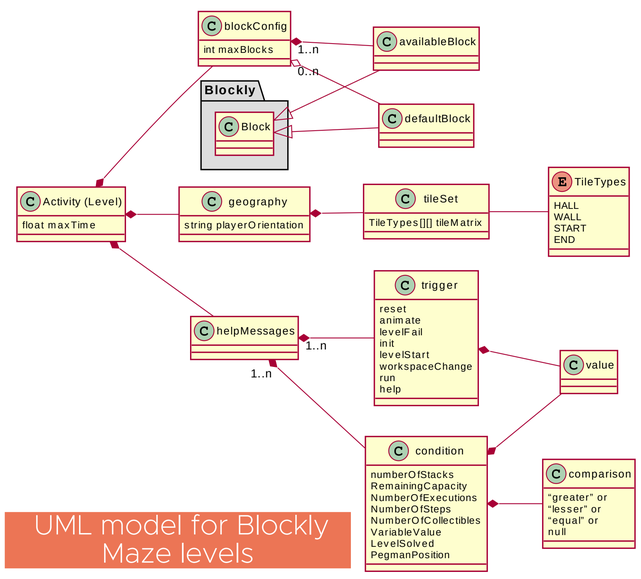
Goal: modelling Blockly Maze levels, and providing an authoring tool

The main features of this Blockly Maze-level model are:

- Simple variables: the maze, the starting conditions (orientation, number of blocks available, etc.)
- A complex system: the helpMessages system based on players actions. Provides a conditions system and a trigger system.

To contribute verifying this model, we have successfully described again all the ten Blockly Maze levels by instantiating them in JSON files and made Blockly Maze able to read them (pull request is going to be submitted to Google).

We are working on an authoring tool based on this model (and JSON file).



UML model for Blockly Maze levels

Monitoring Model

Goal: providing feedback on whether a specific adaptation made by a teacher is relevant

For our monitoring, 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 : triplets actor verb object) which are stored in a Learning Record Store.

We identified two different levels of monitoring:

- within the level,
- between the levels.

With the teachers help, we designed statements:

- 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 Blockly Maze's source code.

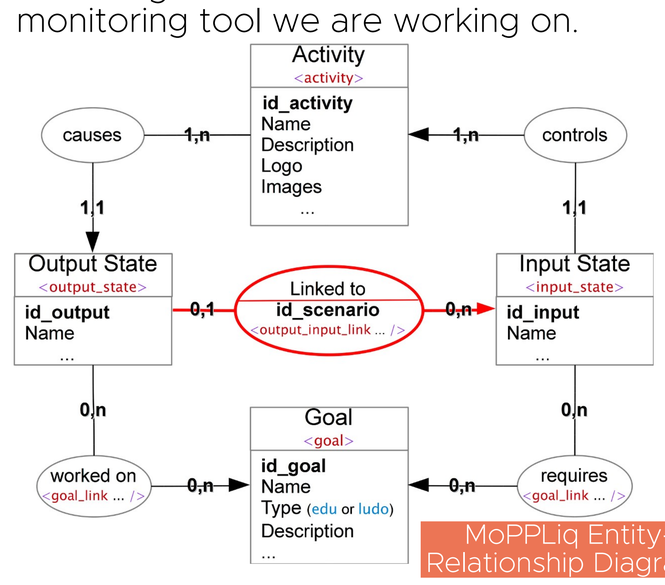
Scenario Model

Goal: adapting/improving MoPPliq & APPliq

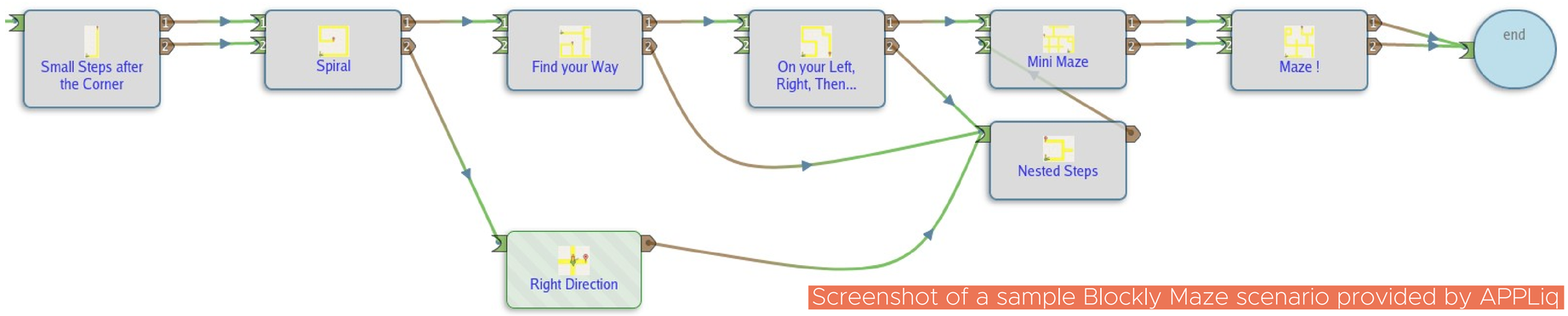
APPLiq enables teachers to prepare and provide learners with a succession of levels (called activities) in a learning game based on prerequisites and worked on pedagogical objectives. APPliq is relevant for meta-design (Marne, 2014).

It is based on MoPPliq, a model of the scenario, in which the activities are black boxes, meant to let players work on some specific sets of pedagogical and playful goals. Each activity can have several output states depending on the set of goals effectively worked on. Each activity can also have several input states restricting the connection possibilities in the scenario. Therefore, a scenario is described as a sequence output state/input state links.

We are improving APPliq to interact with our authoring tool for levels and to use the monitoring tool we are working on.



MoPPLiq Entity-Relationship Diagram



Screenshot of a sample Blockly Maze scenario provided by APPLiq

Meta-DeCT Project
(**Meta-De**sign for **C**omputer Science **T**eaching)
Team: Bertrand Marne, Mathieu Muratet, Karim Sehaba
bertrand.marne@ens-lyon.fr, mathieu.muratet@lip6.fr, karim.sehaba@liris.cnrs.fr