Multi-level Game Learning Analytics for Serious Games

Ivan J. Perez-Colado
Dept. of Software Engineering and AI
Facultad de Informática, Universidad Complutense de Madrid
Madrid, Spain
ivanjper@ucm.es

Dan Cristian Rotaru
Dept. of Software Engineering and AI
Facultad de Informática, Universidad Complutense de Madrid
Madrid, Spain
drotaru@ucm.es

Manuel Freire-Moran
Dept. of Software Engineering and AI
Facultad de Informática, Universidad Complutense de Madrid
Madrid, Spain
manuel.freire@fdi.ucm.es

Ivan Martinez-Ortiz
Dept. of Software Engineering and AI
Facultad de Informática, Universidad Complutense de Madrid
Madrid, Spain
imartinez@fdi.ucm.es

Baltasar Fernandez-Manjon
Dept. of Software Engineering and AI
Facultad de Informática, Universidad Complutense de Madrid
Madrid, Spain
balta@fdi.ucm.es

***Abstract*— Serious games are usually used or deployed in an educational setting as an isolated or individual activity, disconnected from other curricular activities. However, to really increase the adoption of serious games in different educational scenarios, the combination and integration of games into the educational flow should be simplified. We envision Serious Games as new type of educational activity that can be combined as parts of other games (e.g. minigames integrated in larger games), integrated into other online activities, or even mixed with both game and non-game activities. In addition, if we want to make the most from serious games, a learning analytics system must be in place to harvest and analyze interactions, providing metrics and insights to instructors regarding the gameplay sessions. Moreover, if a course-level learning analytics strategy is designed, it must be aligned with the game learning analytics. This approach requires communication between games and educational activities used during the educational experience. From a game learning analytics standpoint, gaining insights from these integrated experiences introduces new requirements within potentially complex multi-level or hierarchical activities. Moreover, the analysis required to generate these metrics should be both efficient and provide insight in an understandable way and for different stakeholders. This paper describes an approach to multilevel game learning analytics from the perspectives of data model, implementation architecture, and result visualization in teacher-oriented dashboards.**

Keywords—game learning analytics; multi-level games; teacher-oriented dashboards; location-based games; learning analytics models; xAPI; serious games;

I. INTRODUCTION

Serious games defined as "video games whose primary goal is not that of entertainment" are becoming popular in different educational and training settings. As a medium, they build upon the popularity of video games, which are played by a large and growing percentage of the population, well beyond their initial teenage-male demographic. Games present advantages for education, since they have greater potential engagement and interactivity and immersion [1]. We consider that a key element in a more general serious game adoption is the inclusion of Game Learning Analytics (GLA), where in-game user interactions are collected and analyzed to gain insights and simplify game usage in different domains [2].

However, games are not frequently integrated in e-learning systems. Games are not easily composable with other educational activities. For instance, it is complex to include games in hierarchies as the ones found in Learning Management Systems (LMS) that support highly structured contents such as SCORM [3] packages. Also, it is complex to communicate the games with the rest of the educational content and include them in the analytics. But from the learning analytics perspective the Experience API (xAPI) [4] specification simplify to track rich interaction data, generated by any educational activity including games. However, xAPI is mainly a format for structuring and collecting data, but it is does not impose nor provide guidance or default strategy about what kind of analysis and visualizations can or should be performed with the data once collected.

To address these identified limitations, we have extended our previous serious games analytics system to support multi-level games. This evolved model supports new educational scenarios where, for example, a geolocated game can launch different minigames depending on the player's position; or even launch another game composed of minigames. Moreover, this approach supports other low-interaction and even teacher-reported offline activities. We have built our approach based on the concept of Learning Analytics Models (LAMs) [5], that model how the inputs/signals/events generated from an activity are selected, aggregated, reported and evaluated. We also adopted the concept of Meta-LAMs [6] which are analytics models built using individual, per-activity LAMs, as building blocks. Fig. 1 briefly illustrates their relationship.

This paper describes a working implementation of multi-level analytics. Section II describes the conceptual model. Section III provides an overview of the system's architecture, Section IV describes the application of the described conceptual model, and Section V presents the resulting visualizations available for teachers when using the system. Finally, we Section VI contains our conclusions and outlines future work.

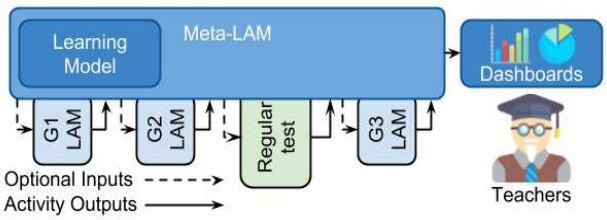

Fig. 1 Learning Model built using a Meta-LAM where multiple LAMs provide outputs as assessment resources and dashboards for teachers.

II. Conceptual Model

As defined in [2], a Learning Analytics Model (LAM) is a multidisciplinary effort, involving educators, designers, and developers to describe everything needed in order to assess a serious game. LAMs specify the player information that will be sent during gameplay as traces to the analytics server, and how these traces should be interpreted (analyzed) and displayed in different dashboards for different stakeholders. A Meta-LAM generalizes this approach and describes how a LAM should be applied to a multi-level game structure.

Meta-LAMs are built using outputs of LAMs as building blocks as illustrated in Fig. 1. This allows games to act as black boxes. To achieve this goal, outputs and aggregations of each level should be defined. Meta-LAMs should provide answers to the following questions:

1. What are the learning goals of the multi-level game? The Meta-LAM uses the outputs of the (mini)games as its inputs to satisfy learning goals.
2. How is the multi-level game hierarchy structured? Analysis needs a structure to aggregate learning goal progresses. The more complex the structure, the harder to use to combine individual game results, to design, and understand.
3. How should the data be aggregated through the structure, and what consequences can be extracted from this aggregation? Determining, when an activity is considered successful, its contributions and the common language. In a hierarchy, parent nodes are designed to receive updates from their children and its handling should be defined in the Meta-LAM.

We have built up our default Meta-LAM from a simplified version of SCORM 2004 4th Edition Sequencing and Navigation (SN) [3], which already includes similar concepts of aggregation. SN can be very briefly characterized by its use of a tree of activities, where each activity communicates with its parents, which aggregate activity results to update the LOs in a process termed rollup.

Answering question 1, designed Meta-LAM limits LOs to only learning goals and competencies to simplify the analysis. As Meta-LAM is generic, LOs and competencies are defined by the aggregation of the ones from every LAM.

Answering question 2, our adaptation maintains the tree structure, simplifying the resulting aggregation process as relationships are well-defined and easy to handle.

Answering question 3, our analysis uses SN rollup rules, where results for any activity node are propagated through its ancestors up to the root. This allows parent nodes to determine their progress and completion status. Also, analysis is parametrized with meta-data that includes optional *limits* and *contributions*. Where *limits* include thresholds that allow to accommodate the difficulty as minimum requirements of an activity and *contributions* include a LOs dictionary that describes how much each one should be incremented. Finally, regarding language, we use xAPI as a standard for all interaction traces, building up on pre-existing glossaries of terms related with Serious Games [7] and Location-based games [8].

III. Architecture for performing multi-level analytics

This section describes the architecture that we use to implement the previously described Meta-LAM. Our LA server used a standalone, generic single-game analysis, called Default Analysis (DA). The DA processes xAPI traces from the actual gameplay according to the xAPI-SG [7] format, and generates a number of metrics and dashboards as a result.

Even through the conceptual model, described in section II, manages the multi-level requirements, it does not store or generate results for the gameplay of each game. We have combined the existing DA with a new Multi-Level Analysis (MLA) that implements the conceptual model of Section II.

To successfully build a Meta-LAM using independent analyses as building blocks, the Meta-LAM architecture designer must answer a few questions:

1. What does the Meta-LAM require from each activity? E.g. games sending xAPI-SG traces require DA, higher level analysis require an overall-storing-analysis.
2. What analysis-to-analysis communication is used? e.g. queue, messaging system, notifications, common storage, etc.

Answering question 1, regarding activity results, the combination of the DA and the MLA will generate, per each of the activities, a results file that contains all its gameplay information along its successors gameplays, meta information related with the achievement of game goals, and results generated based on MLA-generated traces.

However, some of the learning activities may have their own LAM and can use their own analysis to generate the results. To maintain coherency along the multi-level ecosystem, these LAM traces should be stored inside their own analysis results, and translated for their ancestor's analyses to understand them.

Answering question 2, regarding the communication method, all analyses use a common queue to obtain the traces. This way, instead of sending a trace, it is queued with the target node as destination, ensuring that the server is not overloaded. Additionally, every node stores their results into separated documents, causing the root node document to include the whole analysis results.

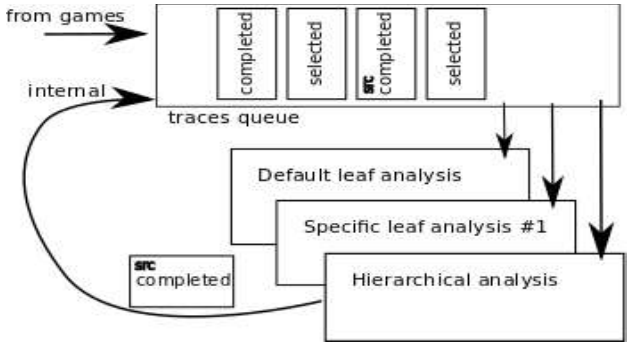

Fig. 2 Diagram of the multi-level game analysis consuming and bubbling traces. The hierarchical analysis propagates "completed" traces to the parent of each originating node, until the root of the hierarchy is reached. Completed trace label "scr" means it is an internal trace.

IV. Multi-Level Analysis Conceptual Model Application

Our system follows a Kappa Architecture [9] by using a stream-based analysis, as we analyze traces sent from games in near real-time. When games send traces to the LA they're added to a queue. Analyses read them from this queue being processed once and only once by all analyses associated with the corresponding game. Analyses can maintain their state by reading and writing from and to datastores, and can add new created statements to the queue. Statements in the queue can be either game-generated (GGS) or analysis-generated (AGS).

By allowing analyses to feedback into themselves, we enable a message system where each statement is targeting a node. GGS are always targeted to leaves, while AGS are used to inform intermediate nodes of knowledge propagation within the hierarchy, bubbling up from leaves from each node to its parent until the root is reached. This process is illustrated in Fig. 2.

To feed our default multilevel dashboard, we perform several types of leaf aggregation: when leaf nodes are considered completed, LOs and competencies gets increased. For each leaf, computes the minimum, maximum and average scores of each player to easily contrast the results. Time-to-completion, and completion history: average time to complete missions and quests by players. Player choices, and choice accuracy: The amount of correct and incorrect alternatives is computed and stored, together with a list of choices over time. Player progress is stored, and an average calculated, so that we can later display average player progress. Performance of students is computed from the score and aggregated by years, months and weeks. When a child trace is completed the parent node progresses accordingly.

When a child node is completed, the parent node progresses accordingly. Furthermore, when all the children have been completed, the parent node completes itself, increasing its related contributions.

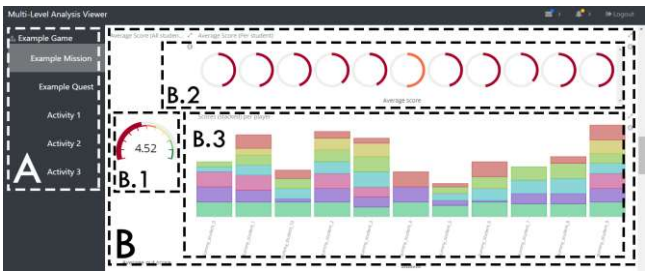

Fig. 3 Web tool used to navigate through game-tree dashboards. Menu list (A) allows to select dashboard while the main frame (B), displays the dashboard, in this case MLA dashboard. Main frame is displaying Scores section with aggregated scores, composed of Average Score of all students (B.1), Average Score per student (B.2) and Stacked scores per player (B.3).

V. User Interface

Regarding data visualization, one important difference between a regular LAM and the Meta-LAM is going from one single dashboard to multiple dashboards. This forces developers to add an upper layer in the UI allowing to navigate through the multi-level game tree. Our approach depicted in Fig. 3, consists of an admin-panel view where the left part (Fig. 3 A) depicts a menu that allows users to select the desired dashboard, while the dashboard are displayed in the main frame (Fig. 3 B).

The root (and in some cases non-leaf node) dashboard has been designed following the conceptual model described in Section II. Aims to be as generic as possible, mostly adhering to xAPI-SG, implying that every game that successfully implements xAPI-SG will fit the Meta-LAM and its data will be correctly analyzed for display in the Meta-LAM dashboard.

By design, every visualization is intended to be meaningful both unfiltered and filtered-by-student, with only few exceptions tog this rule. Our approach aims to be generic enough; however, some cases are not covered when using specific LAM. The more heterogeneous the variety of traces, the more diverse the sections

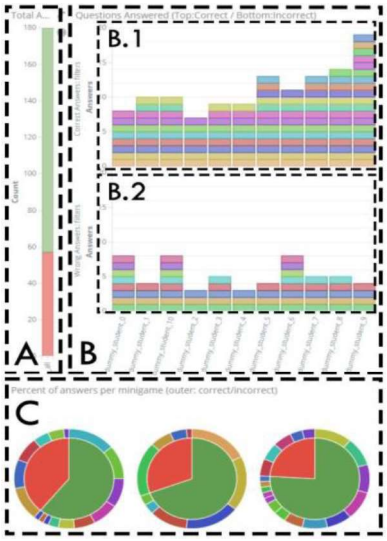

Fig. 4 Answers section showing aggregated answers. Includes total received answers, filtered by correct and incorrect view (A). Specific answers stacked per student (B), where top (B.1) are the correct and bottom (B.2) are the incorrect. And percent of answers received per sub-game (C)

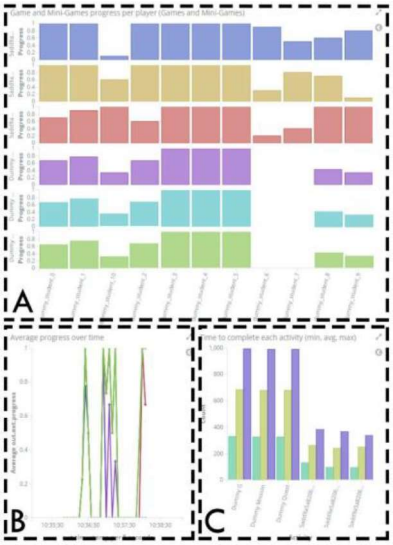

Fig. 6 Progress and duration section. including: a matrix of vertical bar progress visualization where X is student and Y is the activity (A); The average progress of each activity over time (B) and the min, avg and max duration per activity(C).

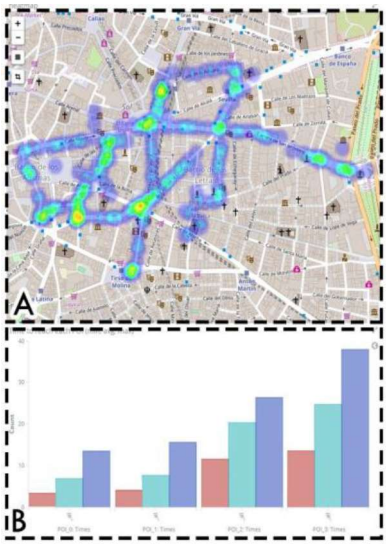

Fig. 5 Location-Based game section. Includes two visualizations: a heatmap of every position tracked by any activity (A); and the minimum, average and maximum duration of the travel between a POI and next POI (B)

included in the dashboard. Even though we could add further visualizations it would likely overwhelm teachers, worsening the user experience. Dashboards can easily be overloaded with too much information, so every visualization must have multiple purposes. Our solution is composed of five sections:

The *General information and configuration section* has been designed to allow teachers to quickly access information regarding whether the lesson is working properly; and allowing them to filter and personalize the dashboard. Includes four visualizations: a student list for dashboard filtering, a session counter for quick comparisons with the expected number of learners, a completion counter to check how many students have already finished, and a time-frame selector.

The *Scores section*, illustrated in Fig. 3, has been created to easily determine if a learner has succeeded or failed, and how well have each learner performed in each multi-level game node. As seen in the previous section, most GLA models can determine if a game has succeeded, and even determine a score that describes how well the learner has played. In the end, this allows educators to transform LA results into grade marks.

The *Answers section*, illustrated in Fig. 4, includes information about the choices that learners have made throughout all sub-games. It is meant to allow the teacher to specifically detect where learners have failed or succeeded, to personalize feedback and provide better assessment. Almost every game allows to make choices, and its answers are usually given with a correct or incorrect result. Those answers can be from a quiz, or can from a potion-mixing minigame, meaning that the potion have been successfully crafted or not.

The *Progress and duration section*, illustrated in Fig. 6, is designed to provide control of the current state of the students by providing and overview of the progress of each student in each activity, their progression, and its duration. It explores the possibilities of MLA by showing non-leaf nodes progress and duration, which are calculated by the analysis (as explained in section IV). It also allows to check when the progress has been made, showing the students' play periods, and providing feedback to understand the homework window available to the teacher, or adapt future games to more appropriate durations.

The *Location-Based games section*, illustrated in Fig. 5, satisfies a frequent use-case of multi-level games, which are Location-based gymkhana-like games, where multiple stops are presented, and each stop requires the player to pass a minigame to continue. This section provides, an overview of where the players have physically been, along the time it took to them to reach each stop. This is made using a heatmap, displaying the position of the players. It helps to understand the most-used routes to travel from stop to stop, together with the time spent in each of the stops.

VI. Conclussions and Future Work

In the previous sections, we have identified and addressed the issues that arise from using multi-level learning analytics for multi-level / hierarchical tree-like activities, focusing on multi-level game scenarios.

In addition, we have also briefly described the main components of our solution, which is designed to be modular and could potentially support several different analyses side-by-side using entirely different Meta-LAMs, by changing the chosen answers to the questions of Section II.

The implementation of the conceptual analysis can be done in several ways, however the driving questions stated in section III lead us to a modular architecture where we defined a default analysis that works at the individual game level; and a multi-level analysis (MLA) that performs all calculations for the intermediate levels. The combination of these two analyses satisfies the needs of Meta-LAM as described in IV.

We have developed a dashboard that displays aggregated results from the MLA, providing a useful tool to track and evaluate multi-level game scenarios. The dashboard is divided into five sections, one for management, three that gather information on aspects needed to assess learners, and a last one for location-based games as a usual multi-level game as described in V.

Although an initial qualitative evaluation of the current dashboards has been positive, based on previous work with non-multilevel teacher dashboards, we feel that many teachers are already overwhelmed with relatively simple dashboards, and multi-level ones may produce additional cognitive overload. We will be collecting additional teacher feedback to lower the cognitive demands of multi-level dashboards

Acknowledgments

This work has been partially funded by Regional Government of Madrid (eMadrid S2013/ICE-2715), by the Ministry of Education (TIN2017-89238-R), by the European Commission (RAGE H2020-ICT-2014-1-644187, BEACONING H2020-ICT-2015-687676, Erasmus+ IMPRESS 2017-1-NL01-KA203-035259).

References

[1] T. M. Connolly, E. A. Boyle, E. Macarthur, T. Hainey, and J. M. Boyle, "A systematic literature review of empirical evidence on computer games and serious games," Comput. Educ., vol. 59, no. 2, pp. 661–686, 2012.

[2] M. Freire, Á. Serrano-Laguna, B. M. Iglesias, I. Martínez-Ortiz, P. Moreno-Ger, and B. Fernández-Manjón, "Game Learning Analytics: Learning Analytics for Serious Games," in Learning, Design, and Technology, 2016, pp. 1–29.

[3] ADL, "SCORM 2004 (4th Edition)," 2009. [Online - Last accessed, April 2018]. Available: http://www.adlnet.gov/research/scorm/scorm-2004-4th-edition/.

[4] B. Y. M. Torrance and C. Wiggins, "What Is xAPI ?," TD Talent Dev., vol. 70, no. 2, pp. 28–31, 2016.

[5] J. M. Baalsrud Hauge et al., "Learning Analytics Architecture to Scaffold Learning Experience through Technology-based Methods," Int. J. Serious Games, vol. 2, no. 1, 2015.

[6] I. J. Perez-Colado, C. Alonso-Fernández, M. Freire-Moran, I. Martinez-Ortiz, and B. Fernández-Manjón, "Game Learning Analytics is not informagic!," IEEE Glob. Eng. Educ. Conf., 2018.

[7] Á. Serrano-Laguna, I. Martínez-Ortiz, J. Haag, D. Regan, A. Johnson, and B. Fernández-Manjón, "Applying standards to systematize learning analytics in serious games," Comput. Stand. Interfaces, vol. 50, pp. 116–123, Feb. 2017.

[8] V. M. Perez-Colado, D. C. Rotaru, F. Mauel, M.-O. Ivan, and B. Fernandez-Manjon, "Learning analytics for location-based serious games," IEEE Glob. Eng. Educ. Conf., 2018.

[9] J. Kreps, "Questioning-the-Lambda-Architecture," 2014. [Online - Last accessed, April 2018]. Available: https://www.oreilly.com/ideas/questioning-the-lambda-architecture.