Alice, Greenfoot et Prog&Play ou Comment Apprendre la Programmation Orientée-Objet par le Jeu

Fahima Djelil[1], Adélaïde Albouy-kissi[1], Benjamin Albouy-kissi[1], Eric Sanchez[2], Jean-Marc Lavest[1]
[1] Université Blaise Pascal, Laboratoire Institut Pascal CNRS-UMR 6602, 63171 Aubière, Clermont-Ferrand, France
{fahima.djelil, adelaide.kissi, benjamin.albouy, j-marc.lavest}@udamail.fr
[2] EducTice, Institut Français de l'Education, Ecole Normale Supérieure de Lyon, 69342 Lyon, France
eric.sanchez@ens-lyon.fr

Résumé. Basé sur la littérature, cet article décrit trois environnements d'apprentissage conçus dans le but d'aider les étudiants novices à acquérir les compétences de la programmation informatique: Alice, Greenfoot et Prog&Play, en vue de présenter leurs mécanismes de représentation des concepts fondamentaux de la Programmation Orienté-Objet, et de vérifier leur adéquation aux profils d'étudiants débutants et leur impact sur l'apprentissage de ces étudiants.

Mots-clés. Alice, Greenfoot, Prog&Play, Programmation Orientée-Objet, Apprentissage par le Jeu

Abstract. Based on the literature, this article describes three learning environments that have been designed to help novice students to develop computer programming skills: Alice, Greenfoot and Prog&Play, in a view to present their principals of how they represent abstract fundamentals of Object-Oriented Programming, and to verify their adequacy for novice students and their impact on the students learning.

Keywords. Alice, Greenfoot, Prog&Play, Object-Oriented Programming, Game Based Learning

1 Introduction

Le paradigme orientée-objet, dont les concepts de base possèdent un haut niveau d'abstraction, est l'un des paradigmes de programmation le plus enseigné dans les cours d'initiation à la programmation et le plus difficilement perçu par les étudiants [1], par conséquent plusieurs environnements ont été conçus visant à réduire la complexité de l'apprentissage et à renforcer la motivation des débutants. Ces environnements emploient des représentations graphiques et permettent de concevoir des scénarios ludiques ou d'interagir avec des jeux existants en manipulant des programmes informatiques.

Notre choix s'est porté sur les environnements Alice [2], Greenfoot [3] et Prog&Play [4] présentant plusieurs similarités dans leur approche de fonctionnement et leurs objectifs pédagogiques. En effet, ces environnements visent à motiver

l'étudiant en lui offrant la possibilité de concevoir des programmes dans lesquels il peut manipuler des entités concrètes visualisées de façon graphique, tout en se divertissant par l'emploi de narrations, simulations et jeux.

Ce travail a pour objectif de répondre aux trois questions suivantes: Comment représentent-ils les concepts fondamentaux de la POO ? Sont-ils adaptés aux profils d'étudiants débutants? En fin, quel est leur impact sur l'apprentissage?

2 Analyse

Alice et Greenfoot représentent les concepts de la POO en employant des visualisations graphiques dans un contexte assez concret et significatif. A titre d'exemple, les objets sont instanciés de façon explicite à partir de graphiques 3D dans Alice, et à partir d'un diagramme de classes dans Greenfoot. Ces environnements offrent la possibilité de visualiser l'état et le comportement d'objets, puis d'accéder au code source leurs correspondants. Greenfoot permet ainsi de manipuler le langage Java, tandis qu'Alice repose principalement sur son propre pseudo-langage. Prog&Play, à l'opposé, n'offre pas de représentations concrètes de ces concepts et se concentre plutôt sur la programmation procédurale, mais permet d'apprendre les langages Java et C++. D'autres notions comme l'héritage et le polymorphisme sont assez bien illustrées dans Greenfoot, grâce notamment au diagramme de classes d'objets constituants sa scène, contrairement à Alice qui ne représente pas de façon claire certaines relations entre classes, comme la composition qui n'est pas nommée et l'héritage qui se traduit par plusieurs manipulations successives et ambiguës. Greenfoot présente néanmoins certains inconvénients. Bien que son utilisateur ait accès au code source correspondant aux classes d'objets représentés dans sa scène, il masque toutes les opérations relatives au programme principal du scénario réalisé. L'adéquation d'Alice et de Greenfoot aux profils d'étudiants débutants a été démontrée dans la littérature [5], [6] , [3]. En effet, Alice réduit la complexité et les détails de programmation grâce, notamment, à son éditeur de scripts qui est manipulable par des glisser-déposer de blocs d'instructions textuelles [2]. Cela permet aux novices de s'affranchir des erreurs de syntaxe et de se concentrer sur les concepts et la logique des programmes [2]. Les représentations 3D apportent également plus de réalisme aux objets manipulés et facilitent la compréhension des concepts abstraits [5]. Cette caractéristique est également présente chez Greenfoot qui permet de visualiser les concepts fondamentaux de la POO, avant de manipuler le code source correspondant, ce qui est très utile pour les étudiants novices [3]. Les retours d'usage de Prog&Play rapportent la difficulté rencontrée par les enseignants à l'intégrer dans leur enseignement, en raison du temps nécessaire à sa prise en main [4]. Les enseignants l'ayant utilisé parlent d'un système adapté aux étudiants souhaitant pratiquer et approfondir leurs connaissances en programmation [4]. Certaines évaluations menées sur Alice, ont montré sa contribution à l'amélioration des performances des étudiants [2], [5], [6]. Les observations ont montré que les étudiants, n'étant pas confrontés aux erreurs de syntaxe, assimilent aisément certaines notions comme l'objet et l'invocation de méthodes, parviennent à comprendre et à corriger rapidement leurs erreurs et consacrent plus de temps à leurs projets [5], [6].

Quant à Greenfoot, aucune étude formelle de son impact sur l'apprentissage n'a été entreprise, bien qu'il ait fait l'objet de plusieurs utilisations en classe [3]. Les expérimentations menées également avec Prog&Play n'ont pas abouti à des résultats suffisamment valables pouvant servir à attester de son efficacité sur l'apprentissage [4]. Par ailleurs, les retours d'usage de ces trois environnements révèlent que les étudiants les trouvent très divertissants et engageants [5], [3], [4]. Ces retours d'expérience ont également montré la contribution de ces trois environnements au maintien de la motivation des étudiants [5], [3], [4].

3 Conclusion

Suite à cette analyse, nous distinguons dans ces environnements une relation permanente qui existe entre les concepts abstraits de la POO et les entités graphiques visualisées dans un contexte concret, significatif et ludique. Cette relation facilite la compréhension de ces concepts par les étudiants novices : plus ces environnements illustrent ces concepts, mieux ils s'adaptent aux profils d'étudiants débutants. Une autre propriété directement liée à cette relation, qui n'est cependant pas présente dans Prog&Play contrairement à Alice et à Greenfoot, concerne une phase primaire dans laquelle les concepts abstraits sont introduits par le biais de ces représentations graphiques, précédant une autre phase de manipulation de programmes dans leur syntaxe avancée. Cette manipulation se fait souvent de façon simplifiée en utilisant des fractions de code préalablement fournies. Cette propriété est assez intéressante pour les étudiants débutants, puisqu'il a été démontré que les étudiants novices comprennent mieux les concepts abstraits grâce à cette première phase de découverte, dans laquelle ils sont amenés à manipuler ces concepts de façon graphique et visuelle. Ceci constitue les principales forces de ces environnements, qui impactent de manière positive l'apprentissage des étudiants novices et le maintien de leur motivation.

Références

1. Börstler, J., Nordström, M., Westin, L. K., Moström, J.-E., Eliasson, J.: Transitioning to OOP/Java—A Never Ending Story. Reflections on the Teaching of Programming. Springer (2008) 80-97
2. Conway, M., Audia, S., Burnette, T., Cosgrove, D., Christiansen, K.: Alice: Lessons Learned from Building a 3D System for Novices. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (2000) 486-493
3. Kölling, M.: The Greenfoot Programming Environment. ACM Transactions on Computing Education (TOCE), vol. 10, n° 14, (2010)14-35
4. Delozanne, E., Jarraud, P., Muratet, M.: Un Projet Jeu Sérieux pour Approfondir l'Apprentissage de la Programmation en Première année à l'université. Pratiques et Enjeux Didactiques. Actes du Quatrième Colloque International. DIDAPRO 4-Dida&Stic (2011)
5. Cooper, S., Dann, W., Pausch, R.: Alice: a 3D Tool for Introductory Programming Concepts. Journal of Computing Sciences in Colleges, vol. 15, n° 15 (2000)107-116
6. Cooper, S., Dann W., Pausch, R.: Teaching Objects-First in Introductory Computer Science. ACM SIGCSE Bulletin, vol. 35, n° 11, (2003)191-195