# Using the Petri Nets for the Learner Assessment in Serious Games

Amel Yessad, Pradeepa Thomas, Bruno Capdevila, and Jean-Marc Labat

Laboratoire d'Informatique de Paris 6
Université Pierre et Marie Curie
4, place Jussieu,
75005 Paris, France
{amel.yessad,pradeepa.thomas,bruno.capdevila,
jean-marc.labat}@lip6.fr

**Abstract.** Game-based learning or serious games is becoming an important trend in the e-learning research area and seems address several typical e-learning problems such as high dropout rates, due to the lack of motivation to continue studying. In serious games, it is very hard to define and mix the learning situations with the game characteristics, and to integrate an assessment and guidance process of the learner without disturbing the game progress and maintain the intrinsic characteristics of the video game: fun, player motivation, immersion and interaction. In this paper, we consider the serious game as an asynchronous and concurrent system, and we propose an approach based on a Petri net to assess learners and detect misconceptions. In the game design stage, a discussion between domain experts, learning experts, and game designers is engaged in order to identify the actions in the game that imply knowledge acquisition and allow achieving the learning objectives of levels. Therefore, in our approach, the Petri net models only game actions allowing the learner to acquire knowledge. We use the reachability graph of the Petri net to track the learner in order to detect, in real time, the learner's misconceptions, improve learner assessment and provide an accurate feed back for both the learner and the instructor.

**Keywords:** Game-based learning, serious games, learner misconception, learner assessment, Petri nets, reachability graph.

## 1 Domain Overview

Game-based learning or serious games are video games that allow one to achieve an educational goal while remaining fun and allowing both a high interaction with players and interactivity in the game. It is usual that the game allows solving problems in an artificial environment with well defined rules. The benefits of serious games compared with classical elearning systems is motivating learners, emerging them in an artificial world and providing them outcomes and feedback in real-time.

There are several learning approaches that can be used in a game, such as learning by doing, learning from mistakes, goal-oriented learning, role playing, and constructivist learning [1, 2].

X. Luo et al. (Eds.): ICWL 2010, LNCS 6483, pp. 339–348, 2010.

According to [3] « For a game to be a serious one, some features are required [4, 5]:

- it has to start with a problem to be solved
- it has to be unproductive itself; it does not generate any property or wealth. The drive is the gaming activity itself
- it has one correct solution, at least
- it should have something to be learnt by the user/player, while introducing new knowledge, fixing previous acquired knowledge, training skills, sharing experiences, discovering new concepts, developing outcomes»

In our work, we retain this serious game definition and, moreover, we consider a serious game as a dynamic system in which parallel and concurrent events appear in response to the actions of the player and other game objects, changing the state of the game.

Examples of video games used for learning include games like StarBank, blossom Flowers, SimCity, Flight Simulator, Civilization, Climway, etc.

All these serious games allow the player freedom; he/she can often perform many game actions from the beginning of the game without interruption or feedback from the instructor. This situation causes two problems. On one hand, some players may find themselves in a situation of disorientation and cognitive overload being incapable to choose the right action and they could even abandon the game. on the other hand, the instructor is in a position of uncertainty about the actual learning of the player and so it is very difficult to assess the latter. Indeed, the player who has achieved the game objectives can still have misconceptions about domain knowledge.

In this paper, we propose an approach that enables not only the instructor to detect these misconceptions and so improve the assessment of the learner, but also to provide an accurate feedback to the learner about her/his progression in the level.

We do not aim to modify the player's path in the game but to propose a mechanism to assess the learner in serious games best. Our approach is based on the construction of a Petri net of the serious game. This Petri net models only game actions that reveal good knowledge acquisition. Once the Petri net has been constructed, we can automatically calculate the reachability graph that models the evolution of the game states following the leaner's actions. These states are coherent with rules defined in the domain knowledge.

Many researches have used Petri nets for simulating real systems because they are adapted to model and evaluate asynchronous and concurrent systems. However, there is little research that addresses the use of Petri nets for serious games and even less for learner assessment. In [6] and [7] the Petri nets are used to analyze the games formally and to detect unwanted behaviours.

We experiment our approach on the game StarBank[1], which is a serious game about how to develop a bank. StarBank was developed by KTM-advance[2] for the bank BNP Paribas. This game aims at teaching the bank employees and the new comers to the bank different trades and businesses of the bank, the banking model and the development of the city around the bank.

---

[1] http://starbankthegame.bnpparibas.com/
[2] http://www.ktm-advance.com/

This paper is organized in two parts: In the first part, we present our learner assessment and detecting misconception approach; and in the second one, we apply our approach to the game StarBank in order to illustrate it.

We note that in this paper we use interchangeably the terms player and learner.

## 2 Petri Net and Reachability Graph

A Petri net is a bipartite graph, formally defined by a 5-tuple *PN* = (P, T, Pre, Post, $M_0$) [8], where P $=\{p_1, p_2, \ldots, p_n\}$ is a finite set of places, T $= \{t_1, t_2, \ldots, t_m\}$ is a finite set of transitions, Pre : P ×T →N is the input incidence function, Post : T×P→N is the output incidence function and $M_0 : P \rightarrow N$ is the initial marking. In the graphical representation, circles denote places and bars denote transitions, tokens are represented with dots inside places. The Pre incidence function describes the oriented edges connecting places to transitions. It represents for each transition t the fragment of the state in which the system has to be before the state change corresponding to t occurs. Pre(p, t) is the weight of the arc (p, t), Pre(p, t) = 0 denotes that the place p is not connected to transition t. The Post incidence function describes arcs from transitions to places. Analogously to Pre, Post(t, p) is the weight of the arc (t, p). The vectors Pre(., t) and Post(t, .) denote respectively all input and output arcs of transition t with their weights (cf. fig. 1).

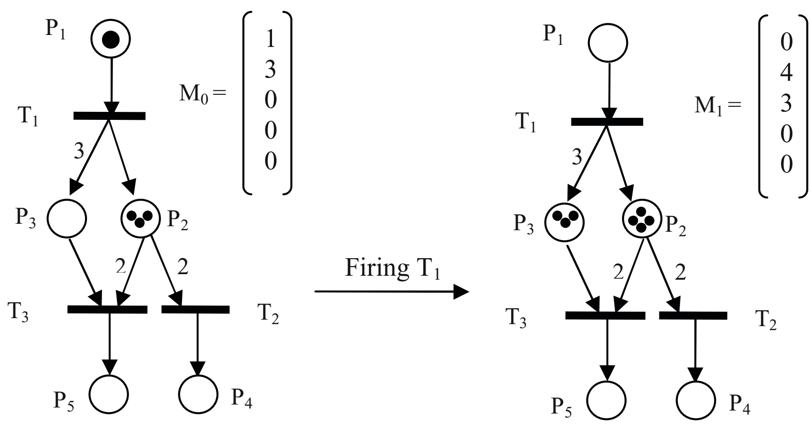
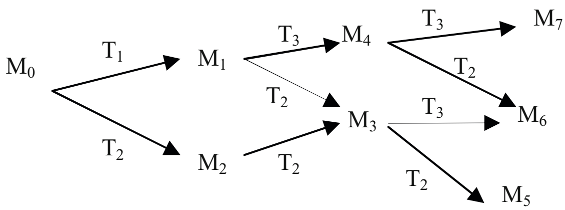



**Fig. 1.** Petri net example (firing the transition T1)

The Petri net marking changes by firing enabled transitions and corresponds to a state change of the system modelled by the Petri net. A transition t is enabled for a marking M, if $M \geq Pre(., t)$. This enabling condition is equivalent to $\forall p \in P, M(p) \geq Pre(p, t)$. Only enabled transitions can be fired. If M is a marking of *PN* enabling a transition t, and M′ is the marking derived by the firing of a transition t from M, then $M' = (M - Pre(., t)) + Post(t, .)$. The firing is denoted as $M \xrightarrow{t} M'$. Firing of a sequence of transitions s = $<t_1 \ldots t_n>$ is defined as $M_0 \xrightarrow{s} M_n = M_0 \xrightarrow{t1}$

$M_1 \xrightarrow{t2} \ldots \xrightarrow{tn} M_n$, where $M_i$, $i = 0. . . n - 1$ is a marking of *PN* enabling a transition $t_{i+1}$ and $M_{i+1}$ is a result of firing $t_{i+1}$ from marking Mi. In a Petri net, a marking $M_g$ is reachable from a marking M if there exists a sequence of transitions s such that $M \xrightarrow{s} M_g$. We call the reachability graph *RG*, the oriented graph where the vertices are the markings reachable from the initial marking $M_0$ and the arcs are the transitions that relate the reachable markings.




$M_o$=(1,3,0,0,0), $M_1$=(0,4,3,0,0), $M_2$=(1,1,0,1,0), $M_3$=(0,2,3,1,0)
$M_4$=(0,2,2,0,1), $M_5$=(0,0,3,2,0), $M_6$=(0,0,2,1,1), $M_7$=(0,0,1,0,2)

**Fig. 2.** The reachability graph of the Petri net of the fig. 1

## 3 Learner Assessment and Detecting Misconceptions by Using Learning Petri Nets

The Petri net is a formal state transition model used to analyse, model, and evaluate asynchronous and concurrent systems. It allows checking properties of the dynamic systems (safety, liveness, deadlock, etc.). In our work, we use the Petri net in order to perform an efficient learning tracking of the player in order to assess his progress, and generate for him an accurate feedback. In our approach, the learner tracking is performed in real time during the game progress by checking the reachability of the current state of the serious game, based on the reachability graph.

In our approach, the Petri net is constructed from domain knowledge, which contains concepts, semantic properties between concepts and domain rules. This construction is manual and conducted jointly by learning experts, domain experts, and modeling experts.

All learning objectives of a serious game are based on domain knowledge and are defined by the actors (game designers, learning experts, domain experts, etc.) during the game design stage of the game development. Thus, when the game level is constructed, the learning expert defines the learning objectives of the level that are projected onto the knowledge domain in order to filter concepts, properties and rules that allow achieving learning objectives. Based on this partial knowledge domain, the Petri net is manually constructed by the modeling expert. Thus, in our context, the Petri net 5-tuple (P, T, Pre, Post, $M_0$) is defined by:

- P is a set of game characteristics that are relevant to observe in order to track and assess the player, For instance in adventure games, a place can be a game characteristic such as resources (life points, skills, arms), the defeated monsters, the visited countries, etc.

- T is a set of game actions that allow acquiring domain knowledge, such as choose a weapon, make decision, go to a place
- Pre is the function that defines for each transition the preconditions necessary to fire it. These preconditions have to be verified only on the learning side and not necessarily in the game. For instance, it is better to fight the monster with a powerful weapon
- Post is the function that defines for each transition the post conditions resulting from firing it. For instance, the monster is defeated and the country is freed
- $M_0$ is the initial state of the serious game. Naturally, only the game characteristics relevant to learner tracking and assessment are represented. It can be the list of the weapons of the player, the number of monsters, the number of life points

We call this Petri net a learning Petri net from which we calculate the reachability graph (cf. fig. 3). The reachability graph represents the critical paths of the player in the game. It informs us about the player's progress and knowledge acquiring.

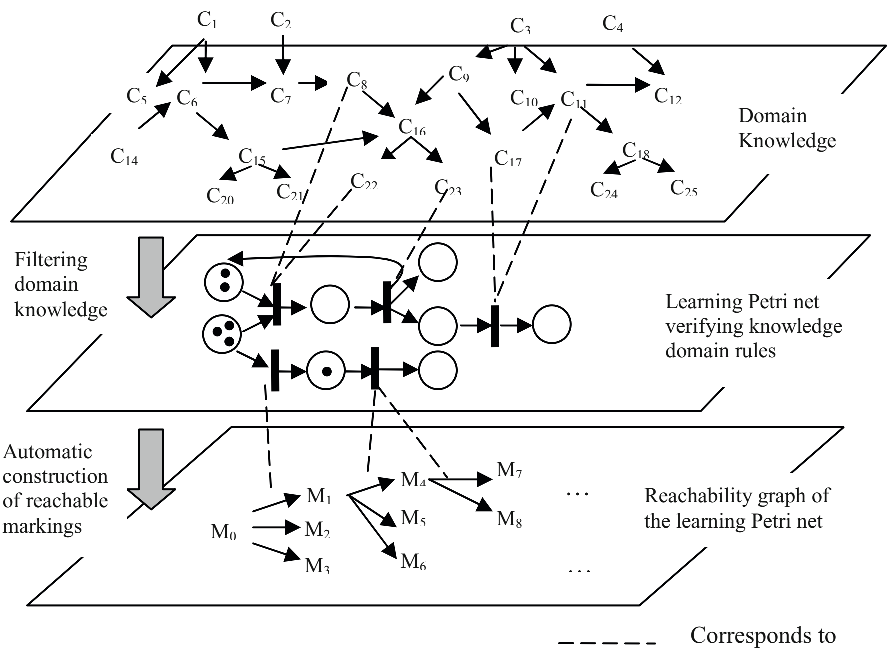



**Fig. 3.** Learning Layers of serious games in our approach

In our approach, detecting the learner gap from the learning objectives consists in detecting when the learner performs a game action that does not corresponds to an enabled transition in the Petri net, i.e. this game action is not an output arc from the current marking of the reachability graph. It means the player has a mental representation of the knowledge domain that is at odds with the educational objectives and the domain representation of experts. However, the player is free to perform this game action because we want to maintain freedom in the game.

Our approach based on the use of the reachability graph of the learning Petri net consists of:

- Tracking the learner in order to:
  - verify if he/she progresses in the game, i.e. he/she performs game actions that correspond to enabled transitions of the learning Petri net marking;
  - detect learner gaps from the knowledge acquisition process representing a misconception, i.e. he/she performs a disabled transition of the Petri net marking;
- If the learner progresses in the game we update his/her learner model. Otherwise, we dynamically recalculate the reachability graph from the current state representing a misconception in order to follow the reasoning of the learner. The idea is not to punish immediately the misconception but to leave the player the chance to progress again, learning from his mistakes. Here, we adopt the same method used by mathematic instructors who meet a miscalculation and penalize the learner for the error but continue to evaluate the rest of the exercise with the miscalculation.
- During or at the end of the game, proposing to the learner an accurate feedback in order to explain to him misconceptions or the reasons for his failure.

### 3.1 Detecting Learner Misconceptions

Every time the player performs a game action, we verify if this action corresponds to a Petri net transition. If so, we verify if the Petri net marking that models the current state of the serious game allows firing the transition. If so, we note that the player is progressing and that he has acquired the knowledge associated to the game action and so we update the learner model. Otherwise, we note that the player has a problem in his mental representation of the knowledge domain; this is a misconception.

When a misconception is detected, we consider the current state of the serious game as the initial marking of the Petri net; and we recalculate the reachability graph from this initial marking. The learning expert has to define the number of misconceptions allowed before stopping the level and declaring the player's failure to achieve the learning objectives.

In order to save computing and memory resources, the system constructs the reachability graph dynamically and gradually as the learner progresses through the level.

### 3.2 Feedback to Learners

We propose two strategies for producing feedback to the learner; either the system gives him guidance during the game level or explanations at the end of the game level.

During the game, the player performs game actions allowing him either to acquire knowledge (transitions of the Petri net) or simply to bring fun and pleasure. When the player performs actions of the second category (just for the fun), the marking of the learning Petri net is not changed. After awhile, we can assume he/she is lost and does not know what do. In this case, the system gives advice to the learner indicating the actions that he/she could perform in order to modify the marking of the Petri net.

At the end of the level two situations are possible: the player has either won or lost. When the player reaches the final marking of the Petri net, it means that learning objectives have been achieved and therefore the knowledge of the level has been validated. During the level, if the player performs disabled transitions, the system saves this misconception and recalculates the reachability graph. Given that each transition of the learning Petri net is associated with concepts of the knowledge domain; it is easy to explain the learner misconception automatically.

Similarly, when the player loses, the system knows the transitions that cause misconceptions or enabled transitions not performed by the player and an accurate explanation about the reasons for failure is transmitted to both the instructor and the player in order to debrief the level.

# 4 Experimentation on the Serious Game StarBank

StarBank is a serious game developed by Ktm-Advance for a Bank, BNP Parisbas, in order to train the newcomers and the promoted employees (cf. fig.5). StarBank takes place in the space where the player plays a role of a banker and invests in financial services. He participates in the economic growth of the city in outer space and discovers the rules and the organization of a bank and the principles of profitability.

In StarBank, the learner must:

- Create agencies for individual clients
- Create business centers for corporate customers
- Create the service account management for businesses
- Create and develop placement services for businesses
- Create financing services
- Invest in the stock market

In order to achieve the level objectives, the player's actions must be coherent with the domain model of the bank. The player has an overview of the company's key performance indicators such as available cash, number of employees, quoted prices of the company and can see the immediate effects on them of his/her decisions.

## 4.1 Tracking and Assessing Learners with SeGAE API

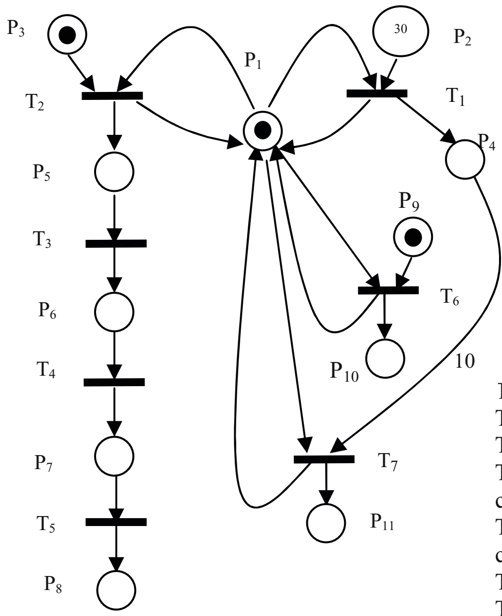
In the game StarBank, the learner learns to manage and develop the bank and participate in the city's development. StarBank is organized in levels with challenges. In order to achieve the level objectives, the learner has to acquire skills (increasing the number of customers, making the bank grow, etc.). While the player acquires the skills, the city grows reflecting his progress in the game.

The approach proposed in this paper allows assessing the learner's progression and his mental representation of the knowledge domain. Fig. 4 represents an excerpt of the learning Petri net of the level 2 of StarBank. The rules extracted from the knowledge domain and represented in this Petri net excerpt are:

- The player must create agencies for individual clients. At the end of level 2, he must create thirty agencies in the city.
- He/she has to create a business center for businesses.

- He/she has to create services in the business center: account management services, investment services, financing services.
- The player has to create a customer multi channel. He/she can create a multi channel from ten agencies created.

We designed and developed an authoring environment, SeGAE (Serious Game Authoring Environment) [9], which allows, among other features, instructors to define the game rules and the game actions (the transitions of the learning Petri net) which allow knowledge acquisition. In SeGAE, the reachable markings from the current marking of the Petri net are automatically calculated. In order to save system resources, only the markings directly reachable are calculated. An API is under development in order to exchange data between SeGAE and the game engine of StarBank. SeGAE listen events emitted when a Petri net transition is fired by the player in the game level. It also transmits to the game engine the guidance information during the level and a debriefing at the end of the level. SeGAE maintains a database about the misconceptions and the learner model used for the learner assessment.




$P_1$ : the level 2 is being
$P_2$ : the number of agencies to create
$P_3$ : the number of agencies created
$P_4$ : ready to create a business center
$P_5$ : business center created
$P_6$: account management service created
$P_7$: investment service created
$P_8$: financing service created
$P_9$ : ready to create the risk department
$P_{10}$ : the risk department is created
$P_{11}$ : the number of customer channels

$T_1$ : create bank agency
$T_2$ : create a business center
$T_3$ : create an account management service
$T_4$ : create an investment service for corporate customer
$T_5$ : create a financing service for corporate customer
$T_6$ : create a risk department
$T_7$ : create a customer multi channel

**Fig. 4.** An excerpt of the learning Petri net of the serious game StarBank

### 4.2 Detecting Learner Misconceptions

As presented in the former part of this paper, the reachability graph is used to detect if there is a gap from the learning objectives of the level, i.e. the learner performs a game action that does not correspond to an output arc from the current marking of the reachability graph. A gap means either the learner's action is not relevant to achieve the learning goals of the level; or given the game state, it is not recommended for the learner to perform this action. For example, in the Petri net of fig.4 the player can not create a customer multi channel before creating at least ten agencies. This rule is not specified in the game design of the level in order to increase the player’s freedom in the game. However, if the player performs this action, he will find himself in a game state that is incoherent with the learning objectives. In this case, we change the initial marking of the Petri net and regenerate a new reachability graph which will be used from now on to track the player. In StarBank, we apply this strategy up to three times and then we consider that the player has failed in the level.


**Fig. 5.** A screenshot of StarBank (left), a feedback about a misconception (right)

## 5 Conclusion

In this paper, we present an approach for assessing the progress of the learner and detecting misconceptions in serious games. The learner assessment is done in real time during the game. Our approach consists in tracking the learner, detecting when the learner performs or not a game action of the Petri net, the gap between the current state and the learning paths; and generating advice in order to help the learner to improve his knowledge acquisition. The central model of our approach is a learning Petri net; an abstraction of the serious game dynamic. This abstraction contains only game actions that allow knowledge acquisition. Our approach enables not only the instructor to detect these misconceptions and so improve the assessment of the learner but also to provide an accurate feedback to the learner about her/his progression in the level.

In order to experiment our approach, we applied it to the serious game StarBank. We designed and developed SeGAE, a Serious Game Authoring Environment that allows specifying domain rules of the StarBank and tracking the learner in order to assess her/his progress and detect misconceptions. For this purpose, we are developing an API allowing communication between SeGAE and the StarBank game engine in order to track the learner during the level and transmit guidance and debriefing to him/her.

Our experiments give good results on StarBank; but it is mandatory to continue experiments on other serious games. Currently, we are studying rules allowing the construction of the learning Petri net from the knowledge domain. We aim to make this hard and complex task semi-automatic.

## References


1. Burgos, D., Tattersall, C., Koper, R.: Re-purposing existing generic games and simulations for e-learning. Computers in Human Behavior 23(6), 2656–2667 (2007)
2. Prensky, M.: Digital Game Based Learning. McGraw-Hill, New York (2001)
3. Burgos, D., Tattersall, C., Koper, R.: Can IMS Learning Design be used to model computer-based educational games? BINARIA Magazine 5 (2006)
4. Sutton-Smith, B.: The ambiguity of play. Harvard University Press, Cambridge (2001)
5. Salen, K., Zimmerman, E.: Rules of play: game design fundamentals. The MIT Press, Cambridge (2003)
6. Natkin, S., Vega, L.: A Petri Net Model for Computer Games Analysis. Int. J. Intell. Games & Simulation 3(1), 37–44 (2004)
7. Araújo, M., Licínio, R.: Modeling Games with Petri NetsDigital. In: Proceedings of Digital Games Research Association, DiGRA (2009)
8. Küngas, P.: Petri net reachability checking is polynomial with optimal abstraction hierarchies. In: Zucker, J.-D., Saitta, L. (eds.) SARA 2005. LNCS (LNAI), vol. 3607, pp. 149–164. Springer, Heidelberg (2005)
9. Yessad, A., Labat, J.M., Kermorvant, F.: SeGAE: a Serious Game Authoring Environment. To appear in Proceeding of The 10th IEEE International Conference on Advanced Learning Technologies, Sousse (2010)