Acquisition de connaissances de programmation en fonction des stratégies d'apprentissage : une étude empirique du micromonde PrOgO.

Fahima Djelil, Eric Sanchez, Benjamin Albouy-Kissi, Adélaïde Albouy-Kissi

Acquisition de connaissances de programmation en fonction des stratégies d'apprentissage : une étude empirique du micromonde PrOgO

Fahima Djelil[1], Eric Sanchez[2], Benjamin Albouy-Kissi[3], Adélaïde Albouy-Kissi[3]

[1] Université de Haute-Alsace, UHA 4.0, LMIA, 68200 Mulhouse, France
fahima.djelil@uha.fr

[2] Université de Fribourg, Laboratoire d'Innovation Pédagogique, 1700 Fribourg, Suisse
eric.sanchez@unifr.ch

[3] Université d'Auvergne, Laboratoire Institut Pascal, 43009 Le puy en Velay, France
benjamin.albouy@udamail.fr
adelaide.kissi@udamail.fr

Résumé. Cet article a pour objectif de présenter notre méthodologie employée pour l'étude des liens entre les comportements d'apprenants et la progression de leurs scores à une évaluation pré/post des effets de l'usage du micromonde PrOgO sur l'apprentissage des fondamentaux de la Programmation Orientée-Objet. Cette méthodologie relève de l'analyse de l'apprentissage par la collecte et l'analyse de traces d'interaction. Nous nous basons sur l'Analyse en Composante Principale pour le calcul des corrélations entre les actions d'apprenants dans l'interface de PrOgO et la progression de leurs scores observée lors d'une évaluation pré/post. Nous concluons sur les actions d'apprenants qui peuvent influencer l'acquisition des connaissances et en particulier l'importance de celles qui concernent l'édition de code.

Mots-clés. PrOgO, Micromonde de Programmation, Analyse de l'apprentissage, Traces d'interaction, Apprentissage de la programmation, Didactique de l'Informatique

Abstract. This article aims to present our methodology used to study the link between learners' behaviors and their score progression in a pre/post evaluation of the effects of the PrOgO microworld usage on learning Object-Oriented Programming basics. This methodology belongs to Learning Analytics by the collection and the analysis of interaction logs. We rely on the Principal Component Analysis to calculate correlations between the learners' actions in the PrOgO interface and their score progression observed in a pre/post evaluation. This leads to determine the learners' actions that may influence knowledge acquisition, particularly the relevant actions with regard to code editing.

Keywords. PrOgO, Programming microworld, Learning Analytics, Interaction logs, Computer Science Education Research

1 Introduction

Les micromondes de programmation sont des environnements restreints et interactifs, qui favorisent le développement et l'assimilation de connaissances abstraites par l'emploi de visualisations graphiques ou d'objets tangibles, voire l'intégration d'éléments ludiques et l'utilisation de métaphores [1]. Les travaux qui ont été conduits sur l'usage des micromondes portent principalement sur l'évaluation de leur potentiel en termes d'apprentissage, sans véritablement prendre en compte la manière dont ils sont utilisés [2], [3], [4].

Cet article a pour objectif de proposer une nouvelle méthodologie d'évaluation facilement reproductible. Elle diffère des pratiques les plus courantes en permettant d'aller au-delà d'un examen élémentaire de l'évaluation des scores des apprenants obtenus aux tests, et cela en caractérisant les stratégies des apprenants. Il s'agit d'examiner la nature des interactions des apprenants et leur rôle dans la progression des scores lors d'une évaluation pré/post de l'effet du micromonde PrOgO sur l'acquisition des connaissances de programmation. PrOgO est un nouveau micromonde que nous avons conçu dans le but d'aider les étudiants débutants à apprendre les fondamentaux de la Programmation Orientée-Objet (POO) dans un contexte de jeu. L'étude expérimentale menée avec PrOgO relève du recueil et de l'analyse de traces d'apprentissage (*Learning Analytics*) [5], et contribue aux méthodes d'évaluation des micromondes de programmation. Notre méthodologie repose sur l'Analyse en Composante Principale (ACP) au travers de traces d'interaction, pour l'identification des actions des apprenants déterminant leur progression lors d'une évaluation pré/post.

Par ailleurs, ce travail ne s'inscrit pas dans un paradigme méthodologique comparatiste. Notre objectif est de caractériser les interactions des apprenants avec l'interface de PrOgO afin de les discuter au regard de l'apprentissage, plutôt que de comparer cet environnement à d'autres ressources d'enseignement.

Cet article est organisé comme suit. La Section 2 décrit les fondements conceptuels de l'environnement PrOgO comprenant les choix didactiques retenus, influençant l'apprentissage. La section 3 décrit notre méthodologie d'analyse dans la recherche des actions déterminant la progression de scores obtenus lors d'une évaluation pré/post. Cela comprend des éléments de contexte et des détails sur la collecte et l'analyse de données, issues à la fois de tests de vérification de connaissances et de traces d'interaction. Les résultats obtenus sont également présentés et discutés. Enfin la Section 4 conclut sur notre méthodologie d'évaluation.

2 Fondements Conceptuels du Micromonde PrOgO

Les fondements conceptuels de PrOgO influencent l'apprentissage et les stratégies que mettent en œuvre les apprenants. Le premier concerne le *jeu-play*[1], un concept

[1] Formulation utilisée pour lever l'ambiguïté du terme « jeu » en langue française, qui désigne l'artefact ou la situation. Cette ambiguïté est inexistante en langue anglaise, le « jeu-play » est désigné par « play » et le « jeu-game » est désigné par « game ».

que nous reprenons de [6] pour désigner la situation qui résulte des interactions de l'apprenant avec le jeu en tant qu'artefact ou système (jeu-*game*). Ce sont ces interactions qui sont à l'origine du développement des connaissances. Notre point de vue est qu'un jeu ne résulte pas de la combinaison d'éléments qui auraient des propriétés ludiques intrinsèques (des points, des badges…) mais qu'il émerge d'une situation au sein de laquelle un apprenant, accepte de jouer avec un jeu-game et, ce faisant, à exercer sa créativité et son imagination (on parle alors d'attitude ludique). Nous veillons, néanmoins, à ce que certains attributs soient soigneusement intégrés dans la structure de l'artefact informatique. Nous prenons particulièrement en compte les modalités de représentation qui doivent être conçues de manière fidèle au modèle de connaissances considéré, ainsi qu'une certaine liberté offerte à l'apprenant quant aux actions qu'il peut effectuer. PrOgO est ainsi conçu de façon à permettre un jeu-play et jouer avec PrOgO consiste à imaginer, à créer des constructions 3D significatives et à les animer[2].

Un second concept sur lequel repose PrOgO est celui d'une *approche didactique par emboîtement hiérarchique* des concepts fondamentaux de la POO, qui découle directement de l'approche Objet-D'abord (*Object-First*). Cette dernière s'est répandue dans la communauté anglo-saxonne et se déroule en pratique sur trois phases successives [7] : 1) utilisation d'objets, 2) création de classes, 3) création de systèmes Orientés-Objets (OO). Nous voyons dans cette approche un emboîtement de trois catégories de concepts. A chaque étape l'apprenant est amené à interagir avec une catégorie de concepts, tout en lui masquant les détails des notions introduites à l'étape suivante. Cela permet de s'affranchir de la complexité inhérente aux concepts OO, qui sont fortement liés. PrOgO implémente cette approche didactique à travers deux modes. Un premier mode consiste en l'utilisation d'objets. Il vise à amener l'apprenant à développer des connaissances en lien avec le concept d'objet (caractéristiques d'objet, état d'objet). Un second mode se consacre à la création de classes, dans lequel l'apprenant accède à des notions en lien avec la classe (caractéristiques de classe, constructeur de classe, encapsulation, utilisation de classe), un concept volontairement occulté dans le premier mode.

Un troisième concept est celui de *système de représentation transitionnel*, formulation que nous proposons pour désigner le système de représentation propre à un micromonde de programmation. Cette formulation découle de l'expression *objets transitionnels* [8], [9], qui désigne les éléments constitutifs d'un micromonde de programmation. Ce sont des objets visuels ou tangibles, directement manipulables par l'apprenant. Ils sont dits transitionnels car ils possèdent des propriétés communes, à la fois avec des concepts de programmation formels et des objets plus concrets de l'expérience sensible. L'interaction de l'apprenant avec ces objets transitionnels, lui permet de développer des connaissances sur les concepts de programmation que ces objets permettent de représenter. PrOgO implémente un système de représentation transitionnel, au sein duquel des concepts fondamentaux de la POO sont représentés par des composants graphiques 3D visuels et interactifs.

L'interface de PrOgO comprend essentiellement un environnement virtuel restreint et un éditeur à auto-complétion qui sont en constante interaction. Les graphiques 3D qui se créent et se manipulent à l'interface sont des objets transitionnels, qui illustrent

[2] http://progo.iut-lepuy.fr

une métaphore de construction et d'animation, décrivant de manière fidèle et consistante les concepts de POO. Les composants graphiques dans la scène 3D ont des propriétés communes avec celles d'objets informatiques, et les constructions 3D réalisées sont des représentations visuelles de systèmes OO (Fig. 1). Les actions directes de l'apprenant sur les graphiques 3D sont traduites en instructions de code C++, et les résultats des actions d'auto-complétion sont instantanément visibles sur les graphiques 3D. L'apprenant construit ses connaissances en produisant du sens sur les concepts OO, suite à ses interactions avec l'interface de PrOgO lors du jeu-play. Enfin, PrOgO permet des études expérimentales de ses usages, en générant des traces numériques d'interaction.

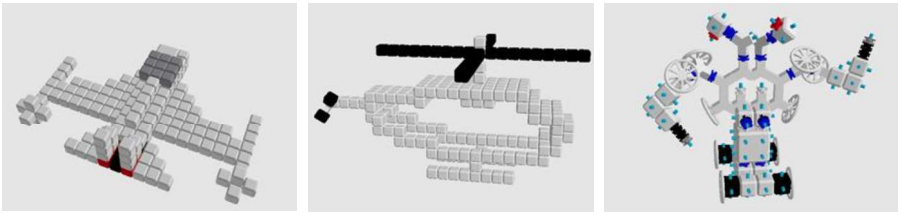


Fig. 1. Constructions 3D représentant des systèmes OO dans PrOgO.

Dans ce qui suit, nous décrivons notre expérimentation visant à déterminer les actions formulées par les apprenants dans l'interface de PrOgO, pouvant influencer l'acquisition des connaissances de programmation.

3 Analyse des Actions Déterminant la Progression des Scores d'une Évaluation Pré/Post de l'Acquisition des Connaissances

3.1. Objectifs et Questions de Recherche

L'objectif de cette expérimentation est double. Il s'agit d'une part, de savoir si un apprenant débutant peut acquérir de nouvelles connaissances à l'aide de PrOgO, et d'autre part, d'identifier les actions de l'apprenant réalisées dans l'interface de PrOgO qui déterminent le gain de connaissances et donc l'apprentissage.

Nous considérons essentiellement les deux questions suivantes : premièrement, les connaissances des apprenants débutants progressent-elles après utilisation de PrOgO ? Deuxièmement, si cette progression est observée, alors quelles sont les actions des apprenants qui sont corrélées avec cette progression. Autrement dit, quelles sont les interactions avec l'interface de PrOgO qui sont de nature à permettre des gains en termes d'apprentissage ?

3.2. Méthodologie

Pour répondre à la première question : les connaissances des étudiants débutants progressent-elles après utilisation de PrOgO ? Nous avons élaboré un pré-test et un

post-test sur la base de la *Taxonomie de Bloom révisée* des objectifs d'apprentissage [10]. En effet, les objectifs d'apprentissage sont préalablement définis et la taxonomie de Bloom a servi pour la formulation des questions d'évaluation de l'apprentissage. Nous avons également calculé des paramètres statistiques appliqués aux scores obtenus aux tests, afin de pouvoir les analyser de façon significative.

Pour répondre à la seconde question : quelles sont les actions des apprenants qui déterminent la progression de leurs connaissances lors de leur interaction avec l'interface de PrOgO ? Nous avons conduit une analyse factorielle de certaines variables de traces d'interaction. Cette partie de la méthodologie relève du domaine des *Learning Analytics* [5]. Afin d'arriver à identifier les actions des étudiants qui sont déterminantes dans la progression de leurs connaissances, on s'est appuyé sur l'ACP de type *pearson(n)*, appliquée à des variables d'analyse constituées à partir de traces d'interaction. Cette analyse a été réalisée sur le logiciel XLSTAT qui s'utilise avec Excel.

Contexte de l'Expérimentation. L'expérimentation a été réalisée le 30 Mai 2016, auprès de 51 étudiants de première année universitaire en DUT Métiers du Multimédia et de l'Internet (MMI) à l'IUT du Puy en Velay (Université d'Auvergne). Au cours de leur première année, les étudiants suivent un module d'Algorithmique et de programmation dans lequel ils sont initiés à l'algorithmique et à la programmation Web avec le langage PHP. Les étudiants n'ont pas de connaissances évoluées en POO et en langage C++.

L'expérimentation a duré 1h40mn et s'est consacrée à l'utilisation d'objets. Les participants ont répondu à un pré-test d'évaluation de connaissances pendant 20mn, puis pendant 1h, ils ont manipulé PrOgO en ayant à disposition un tutoriel d'utilisation. Après cela, les participants ont répondu à un post-test pendant 20mn. Les questions des tests d'évaluation étaient illustrées par des lignes de code de programmes en langages C++. Les participants étaient amenés à donner libre court à leur imagination pour construire et animer des robots virtuels ou des structures mécaniques 3D. Durant cette expérimentation, les traces d'interaction des participants avec l'interface de PrOgO ont été sauvegardées dans des fichiers externes (*.csv).

Collecte et Analyse de Données.

Élaboration du pré-test et du post-test et analyse des scores. Le pré-test et le post-test étaient identiques. Les objectifs d'apprentissage concernent les concepts suivants :

- *Le lien existant entre l'objet et la classe* : un objet est une instance de classe.
- *Les caractéristiques d'un objet* : un objet se caractérise par des attributs et des méthodes.
- *La modification de l'état d'un objet* : modifier la valeur d'un attribut d'objet ou réaliser un appel de méthode d'objet, provoquera la modification de l'état de cet objet.

PrOgO est conçu de façon à introduire l'ensemble de ces concepts au travers de son interface et d'amener l'étudiant à les implémenter en langage de programmation C++. Par conséquent, les tests doivent estimer en terme de scores, la capacité des étudiants à identifier ces différents concepts, les expliquer et les implémenter en langage C++.

Cela correspond aux trois premiers niveaux hiérarchiques de la taxonomie de Bloom révisée [10] : 1) *Reconnaître* : se réfère à la capacité d'identifier et de nommer des faits, des concepts ou des principes. 2) *Comprendre* : sous-entend qu'un apprenant est capable de construire du sens à partir de ce qu'il connaît et d'utiliser ses propres mots pour formuler cette connaissance. 3) *Appliquer* : suppose que l'apprenant connaît et comprend quelque un concept avant de l'utiliser. Cette taxonomie a servi à constituer le pré-test et le post-test (Tableau 1).

Tableau 1. Description des questions constituant le pré-test et le post-test.

Niveau de la taxonomie	Question
Niveau 1 : Reconnaître	- Lister des objets par leurs noms dans un programme édité dans PrOgO - Identifier pour chaque objet listé sa classe correspondante - Faire correspondre une syntaxe donnée à un changement de valeur d'attribut - Faire correspondre une syntaxe donnée à un appel de méthode - Faire correspondre une syntaxe donnée à un appel de méthode en identifiant les objets appelants - Identifier pour chaque objet les instructions qui modifient son état
Niveau 2 : Comprendre	- Conclure sur le lien existant entre l'objet et sa class - Conclure sur le rôle du nom d'objet - Conclure sur les rôles des attributs et méthodes d'objet - Conclure sur la modification de l'état d'objet - Interpréter à partir d'écritures générales les instructions d'instanciation de classes - Interpréter à partir d'écritures générales les instructions de changement de valeurs d'attributs - Interpréter à partir d'écritures générales les instructions d'appels de méthodes
Niveau 3 : Appliquer	- Implémenter une instanciation de classe - Implémenter une modification de valeur d'attribut - Implémenter un appel de méthode

Afin de comparer de manière significative les scores obtenus au pré-test et au post-test, trois paramètres statistiques sont calculés, à savoir un *indice de difficulté*, un *indice de discrimination* et un *indice de fiabilité*. L'indice de difficulté est la proportion d'étudiants répondant correctement au test [11]. Il est équivalent à la moyenne du test. Idéalement, il convient d'avoir une valeur comprise entre 0.3 et 0.8 [12]. L'indice de discrimination sert à estimer combien le test a permis de discriminer les étudiants ayant obtenu des scores élevés de ceux ayant obtenu des scores faibles [13], [11]. Idéalement, il convient d'avoir une valeur supérieure à 0.39 [13]. L'indice de fiabilité donne la probabilité d'obtention de résultats similaires quand le même test est utilisé auprès d'autres étudiants de profil similaire. Une telle mesure peut être donnée par le *coefficient alpha de Cronbach*. Il convient d'avoir une valeur comprise entre 0.6 et 0.8 [11].

Collecte et analyse de traces numériques d'interaction. Les traces d'interaction consistent en les actions séquentielles des participants dans l'interface de PrOgO. Cela comprend les actions liées à la création d'objets dans la scène 3D, la modification de valeurs d'attributs d'objets, les appels de méthodes sur les objets, et d'autres actions

comme l'exécution pas à pas de code et la sélection d'items dans la hiérarchie d'objets. Les traces incluent également les actions formulées dans l'éditeur à auto-complétion, comme la sélection et la complétion de code, et d'autres actions comme la modification de noms d'objets et la suppression de lignes de code dans l'éditeur.

Les traces brutes ont été traitées afin de constituer des variables d'analyse de l'ACP. Huit variables d'analyse ont été constituées, donnant, pour chaque participant, le nombre d'actions formulées dans l'interface de PrOgO et le temps total passé sur son interface durant l'expérimentation (Tableau 2). Les actions réalisées dans la scène 3D sont quantifiées et exprimées en nombre d'actions d'instanciation de classes et de connexion d'objets, nombre d'actions de modification de valeurs d'attributs et nombre d'appels de méthodes. Les actions réalisées dans l'éditeur de code sont exprimées en nombre d'actions de sélection de code, nombre d'actions de complétion de code et en nombre d'autres actions de code. Les actions d'exécution pas à pas de code et de sélection d'items dans la hiérarchie d'objets sont également quantifiées. A ces huit variables (dites variables actives), est ajoutée la différence des scores obtenue entre le post-test et le pré-test, comme variable quantitative supplémentaire (Tableau 2).

Tableau 2. Variables d'analyse de l'ACP

Variable	Signification
NB Instancier Connecter	Nombre d'actions d'instanciation de classes et de connexion d'objets dans la scène 3D
NB modifAttributs	Nombre d'actions de modification de valeurs des attributs *couleur* et *angleDeRotation* dans la scène 3D
NB appelsFonctions	Nombre d'appels des méthodes *colorierPendant()* et *tournerPendant()* dans la scène 3D
NB complétionCode	Nombre d'actions de complétion de code
NB autresCodeActions	Nombre d'autres actions réalisées dans l'éditeur de code (sélection de code, modification de noms d'objets et suppression de lignes de code)
NB executerCode pas à pas	Nombre d'actions d'exécution pas à pas de code
NB sélection itemsHiérarchiques	Nombre d'actions de sélection d'items dans la hiérarchie d'objets
temps total	Temps total passé sur l'interface de PrOgO durant l'expérimentation
progression	Variable quantitative supplémentaire indiquant la différence des scores entre le post-test et le pré-test

L'ensemble des participants désignés par leurs identifiants constituent les observations de l'ACP. L'objectif étant d'identifier et de visualiser les corrélations existantes entre la variable supplémentaire progression et les variables d'analyse actives. La corrélation reflète le degré de dépendance entre deux variables. Dans notre cas, elle est mesurée par le coefficient de Pearson ayant une valeur comprise entre -1 et +1. Plus le coefficient est proche de -1 ou de +1, plus la corrélation entre les variables est bonne. La corrélation peut être observée sur le *cercle des corrélations*, qui est un graphique montrant une projection des variables initiales dans un plan factoriel. Une variable est fortement liée à un axe, quand sa valeur absolue sur cet axe est élevée (proche de 1). Lorsque deux variables sont loin du centre du

graphique et qu'elles sont proches les unes par rapport aux autres, alors elles sont significativement positivement corrélées.

3.3. Résultats et discussion

Évaluation Pré/Post de l'Acquisition des Connaissances. Les résultats indiquent une amélioration des scores obtenus au post-test par rapport au pré-test. Pour chacun des trois niveaux *Reconnaître, Comprendre* et *Appliquer* et pour le test dans son ensemble, la moyenne des scores des participants a augmenté dans le post-test (Tableau 3).

L'indice de difficulté P (équivalent à la moyenne) pour chacun des trois niveaux et pour l'ensemble du test, possède une valeur rentrant dans la plage optimale (entre 0.3 et 0.8). Cela indique que la difficulté des tests est modérée. L'indice P a également augmenté dans le post-test. Ses valeurs respectives pour chacun des trois niveaux sont (0.40/0.48, 0.34/0.50 et 0.54/0.76). Ses valeurs pour les deux tests sont (0.40/0.54). Cela indique que le post-test a été perçu moins difficile que le pré-test, ce qui confirme une amélioration dans la compréhension des concepts visés chez les étudiants.

Tableau 3. Paramètres statistiques des scores obtenus au pré-test et au post-test.

	Pré-test				Post-test			
	A_1	A_2	A_3	A	A_1	A_2	A_3	A
Moyenne	0.40	0.34	0.54	0.40	0.48	0.50	0.76	0.54
Ecart type	0.25	0.16	0.36	0.17	0.27	0.22	0.34	0.20
Indice de difficulté P	0.40	0.34	0.54	0.40	0.48	0.50	0.75	0.54
Indice de discrimination D	0.62	0.38	0.94	0.42	0.68	0.54	0.77	0.42
Indice de fiabilité α	0.50				0.63			

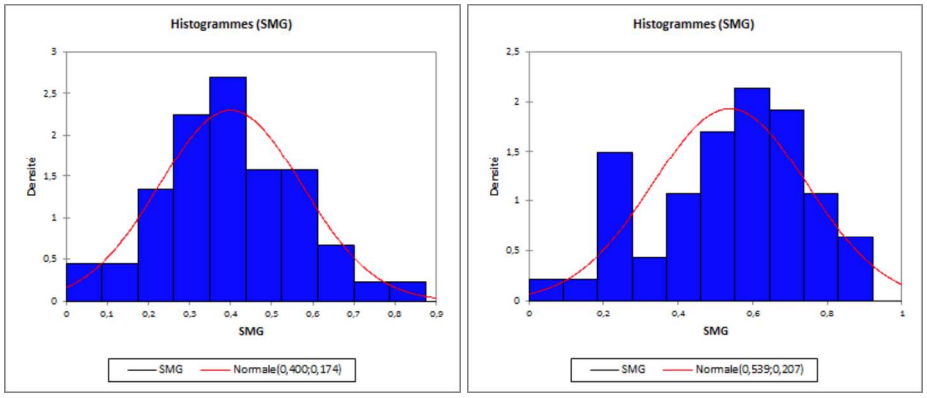
A_i est le score moyen global du niveau i.
i vaut 1 pour Reconnaître, 2 pour Comprendre ou 3 pour Appliquer.
A est le score moyen du test global.

Le pré-test et le post-test possèdent le même indice de discrimination D (0.42). Cela indique que les deux tests ont discriminé de la même façon les participants ayant obtenu des notes élevées de ceux ayant obtenu des notes faibles. L'indice D étant supérieur à 0.39, les deux tests ont très bien discriminé l'ensemble des participants. Au niveau Appliquer, l'indice D de chacun des deux tests, est très grand comparativement aux autres niveaux. Cela indique un grand écart de scores entre les participants ayant répondu correctement aux exercices d'implémentation de ceux n'ayant pas répondu correctement.

Enfin, l'indice de fiabilité α du pré-test possède une valeur inférieure à celle souhaitée (inférieur à 0.6) contrairement à celui du post-test (0.63). Étant donné que le pré-test est identique au post-test, l'indice α montre que le test devient fiable et consistant quand les connaissances des étudiants progressent.

On peut également constater la progression des scores des participants en observant les histogrammes du score moyen global du pré-test et du post-test (Fig. 2).

Les histogrammes indiquent que la fréquence du score moyen global supérieure à la moyenne est plus dense dans le post-test comparativement au pré-test, tandis que la fréquence du score moyen global inférieure à la moyenne est plus dense dans le pré-test comparativement au post-test. Ces résultats montrent que les étudiants ont bien progressé après utilisation de PrOgO.



(a) Score moyen global (SMG) du pré-test. (b) Score moyen global (SMG) du post-test.

Fig. 2. Histogrammes du score moyen global de l'ensemble des participants relatifs au pré-test et au post-test.

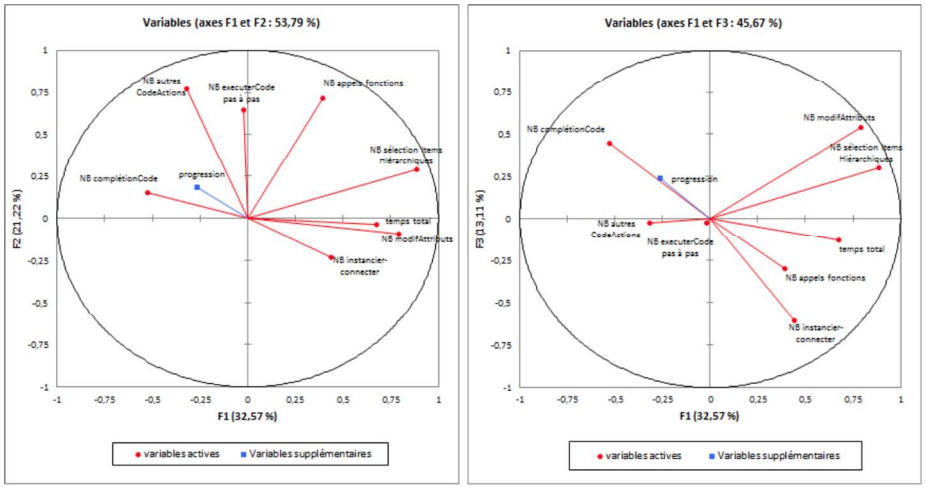
Analyse de traces d'interaction. L'ACP a retourné huit facteurs (composantes principales ou axes) calculés à partir des variables d'analyse issues des traces d'interaction (Tableau 2). Les valeurs propres des trois premiers facteurs ont permis de cumuler 66.89% de la variabilité initiale des données (Tableau 4). Nous avons retenu les trois premiers facteurs et ignoré les cinq autres. Il convient de retenir un nombre restreint de facteurs ayant cumulé un maximum de variabilité [14].

Tableau 4. Composantes principales retournées par l'ACP.

	F1	F2	F3	F4	F5	F6	F7	F8
Valeur propre	2.60	1.70	1.05	0.88	0.72	0.59	0.44	0.02
Variabilité (%)	32.57	21.22	13.11	10.59	9.02	7.40	5.54	0.20
% cumulé	32.57	53.79	66.89	77.84	86.86	94.26	99.78	100

Les cercles de corrélation (F1, F2) et (F1, F3) montrent que la variable progression n'est pas bien représentée sur les axes F1, F2 et F3 (Fig. 3). En effet, la variable progression se retrouve proche du centre de chaque cercle. Le Tableau 5 confirme que la variable progression est faiblement corrélée avec les axes F1, F2 et F3 (respectivement -0.26, 0.18 et 0.24).

Les résultats de l'ACP montrent que la variable *progression* est la mieux corrélée avec la variable *NB autresCodeActions* comparativement au reste des variables (Tableau 5). Cela indique que les étudiants ayant amélioré leurs scores dans le post-test, sont ceux qui ont formulé des actions dans l'éditeur de code : sélection de code, modification de noms d'objets et suppression de lignes de code (Tableau 2).



(a) Cercle des corrélations (F1, F2) (b) Cercle des corrélations (F1, F3)

Fig. 3. Cercles des corrélations montrant la corrélation de la variable progression avec les variables actives sur les axes F1 - F2 et F1 - F3.

Tableau 5. Corrélation de la variable progression avec les trois premiers facteurs et l'ensemble des variables actives.

Facteurs / Variables actives	progression
F1	-0.26
F2	0.18
F3	0.24
NB Instancier Connecter	-0.25
NB modifAttributs	-0.03
NB appelsFonctions	-0.06
NB complétionCode	0.24
NB autresCodeActions	0.37
NB executerCode pas à pas	0.00
NB sélection itemsHiérarchiques	-0.01
temps total	-0.26

Les trois premiers facteurs (F1, F2 et F3), sont représentatifs des actions décrivant et caractérisant le mieux l'ensemble des individus. L'absence de corrélation de la variable progression avec ces trois facteurs, signifie que les individus ont majoritairement réalisé des actions différentes de celles exprimées par la variable *NB autresCodeActions*, qui est la mieux corrélée avec la variable progression.

Il est difficile de conclure avec certitude sur les actions et les comportements des étudiants qui déterminent la différence des scores entre le post-test et le pré-test. Ceci est dû au lien faible observé entre la différence des scores et les actions les plus représentatives de l'ensemble des participants. Néanmoins, la différence des scores présente un lien particulier avec certaines actions réalisées dans l'éditeur de code. Ces actions concernent principalement l'édition de code, et comprennent la sélection de

code, la modification de noms d'objets et la suppression de lignes de code. Cela indique que les actions formulées dans l'éditeur de code seraient les plus à même d'influencer l'acquisition des connaissances.

4 Conclusion

Les résultats obtenus à l'issue de cette expérimentation ont montré que l'usage de PrOgO est en mesure d'aider les étudiants débutants à acquérir de nouvelles connaissances. Cependant, il est difficile de conclure sur le type d'actions et les comportements adoptés par les étudiants qui sont à l'origine de cet apprentissage. Bien que les techniques d'analyse employées offrent un moyen de savoir formellement quelles sont les actions formulées dans l'interface qui sont liées avec un changement de l'état des connaissances des participants, les résultats de notre expérimentation ne permettent pas d'associer de manière indiscutable tel ou tel comportement avec une progression des élèves en termes d'apprentissage.

Nous pensons, que le caractère mitigé de ces résultats n'est pas dû à la méthodologie elle-même, mais plutôt au contexte de l'expérimentation. En effet, l'expérimentation s'est déroulée à la fin de l'année universitaire à un moment où les étudiants étaient peu enclins à s'investir pour découvrir les fondamentaux de la POO. Ce manque de motivation a pu influencer leur comportement et limiter l'intensité des interactions avec l'interface de PrOgO. Il est peu probable d'avoir les mêmes résultats dans un contexte où les apprenants se trouvent très investis dans leur apprentissage. Le choix du contexte expérimental est donc primordial.

Par ailleurs, les fondements conceptuels du micromonde PrOgO comprenant le jeu-play (liberté offerte à l'apprenant de développer sa propre stratégie), l'emboitement hiérarchique des concepts (approche didactique retenue) et le système de représentation transitionnel (lien sémantique des objets 3D avec les concepts de programmation) sont transparents pour l'apprenant. Ils constituent nos choix didactiques et définissent notre approche pédagogique. Ils influencent probablement les stratégies d'apprentissage mises en œuvre par les apprenants, sans qu'il soit possible, à partir des travaux effectués, de trouver une relation de cause à effet.

Nous soulignons également, les limites de notre méthodologie de collecte de données, qui est entièrement fondée sur les traces d'interaction. Ces traces constituent un recueil partiel des variables signifiantes du point de vue de la caractérisation du comportement des apprenants. En particulier, avec ces traces, il n'est pas possible de savoir à quel moment, un apprenant dirige son attention sur un élément donné de l'interface. Cette limite dans la collecte de données pourrait être levée grâce aux techniques de suivi du regard pour l'analyse des interactions utilisateurs, *eye-tracking* [15]. Ces techniques peuvent apporter d'avantage d'informations relatives aux attitudes d'apprenants et améliorer notre méthodologie de recueil de données.

Enfin, notre contribution nous semble consister dans notre méthodologie d'évaluation qui est facilement reproductible. Elle répond en effet, au manque de méthodologies d'évaluation d'outils d'instrumentation de l'apprentissage de l'informatique et de la programmation en particulier [16]. Elle présente également, une avancée par rapport aux méthodologies jusque-là employées pour évaluer des

micromondes de programmation, dans la mesure où elle prend en compte l'analyse des comportements effectifs des apprenants plutôt que de s'en tenir au recueil de leurs points de vue ou à la progression de leurs scores mesurés à l'aide de questionnaires ou de tests. De plus, cette méthodologie nous paraît constituer une alternative aux approches comparatistes : plutôt que de comparer l'efficience d'un dispositif numérique nous procédons à la caractérisation des usages effectifs.

Références

1. Djelil, F., Albouy-Kissi, A., Albouy-Kissi, B., Sanchez, E., Lavest, J.-M.: Microworlds for learning Object-Oriented Programming : Considerations from research to practice. Journal of Interactive Learning Research, vol. 27, n° 13, (2016), 265-284
2. Xinogalos, S., Maya, S., Vassilios, D. : An introduction to Object-Oriented Programming with a didactic microworld: ObjectKarel. Computers & Education , vol. 47, n°12, (2006), 148-171
3. Cooper, S., Dann, W. , Pausch, R. : Teaching Objects-First in introductory computer science. ACM SIGCSE Bulletin, vol. 35, n°11, (2003), 191-195
4. Dann, W., Cosgrove, D., Slater, D., Culyba, D., Cooper, S.: Mediated transfer: Alice 3 to java. Proceedings of the 43rd ACM technical symposium on Computer Science Education, (2012)
5. Siemens, G., d Baker, R. S. : Learning analytics and educational data mining: towards communication and collaboration. Proceedings of the 2nd international conference on learning analytics and knowledge, (2012)
6. Sanchez, E., Emin-Martinez, V., Mandran, N.: Jeu-game, jeu-play, vers une modélisation du jeu. Une étude empirique à partir des traces numériques d'interaction du jeu Tamagocours. Sciences et Technologies de l'Information et de la Communication pour l'Education et la Formation (STICEF), vol. 22, (2015)
7. Bennedsen, J., Schulte, C. : What does Objects-First mean? : An international study of teachers' perceptions of Objects-First. Seventh Baltic Sea Conference on Computing Education Research, (2007)
8. Papert, S.: Mindstorms : Children, computers, and powerful ideas. Basic Books, (1980)
9. Lawler, R. W. : Artificial Intelligence and Education: Learning environments and tutoring systems. Learning Environments. Now, Then and Someday, vol. 1. Intellect Books (1987)
10. Anderson, L. w., Krathwohl, D. R : A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives. New York: Addison Wesley Longmann, (2001)
11. Yuan, W., Deng, C., Zhu, H., Li, J. : The statistical analysis and evaluation of examination results of materials research methods course. Creative Education, vol. 3, n° 17, (2013)
12. McDonald, M. E. : The nurse educator's guide to assessing learning outcomes. Jones & Bartlett Publishers, (2013)
13. Kelley, T. L. : The selection of upper and lower groups for the validation of test items. Journal of Educational Psychology, vol. 30, n° 11, (1939)
14. Jolliffe, I. : Principal component analysis. Springer, (2002)
15. Conati, C., Merten, C. : Eye-tracking for user modeling in exploratory learning environments : An empirical evaluation. Knowledge-Based Systems, vol. 20, n°16, (2007), 557-574
16. Djelil, F., Boisvert, C., Peter, Y., Secq, Y., Broisin, J., De La Higuera, C. : Vers une massification de l'apprentissage instrumenté de l'informatique, et une intégration des instruments et de leur évaluation. Les Grands Challenges des ORPHEE-RDV' 2017.