Understanding Learners' Behaviors in Serious Games

Mathieu Muratet, Amel Yessad, Thibault Carron

Understanding learners' behaviors in serious games

M. Muratet[1,2], A. Yessad[1], and T. Carron[1,3]

[1] Sorbonne Universités, UPMC Univ Paris 06, CNRS, LIP6 UMR 7606, 4 place Jussieu 75005 Paris, France
[2] INS HEA, 58-60 Avenue des Landes, 92150 Suresnes, France
[3] Université Savoie Mont Blanc, 73376 Le Bourget du Lac, France

{mathieu.muratet, amel.yessad, thibault.carron}@lip6.fr

Abstract. Understanding play traces resulting from the learner's activity in serious games is a challenged research area. Especially, when the serious game is characterized by a large state space and a large amount of free interactions, the play traces become complex and thus hard to analyze and to interpret by teachers. In this paper, we present a framework that assists designers to build a model of an expert's solving process semi-automatically. Based on this model, we propose an algorithm that analyzes player's traces in order to generate pedagogical labels about the learner's behavior. We carried out an experimental study which aimed to evaluate the effectiveness of the labeling algorithm. From a usability point of view, we also use the experiment to validate the acceptation and readability of pedagogical labels by the teachers.

Keywords: Learner Assessment, Serious Game, Feedback, Behavioral Model, Petri Net, Reachability Graph

1 Introduction

Learner assessment is considered as a key issue in Technology-Enhanced Learning (TEL). Learners are not all alike and their play traces are often very complex. Thus, it is interesting to analyze their behavior in order to provide an efficient and readable information to the teachers. Our approach aims (1) to be seamlessly integrated with a LG, rather than presented as a separate artificial assessment disconnected from the nature of the task and (2) to provide pedagogical information about the learner's behavior by comparing it with experts' solving model.

Our research focuses on serious games which simulate process (physical, industrial, business, etc.). In this kind of complex systems with a large amount of freedom in interaction, the learner's behavior is hard to understand and to analyze by a teacher. Thus, our objective is to generate a description of learner's behavior, readable by teachers or anyone interested in understanding it (tutors, game designers, learning designers, the player himself/herself, etc.).

The overall architecture is composed of two parts: the first part consists in a workflow enabling the modeling of experts' solving and the second one is a learner assessment system called Laalys V2 (Learner Activity AnaLYSer). In this paper, we focus on the labeling algorithm that is used to generate pedagogical labels which tag the learner's

actions and calculate a global score based on these labels. The generated pedagogical labels enable the teachers to have information about the learner's behavior such as the correct actions, the erroneous actions, the too late or too early actions and give them many others information reflecting gaps between the player's behavior and the solving process expected or recommended by experts. An experimental study was carried out with students and teachers in order to evaluate the effectiveness of the labeling algorithm.

In the next section, we present a review of the existing research and the recent related work. Then, we present the framework that assists us to formalize the expert's solving process in serious games. In section 4, we present the main contribution of this paper: the labeling algorithm that generates the pedagogical labels of the learner's actions. We present the experimental protocol and results in section 5. Finally, we conclude this paper and present some future work.

2 Positioning

The work presented in this paper contributes to research in learner assessment. Data mining and intelligent techniques have been used to analyze learner data and generate feedback to teachers in TEL [1, 2]. These research remain very specific to e-learning and are not designed to simulation based systems including serious games.

For these kind of systems, research in the domain of information visualization [3] have been interested to provide the means for teachers to explore graphical representations and discover for themselves what information is needed for their pedagogical activities. For instance, research presented in [4, 5] have been focused on visual game analytics and consist in extensively collect in-game data of players in order to guide decision-making, gameplay analysis and sequence and time-based analysis of players execution traces. The proposed techniques have not been interested in defining indicators related to the learner's behavior in the education field which is a key point of our approach. In our research, the system takes decisions about learner's behavior by tagging its actions with pedagogical labels. We want to provide a efficient and relevant feedback to teachers about learner's behavior.

Research similar to ours has been proposed by [6]. The authors propose a methodology for extracting conceptual features from student's log and use these features to perform clustering of student solutions. Our approach differs from their that considers only the result of the learner's solving and not its behavior along the solving. In addition, their expert's solving is machine learned rather than defined by experts and their approach is focused on puzzle games such as *RumbleBlocks*[4] or *Refraction*[5] and seems not to be well-adapted to simulation-based serious games. In [7] the authors added a system of state machines to the game, i.e. predefined situations that the teacher wants to observe. The teacher sets the states he wants to trace in the monitoring system. This approach is interesting but requires from the teacher a great effort of interpretation to analyze the provided game indicators.

[4] RumbleBlocks: http://rumbleblocks.etc.cmu.edu/ accessed April 4, 2016

[5] Refraction: http://games.cs.washington.edu/refraction/refraction.html, accessed April 4, 2016

Other close research used Petri nets to describe experts' solving of "Case study" games and proposed an algorithm to label learners' actions [8, 9]. However, this algorithm is adapted to simple case studies and are not at all suitable for serious games with large state spaces and a large amount of freedom in interaction. Our research is a follow-up to these reseach but aims to propose a scalable and generalizable framework giving more accurate pedagogical information about the learner's behavior. The pedagogical labels are more semantic and are based on the comparison between the learner's behavior and the expert's solving of a game level. Another important contribution of our approach is the semi-automatic building of the expert's solving model from a high level declarative description of game levels.

3 Workflow to build the expert's solving model

The work presented in this paper shows how to analyze the learner's behavior when he or she interacts with the LG. The assessment system that we propose is based on the comparison between the play traces of learners and the expert's solving of the game levels. A workflow is proposed in order to assist designers to model the expert's solving by a Petri net.

3.1 Brief presentation of the Petri net formalism

Formally, a Petri net is a valued bipartite graph composed of two types of vertices: places and transitions. Each edge connects a place to a transition or a transition to a place and each place contains a number of tokens greater than or equal to 0. The vector that contains the number of tokens of each place is called marking. It represents the state of the system modeled by the Petri net, at any time. A transition of a Petri net may fire if it is enabled to, i.e., if there are sufficient tokens in all of its input places; when the transition fires, it consumes the required input tokens, and creates tokens in its output places. The set of all markings reachable from the initial marking of Petri net by firing the transitions represents the state space of the system and is called the reachability graph. More details about the Petri net formalism can be found in [10].

Petri net is a powerful modeling formalism which combine a well-defined mathematical theory with a graphical representation of the system's behavior. The game simulation, we are interested in, is complex and require adapted models to represent it. Petri nets have proved to be adapted to modelise the behavior of this class of systems. In our context, the places represent the states of game objects and the transitions represent the actions that the player can perform in the game, changing the game objects' state.

3.2 Semi-automatic building of Full and Filtered Petri nets

In order to implement the automatic learner assessment, we construct two Petri nets semi-automatically. The first Petri net is called "Full Petri net" (FullPn) and includes all actions that learners can perform in the game. A FullPn models game simulation and its marking depicts the state of the simulation. The second Petri net, called "Filtered Petri net" (FilteredPn), is a part of the FullPn and includes only actions used by experts

to solve the current level. It embeds the solving of experts, recommended to solve the game level.

The building of the FullPn is a difficult, expansive and challenging task due to the high number of actions that the learner can perform in each state of the game; the state space grows exponentially with the number of actions. Thus, this building process has to be automatic or at least semi-automatic. More details about this assistive module of Laalys V2 can be found in [11].

3.3 Workflow overview

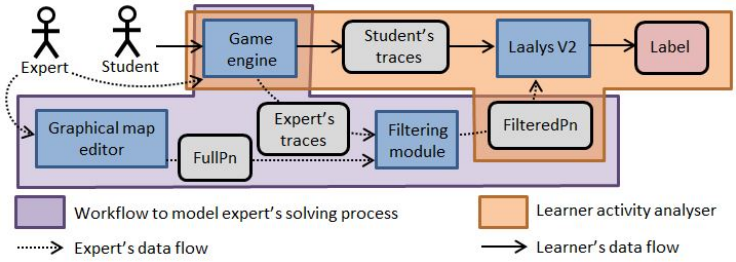

Fig. 1. Global architecture of the assessment framework.

As depicted in Fig. 1, the designers start by using a user-friendly graphical tool to build a game level and its associated Petri net automatically (the FullPn). An expert can play the level (several times if several solutions are available) and the game engine traces the expert's actions. These traces are used to filter the FullPn and build the FilteredPn. We notice that a non-expert's trace could be used to filter the FullPn, for example an original and correct solving made by a learner and positively assessed by teacher can be added to the expert's traces to enlarge the FilteredPn.

Once both FullPn and FilteredPn are generated, they can be validated or completed by expert/designer manually in order to include constraints not configurable with the graphical editing tool. Finally, the validated FilteredPn is used by the labeling algorithm to label learners' actions pedagogically.

4 Learner assessment based on Petri nets

As defined in section 3, Laalys V2 is based on two Petri nets (FullPn and FilteredPn): the first one (FullPn) models the game simulation and the second one (FilteredPn) is filtered from the first one and models only actions recommended by the experts in order to solve the game level.

4.1 Labeling algorithm

The inputs of the labeling algorithm is the FilteredPn and a learner's trace. The algorithm base is to analyze the current state game resulting from the learner's action. The

labeling algorithm deals with the following three cases: case 1 corresponds to the cases where the player performed actions that are refused by the game and so don't change the game state; case 2 corresponds to the case where the player performed actions that are accepted by the game and change its state, these actions may match with the expert's solving or not; and finally case 3 corresponds to the missing actions whilst the game is over. Each of these cases leads to a description that characterizes the learner's action by one and only one pedagogical label.

Case 1 - actions that don't change the game state: this case occurs when the learner tries to perform an action that is not allowed by the current state of the LG, for instance, if the player tries to open a door without the right key. We identified four interesting labels to characterize these actions: (1) the action was enabled in the reachability graph (the state space) of the FilteredPn, the label generated is "Too late", (2) the action will be enabled in the reachability graph of the FilteredPn, the label generated is "Too early", (3) the action was enabled and will be enabled another time in the FilteredPn, the label generated is "Unsynchronized", or (4) the action is never enabled in the FilteredPn, i.e. it is never performed by the experts in the level solving, the label generated is "Erroneous".

Case 2 - actions that change the game state: this is the most common case, but all the actions performed by the learner are not necessarily correct compared to the expert's solving. Indeed, most of LGs allow learners to make mistakes. This trial and error principle is at the heart of video games. The opportunity to perform useless, redundant or incorrect action sequences in a LG [12] is fundamental to catch learners' attention and hopefully achieve pedagogical objectives. The algorithm identifies whether the current game state leaves the filtered state space (f-space), comes back to the f-space, moves inside the f-space or moves outside of the f-space:

- *Case 2.1 - exit from the f-space:* this can happen when the learner plays an action that was not performed by experts. If the game state is out of the f-space, this means that the generated state of the game doesn't correspond to any state of the f-space. The label used to tag this action is "Erroneous".
- *Case 2.2 - come back to the f-space:* this happens when the learner corrects previous errors and comes back into the f-space. In this case, the algorithm (1) checks whether the game state resulting from the learner's action is included in the history of states already reached by the player, (2) computes the length of the shortest path between this new state and a final state and (3) computes the minimal length of shortest paths between each state of the history and a final state. Depending on results of these three criteria, following labels are generated for the player's action: "Recovery" ($1 \wedge (2 = 3)$), "Leap backward" ($1 \wedge (2 > 3)$), "Leap forward" ($\neg 1 \wedge (2 < 3)$) or "Non optimal" ($\neg 1 \wedge (2 \geq 3)$).
- *Case 2.3 - move in the f-space:* in this case, the algorithm checks if the player progresses towards a final state or digresses from it. Indeed, the player digresses when he/she performs a non-expert action that produces a game state in the f-space but not nearest from a final state. Then, the algorithm (1) computes the length of the shortest path between the new state and a final state, (2) computes the length of the shortest path between the previous state and a final state, (3) checks if these two paths are different, (4) checks if the action played is an expert's action and (5)

checks if the new state is a direct successor of the previous state in the f-space. Depending on the results of these five criteria, following labels are generated for the player's action: "Correct" ($(1 < 2) \wedge 4$), "Equivalent" ($(1 < 2) \wedge \neg 4 \wedge 5$), "Leap forward" ($(1 < 2) \wedge \neg 4 \wedge \neg 5$), "Useless" ($(1 = 2) \wedge \neg 3$), "Non-optimal" ($(1 = 2) \wedge 3$), "Leap backward" ($1 > 2$).

– *Case 2.4 - move outside the f-space:* this happens when the player performs an action which keeps the game in a state out of the f-space. In this case, the algorithm tries to compute a new f-space by initializing the FilteredPn to the current game state. If a final state is reachable from the current state, the algorithm processes this action as in Case 2.3. Otherwise, the algorithm determines if the new state is closer (the label generated is "Becoming closer"), farther (the label generated is "Farther") or equidistant (the label generated is "Stagnation") to a final state of the level than the preceding state.

Case 3 - missing actions while the game is over: if the learner's last action doesn't allow him or her to reach a final game state, the algorithm looks for the last state in the f-space reached by the learner and calculates the shortest path to reach the end of the level from this state. Then, the algorithm labels the actions of this path as "Missing".

4.2 Score computing

The labeling algorithm, presented above, produces 15 labels. We associate a coefficient for each label in order to compute a score (see Table 1).

Table 1. Positive and negative labels coefficients

Positive labels	Coeff.	Negative labels	Coeff.
Correct	1	Useless	-0.1
Equivalent	0.8	Farther	-0.2
Leap forward	0.6	Missing	-0.3
Becoming closer	0.4	Erroneous	-0.5

First, we define as positive labels the ones that guarantee a progression towards a final state. We define the positive coefficients in $]0, 1]$. For this class, we retain the following labels: "Correct" reflects a right behavior in comparison with the expert's solving (coeff = 1); "Equivalent" is similar to "Correct" action, but makes reference to a non-expert action (hence a lower coeff = 0.8); "Leap forward" means that the player follows a non-expert resolution process but it is close to the solution (coeff = 0.6); and "Becoming closer" is the lowest positive label because it occurs when learner moves outside the f-space (coeff = 0.4).

Then, we define as negative labels the ones that represent a deviating from the expert's solving. In order to give more weight to the positive labels, we define the negative coefficients in $[-0.5, 0[$. For this class, we retain the following labels: "Erroneous" is the opposite of "Correct" (coeff = -0.5); "Missing" occurs when learners drop the level, we chose to strongly weighting this label to degrade student's scores in case of abandonment (coeff = -0.3); "Father" is the opposite of "Becoming closer" (coeff = -0.2);

and “Useless” doesn’t characterize a major error and so has a minor influence on the score (coeff = -0.1).

Finally, the remaining labels (Non-optimal, Recovery, Leap backward, Stagnation, Too early, Too late and Unsynchronized) are classified in neutral labels. Even if their coefficients are set to 0, they contribute to the score computation by increasing the size of the trace.

We calculate the score of a learner’s trace by the formula (1), the score is defined in $[-0.5, 1]$. For a trace, we note $L = \{l_1, l_2, ..., l_n\}$ the set including the number of each kind of label and $C = \{c_1, c_2, ..., c_n\}$ the set defining coefficient of each kind of label.

$$score = \frac{\sum_{i=1}^{n} l_i * c_i}{\sum_{i=1}^{n} l_i} \tag{1}$$

5 Experimental study

The overall architecture has been evaluated on the LG called “Les Cristaux d’Ehere”, designed to teach concepts of physics. The goal for each level is to solve problems about competences related to water state changes. Learners must move an avatar to interact with game objects and reach a solution concerning physics-related topics. All game levels (18) were designed with the framework we presented in this paper. Then, both FullPn and FilteredPn for each level had been available.

We present here results about two levels: “The thermometer” and “The ice wall”. In the first one, “The thermometer”, the learner works on the competence “reading the temperature accurately on an analog thermometer” and in the second one, “The ice wall”, he works on competences about the concept of water melting. Fig. 2 and 3 show the screenshot and the FilteredPn (built automatically) of the level “The ice wall”.

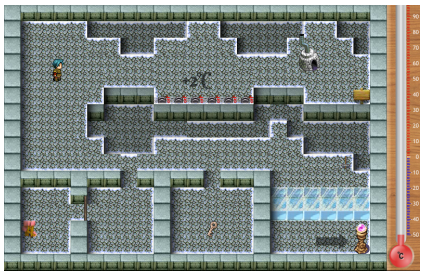

Fig. 2. Screenshot of the level “The ice wall”

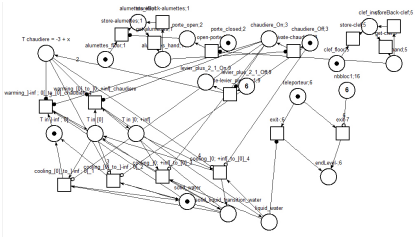

Fig. 3. Filtered Petri net (automatically built) of the level “The ice wall”

5.1 Experimental protocol

The objective of this experiment is to determine whether the labels and the label-based score computed by the algorithm would give a good idea about the learner’s behavior and are consistent with the teachers’ scores.

We carried out a qualitative experiment with nine students and four teachers of physics from junior high school. All students were familiar with video games and the teachers had an experience of using TEL in classroom. We wanted to avoid a bias related to difficulty of students with the use of video games and to focus on their level solving.

An experimental protocol was followed in order to collect the opinions of the teachers about the effectiveness of the labeling algorithm. First, we filmed the students during playing sessions. Then, we presented the game levels to the teachers. Two qualitative assessments of learners' activities were carried out:

- We asked three teachers from the four to visualize videos of the learners during the levels "The thermometer" and "The ice wall". We asked them to assess three learners' behavior with an overall score on a scale of 1 to 3: 1 for a student who failed or met big difficulties to solve the level, 2 for a student who met some difficulties and 3 for a learner who showed a good understanding of the domain concepts underlying the level solving. Then, we provided the list of the labels and their meaning. We asked to these teachers to tag the actions of the students manually by using the list of labels. Once, they tagged the students' actions, we presented them the labels generated by the labeling algorithm. Finally, we submitted a questionnaire to the teachers in order to assess the effectiveness of the labels generated automatically.
- We asked the fourth teacher to visualize videos of all the learners during the two levels and we asked him to assess the learners' behavior with the same scale score as the other three teachers. Finally, we compared the automatic score generated by Laalys V2 with the grades given by this teacher.

5.2 Results and discussion

Teachers' opinion about the labeling algorithm results The three teachers were able to label the traces of students within 3 to 10 minutes, depending on the length of the trace. They considered that the labels are meaningful and bring a good understanding of the students' behavior.

When we compared the automatic labeling with the ones performed by the teachers manually, we noted a great similarity. Nevertheless, they made confusion between the labels "Recovery" and "Leap backward". We think that these labels are very similar but different and would require more explanation. One teacher didn't understand the meaning of the label "Unsynchronized" and so didn't use it. After discussion with this teacher, we think this label can be merged with the label "Too early" because the fact that an action was enabled in the past is not important and the most essential feature for the teacher is to know if an action will be enabled in the future.

Finally, the three teachers considered that the labels choosen by algorithm were more accurate than their own choices and represent a substantial saving of time when compared to the video viewing. They propose to identify patterns of labels automatically: for instance, a sequence of many actions labeled "Too early" or "Too late" could show that the learner doesn't understand the temporal constraints of the process simulated by the game.

Comparing between scores of Laalys V2 and the grades given by the teacher Table 2 shows experimental data on levels "The thermometer" and "The ice wall". For both levels, we present data for the nine students. The two first rows gives for each students the automatic score (shifted and discretized on $[1, 2, 3]$ values) based on labels and the teacher's score. The last row gives the trace length.

Table 2. Scores comparison for each student and level ("The thermometer" and "The ice wall")

	The thermometer									**The ice wall**								
	S1	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**	**S9**	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**	**S9**
Algorithm's score	2	1	1	3	1	1	1	1	1	3	1	2	3	1	3	1	3	1
Teacher's score	2	1	1	3	1	3	1	2	1	3	2	2	3	1	2	1	3	1
Trace length	8	27	18	6	23	18	33	35	32	9	22	13	13	20	12	13	9	42

The correlation coefficient between the teacher's grades and the automatic scores generated by Laalys V2 is good, 0.75 for both levels. We notice divergence with students S6 and S8 on "The thermometer" and students S2 and S6 on "The ice wall":

- Thermometer S6: This case is the biggest gap between the grades given by the teacher (3) and the automatic score (1). The teacher noticed that the student groped for the solution, tried lots of things and checked the temperature after each action. The algorithm detected this behavior by tagging several labels "Stagnation" in the student's trace. However, the teacher pointed a bug in the game, the aid tool was not activated during the activity of S6. The teacher took into account this problem and this explains the gap between the assessments and his/her indulgence.
- Thermometer S8: The teacher notices that this student plays sticks without testing the exit each time. This is a good behavior from the teacher's point of view, but the game engine doesn't trace this information. If that were the case the computed score would increase regarding on other students who had not this behavior.
- Ice wall S2: The teacher justify its grade with the same explanations as the ones of the student S5 but didn't give the same score. From the automatic analysis point of view, the two traces are very similar and the score too. This seems to be a more consistent result than teacher's evaluation.
- Ice wall S6: The teacher noticed that the student followed the best procedure like other students who get 3 (S1, S4 and S8). He didn't explain the differences between these students. In this case, automatic analysis seems to be more consistent.

6 Conclusion and future research

The work presented in this article deals with the assessment of learners' behavior in serious games. The main contribution is the algorithm to label learner's actions pedagogically. The overall architecture is used to design the 18 levels of the LG "Les Cristaux d'Ehere" and produced full and filtered Petri nets of these levels automatically. The labeling algorithm was experimenting in a qualitative study with nine students and four teachers from junior high school. The evaluation gave positive results both on the label

effectiveness and the relevance of the calculated score. This research work has also identified several future research lines: (1) using the labels in order to implement adapted online feedback to learners, (2) comparing the Petri nets (FilteredPn) built by filtering the full Petri nets automatically (FullPn) with the Petri nets built by experts manually and (3) implementing the propagation of the labels into a learner model.

References

1. Essam Kosba, Vania Dimitrova, and Roger Boyle. Adaptive feedback generation to support teachers in web-based distance education. *User Modeling and User-Adapted Interaction*, 17(4):379–413, 2007.
2. Brijesh Kumar Baradwaj and Saurabh Pal. Mining educational data to analyze students' performance. *CoRR*, abs/1201.3417, 2012.
3. Luca Mazzola and Riccardo Mazza. Supporting learners in adaptive learning environments through the enhancement of the student model. In *Proceedings of the 13th International Conference on Human-Computer Interaction. Part IV: Interacting in Various Application Domains*, pages 166–175, Berlin, Heidelberg, 2009. Springer-Verlag.
4. Ben Medler, Michael John, and Jeff Lane. Data cracker: Developing a visual game analytic tool for analyzing online gameplay. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 2365–2374, New York, NY, USA, 2011. ACM.
5. G. Wallner and S. Kriglstein. Technical section: Plato: A visual analytics system for gameplay data. *Comput. Graph.*, 38:341–356, February 2014.
6. Erik Harpstead, Christopher J. MacLellan, Kenneth R. Koedinger, Vincent Aleven, Steven P. Dow, and Brad A. Myers. Investigating the solution space of an open-ended educational game using conceptual feature extraction. In *Proceedings of the 6th International Conference on Educational Data Mining, Memphis, Tennessee, USA, July 6-9, 2013*, pages 51–58, 2013.
7. Ángel del Blanco, Javier Torrente, Eugenio J. Marchiori, Iván Martínez-Ortiz, Pablo Moreno-Ger, and Baltasar Fernández-Manjón. Easing assessment of game-based learning with e-adventure and lams. In *Proceedings of the Second ACM International Workshop on Multimedia Technologies for Distance Leaning*, MTDL '10, pages 25–30, New York, NY, USA, 2010. ACM.
8. Amel Yessad, Pradeepa Thomas, Bruno Capdevila Ibáñez, and Jean-Marc Labat. Using the petri nets for the learner assessment in serious games. In *ICWL*, pages 339–348, 2010.
9. Pradeepa Thomas, Jean-Marc Labat, Mathieu Muratet, and Amel Yessad. How to evaluate competencies in game-based learning systems automatically? In *Intelligent Tutoring Systems*, pages 168–173. Springer Berlin Heidelberg, 2012.
10. J. L. Peterson. *Petri Net Theory and Modeling of Systems*. Prentice Hall, Reading, Mass, 1981.
11. Mathieu Muratet, Amel Yessad, and Thibault Carron. Framework for learner assessment in learning games. In Mike Sharples, Katrien Verbert, and Tomaž Klobučar, editors, *Adaptative and Adaptable Learning: 11th European Conference on Technology Enhanced Learning, EC-TEL 2016*. Springer, 2016.
12. Pradeepa Thomas, Amel Yessad, and Jean-Marc Labat. Petri nets and ontologies: Tools for the "learning player" assessment in serious games. In *ICALT*, pages 415–419, 2011.