# Prog&Play: un jeu sérieux instrumentalisé pour l'apprentissage de la programmation

A. Yessad[1], M. Muratet[1,2], B. Bontemps[1], and S. Meresse[1]

[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
{mathieu.muratet, amel.yessad}@lip6.fr

**Abstract.** Prog&Play est un jeu sérieux basé sur un moteur de jeu de stratégie temps réel qui a pour objectif de faire travailler des apprenants sur des concepts de programmation et d'algorithmique. Prog&Play est utilisé depuis plusieurs années avec des étudiants de L1 à l'UPMC et l'université de Toulouse et plusieurs besoins nous ont été remontés du terrain. Pour répondre à ces besoins, nous avons intégré à ce jeu (1) un système auteur qui répond à un besoin d'instrumentalisation du jeu par les enseignants et (2) un système d'analyse des traces de résolution de l'apprenant permettant de fournir automatiquement un feedback à ce dernier sur sa résolution.

**Keywords:** Jeu sérieux, Système auteur, Prog&Play, Feedback, Analyse de traces

## 1 Introduction

Dans le domaine des EIAH, de nombreux systèmes peuvent être développés et proposés aux enseignants pour les assister à mettre en place une pédagogie différente. Or, les enseignants ont besoin d'une part de comprendre ces outils et d'autre part de se les approprier. L'approche instrumentale de Rabardel [1] repose sur le concept de genèse instrumentale qui consiste en l'élaboration de l'instrument à partir de l'artefact par l'utilisateur au cours de l'activité. Rabardel définit l'instrumentalisation comme le fait que les utilisateurs modifient les artefacts pour les adapter à leurs besoins et l'instrumentation comme le fait que l'utilisateur lui-même se modifie en apprenant à maîtriser l'artefact.

Prog&Play est un jeu sérieux qui a pour objectif de faire travailler des étudiants, engagés dans des filières scientifiques, sur des concepts d'algorithmique et de programmation. Il s'agit d'un jeu de stratégie temps réel (STR) basé sur le moteur Spring Engine dans lequel le joueur contrôle des entités (appelées unités) afin de réaliser des missions. Ce jeu se différencie des autres STR essentiellement par son mode d'interaction avec les unités : dans un STR classique (comme Warcraft ou Age of Empire) les unités sont contrôlées en utilisant la souris et le clavier, tandis que dans Prog&Play, elles sont contrôlées par du code source faisant appel à une API permettant d'interagir avec le jeu

(sélectionner des unités, les déplacer, les attaquer, les réparer, etc.). Ce code source peut être écrit en C, Java, OCaml, Python, Ada ou Scratch[3] (voir Fig. 1).

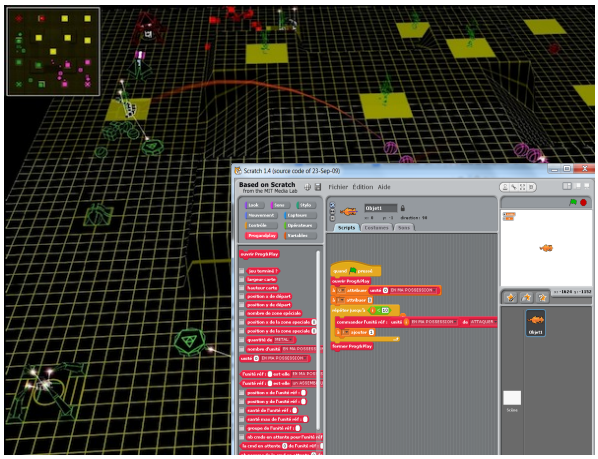

**Fig. 1.** Une capture d'écran du jeu sérieux Prog&Play et son API Scratch

Initialement, le jeu Prog&Play n'est composé que de huit missions distinctes (également appelées niveaux) linéairement scénarisées allant d'un simple déplacement d'unités à la gestion avancée d'une bataille contre l'ordinateur. La modification ou l'ajout de niveaux ne peut se faire que par l'écriture de code source en Lua, ce qui constitue sa limitation principale, d'autant plus que l'API fournie par Spring est peu documentée.

Dans la perspective de rendre intelligible (démarche d'instrumentation) le jeu sérieux Prog&Play pour les enseignants et permettre son appropriation par ces derniers (démarche d'instrumentalisation), nous avons conçu et développé SPRED (SPRing EDitor). L'objectif de ce premier travail était donc de concevoir un outil auteur à destination des enseignants. Un tel outil leur permettra d'adapter le jeu à leur contexte en créant de nouveaux niveaux et scénarios ou simplement en modifiant les niveaux déjà existants. Le travail a consisté donc à proposer un éditeur interactif intégré au moteur Spring (il ne s'agit donc pas d'un logiciel séparé) permettant de définir les contraintes des niveaux sans avoir à écrire la moindre ligne de code, puis de créer une scénarisation à partir des niveaux créés.

Ensuite, nous avons voulu intégrer un module d'analyse de traces à Prog&Play permettant la génération automatique de feedbacks à destination des apprenants pour les renseigner sur la qualité de leurs solutions. Etant donné que Prog&Play est compatible avec plusieurs langages de programmation et n'impose pas d'environnement de programmation, une approche qui serait basée sur l'analyse statique des programmes impliquera nécessairement le développement de modules spécifiques à chaque langage

[3] Scratch est un logiciel de programmation graphique à base de blocs s'imbriquant les uns dans les autres. Accédé le 12 octobre 2016 : `https://scratch.mit.edu/`

et environnement de programmation. L'approche que nous avons envisagée ne consiste donc pas à analyser la production de l'étudiant (son code source) mais à analyser la trace d'exécution de son programme afin d'être indépendant du langage et de l'environnement de programmation utilisé. Cependant, de telles traces peuvent être volumineuses et sont souvent pauvres d'un point de vue sémantique car seuls les appels aux fonctions de la bibliothèque Prog&Play sont tracés. Nous avons ainsi dégagé deux questions de recherche :

1. Comment reconstituer au mieux la sémantique du code source d'un programme informatique écrit par un apprenant à partir de sa trace d'exécution brute et partielle ?
2. Comment, à partir d'une trace d'exécution ainsi reconstituée, générer automatiquement un feedback pertinent et utile pour un apprenant ?

## 2 Architecture de la chaîne d'outils Prog&Play

La Fig. 2 schématise l'architecture de la chaîne d'outils associé à Prog&Play.

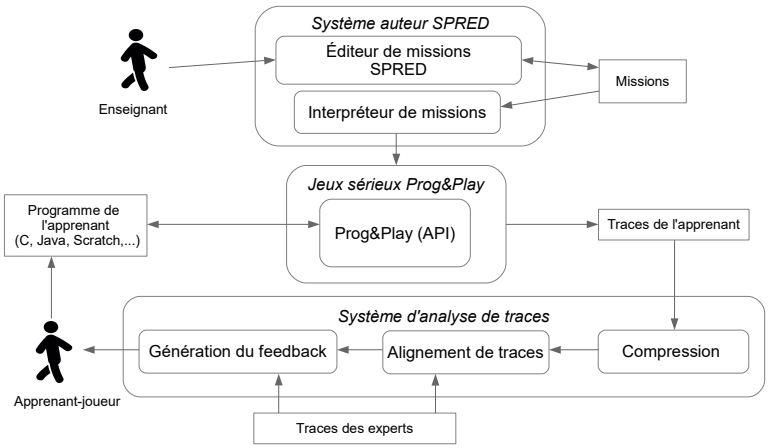


**Fig. 2.** Architecture de la chaîne d'outils Prog&Play

### 2.1 SPRED: le système auteur de Prog&Play

La conception de l'interface de SPRED est centrée utilisateurs. Nous avons commencé par lister la plupart des fonctionnalités à implémenter, puis chaque élément d'interface a été soumis à un prototype papier, et enfin nous avons réalisé des tests utilisateurs précis auprès de trois enseignants. Nous avons implémenté plusieurs fonctionnalités, parmi lesquelles : (1) sauvegarde et chargement des niveaux, (2) instanciation d'unités sur le terrain avec la possibilité de définir le type, l'équipe, la position, la rotation, le nombre

de points de vie initiaux, etc., (3) définition des déclencheurs (appelés événements dans notre cas et composés de couples conditions/actions) pour définir le comportement des unités, (4) création de zones logiques ou de groupes d'unités pouvant être utilisées dans le système de déclencheurs, (5) paramétrisation des équipes de joueurs : activation ou désactivation d'une équipe, contrôle par un joueur ou par une intelligence artificielle, (6) scénarisation des niveaux créés, et (7) exportation des niveaux et de leur scénarisation sous la forme d'un mod Spring.

### 2.2 Le système d'analyse de traces de Prog&Play

Le travail de recherche qui a été mené pour analyser les traces d'exécution des programmes informatiques construits par les apprenants comporte deux questions de recherche principales explicitées dans la section 1.

Pour répondre à la première question, nous nous sommes inspirés du travail de Hamou-Lhadj et Lethbridge [2] pour mettre au point un algorithme capable de compresser une trace brute produite lors de l'exécution d'un programme rédigé par un apprenant.

En ce qui concerne la seconde question, nous avons développé un algorithme d'alignement de traces compressées adapté de celui de Needleman et Wunsch [3]. Cet algorithme est utilisé pour identifier parmi un ensemble de traces compressées expertes de la mission jouée celle qui permet d'obtenir le meilleur alignement avec la trace compressée de l'apprenant. Grâce à cet alignement, il est ainsi possible d'identifier les points de divergences entre les deux structures de programme reconstituées afin de générer automatiquement un *feedback* à destination de l'apprenant. La qualité de ce *feedback* repose bien évidemment sur la qualité de la compression des traces d'exécution, réalisée de façon heuristique, mais également sur le niveau de contribution des experts qui ont la possibilité d'injecter des informations sémantiques pouvant améliorer la fiabilité du processus d'analyse.

## 3 Conclusion

Cet article décrit la chaîne d'outils associée au jeu sérieux Prog&Play. Cette chaine d'outils émane de besoins recueillis auprès d'enseignants qui utilisent le jeu depuis plusieurs années pour faire travailler des étudiants de L1 sur des concepts de programmation et d'algorithmique.

## References


1. Pierre Rabardel. Les hommes et les technologies; approche cognitive des instruments contemporains. 1995.
2. Abdelwahab Hamou-Lhadj and Timothy C Lethbridge. Compression techniques to simplify the analysis of large execution traces. In *Program Comprehension, 2002. Proceedings. 10th International Workshop on*, pages 159–168. IEEE, 2002.
3. Saul B Needleman and Christian D Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453, 1970.