Sticef

Sciences et technologies de l'information et de la communication pour l'éducation et la formation

Volume 28, numéro 3, 2021

numéro spécial

Technologies pour l'apprentissage de l'informatique de la maternelle à l'Université

sous la direction de
Laetitia BOULC'H,
Julien BROISIN,
Yvan PETER,
Yann SECQ

ASSOCIATION DES TECHNOLOGIES DE L'INFORMATION POUR L'ÉDUCATION ET LA FORMATION

Sticef

Volume 28
numéro 3, 2021

numéro spécial

Technologies pour l'apprentissage de l'informatique de la maternelle à l'Université

ISBN 978-2-901384-05-2
DOI: 10.23709/sticef.28.3 en ligne sur www.sticef.org

Sommaire

Éditorial du numéro spécial « Technologies pour l'apprentissage de l'informatique de la maternelle à l'Université »

▶ **Laetitia BOULC'H** (EDA, Université Paris Cité), **Julien BROISIN** (Irit), **Yvan PETER**, **Yann SECQ** (Université de Lille, CNRS, Centrale Lille, UMR 9189 CRIStAL)

Ce numéro spécial « Technologies pour l'apprentissage de l'informatique de la maternelle à l'Université » est le résultat d'une belle aventure démarrée en 2017, qui a pu se développer grâce à la communauté EIAH.

En effet, fin janvier **2017**, les rendez-vous Orphée ont accueilli (http://www.atief.fr/content/orphee-rendez-vous-2017-0) plusieurs ateliers, dont celui intitulé « Apprentissage instrumenté de l'informatique » qui a regroupé une quinzaine de personnes d'horizons différents, mais partageant la même préoccupation au sujet de l'enseignement de l'informatique. Ce petit groupe a commencé à faire connaissance, partager leurs projets et identifier des points critiques pour le passage à l'échelle de l'enseignement de l'informatique, un peu avant sa réintroduction dans les programmes scolaires. Cette nouveauté dans les programmes allait fondamentalement poser la question de la formation des nombreux collègues qui seraient chargés de son enseignement dans les cycles initiaux du primaire jusqu'à l'Université. À l'issue de ces trois jours d'échanges, la question des suites éventuelles à donner s'est posée. Quelques-uns d'entre nous se sont mobilisés pour proposer un atelier lors de la conférence EIAH'17 à Strasbourg qui a constitué la première édition des ateliers « Apprentissage de la pensée informatique de la maternelle à l'Université » (Apimu, 2017) au sein de la communauté EIAH. Regroupant vingt-cinq participantes et participants, cette rencontre strasbourgeoise a donné lieu à de nouveaux et enrichissants échanges qui ont permis d'élargir le cercle initial formé initialement lors des rendez-vous Orphée.

Fort de cette dynamique grandissante, la proposition d'un deuxième atelier Apimu en **2018** (Apimu, 2018) lors des RJC-EIAH'18 s'est concrétisée par une nouvelle session qui a regroupé cette fois-ci une vingtaine de collègues.

L'année **2019** a permis un ancrage plus formel au sein de la communauté EIAH, grâce à la mise en place d'un groupe de travail GT-Apimu au sein de l'Atief (http://www.atief.fr/content/gt-apprentissage-informatique). L'objectif principal des personnes impliquées dans ce groupe est le développement et l'animation de cette communauté (re)naissante de collègues préoccupés par l'enseignement de l'informatique, soit de par leurs pratiques, soit de par leurs recherches. Cette dualité de préoccupations nourrit une riche dialectique dans la dynamique des échanges et débats entre collègues issus de la communauté EIAH et plus spécifiquement de l'informatique et des sciences de l'éducation. La troisième édition de l'atelier Apimu (Apimu, 2019) lors du colloque EIAH' 19 a regroupé cette fois-ci plus d'une trentaine de collègues, à un moment où nos préoccupations faisaient écho à la réintroduction de l'informatique dans les programmes scolaires en France ainsi que dans certains cantons romands suisses. Cet atelier a également permis d'évoquer le diplôme interuniversitaire (DIU) « Enseigner l'informatique au lycée », initié par des membres d'Apimu avec d'autres collègues et qui a été reconnu par le ministère de l'Éducation nationale pour la formation continue des enseignants de la nouvelle spécialité « Informatique et sciences du numérique ».

L'année **2020** aura été marquée par l'absence d'atelier Apimu afin de nous concentrer sur l'organisation d'un temps fort de la communauté en didactique de l'informatique francophone : la conférence du colloque Didapro 8 à Lille (https://www.didapro.org/8/). En effet, parallèlement aux ateliers Apimu au sein de la communauté EIAH, un autre courant cherchait à maintenir vivant l'intérêt pour l'enseignement et l'apprentissage de l'informatique, notamment au sein des colloques Didapro (colloques francophones de la didactique de l'informatique). La dernière édition ayant eu lieu en 1996 (juste après la disparition de l'option informatique des lycées français), le colloque Didapro de 2003 visait à « présenter des contributions de chercheurs investis dans l'enseignement des outils bureautiques, de formateurs concernés par les problèmes didactiques rencontrés et de professionnels de la publication ou de chercheurs en typographie numérique » (argumentaire du premier colloque Didapro, 2003). Les différentes éditions du colloque depuis cette date affirmaient en

substance, alors que l'école ne parlait plus du tout d'apprendre l'informatique, mais uniquement de « l'outil informatique », que même les outils logiciels ont besoin d'être appris et que ces apprentissages peuvent être étudiés par des chercheurs en éducation, en informatique et d'autres disciplines. Le retour, de plus en plus affirmé dans l'école française, d'enseignements scolaires de l'informatique, jusqu'à l'introduction récente de spécialités, options et plus encore de concours de recrutements, a progressivement conduit à une centration moins nette des colloques sur les usages des progiciels et leurs apprentissages afférents, pour traiter un plus large éventail de questions liées aux usages et technologies pour la didactique de l'informatique.

Le colloque Didapro 8 **2020** se situait donc au confluent de ces deux courants, l'usage de l'informatique en éducation et la didactique de l'informatique, et marquait de ce fait, une étape dans la reconstruction d'un champ scientifique étudiant l'ensemble des problématiques liées aux contenus, savoirs en jeu, activités intégrant l'informatique, de l'école maternelle jusqu'à l'université. Ce colloque a cristallisé les fortes dynamiques autour de ce renouveau de l'informatique scolaire avec pas moins de 250 participantes et participants et plus de 20 ateliers lors de la journée d'ouverture.

Ce numéro spécial est fortement induit par cette édition 2020 du colloque Didapro. Il rassemble une sélection de travaux de recherche, en informatique et sciences humaines et sociales, en lien avec une thématique centrale de la revue STICEF, celle de l'instrumentation des apprentissages. Mettant l'accent sur les principaux défis actuels et futurs de l'apprentissage et l'enseignement de l'informatique de la maternelle à l'université, il permet de réfléchir aux aides que les technologies peuvent apporter pour la formation initiale et continue des enseignants du primaire, du secondaire et du supérieur qui s'intéressent aux outils et approches pédagogiques innovantes susceptibles de soutenir les apprenants dans leur apprentissage de l'informatique.

Les deux premiers chapitres se focalisent sur l'enseignement de l'informatique à l'école maternelle. Le texte de Katell Bellegarde, Julie Boyaval et Julian Alvarez s'intéresse aux médiations cognitives à l'œuvre dans les dispositifs pédagogiques visant à initier des élèves de maternelle (5-6 ans) à la robotique et à l'informatique. Il permet de comprendre comment les instruments influent sur l'agir enseignant. Celui de Marielle Léonard, Yvan Peter, Yann Secq, Julian Alvarez et Cédric Fluckiger présente

un dispositif pédagogique destiné à initier des élèves de même âge (5-6 ans) à la notion de répétition en insistant sur la capacité d'identification de motifs. Ces auteurs montrent comment il est possible d'introduire cette notion de répétition dès 5 ans, à la condition de contourner les difficultés liées au repérage spatial. L'article permet aussi de comprendre comment les enseignantes et enseignants de maternelle formés à la pensée et aux concepts fondamentaux en informatique peuvent plus facilement s'approprier et faire évoluer les dispositifs pédagogiques.

Les chapitres suivants focalisent leur propos sur le niveau élémentaire et plus particulièrement le cycle 3 et la transition école-collège. Olivier Grugier montre de quelle façon des élèves de CM1 (9-10 ans), s'approprient les robots programmables et comment leur utilisation collective en classe les incite à s'interroger sur leur fonctionnement et leur utilisation. L'auteur montre notamment qu'en manipulant les robots, les élèves réussissent à construire des schèmes d'utilisation et plus généralement une culture technique. Dans la continuité, Matthieu Branthôme se questionne sur la manière de concevoir une ingénierie didactique permettant d'accompagner la transition collège-lycée dans l'apprentissage de la programmation. Sa réflexion s'appuie sur la mise en œuvre et l'analyse d'une séquence de résolution de défis à travers la programmation d'une carte Micro:bit proposées à des élèves de 3e hors-temps scolaire.

Les deux chapitres suivants concernent à la fois les niveaux primaire et secondaire. L'étude de Fanny Boraita, Anne-Sophie Collard et Julie Henry analyse les métaphores spontanées présentes dans les discours des enseignants et des apprenants de 3 à 15 ans lors d'activités d'initiation à la robotique. Les chercheuses font émerger les représentations des élèves, et apportent des éléments de réflexion sur les problématiques à aborder dans le cadre d'une éducation critique à la technologie. La recherche de Charline Carlot, Audrey Kumps et Bruno De Lièvre, quant à elle, propose d'intéressantes pistes de réflexion pour la formation initiale et continue des enseignants du primaire et des enseignants de mathématiques du secondaire. Ces auteurs décrivent de quelle manière les profils des futurs enseignants impactent leurs perceptions et leur intention pédagogique dans l'enseignement de la programmation.

Dans le supérieur cette fois, Pierre Bellet, Rémi Venant, Chrysta Pélissier, Stéphanie Mailles Viard Metz et Julien Broisin s'intéressent à la manière dont l'usage des technologies, et plus spécifiquement le recours à un laboratoire virtuel et distant pour l'apprentissage de l'informatique,

impactent le processus d'entraide entre étudiants. Les auteurs montrent l'influence « du calibrage et de la difficulté de la tâche sur la qualité des sessions d'entraide » et soulignent « l'apport intéressant du chat intégré au laboratoire et de l'outil de consultation du terminal d'un pair pour stimuler les interactions entre apprenants ». Enfin, Jean-Baptiste Raclet, Franck Silvestre et Mika Pons, se focalisant sur le contexte particulier de l'enseignement du génie logiciel, présentent un tableau de bord pour les enseignants (Git4School), « offrant des visualisations s'appuyant sur des données extraites des dépôts Git des apprenants, combinées à des informations contextuelles temporelles ». Les auteurs montrent en quoi cet outil permet de soutenir les interventions des enseignants en fonction de l'activité des étudiants : il les aide à prendre des décisions pendant un cours et à identifier les faiblesses des conceptions des situations.

Le numéro se conclut par une rubrique de Christophe Declercq où est présentée une liste riche, mais non exhaustive, de travaux en didactique de l'informatique pouvant être utilisés pour la formation des enseignants d'informatique au lycée.

Ce numéro spécial, qui fait suite au colloque Didapro 2020, constitue un point d'étape important pour cette communauté de chercheurs et praticiens intéressés par la question de l'enseignement et l'apprentissage de l'informatique. Les rencontres et les nombreux échanges se poursuivent, avec, en 2021, la quatrième édition d'un atelier Apimu (Apimu, 2021) accueilli au sein d'EIAH' 21 à Fribourg. L'année 2022 sera marquée par la neuvième édition du colloque Didapro au Mans (https://www.didapro.org/9/), avec quelques membres du GT-Apimu en première ligne sur l'organisation, comme lors de l'édition lilloise. Nous espérons qu'il permettra, lui aussi, de riches échanges, des débats et de belles rencontres.

Pour l'animation du GT Apimu : Julien Broisin, Christophe Declercq, Cédric Fluckiger, Yvan Peter, Yann Secq.

Pour la coordination du numéro spécial : Laetitia Boulc'h, Julien Broisin, Yvan Peter, Yann Secq.

RÉFÉRENCES

Apimu (2017, 6 juin). *Apprentissage de la pensée informatique de la maternelle à l'Université : recherches, pratiques et méthodes* [atelier]. Colloque EIAH, Strasbourg, France. https://wikis.univ-lille.fr/computational-teaching/wiki/actions/2017/aii-eiah/home

Apimu (2018, 6 avril). *Organisation et suivi des activités d'apprentissage de l'informatique : outils, modèles et expériences* [atelier]. Colloque RJC-EIAH'18, Besançon, France. https://wikis.univ-lille.fr/computational-teaching/wiki/actions/2018/rjceiah/home

Apimu (2019, 4 juin). *Apprentissage de la pensée informatique de la maternelle à l'Université : retours d'expériences et passage à l'échelle* [atelier]. Colloque EIAH' 19, Paris, France. https://wikis.univ-lille.fr/computational-teaching/wiki/actions/2019/eiah19/home

Apimu (2021, 7 juin) *Apprentissage de la pensée informatique de la maternelle à l'Université : retours d'expériences et passage à l'échelle* [atelier]. Colloque EIAH' 21, Fribourg, Suisse. https://wikis.univ-lille.fr/computational-teaching/wiki/actions/2021/eiah21/home

Initier des élèves de maternelle à la robotique/informatique : quand les supports médiateurs impactent la grammaire de l'agir enseignant

▶ **Katell BELLEGARDE** (Univ. Lille, EA 4354 – Cirel), **Julie BOYAVAL** (Circonscription de Montigny-en-Ghoelle, Académie de Lille), **Julian ALVAREZ** (DeVISU, Université Polytechnique Hauts-de-France – Cristal-Noce, Université de Lille, Ludoscience)

■ **RÉSUMÉ** • Cette contribution rend compte d'une étude comparative s'intéressant à des médiations cognitives à l'œuvre dans un dispositif pédagogique visant à initier des élèves de grande section de maternelle à la robotique/informatique. Une séance ludopédagogique utilisant le dispositif *Blue Bot* a été déclinée sur supports corporel, robotique et tablette numérique. Cette étude éclaire la manière dont les instruments influent sur l'agir enseignant interrogé à partir de l'identification, chez les praticiens, de leurs gestes, postures et conceptions professionnels.

■ **MOTS-CLÉS** • robotique pédagogique, médiation cognitive, analyse de l'activité, élèves de grande section de maternelle, grammaire de l'agir enseignant, gestes et postures professionnels.

■ ***ABSTRACT*** • *This paper presents a comparative study of cognitive mediations between pedagogical devices dedicated to introduced robotic/computer science to last section of kindergarten learners. The ludo-pedagogical session "Blue Bot" used for the experimentations was declined in three modalities: the body, the robot and digital tablets. This study highlights how mediator instruments influence acts of teaching through the identification of professional gestures, postures and conceptions.*

■ ***KEYWORDS*** • *educational robotics, cognitive mediation, activity analysis, klast section of kindergarten, teacher acts, professional gestures and postures.*

1. Introduction

Le projet de recherche *Blue Bot* s'est déroulé entre 2016 et 2019 et s'est attaché à étudier plusieurs aspects en lien avec les apprentissages du codage informatique auprès d'enfants de maternelle. Il a impliqué trente-cinq enseignants, deux cent trente élèves de 5 ans répartis dans vingt-huit écoles du Nord de la France.

La question centrale du projet consistait initialement à étudier l'influence de la robotique et du numérique sur l'apprentissage d'élèves de grande section de maternelle (GSM). Ce questionnement s'inscrivait pour nous, dans la continuité des travaux de recherche en lien avec la robotique pédagogique initiée par Papert (1981) dès les années 1970 et qui a suscité de l'engouement, mais également des interrogations (Cohen et Mialaret, 1987) sur leurs apports en termes de développements cognitifs chez l'enfant. En parallèle, il s'agissait d'étudier si la tablette numérique, dont les premières introductions en maternelle remontent au début des années 2010 (Mélot *et al.,* 2018), constituait une modalité facilitant la médiation cognitive dans l'acquisition de la programmation séquentielle chez de jeunes élèves.

Après échanges avec les enseignants et les inspecteurs académiques impliqués dans ce projet, il a été décidé d'étudier également les apports du corps qui constitue pour les enseignants de maternelle la modalité de référence pour prodiguer leurs enseignements en classe. C'est ainsi qu'a été mise en place une étude comparative basée sur des séances pédagogiques dans lesquelles le jeu est utilisé comme technique d'apprentissage (Alvarez, 2018). Déclinées selon trois modalités (Corps, Robot-jouet et Tablette), ces séances ludopédagogiques présentaient le même scénario d'apprentissage visant à initier à la pensée informatique des enfants de 5 ans inscrits en GSM.

Du point de vue de la recherche, nous avons étudié l'influence des trois modalités en jeu sur l'apprentissage des élèves à partir de l'identification de leurs stratégies, performances et conceptions (Alvarez *et al.,* 2021 ; Bellegarde *et al.,* 2019). Cependant, une autre dimension, à laquelle nous ne pouvions nous soustraire, est apparue nécessaire à interroger : la grammaire de l'agir enseignant (Bucheton et Soulé, 2009) dans le cadre d'une initiation à la programmation. Concrètement, quels gestes et postures professionnels les enseignants ont-ils développés dans le cadre d'une initiation à la programmation ? En parallèle, dans quelles mesures les modalités utilisées ont-elles influé leurs pratiques d'enseignement ? Ce présent article explore ainsi l'influence des différentes modalités sur l'agir enseignant interrogé à

partir de l'identification, chez les praticiens ayant participé à l'expérimentation *Blue Bot,* de leurs gestes, postures et conceptions professionnels.

Après avoir mené une réflexion théorique sur l'enseignement de l'informatique en maternelle, nous présenterons tout d'abord la méthodologie déployée pour procéder à l'analyse de l'activité enseignante dans le cadre du projet *Blue Bot,* puis nos analyses, qui nous conduiront à identifier des gestes et postures professionnels particuliers en fonction des trois modalités en jeu.

2. Enseigner la programmation à des élèves de cycle 1

Cette première partie rend compte des investigations théoriques menées en vue d'approcher la question de l'enseignement de la programmation à des élèves de GSM. Nous examinerons les trois considérations scientifiques suivantes.

– Programmer en GSM questionne la place et le statut à donner aux sciences informatiques à l'école : s'agit-il d'un objet d'enseignement à part entière ou seulement d'un outil d'enseignement ?

– L'usage de la robotique pédagogique permet-elle aux jeunes élèves de commencer à appréhender des concepts de programmation ?

– Initier de jeunes élèves à la robotique pédagogique interroge le rôle de l'agent médiateur humain, ses gestes et postures professionnels.

2.1. Les sciences de l'informatique à l'école maternelle : un objet d'enseignement au service d'autres apprentissages

À l'école maternelle, il n'existe pas de compétences à proprement parler qui visent la programmation et le codage informatique. En effet, la seule référence du programme de l'école maternelle en France, en lien avec la programmation et le codage informatique, concerne l'identification du principe d'un algorithme et la poursuite de son application (BO spécial n° 2 du 26 mars 2015). Mais, il s'agit ici de compléter une suite à partir de critères préalablement identifiés. Cette simple référence à l'algorithme peut néanmoins justifier l'initiation à la robotique/informatique en classe de GSM. Dans cette perspective, initier des élèves à la programmation apparaît cohérent avec la réintroduction, en 2016, de la science informatique dans les programmes de L'Éducation nationale à l'école élémentaire et au collège et avec le nouveau socle commun de compétences. Et pourtant, si

l'on considère l'école maternelle comme la base sur laquelle va se construire l'ensemble des apprentissages de la vie de l'élève, on sent bien la nécessité d'amorcer, dès le cycle 1, certains savoirs, qui se concrétiseront par un apprentissage « systématisé » quelques années plus tard, notamment en cycles 3 et 4. En effet, la continuité des apprentissages et la fluidité des parcours sont deux aspects importants dans la scolarité des élèves, de l'école primaire au supérieur. Les besoins de compétences numériques des élèves ont d'ailleurs été réaffirmés avec l'introduction de nouveaux enseignements au lycée à la rentrée 2019 et par la mise en place d'un cadre de référence des compétences numériques (CRCN). Ce référentiel, outil de positionnement et de certification, est paru au Journal officiel le 30 août 2019 (décret n° 2019-919). Il donne lieu à une certification des compétences numériques (https://pix.fr/), dont la programmation informatique en fin de cycle 4 et au cycle terminal du lycée. Ainsi, une initiation précoce aux concepts de robotique et de programmation s'inscrit, selon nous, dans une démarche cohérente au regard des compétences numériques à acquérir dans les cycles supérieurs.

Un débat persiste cependant autour de l'apprentissage de l'informatique à l'école : s'agit-il d'un outil d'enseignement au service d'autres disciplines ou d'un objet d'enseignement à part entière ? S'agit-il d'apprendre à programmer ou de programmer pour apprendre (Baron et Drot-Delange, 2016 ; Béziat, 2012) ? Nous adopterons, dans cet article, un positionnement intermédiaire qui envisage la programmation comme un objet d'enseignement, qui favoriserait également l'acquisition de certains apprentissages premiers, langagiers et culturels, qui dépassent les notions algorithmiques[1] : la communication orale, la construction du nombre, la structuration de l'espace et du temps, la résolution de problèmes, la collaboration ou encore l'abstraction (Greff, 2004).

2.2. La robotique pédagogique : vers une initiation précoce à la pensée informatique

La robotique pédagogique, initiée par Papert (1981) au début des années 1970 avec la tortue de plancher Turtle, associée au langage de programmation Logo, s'inscrit dans le modèle constructiviste des apprentissages (Komis et Misirli, 2011). En tant qu'« *objets pour penser avec* »

[1] Les travaux de Roméro (Romero, 2016) confirment, par ailleurs, le caractère interdisciplinaire des sciences informatiques à l'école. Cinq compétences clés pour le XXIe siècle y sont identifiées : la pensée critique, la collaboration, la résolution de problèmes, la créativité, la pensée informatique.

(Papert, 1981, p. 23), les robots permettent une manipulation et une expérimentation à partir de situations réelles, dans un contexte de résolutions de problème et de développement de la pensée algorithmique : « *L'élève est l'artisan de sa propre formation en se posant lui-même ses problèmes. [...] Un système comme LOGO est un des moyens de donner à l'élève un comportement actif dans un processus d'acquisition de concepts ou de prise de conscience d'un mécanisme.* » (Vivet, cité par Denis, 2000, p. 196). Considérés comme « *les bâtisseurs* [de leurs savoirs,] de leurs propres structures intellectuelles » (Papert, 1981, p. 17), les enfants sont également « *épistémologues* » (Papert, 1981, p. 31) dans le sens où ils seraient amenés à entrer dans une étude critique de leur propre réflexion. La robotique pédagogique contribuerait ainsi à débarrasser la notion d'erreur du sentiment de sanction intellectuelle : la recherche du bug du programme, son analyse, sa compréhension et sa correction font partie intégrante de l'activité de programmation et constituent une étape du processus d'apprentissage.

Initier de jeunes élèves à la programmation via la robotique pédagogique constitue alors une mise en pratique intéressante de la pensée informatique (*computational thinking*) (Wing, 2008) ; la démarche intellectuelle pouvant faire l'objet d'une application concrète, d'une confrontation au réel. À l'instar de Wing (2008), nous pensons que « *la pensée informatique est un ensemble d'attitudes et d'acquis universellement applicables* [la logique, l'algorithmique, la décomposition, la recherche de similitudes entre les sous-problèmes, l'abstraction et l'évaluation] *que tous, et pas seulement les informaticiens, devraient apprendre et maîtriser* » (p. 1). Dans cette perspective, Romero (2018) invite à dépasser l'enseignement du codage (au sens de coder avec un langage informatique) pour s'inscrire dans une démarche plus large qui engage les apprenants dans un processus critique et créatif de la résolution de problème. Cette programmation créative fait ainsi appel aux concepts et processus informatiques, à la pensée informatique : « *Il ne s'agit pas de coder pour coder, ou d'écrire des lignes de code les unes après les autres, mais de développer une approche de résolution de problèmes complexes qui engage dans une analyse réflexive et empathique de la situation, de sa représentation et de l'opérationnalisation d'une solution qui profite des stratégies métacognitives liées à la pensée informatique* » (p. 72).

Le robot programmable *Blue Bot* que nous avons choisi dans le cadre des expérimentations que nous présentons ici autorise une appréhension précoce des concepts de robotique et de programmation et plus largement, une entrée dans la pensée informatique. En effet, véritable outil de médiation, le jouet programmable, de par ses aspects ludiques et tangibles, favorise la motivation des élèves et leur implication dans des activités porteuses de significations (Komis et Misirli, 2013) tout en leur permettant de s'y identifier par « effet miroir ». Linard (1996) nous met cependant en garde contre le mythe de l'autogenèse cognitive qui consisterait à négliger le rôle de la médiation humaine vis-à-vis des médiations techniques. Or, le soutien humain sera toujours nécessaire pour relayer l'information médiatisée et ainsi aider les élèves à faire, à penser, à comprendre, à réfléchir sur leurs actions et finalement, les aider à apprendre (Leroux, 2002 ; Vivet, 2000).

2.3. La médiation humaine au service de la robotique pédagogique : la grammaire de l'agir enseignant en question

S'approprier un savoir suppose toujours un processus d'objectivation qui n'est possible que par l'action d'un système médiateur qui va jouer le rôle d'intermédiaire entre le sujet et le savoir. Composé d'aspects très hétérogènes, ce système médiateur comprend des personnes ayant des statuts différents (médiateur ou apprenant) et des artefacts culturels (Weil-Barais et Resta-Schweitzer, 2008). La fonction de médiation de l'enseignant renvoie à celle d'étayage, à la manière dont un adulte, plus expert, organise le monde pour l'enfant dans l'optique d'assurer la réussite de ses apprentissages (Bruner, 1983).

La recherche que nous présentons ici implique l'utilisation d'artefacts de type robotique, tablette numérique et corporel qui suppose une activité de didactisation exercée par l'enseignant à travers ces artefacts. Dans cette perspective, nous envisageons la médiation comme un processus résultant soit de l'action directe d'une personne, soit de son action indirecte exercée par le biais des instruments. À l'instar de Rézeau (2002), nous proposons de penser le système médiateur dans le projet de recherche *Blue Bot* (figure 1) sous la forme d'un carré pédagogique qui met en évidence le rôle d'agents-médiateurs joué à la fois par l'instrument et l'enseignant.

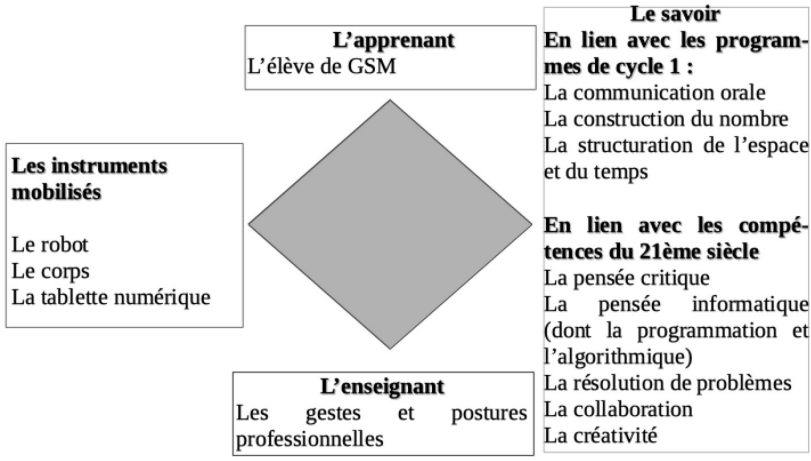


Figure 1 · Le système médiateur dans le projet *Blue Bot*

La réflexion que nous souhaitons mener dans cet article focalise son attention sur trois pôles du carré, l'instrument, l'enseignant et le savoir. Il s'agit ainsi de comprendre l'influence de la tablette numérique et de la robotique sur l'agir enseignant, agir questionné dans le cadre d'une initiation à la programmation. Le modèle théorique du multiagenda a constitué un cadre d'analyse de ces situations didactiques, appréhendées dans leur dimension située et à travers l'identification des différentes configurations de gestes professionnels enseignants et de leurs effets sur les dynamiques cognitives et relationnelles dans la classe. Dans une perspective heuristique, il s'agit alors « *de dérouler la pelote de ce qui se joue dans la classe pour pouvoir mieux rendre compte de la complexité et l'épaisseur de l'agir du maître* » (Bucheton et Soulé, 2009, p. 32).

Le modèle du multiagenda dessine la matrice de l'activité de l'enseignant à travers la conjugaison de cinq préoccupations majeures :

– piloter et organiser le déroulé de la leçon (« le pilotage ») ;

– conserver un espace de travail et de collaboration langagière et cognitive (« l'atmosphère ») ;

– tisser le sens de ce qui se déroule (« le tissage ») ;

– étayer le travail en cours de réalisation (« l'étayage ») ;

– avec pour objectif un apprentissage (« les objets de savoir et les techniques »).

Présentes de la maternelle à l'université, ces cinq macro-préoccupations constituent les invariants de l'activité enseignante, l'essence des gestes

professionnels. Bucheton et Soulé appréhendent la grammaire de l'agir enseignant comme « *l'organisation modulaire, systémique et dynamique des gestes professionnels (le jeu des postures d'étayage des maîtres)* » (2009, p. 45). À leur instar, nous appréhendons l'agir enseignant sous le prisme d'un ajustement permanent des gestes et postures en fonction des situations. Cet ajustement est un principe fondateur du geste professionnel.

L'approche comparative développée dans ce travail de recherche nous conduit à observer, analyser et comprendre les changements de gestes et postures en fonction des instruments en jeu (robot, corps, tablette).

3. Présentation de l'enquête de terrain

Au démarrage de cette recherche, nous avons fait l'hypothèse que la robotique, par sa dimension tangible et extracorporelle et les possibilités de décentration et de manipulation de l'objet-robot qu'elle procure à l'élève (Béziat, 2012 ; Komis et Misirli, 2011), conduirait particulièrement à une mise en retrait[2] de l'enseignant au profit d'apprentissages autodirigés chez les élèves. Nous étudierons l'agir enseignant sous le prisme des gestes, postures et conceptions professionnels construits à travers la conduite de séances d'initiation à la programmation.

Les informations exposées ci-dessous entendent procurer au lecteur les éléments de repérage sur la manière dont a été mise en œuvre l'enquête de terrain.

3.1. Dispositif expérimental mis en œuvre

Les expérimentations *Blue Bot* ont été menées durant l'année 2017 et ont concerné trente-cinq classes de GSM et vingt-huit écoles du Nord et du Pas-de-Calais. En amont des expérimentations, une série de séminaires associant enseignants, inspecteurs académiques de l'académie de Lille, représentant de la délégation académique du Numérique pour l'École (Dane), représentants de l'Institut national du professorat et de l'Éducation (INSPE) et enseignants-chercheurs a été organisée durant le dernier semestre 2016. C'est lors de ces échanges qu'il est apparu que la majorité des enseignants n'avaient jamais fait l'usage de robots en classe. Pour répondre à cette situation, une séance d'initiation a été organisée pour permettre à l'ensemble des enseignants de se familiariser avec le robot *Blue bot*, la barre

[2] Cette mise en retrait de l'enseignant renvoie à la posture de lâcher prise identifiée par Bucheton et Soulé. Dans cette situation, « *l'enseignant assigne aux élèves la responsabilité de leur travail et l'autorisation à expérimenter les chemins qu'ils choisissent* » (Bucheton et Soulé, 2009, p. 40).

de programmation associée d'une part, et l'application *Blue Bot app* fonctionnant sur tablette numérique, d'autre part.

Élaboré selon une progression en trois temps, les scénarios expérimentés en classe ont été pensés de manière à être transposables aux trois supports (robot, tablette, corps). Le tableau ci-dessous est une synthèse des différents temps de l'expérimentation[3].

Tableau 1 · Vue générale du scénario pédagogique

1er temps : Introduction à l'algorithmique et aux instructions de programmation		
Séance 1 : déplacement sur plateau vierge à partir d'instructions verbales (étape 1) + traduction de ces instructions verbales en codage (étape 2)		
2e temps : Créations de séquences de codage		
Séance 2 : Parcours proposés uniquement avec la commande « avancer »	Séance 3 : Parcours proposés avec l'introduction de la commande « tourner à droite »	Séance 4 : Parcours proposés avec l'introduction de la commande « tourner à gauche ».
3e temps : Création de séquences de codage avec contraintes supplémentaires		
Séance 5 : Parcours proposés avec des cases par lesquelles passer	Séance 6 : Parcours proposés avec des obstacles fixes	Séance 7 : Parcours proposés avec des cases par lesquelles passer et des obstacles fixes

Dans le cadre de l'expérimentation *Blue Bot*, la même séance ludopédagogique convoquant la thématique du robot dans un objectif d'initiation à la robotique/informatique, et au codage en particulier, a été mise en œuvre au sein des classes selon les trois modalités suivantes :

– utilisation du corps : un enfant incarne le robot et doit se déplacer sur un damier reproduit au sol. Un autre enfant lui dicte les instructions à effectuer (figure 2 – gauche) ;

[3] L'intégralité du volet pédagogique du projet Blue Bot se situe à l'adresse suivante : http://ja.games.free.fr/BlueBot/PROJETDERECHERCHEBLUEBOTV2b-3.docx.

– utilisation du robot : les enfants programment le robot jouet *Blue Bot* qui se déplace sur un damier imprimé sur un tapis en plastique posé sur une table (figure 2 – milieu)[4] ;

– utilisation d'une tablette numérique : le jeu est reproduit à l'identique dans un environnement entièrement virtuel qui se joue sur tablette (figure 2 – droite).

Figure 2 · Modalités Corps, Robot, Tablette

Les situations-problèmes proposées aux élèves sont construites autour de l'histoire de Vibot – le robot (Romero, 2016), personnage que l'enfant doit programmer pour le conduire en différents lieux. Les activités de programmation se réalisent sur un damier de 24 cases (6 x 4) avec une barre et des cartes de programmation. L'élève conserve alors la trace du programme et un travail sur l'erreur est possible. Les élèves réalisent les activités par groupe de quatre, maximum, sur un temps d'environ trente minutes.

Afin de mesurer l'influence des instruments (robot, tablette, corps) sur l'agir enseignant, l'expérimentation a été réalisée de façon à ce que les enseignants mettent en œuvre le scénario pédagogique en expérimentant les supports dans des ordres différents[5].

[4] Une documentation technique du robot Blue Bot est consultable à l'adresse : https://www.generationrobots.com/media/tts/Blue-Bot-Manual.pdf.

[5] Nous avons conscience que l'ordre d'expérimentation des supports par les élèves peut impacter leurs apprentissages. Mais, l'approche comparative développée dans le cadre de cette recherche nous a conduits à proposer aux classes des ordres différents. Toutefois, dans le cadre de l'atelier "Apprendre la Pensée Informatique de la Maternelle à l'Université" (APIMU), nous avons pu discuter cette question (Alvarez *et al.*, 2021).

3.2. Modalités d'élaboration du corpus : l'analyse de l'activité enseignante au cœur de l'investigation

Cette étude s'inscrit dans une approche compréhensive de la grammaire de l'agir enseignant dans le cadre de séances d'initiation à la programmation. L'analyse de l'activité enseignante a constitué le cœur de la démarche d'investigation mise en œuvre. Trois modalités d'élaboration du corpus ont été retenues.

D'abord, des captations vidéo des situations éducatives ont été réalisées de manière à déconstruire-reconstruire la dynamique de la situation d'enseignement et à identifier les invariants significatifs de l'agir enseignant. Sept enseignants ont été filmés à trois temps particuliers du scénario pédagogique (séance 1 – étape 2, séance 4, séance 7 (voir tableau 1) et, ceci pour chaque modalité mise en œuvre, soit un total de 63 séances filmées.

Ensuite, des documents de suivi ont été mis à la disposition des enseignants dans l'optique de saisir leur regard porté sur leur activité et celle des élèves. Dix-sept évaluations intermédiaires et quatorze carnets de bords ont pu faire l'objet d'une exploitation[6].

Enfin, dans cette même visée, des entretiens semi-directifs ont été réalisés avec les sept enseignants filmés après l'expérimentation de chacun des supports dans leur classe (soit un total de 21 entretiens).

Notre échantillon d'enquête a été constitué de manière aléatoire sur la base du volontariat des enseignants. Aucun critère de sélection, lié à la culture numérique des enseignants, par exemple, n'a pu être retenu. Par ailleurs, nous ne possédons pas de données descriptives liées à la totalité de notre échantillon. Seuls les sept enseignants, dont nous avons filmés les séances d'initiation à la programmation et avec lesquels nous avons réalisé des entretiens, ont fait l'objet de questionnements liés à leur profil.

Ainsi, cet échantillon de sept enseignants est composé de six femmes et un homme. Tous enseignent à l'école maternelle depuis au moins dix années et trois enseignants depuis plus de vingt ans. Trois sont directeurs d'école, un possède le certificat d'aptitudes aux fonctions d'instituteur ou de professeur des école maître formateur (CAFIPEMF) et deux sont inscrits

[6] L'intégralité du volet scientifique du projet Blue Bot comprenant les outils de suivi mis à disposition des enseignants (évaluations intermédiaires et carnet de bord) se situe à l'adresse suivante : http://ja.games.free.fr/BlueBot/PROJETDERECHERCHEBLUEBOTV2b-3.docx.

en master à l'INSPE dans l'optique d'obtenir cette certification. Deux d'entre eux déclarent avoir pratiqué ou pratiquer la programmation : le premier a programmé dans les années 1980 dans le cadre de ses études (DEUG Maths, Informatique et Sciences sociales – MIS) et le second, autodidacte et « *féru d'informatique* », programme en lien avec la gestion de sites internet. Leurs pratiques de classe montrent un usage quotidien du numérique : six utilisent les outils numériques tous les jours et un, une fois par semaine. Ils possèdent au moins un ordinateur dans leur classe ; les jeux éducatifs et le traitement de texte constituent les deux activités mentionnées. Cinq possèdent par ailleurs un tableau blanc interactif (TBI). Aucun ne déclare de difficultés particulières en lien avec l'usage des outils numériques. Au moment de l'expérimentation *Blue Bot,* ils n'ont jamais enseigné la programmation et fait usage de la robotique en classe.

4. Enseigner la programmation à des élèves de GSM : des gestes, postures et conception professionnels en « (trans)formation »

L'analyse de l'activité enseignante met en évidence la grammaire de l'agir des professionnels confrontés à l'enseignement de la programmation en classe de GSM, à travers des gestes, postures et conceptions professionnels qui se (trans)forment au fil de l'expérimentation. Elle souligne également l'influence des instruments sur ces configurations enseignantes.

4.1. Des gestes professionnels de soutien aux activités de programmation des élèves

À l'instar de Bucheton (2014), nous envisageons le geste professionnel comme un ensemble d'actions, de mouvements et d'opérations mentales, articulés et coordonnés, visant la réalisation d'une tâche d'enseignement. Celui-ci requiert la mobilisation de compétences professionnelles.

Six gestes professionnels majeurs ont été observés chez les enseignants dans le cadre de notre expérimentation. Ceux-ci sont orientés vers le soutien des activités de programmation des élèves.

4.1.1. Le rappel

Le rappel est le premier geste professionnel observé chez les enseignants à travers la verbalisation des consignes, règles et principes propres à l'activité de programmation (par exemple, l'usage de la touche « X » pour effacer le programme ou encore la rotation sur place exercée par le robot).

Ce geste professionnel apparaît également dans l'agir enseignant au travers du rappel de ce qui a été fait lors des séances précédentes.

À travers le rappel, il s'agit pour l'enseignant de « dire » au sens de rendre explicite pour les élèves ce qu'ils doivent faire et comment. Il s'agit également de rendre disponibles, pour les élèves, les connaissances antérieures qui devront être mobilisées dans l'activité. On se situe, ici, dans une action de « tissage » (Bucheton et Soulé, 2009) ; en début de séance, l'enseignant met en relation la tâche en cours avec celles qui précèdent.

« *J'ai eu des actions de rappel, je commence toujours par un rappel de ce qui a été fait les fois précédentes. [...] J'étais vraiment là pour donner des consignes.* » (Entretien enseignante M, Robot_1[7])

4.1.2.La reformulation

Le geste de reformulation observé chez les enseignants prend trois formes :

– celle d'une reformulation des consignes, des règles et principes de la programmation ;

– celle d'un modèle verbal suscité par l'enseignant ;

– celle de la reprise des propos d'un élève pour les mettre en relief et ceci, notamment pour faire avancer le raisonnement de l'ensemble du groupe.

Dans le second cas, il peut s'agir d'insister sur les connecteurs de temps pour travailler la chronologie des actions en utilisant un vocabulaire précis.

Ce qui est alors visé par l'enseignant à travers ce second geste professionnel, c'est de mettre l'accent sur des éléments de la tâche jugés importants pour comprendre, réfléchir, réussir et surtout apprendre. L'enseignant se situe ici dans une action de « pilotage ».

« *Reformulation du parcours en insistant sur les connecteurs de temps : d'abord, après, enfin.* » (Évaluation intermédiaire enseignant H, Robot_3)

[7] Le nombre indiqué à côté du support correspond au moment où a été utilisé le support. Par exemple, dans le cas de l'enseignant M, le support robot a constitué le premier support mis à sa disposition.

4.1.3. Le questionnement

Le questionnement est un geste professionnel tout particulièrement convoqué par les enseignants. Quatre visées principales de ce geste ont été observées :

– solliciter chez les élèves la verbalisation de leurs stratégies, les amener à communiquer avec leurs camarades sur leurs manières de faire et de penser, notamment pour des élèves qui auraient tendance à réaliser l'activité sans parler ;

– les inciter à convoquer des savoirs scolaires, par exemple, utiliser les notions de « droite » et de « gauche » pour donner les instructions quand il est plus facile d'indiquer la direction avec la main ;

– les inviter à utiliser un procédé particulier, notamment en leur demandant le chemin qu'ils ont choisi pour les encourager à le montrer avec le doigt avant de coder le programme.

Par ce geste professionnel du questionnement, l'enseignant va chercher à ce que les élèves rendent explicite ce qu'ils font, leur raisonnement, mais également à ce qu'ils utilisent des savoirs et savoir-faire scolaires et enfin à ce qu'ils réfléchissent à ce qu'ils font pour accompagner la réussite de l'activité. L'enseignant met ici en œuvre une action d'« étayage » orientée vers « les objets de savoir ».

« *Le maître doit encourager les élèves à parler, les questionner pour les amener à expliquer ce qu'ils font et comment ils font. Il faut aussi veiller à interroger les élèves qui font sans parler.* » (Carnet de bord enseignant H-B, Tablette_1)

« *Je leur demande de changer le message de celui qui guide sans changer les pict*ogrammes] *pour les amener à utiliser le nombre.* » (Évaluation Intermédiaire enseignant C, Corps_2)

4.1.4. L'aide dans l'activité

Bienveillants et soucieux de la réussite de leurs élèves, les enseignants ont également développé un certain nombre de gestes afin de les aider dans la réalisation de l'activité :

– la simplification de la tâche (retrait d'une contrainte, d'un déplacement, etc. ou ajout d'un outil supplémentaire) dans l'optique de permettre sa réussite ;

– la prise en charge d'une partie de la tâche (« *le faire avec* » ; « *je commence et tu finis* ») ; ou

– sa totalité (« *le faire à la place de* ») à travers le geste du « *contre-étayage* », l'enseignant pour avancer plus vite, si la nécessité s'impose, pouvant aller jusqu'à faire à la place de l'élève ou donner la réponse.

Tous ces gestes permettent à l'enseignant de guider les élèves dans la réussite de l'activité, ne pas les mettre en situation d'échec et redonner à l'erreur un statut positif, dans une « atmosphère » propice aux apprentissages.

« *Les cartes "tourne à gauche" ont été enrichies d'une gommette colorée afin de rappeler le chouchou porté par les élèves au poignet gauche*[lors des activités préliminaires de repère dans l'espace mettant en scène le corps de l'élève proposées aux élèves en amont du travail de programmation] *et faciliter ainsi le transfert.* » (Carnet de Bord enseignant B., corps_1)

« *Aidée par l'enseignant pour la verbalisation de chaque déplacement.* » (Évaluation Intermédiaire enseignant SO, tablette_1)

4.1.5. Le maintien dans l'activité

Un autre geste particulièrement observé chez les enseignants est celui du maintien dans l'activité des élèves. Ces praticiens encourageaient les élèves, valorisaient leurs réussites, régulaient le travail collectif à travers la distribution des tâches (par exemple en veillant à ce que chacun joue son rôle dans l'activité) ou proposaient une différenciation pédagogique en fonction du niveau des élèves (simplification ou complexification de la tâche). Derrière tous ces gestes de maintien dans l'activité, l'objectif est celui de conserver et renouveler la motivation des apprenants, de maintenir « *une atmosphère* » propice à un espace de travail et de collaboration langagière et cognitive (Bucheton et Soulé, 2009).

« *C'est aussi le rôle de l'enseignant de dire "c'est bien, tu as bien travaillé aujourd'hui. C'est un peu difficile, on reprendra demain". [...] À chaque fois c'était une évaluation positive. Je disais "tu t'es trompé, mais vas-y corrige toi, tu vas y arriver". Donc, j'ai essayé d'être bienveillante. [...] Le fait de les encourager, c'est vraiment important.* » (Entretien enseignant W, robot_1)

4.1.6. La mise en retrait

Le dernier geste professionnel observé est celui de la mise en retrait volontaire de l'enseignant. Ce dernier se met alors en position d'observateur et invite les élèves à collaborer, à co-construire la solution au problème de programmation posé, ce travail entre pairs autorisant une analyse et une correction du programme par les élèves eux-mêmes. Ce

geste, expert, de « *pilotage* » permet à l'enseignant d'observer ses élèves et d'entrer dans un rôle de modérateur.

« *J'essaie toujours de me mettre en retrait, mais j'essaie que ce soit vraiment les interactions des enfants qui leur permettent de trouver eux-mêmes les solutions aux problèmes qui leur sont proposés. Donc, moi, je suis là plus pour réorienter leurs réflexions, mais j'essaie de pas trop les guider.* » (Entretien enseignant MLP, tablette_2)

4.2. Des postures professionnelles différenciées en fonction des instruments

Les gestes professionnels dégagés précédemment s'organisent autour de trois postures professionnelles endossées par les enseignants dans le cadre d'activités de programmation. En référence aux travaux de Bucheton (2014), nous envisageons une posture professionnelle comme une manière cognitive et langagière de s'emparer d'une tâche en fonction des obstacles liés à l'acquisition du savoir ou aux difficultés ressenties par les élèves.

4.2.1. Le contrôle et le support corps

Une première posture observée est celle du contrôle. Pour celle-ci, le pilotage de l'enseignant est très serré et en synchronie, c'est-à-dire que les élèves doivent travailler au même rythme. L'atmosphère est plutôt tendue et hiérarchique (Bucheton et Soulé, 2009). Les gestes professionnels les plus représentés pour cette seconde posture sont ceux de l'aide dans l'activité de programmation. L'enseignant va, par exemple, inciter voire imposer un procédé à suivre, faire avec ou à la place de l'élève ce qui suppose bien une forme de contrôle sur l'activité de l'élève.

« *Et alors ? qu'est-ce qu'on fait maintenant ? Il faut tourner ? On n'a pas dit de tourner ?* [l'enseignante intervient alors corporellement pour replacer l'élève dans le bon sens]. » (Extrait captation vidéo, Enseignante M, corps_1)

Quelle que soit la place du support dans le processus d'apprentissage, on peut remarquer une prédominance de la posture du contrôle sur le support corps avec une intervention verbale et corporelle davantage marquée des enseignants.

4.2.2. L'accompagnement et le support robot

La posture d'accompagnement se réalise dans le cadre d'un pilotage de classe souple et ouvert et dans une atmosphère détendue et collaborative. L'enseignant apporte une aide de manière ponctuelle individuelle ou

collective en fonction des obstacles à surmonter et de l'avancée de la tâche. Il évite de donner la réponse, il provoque les discussions entre les élèves. Il se retient d'intervenir, observe plus qu'il ne parle. Les élèves, quant à eux, sont amenés à faire et à discuter sur ce qu'ils font et donc à s'inscrire dans une posture réflexive et créative (Bucheton et Soulé, 2009). Les gestes professionnels les plus représentés ici sont le rappel (le plus souvent réalisé par l'élève accompagné par l'enseignant), la reformulation et le questionnement.

« *On n'avait même pas besoin de gérer le groupe, il se gérait tout seul. [...] Y a peut-être des groupes où c'était un petit peu plus difficile donc, ceux-là, plus les accompagner. [...] Un rôle de pilotage. Si on voit que ça ne part pas dans le bon sens on essaie de corriger.* » (Entretien enseignant V, tablette_2)

L'accompagnement dans toutes ses dimensions prédomine sur le support robot. Les enseignants évoquent un pilotage de la classe facilité par ce support qui s'intègre facilement au système de fonctionnement en atelier. L'adoption de cette posture est encouragée par une prise en main facile de l'instrument par les enfants et le système de réglette qui autorise un repérage autonome et aisé de l'erreur par les élèves.

« *J'ai utilisé la stratégie du recours aux pairs prioritairement et puis après effectivement refaire avec le robot, refaire le parcours avec l'accompagnement de l'enseignant.* » (Entretien enseignant C, robot_1)

4.2.3. Le lâcher-prise et le support tablette

Enfin, la posture du lâcher-prise se caractérise par un pilotage confié au groupe, avec une atmosphère de confiance et de refus d'intervention du maître. L'enseignant confie aux élèves la responsabilité de leur travail, les autorise à expérimenter et les laisse libres de leurs choix. Les élèves sont, ici, dans une posture réflexive et « du faire » puisqu'ils sont amenés à agir, créer et à développer une réflexion sur leur propre action (Bucheton et Soulé, 2009). Le geste professionnel le plus représentatif de cette posture est la mise en retrait.

« *J'essaie toujours de me mettre en retrait, mais j'essaie que ce soit vraiment les interactions des enfants qui leur permettent de trouver eux-mêmes les solutions aux problèmes qui leur sont proposés. Donc, moi, je suis là plus pour réorienter leurs réflexions, mais j'essaie de pas trop les guider.* » (Entretien enseignant MLP_tablette_2)

La posture du lâcher-prise est plus prégnante sur support tablette. Le caractère intuitif de ce support, mais aussi son format réduit implique une autonomie des élèves et l'adoption par l'enseignant d'un rôle d'observateur, de modérateur.

4.3. Des conceptions de la programmation signe d'une appropriation des instruments par l'enseignant

L'analyse des données nous ont également permis d'approcher les conceptions construites par les enseignants à propos de l'enseignement de la programmation en maternelle.

Les objets robot et tablettes sont tout d'abord plébiscités pour leurs aspects ludiques, dimension particulièrement importante aux yeux des enseignants intervenants auprès de jeunes élèves. En effet, « jouer pour apprendre » s'inscrit dans les orientations des programmes et recommandations d'enseignement à l'école maternelle qui considèrent le jeu comme essentiel au bon développement physique, psychologique et social de l'enfant (Bulletin officiel n° 31 du 30 juillet 2020). Pour autant, selon les enseignants, cet aspect ludique ne serait pas suffisant en lui-même et des objectifs pédagogiques doivent être clairement identifiés. Nous avons dans un précédent article (Bellegarde *et al.,* 2019), mis en évidence la difficulté pour des jeunes élèves de cycle 1 d'adopter, dans un contexte ludopédagogique, une posture de secondarisation (Bautier et Goigoux, 2004). Ils traitent alors les tâches scolaires sans être capables d'en saisir la signification, c'est-à-dire ce qu'elles leur permettent d'apprendre. On observe chez ces élèves une certaine centration sur « le faire », sur le plaisir de réaliser et réussir l'activité, attitude qui pourrait tout particulièrement être renforcée par une séance ludopédagogique auprès de jeunes élèves de cycle 1. Ainsi, on note chez eux une impossibilité d'énoncer les apprentissages contenus dans les activités de programmation. Une partie des enseignants soulignent d'ailleurs l'importance d'accompagner les élèves dans l'institutionnalisation du savoir afin de permettre aux élèves d'identifier les enjeux cognitifs de la situation d'apprentissage.

« *Je pense que c'est un outil [le robot] qui est amusant pour les enfants. C'est l'aspect ludique de l'outil qui est intéressant. Cela correspond à notre société de maintenant aussi. Je pense qu'il faut y trouver des objectifs pédagogiques pour l'utiliser.* » (Entretien enseignant A, robot_2)

« *Il faut que l'enfant arrive à se détacher du matériel et à ne pas rester centrer sur le "Blue bot"... il y a des enfants qui ont encore besoin de manipulation et de "toucher" et ça peut être un frein s'il en reste là... c'est à l'enseignant de les aider à passer au-delà.* » (Entretien enseignant C, robot_1)

Ainsi, conscients que les activités de programmation ne sont pas au programme du cycle 1, les enseignants y voient la possibilité de réinvestir les compétences travaillées dans d'autres domaines d'apprentissage ce qui justifie en soi cet enseignement. On se situe ici dans « programmer pour apprendre », une initiation précoce à la programmation s'inscrivant dans la logique de parcours et de continuité des apprentissages promue par l'école :

« *Ça leur permet aussi de travailler toutes ces compétences de [...] mise en espace, mais réflexif. Ils réussissent à imaginer ce que peut être une forme, une direction sans forcément l'avoir tout de suite sous les yeux. [....] Le fait de développer la logique des élèves, le retour sur la réflexion, etc.* » (Entretien enseignant H, robot_1)

« *Dans les évaluations nationales et celles de notre bassin qui est quand même défavorisé, on voit que dans les évaluations CE1 ce qui pêche, c'est la résolution de problèmes. Donc, je me dis plus tôt on met en place des activités qui vont mettre en route des images mentales, des schémas chez l'élève, plus tôt, on va l'entraîner à ça, plus vite ça portera ses fruits. Et c'est ça la grande force de ce projet, c'est la construction des images mentales, la résolution de problèmes. Il est pas du tout illogique de travailler des compétences de codage puisqu'elles seront reprises des années plus tard.* » (Entretien enseignant C, robot_1)

Les limites à initier de jeunes élèves à la programmation sont moins évoquées par les enseignants. La question du rapport entre le niveau cognitif de l'enfant et les tâches à accomplir (notamment les tâches qui ont attrait à l'abstraction), est néanmoins soulevée par un enseignant. Il ne faudrait pas trop anticiper certains apprentissages ou, du moins, ne pas forcer l'entrée dans ces apprentissages, le propre de l'école maternelle étant aussi cette faculté de « laisser du temps » aux enfants, de les préparer aux apprentissages futurs, mais sans les anticiper trop prématurément en les mettant en situation d'échec :

« *Peut-être certains se sentiraient en échec face à ça et ensuite, ça se répercuterait sur différentes activités qu'on leur proposerait pour la suite de leur scolarité. En même temps, j'ai pas vu trop d'élèves en échec sur ce qu'on a fait.* » (Entretien enseignant H, robot_1)

Le pilotage de la classe à travers l'accompagnement du travail entre pairs constitue, en revanche, une difficulté mentionnée par plusieurs enseignants :

« *Les inconvénients, c'est que c'est un travail par petits groupes [...]. On doit le faire en dirigé pour le mettre en place, tout au moins au début...parce qu'ils font par essaie-erreur je pense... Donc, ils peuvent peut-être être plus en autonomie, mais on sait bien qu'avec certains enfants cela ne va pas être facile à faire. C'est donc plutôt des difficultés de l'ordre du pilotage de la classe.* » (Entretien enseignant L, robot_1)

Notons également que les enseignants de maternelle n'ont pas été formés à l'enseignement de la programmation ; leurs connaissances théoriques liées aux sciences informatiques apparaissent alors fragiles.

« *J'ai fréquemment utilisé le mot "programme", mais beaucoup moins les termes "instruction" et "code". Je pense ne pas être au clair moi-même avec les nuances entre ces termes.* » (Carnet de Bord enseignant C, robot_3)

4.4. Comprendre l'influence des médiations cognitives sur la grammaire de l'agir enseignant à partir de la matérialité différente des supports

Précédemment, nous avons procédé à l'analyse de la grammaire de l'agir enseignant confronté à la mise en œuvre de scénarios pédagogiques d'initiation à la programmation auprès d'élèves de GSM. Nous avons alors analysé leurs gestes, postures et conceptions professionnels construits lors de ces séances dédiées à la programmation. Nous avons ainsi identifié la mise en œuvre de trois postures professionnelles différentes : la posture de contrôle sur support corps, la posture d'accompagnement sur support robot et la posture de lâcher prise sur support tablette. À chacune de ces postures ont été adossés des gestes professionnels particuliers. Maintenant, nous proposons de comprendre plus particulièrement l'influence des instruments en jeu sur cette grammaire de l'agir enseignant à travers les configurations des postures et gestes professionnels observés.

La matérialité des instruments constitue un élément de compréhension d'un agir enseignant différencié dans les activités de programmation. À l'instar de Laparra et Margolinas, nous pensons que « *la matérialité d'une situation d'apprentissage n'est jamais indifférente* » (2016, p. 30), la dimension matérielle des tâches et objets scolaires structurant l'organisation de l'espace de classe et les conduites des élèves et enseignants au sein de cet espace. Le tableau 2 est une représentation synthétique des matérialités

propres à chacun des supports pédagogiques appréhendés au travers de leurs dimensions tangible (robot et corps) *versus* virtuelle (tablette), intracorporelle (corps) *versus* extracorporelle (robot et tablette) et de leur écart à la réalité physique de l'élève (Greff, 2004).

Tableau 2 · Matérialités des supports-médiateurs

Matérialités/Supports	Corps	Robot	Tablette
Tangible	x	x	
Virtuel			x
Intracorporel	x		
Extracorporel		x	x
Écart à la réalité physique de l'élève réduit			x
Écart à la réalité physique de l'élève important	x	x	

Dans de précédentes analyses (Bellegarde *et al.,* 2019), nous avons montré les incidences de la matérialité des supports médiateurs sur l'appropriation de la programmation chez des élèves de GSM en termes d'activités cognitives prises en charge et de compétences travaillées par les élèves. Demandons-nous maintenant en quoi la matérialité des instruments peut transformer les qualités de la situation d'enseignement/apprentissage du point de vue de l'enseignant.

D'une part, la dimension tangible, caractéristique des modalités corps et robot, a autorisé une intervention de l'enseignant qui peut agir corporellement sur le robot *Blue Bot* ou l'enfant-robot : par exemple, les repositionner dans le bon sens sur la case « départ ». À l'inverse, la dimension virtuelle du support tablette permet une moindre intervention de l'enseignant sur l'objet qui ne peut être saisi, sur lequel il ne peut influer.

« *Je les ai laissés libres de manipuler. Je leur rappelais le scénario et puis comme ils étaient en binôme en fait, ils se débrouillaient tout seul.* [...] [C'était] *du tutorat et puis j'écoutais si la verbalisation était correcte.* » (Entretien enseignant L, tablette_3)

La dimension intracorporelle du support corps est venue également renforcer le positionnement interventionniste de l'enseignant. Les schèmes d'utilisation de ce dispositif à travers un élève qui joue le rôle de l'enfant-robot explique cette posture de contrôle : on observe alors des interventions corporelles et verbales plus marquées pour inciter l'enfant-robot à suivre les instructions données.

« *Il y a eu un peu plus d'interventions de ma part* [...] [sur support corps] *puisque là, le robot est faillible par essence puisque c'est un enfant. Donc, j'avais ce rôle de vérificateur.* » (Entretien enseignant H, corps_3)

Inversement, la dimension extracorporelle des supports robot et tablette, leurs schèmes d'utilisation invitent à une mise en retrait de l'enseignant. « *Ces objets pour penser avec* » (Papert, 1981, p. 23) inscrivent les élèves dans un contexte de résolution de problèmes, la recherche, le contrôle et le débogage du programme par les élèves, étant facilités par la mise en œuvre du programme par les dispositifs pédagogiques eux-mêmes et le système de réglette adossé.

« *Le système réglette permet à l'enfant de revoir sa programmation parce qu'elle reste là.* [Sans], *il n'aurait pas vu le résultat de sa programmation, où est-ce qu'il se serait trompé. [...] Là, au moins, y avait une trace de son essai.* » (Entretien enseignant W., robot_1)

Enfin, l'écart du média à la réalité physique de l'élève vient renforcer la prévalence des postures enseignantes observées sur les trois supports expérimentés lors des activités de programmation. Le dispositif tablette, de par sa taille réduite, apparaît proche de la réalité de l'élève, il la tient entre ses doigts, juste devant ses yeux. À l'inverse, on observe, à travers la grandeur du quadrillage sur support robot et particulièrement sur support corps, un écart plus important entre ces modalités et la réalité physique de l'enfant. Dans le premier cas, l'enseignant se trouve naturellement mis en retrait de l'activité de programmation de l'élève, dans le second, l'enseignant est autorisé à s'impliquer, intervenir dans cette activité.

5. Conclusion

Dans le cadre de ce travail, nous avons souhaité étudier l'agir enseignant pour disposer, à terme, de clés permettant d'analyser plus finement ses effets sur les apprentissages opérés par des élèves de GSM dans le cadre d'une initiation à la pensée informatique. Cette analyse permettrait par la suite, de montrer dans quelles mesures les gestes et postures de l'enseignant influent sur les pratiques des élèves, mais, également, comment les stratégies et postures

d'apprentissages développées par les élèves impactent l'agir enseignant. À l'instar de Bucheton et Soulé (2009), nous envisageons cette dynamique sous le prisme d'un « *ajustement réciproque des postures des enseignants et des élèves* » (p. 42). Pourrait alors être interrogée, l'influence des instruments en jeu (corps, robot, tablette) sur « *cet ajustement réciproque* ».

Dans le cadre de cette contribution, trois dimensions relatives à la grammaire de l'agir enseignant (gestes, postures, conceptions) ont été étudiées lors d'une initiation à la programmation en GSM. Des configurations différentes de l'agir enseignant ont alors été observées, puis, analysées sous le prisme des matérialités propres aux instruments (corps, robot, tablette) expérimentés en classe (tableau 2). Toutefois, dans le cadre de ce projet, nous n'avons pas pu recueillir de données relatives aux enseignants : connaissance de leur culture numérique ou de leur grammaire de l'agir enseignant dans le cadre de leurs activités quotidiennes de classe, par exemple. Ces variables auraient pu être intéressantes à identifier pour comprendre leurs effets sur les pratiques d'enseignement de la pensée informatique auprès de jeunes élèves.

En outre, les conditions d'observation avec la présence de caméras et de chercheurs peuvent nécessairement influencer les agir enseignants, ne serait-ce que par le biais des désirabilités sociales qui poussent tout un chacun à se montrer sous son meilleur jour. Ainsi, les enseignants observés et interviewés sont probablement conscients de ce contexte particulier qui peut potentiellement influer sur leurs gestes et postures professionnels. Cela nous invite donc à éprouver et repenser nos protocoles pour de prochaines expérimentations. Inscrire les observations sur de plus longues durées pourrait, par exemple, constituer une manière d'atténuer les effets de la présence du chercheur sur le terrain et donc leurs influences sur les pratiques de classe de l'enseignant.

Nos analyses ouvrent néanmoins des perspectives qu'il conviendrait à présent d'investiguer davantage. En effet, au démarrage de ce travail, nous avons fait l'hypothèse que le support robot conduirait particulièrement à une mise en retrait de l'enseignant au profit d'apprentissages autodirigés chez les élèves. Nos premières analyses nuancent ce positionnement supposé de l'enseignant. Nos observations relèvent tout d'abord la manière dont les environnements informatiques d'apprentissage humains (EIAH), ici, les modalités robot et tablette numérique, constituent des sources de guidance pour l'élève, une aide à la résolution de problèmes. La mise en œuvre du programme par les environnements eux-mêmes, les traces sur l'activité de

l'élève et les feed-back procurés par le système de réglette accompagnent l'apprenant dans une réflexion sur sa propre action, favorisent une révision et réussite de sa tâche (Bellegarde *et al.,* 2019). Il en résulte pour l'enseignant une facilitation de sa tâche d'accompagnement des élèves dans les activités de programmation et l'adoption, chez ces praticiens, de gestes et postures professionnels qui laissent une place aux interactions entre les systèmes informatiques et les sujets-apprenants, autorisant chez les élèves de GSM une construction active de leurs savoirs. À l'inverse, nous avons montré que, sur support corps, l'enseignant ne disposant de systèmes informatiques de guidage des apprentissages, se trouve dans la nécessité de guider, de contrôler lui-même, l'activité de l'élève ; une posture plus interventionniste a ainsi été observée dans le cadre de la recherche *Blue Bot*. Nous avons également montré à travers la matérialité des supports que, même si le robot procure à l'enfant des possibilités de décentration, de manipulation de l'objet-robot intéressantes, sa dimension tangible et son écart à la réalité physique de l'élève structurent l'agir enseignant autour de gestes professionnels relevant de la posture d'accompagnement. La matérialité du support robotique autorise l'intervention de l'enseignant et moins sa mise en retrait. C'est alors la dimension virtuelle de la tablette et sa proximité à la réalité physique de l'élève qui encouragent l'adoption de la posture du lâcher-prise chez l'enseignant.

À l'instar de Laparra et Margolinas (2016), ces analyses semblent révéler l'intérêt à porter à la matérialité des tâches et objets scolaires, et, notamment, celle des EIAH, dans l'optique de comprendre la manière dont elle peut transformer les qualités des situations scolaires et ainsi influer sur l'agir enseignant. Ces éléments mettent en évidence les enjeux de formation et d'accompagnement des enseignants relatifs à l'introduction des EIAH et des sciences informatiques à l'école.

REMERCIEMENTS

Pour leurs conseils, expertises, le partage de leurs expériences, soutiens, traductions et suggestions, nous tenons à remercier : Véronique Alvarez, Anne Losq, Gilles Brougère, Serge Tisseron, Sylvie Leleu-Merviel, Dorothée Hallier-Vanuxemm, Margarida Romero, Jean-François Condette, Alfonsino Cutillo, Romain Deledicq, Yoann Lebrun, Philippe Leclercq, David Detève, Angelino Mascaro, Patrick Pelayo, Gilles Petit, Yvan Peter, Yann Secq et Marielle Léonard ainsi que tous les partenaires associés au projet de recherche *Blue Bot* : l'Académie Lille et de Dijon, la Dane de Lille, le laboratoire DeVisu de l'université Polytechnique Hauts-de-France et l'INSPE Hauts-de-France ainsi que l'Université de Laval.

RÉFÉRENCES

Alvarez, J. (2018). La ludopédagogie. *Lectures.Cultures, 10*, 29-31.

Alvarez, J., Bellegarde, K., Boyaval, J., Hurez, V., Flahaut, J-J. et Lafouge, T. (2021, juin) *Hypothèse de la « distance » appliquée à la robot-pédagogie pour les enfants en maternelle* [communication orale]. *Xe Conférence « Environnements informatiques pour l'apprentissage humain » [EIAH],* Fribourg, Suisse. https://hal.archives-ouvertes.fr/hal-02129311

Baron, G.-L. et Drot-Delange B. (2016). L'éducation à l'informatique à l'école primaire. *Bulletin de la Société informatique de France, 9*, 73-79. https://www.societe-informatique-de-france.fr/wp-content/uploads/2016/11/1024-no9-Baron-Drot-Delange.pdf

Bautier, E. et Goigoux, R. (2004). Difficultés d'apprentissage, processus de secondarisation et pratiques enseignantes : une hypothèse relationnelle, *Revue française de pédagogie, 148,* 89-100. https://www.persee.fr/doc/rfp_0556-7807_2004_num_148_1_3252

Bellegarde, K., Boyaval, J. et Alvarez, J. (2019). S'initier à la robotique/informatique en classe de grande section de maternelle – une expérimentation autour de l'utilisation du robot Blue Bot comme jeux sérieux. *RESMICTE, 13*(1), 51-72. https://pasithee.library.upatras.gr/review/article/view/3105

Béziat, J. (2012). Informatique, outil ou objet ? Permanence d'une question – Le cas de l'école primaire en France. *Adjectif, 3*, 1-7. https://adjectif.net/spip.php?article177

Bucheton, D. (2014). *L'agir enseignant : des gestes professionnels ajustés.* Ocatres.

Bucheton D. et Soulé, Y. (2009). Les gestes professionnels et le jeu des postures de l'enseignant dans la classe : un multiagenda de préoccupations enchâssées, *Éducation et didactique, 3*(3), 29-48. https://journals.openedition.org/educationdidactique/543

Bruner, J. S. (1983). *Le développement de l'enfant, savoir-faire, savoir dire.* Presses universitaires de France.

Cohen, R. et Mialaret, G. (1987). *Les Jeunes Enfants, la découverte de l'écrit et de l'ordinateur.* Presses universitaires de France.

Denis, B. (2000). Vingt ans de robotique pédagogique. *Sciences et techniques éducatives, 1*(7), 195-206. https://www.persee.fr/doc/stice_1265-1338_2000_num_7_1_1450

Greff, E. (2004). Le corps d'abord ! *Éducation enfantine, 1056,* 62-63.

Kosmis, V. et Misirli, A. (2011). Robotique pédagogique et concepts préliminaires de la programmation à l'école maternelle : une étude de cas basée sur le jouet programmable Bee-Bot. Dans G.-L. Baron, E. Bruillard, V. Komis (dir.), *Sciences et technologies de l'information et de la communication en milieu éducatif : Analyse de pratiques et enjeux didactiques* (p. 271-281). New Technologies Éditions. https://edutice.archives-ouvertes.fr/edutice-00676143/document

Komis, V. et Misirli, A. (2013). Étude des processus de construction d'algorithmes et de programmation par les petits enfants à l'aide de jouets programmables. Dans G.-L. Baron, E. Bruillard, V. Komis (dir.), *Sciences et technologies de l'information et de la communication en milieu éducatif* (p. 271-281). Edutice-00875628. https://edutice.archives-ouvertes.fr/edutice-00875628/document

Laparra, M. et Margolinas, C. (2016). *Les premiers apprentissages scolaires à la loupe : des liens entre énumération, oralité et littératie.* De Boeck.

Linard, M. (1996). *Des machines et des hommes. Apprendre avec les nouvelles technologies.* L'Harmattan.

Leroux, P. (2002). *Machines partenaires des apprenants et des enseignants - Étude dans le cadre d'environnements supports de projets pédagogiques* [HDR, Université du Maine]. https://tel.archives-ouvertes.fr/edutice-00000311/document

Mélot, L., Strebelle, A., Mattens, J. et Depover, C. (2018). Dessiner un bonhomme en maternelle : analyse comparative des dessins réalisés avec des outils traditionnels et avec une tablette tactile. *frantice.net, 14(1),* p. 25-38. http://frantice.net/index.php?id=1461

Papert, S. (1981). *Jaillissement de l'esprit. Ordinateur et apprentissage.* Flammarion.

Rézeau, J. (2002). Médiation, médiatisation et instruments d'enseignement : du triangle au carré pédagogique. *ASP, Varia, 35-36,* 183-200. https://journals.openedition.org/asp/1656

Romero, M. (2016). *Vibot - le robot.* Les publications du Québec.

Romero, M. (2018). Développer la pensée informatique pour démystifier l'intelligence artificielle. *Bulletin de la société informatique de France, 12,* 67-75. https://www.researchgate.net/publication/325810912_Developper_la_pensee_informatique_pour_demystifier_l'intelligence_artificielle

Vivet, M. (2000). Des robots pour apprendre. *Sciences et techniques éducatives, 7,* 17-60. https://www.persee.fr/doc/stice_1265-1338_2000_num_7_1_1441

Weil-Barais, A. et Resta-Schweitzer, M. (2008). Approche cognitive développementale de la médiation en contexte d'enseignement-apprentissage. *La nouvelle revue de l'adaptation et de la scolarisation, 42,* 83-98. https://www.cairn.info/revue-la-nouvelle-revue-de-l-adaptation-et-de-la-scolarisation-2008-2-page-83.htm

Wing, J. (2008). La pensée informatique. *Bulletin of Specif,* 1-4. https://www.cs.cmu.edu/afs/cs/usr/wing/www/ct-french.pdf

MOTIF..MOTIF.. : initier à la notion de « répétition » en maternelle sans mobiliser de repérage spatial

▶ **Marielle LÉONARD** (1 ; 2), **Yvan PETER** (1), **Yann SECQ** (1), **Julian ALVAREZ** (1), **Cédric FLUCKIGER** (2)

(1) Univ. Lille, CNRS, Centrale Lille, UMR 9189 CRIStAL, France

(2) Univ. Lille, ULR 4354 – CIREL – Centre interuniversitaire de recherche en éducation de Lille, F-59000 Lille, France

■ **RÉSUMÉ** • Cet article détaille un dispositif pédagogique destiné à initier des élèves de 5-6 ans à la notion de « répétition ». Celui-ci articule activités débranchées, résolution de problèmes et activités créatives dans un EIAH. L'analyse des observations et des traces numériques obtenues en milieu écologique montre qu'il est possible d'introduire les notions de base en algorithmique en s'appuyant sur l'identification de motifs visuels en évitant les difficultés liées au repérage spatial (mouvements du robot). Nous fournissons également les premiers éléments d'appropriation par des enseignants et des enseignantes de maternelle qui ont été formés à la pensée informatique.

■ **MOTS-CLÉS** • pensée informatique, programmation, apprentissage par le jeu.

■ ***ABSTRACT*** • *This article presents an educational setting intended to introduce 5-6 year olds pupils to the loop concept. It articulates unplugged activities, problem solving and creative activities through an online platform. The analysis of observations and digital traces collected in an ecological setting shows that it is possible to introduce basic algorithmic concepts by relying solely on visual patterns identification, avoiding spatial tracking difficulties (robot movements). We also show elements of appropriation by kindergarten teachers who have been introduced to computational thinking.*

■ ***KEYWORDS*** • *computational thinking, programming learning, game-based learning.*

Marielle LÉONARD, Yvan PETER, Yann SECQ, Julian ALVAREZ, Cédric FLUCKIGER

1. Introduction

Depuis plusieurs années, et particulièrement suite à l'article de Wing (2006), la question de la formation de l'ensemble des élèves à la pensée informatique a suscité de nombreux débats (Drot-Delange *et al.,* 2019) et a participé à de fortes évolutions institutionnelles au niveau des programmes scolaires de plusieurs pays. Ces débats se poursuivent toujours et portent sur le périmètre de cette notion de « pensée informatique », ainsi que sur les éléments qu'il conviendrait d'introduire selon le niveau des élèves afin d'avoir un cursus cohérent. En France, depuis 2015, des réformes successives ont introduit des éléments d'informatique dans les différents programmes du système scolaire : depuis l'école primaire (3 à 10 ans), puis au collège (11 à 14 ans) jusqu'au lycée (15 à 18 ans). Ces modifications significatives des programmes entraînent d'importantes problématiques en termes de didactique, sur ce qu'il est pertinent de réaliser selon les niveaux d'études, mais aussi institutionnelles, pour ce qui est de la formation des enseignants et enseignantes en poste ou en formation.

Dans cet article, nous présentons les travaux que nous effectuons afin de répondre partiellement à certaines de ces problématiques. Nous proposons un scénario pédagogique d'initiation à la programmation pour de jeunes élèves de 5-6 ans et un dispositif associé pour la formation de leurs enseignants et enseignantes. Cette initiation porte sur les notions fondamentales de la programmation avec un focus spécifique sur l'appropriation de la notion de « répétition » et le développement de la capacité de reconnaissance de motifs. Dans le système scolaire français, les élèves de 5-6 ans sont en grande section de maternelle, c'est-à-dire dans l'année qui précède l'apprentissage structuré de la lecture et de l'écriture. Les élèves visés sont donc non-lecteurs, mais entrent progressivement dans la culture de l'écrit. Des problématiques spécifiques sont à appréhender pour cette tranche d'âge, comme la non-maîtrise de l'écrit et l'importance des activités manipulatoires.

Pour répondre à ces problématiques, nous articulons des activités débranchées et des activités sur tablette. Nous mobilisons les activités débranchées (Romero *et al.,* 2018b) afin d'introduire les notions dans un contexte non-technique. Par la mobilisation d'un environnement de programmation simplifié sur une plateforme en ligne, nous visons le renforcement de l'appropriation de ces notions. Nous avons adopté une approche ludopédagogique qui consiste à mettre en situation d'apprendre avec le jeu sous différentes formes.

Nos travaux antérieurs réalisés avec des élèves plus âgés de 8-10 ans (CM1-CM2) (Peter *et al.,* 2019), puis de jeunes lecteurs de 6-7 ans (CP) (Léonard *et al.,* 2020), nous ont permis d'identifier plusieurs paliers de difficultés pour l'apprentissage des bases de la programmation et de la notion de « répétition ». La première expérimentation a permis de conclure que la notion de « répétition » est accessible à des élèves de 8-10 ans et repose sur le développement de leurs capacités de reconnaissance et de synthèse de motifs redondants. Nous avons aussi détecté un palier de difficulté lorsque le motif à identifier est constitué de plusieurs instructions. Cette difficulté a été confirmée par des résultats similaires avec les 6-7 ans. Nous avons également observé sur cette tranche d'âge, un impact positif de la suppression des activités impliquant un repérage dans l'espace (par exemple le déplacement d'un robot), au profit de la reproduction de frises colorées.

Suite à cette dernière expérimentation, nous nous sommes interrogés sur la possibilité de réaliser une séquence similaire avec un public plus jeune et non-lecteur. Est-ce que le contexte proposé permettrait à ces élèves de s'approprier la notion de « répétition » et plus spécifiquement, est-ce que les capacités de reconnaissance de motifs et de leur synthèse peuvent être développées dès 5-6 ans ? Quelles doivent être les évolutions à apporter au niveau des activités débranchées et leur articulation avec la plateforme en ligne ? Quel serait le degré d'appropriation de cette séquence pédagogique par leurs enseignants et enseignantes ?

Cet article vise à apporter des éléments de réponse à ces différentes questions en proposant un scénario ludopédagogique. D'une part, le scénario est testé avec des élèves de 5-6 ans en milieu écologique, au sens d'un milieu ordinaire non contrôlé expérimentalement. D'autre part, ce scénario est présenté à des enseignants de cycle 1 en formation continue. La section 2 présente les travaux proches de nos problématiques et les cadres théoriques mobilisés. La section 3 détaille le scénario pédagogique « MOTIF..MOTIF.. » et les évolutions effectuées pour l'adapter à un plus jeune public. La section 4 analyse les résultats des expérimentations réalisées en classe, à partir d'observations et des traces d'activités des élèves sur la plateforme. La section 5 aborde quelques retours de terrain collectés auprès d'enseignants et enseignantes ayant suivi la formation et ayant expérimenté une séquence avec leur classe. Finalement, la conclusion synthétise les principaux résultats de cette étude et présente les perspectives qui s'ouvrent à l'issue de ces travaux.

2. État de l'art

2.1. Pensée Informatique

Il n'existe pas de définition unanimement acceptée du terme « Pensée Informatique » (PI). Celle-ci se réfère notamment à un certain nombre d'habiletés (abstraction, réflexion algorithmique...) qui ouvrent la voie à un traitement automatisé. On pourra se référer à ce sujet au compte-rendu de la table ronde qui a eu lieu au colloque Didapro 7 (Drot-Delange *et al.,* 2019). On retrouve les premiers fondements de la PI dans les travaux de Papert (1980) et l'intérêt de la recherche et de l'éducation pour ce domaine a été grandement relancé par la prise de position de Wing qui considérait que la PI devait faire partie des enseignements fondamentaux à l'école (2006).

On recense plusieurs approches pour l'apprentissage de la PI qui peuvent être combinées et se compléter : la robotique pédagogique, les activités débranchées et les environnements de programmation dédiés.

2.2. Robotique pédagogique

Papert (1980) a probablement fondé la robotique éducative dès 1969 avec le robot de plancher Turtle et son langage de programmation *Logo* associé (Catlin et Blamires, 2019 ; Catlin et Woollard, 2014 ; Denis, 1987). Ce langage a ensuite donné naissance à plus de 300 autres variantes logicielles (Boytchev, 2014). Parmi ces déclinaisons du langage *Logo*, certaines étaient consacrées à des jouets tangibles tels que le *Logo LEGO* (Jarvinen, 1998 ; Krumholtz, 1998 ; Ocko et Resnick, 1987). Le robot de plancher *Turtle* a inspiré de son côté plusieurs jouets robots, comme *Blue Bot* par exemple, que l'on retrouve mobilisés dans le champ de l'enseignement (pour une étude sur les apports de la robotique pédagogique au regard d'autres modalités, voir Bellegarde *et al.* (2019).

Cependant, si la modalité offerte par l'emploi de jouets robots peut générer des résultats intéressants auprès de jeunes élèves comme ceux de cycle 1 (Alvarez *et al.,* 2018), ou avec des élèves de cycles supérieurs (Nogry, 2019), leur emploi sur le terrain se heurte à des problèmes de repérage dans l'espace (prise de perspective décentrée gauche/droite) (Touloupaki *et al.,* 2019) et qui perdurent chez la moitié des enfants de 11 ans (Romero *et al.,* 2018a). On peut raisonnablement imaginer que cette proportion est plus importante avec des enfants de 5 ans. Komis *et al.* (2011) pointent la difficulté de prise en main des commandes de pivotement. À cela s'ajoute le fait que ce repérage dans l'espace est encore plus complexe chez les

enfants lorsqu'ils sont en présence d'objets en mouvement (déplacements et rotations) (Lurçat, 1979).

En parallèle peuvent être aussi recensées d'autres contraintes comme la possibilité pour le corps enseignant d'accéder simplement à des robots jouets pour conduire leurs enseignements ou à devoir gérer des pannes matérielles par exemple. Face à de telles contraintes, qui ne sont pas exhaustives, Romero *et al.* (2018a) préconisent de se tourner vers les activités débranchées pour enseigner la pensée informatique.

Ces difficultés, rencontrées de manière récurrente dans les expérimentations de robotique pédagogique, nous amènent à ne pas mobiliser de jouet robot et à dissocier l'introduction de notions de pensée informatique de la mobilisation de compétences de repérage dans l'espace.

2.3. Environnements de programmation pour jeunes élèves

Concernant les environnements de programmation adaptés aux élèves de 5-6 ans, ScratchJr (https://www.scratchjr.org/) est le plus répandu. Le concept central pour *ScratchJr* est celui d'animation (Komis *et al.,* 2017). L'analyse de ces auteurs montre qu'en dépit des apparences, la syntaxe de ScratchJr n'est ni simple, ni intuitive pour des jeunes élèves. Aborder la répétition est difficile dans cet environnement avec des élèves de cette tranche d'âge (Touloupaki *et al.,* 2019). Dans l'environnement Scratch (https://scratch.mit.edu/) destiné aux élèves à partir de 8 ans, on retrouve des difficultés liées à la grande quantité de blocs disponibles, dont seulement un petit nombre est mobilisé pour une activité d'initiation. Cette constatation amène Romero *et al.* (2018a) à sélectionner des blocs en amont d'une séquence pédagogique avec l'environnement Scratch. Dans l'environnement de programmation mobilisé pour la présente recherche, seuls les blocs qui font sens pour l'activité proposée sont disponibles sur l'interface.

Dans cette étude, nous recherchons des approches qui articulent des activités débranchées et des activités dans un environnement de programmation par blocs sans mobiliser de commandes de déplacement. Très peu de travaux concernent la tranche d'âge de 5-6 ans qui nous intéresse, ce qui nous amène à élargir notre état de l'art avec les travaux concernant des élèves un peu plus âgés.

2.4. Activités débranchées

Olmo-Muñoz *et al.* (2020) ont étudié l'impact des activités débranchées sur le développement des capacités de PI dans les premières années de primaire. Leur étude montre que l'utilisation d'activités débranchées préalablement aux activités sur support numérique a un impact positif à la fois sur le développement de la PI, mais également sur la motivation. Cette étude récente confirme les résultats de Brackmann *et al.* obtenus lors d'une expérimentation avec des élèves de 10-12 ans. Pour cette tranche d'âge, ils ont montré au moyen de pré-test et post-test que des activités débranchées sur support papier ont un impact équivalent à une séquence sur *code.org* (plateforme qui propose des activités de programmation en ligne) en termes de développement des compétences de PI (2017). Ces derniers évoquent l'articulation entre activités débranchées et activités branchées pour renforcer l'apprentissage de la PI.

Aggarwal *et al.* (2017) quant à eux, ont mesuré l'impact d'une activité manipulatoire en support d'activité de conception de programme pour des élèves de 8-10 ans. Ils la comparent à une activité similaire dans un environnement de programmation par blocs. Ces auteurs concluent que le feedback dynamique de l'environnement de programmation est plus efficace que la manipulation de matériel tangible pour la compréhension des concepts et la représentation de l'exécution du programme. Ils montrent aussi que la manipulation des éléments de langage via du matériel tangible constitue une aide efficace à la conception de programme (encodage). Ils suggèrent donc une utilisation limitée de matériel tangible comme introduction, avant de passer dans un environnement de programmation par blocs.

Nous nous situons dans la même perspective d'articulation d'activités débranchées et d'activités dans un environnement de programmation par blocs, avec un focus sur la notion de « répétition ». Le scénario que nous proposons et qui est détaillé dans la partie suivante repose sur la manipulation de briques de construction au cours d'activités débranchées organisées sous forme de jeux. Utiliser des briques de construction, matériel très courant dans les classes, nous semble pertinent. Saxena *et al.* (2019) décrivent une activité basée sur ce matériel avec des élèves de 3 à 6 ans, qui consiste à continuer une suite logique. Cette activité est placée en amont d'une séquence de robotique pédagogique, sans que l'articulation soit identifiée. Nous explorons cette piste pour notre approche de l'introduction de la PI basée sur la reconnaissance de motifs visuels, en

portant une attention particulière, dans le scénario ludopédagogique, à l'articulation entre les activités débranchées et les activités sur tablette. Ainsi, notre démarche s'inscrit en rupture avec la mobilisation de robots pédagogiques et d'instructions de déplacements, pour introduire des notions de PI. Nous cherchons à appréhender si le dispositif que nous décrivons dans la partie suivante rend la notion de « répétition » accessible à des enfants de 5-6 ans. Parallèlement, nous sommes attentifs au degré d'appropriation d'un tel dispositif par des enseignants et enseignantes d'école maternelle.

3. Élaboration du scénario ludopédagogique

3.1. Ludopédagogie

La ludopédagogie, est une méthode pédagogique où l'on emploie du jeu (Alvarez, 2018) ou encore du jeu sérieux comme médiation pour diffuser des messages (enseignement), dispenser des entraînements (exercices) ou collecter des données (évaluations). Keymeulen (2016) précise pour sa part : « l*e terme ludopédagogie englobe à la fois la pédagogie du jeu et la pédagogie des jeux. Il s'agit d'une part de l'utilisation du jeu et des jeux dans les apprentissages, mais plus encore d'une méthodologie d'apprentissage basée sur le jeu* ». Ainsi le simple fait de convoquer du jeu ou bien encore du jeu sérieux dans une pédagogie, nous inscrit dans la ludopédagogie. Précisons cependant, qu'il existe plusieurs manières d'employer du jeu dans un cadre pédagogique. Ainsi pour Hochet (2013), nous pouvons enseigner « *avec* », « *par* », « *sur* » et « *autour du jeu* ». Dans une approche ludopédagogique moderne, le défi consiste notamment à enseigner « *avec du jeu* » et moins « *par le jeu* ». La nuance étant que, dans le cas où l'on « *enseigne par le jeu* », nous sommes dans l'idée d'utiliser le jeu comme élément de récompense une fois la tâche utilitaire accomplie. En revanche, lorsque l'on cherche à enseigner « *avec le jeu* », il s'agit d'employer le jeu comme une véritable médiation : on s'appuie sur le jeu pour illustrer les propos, pour faire vivre une expérience concrète, pour contextualiser un concept... Le jeu n'est pas accessoire dans cette approche ludopédagogique, il est central.

Dans le cadre de nos expérimentations, c'est l'approche « *avec le jeu* » que nous convoquons au travers de l'emploi de jouets que représentent les briques de construction ou bien encore de jeux sérieux comme l'environnement « MOTIF ART » sur tablette. Ainsi, nous parlerons, dans notre cas, de scénario ludopédagogique pour définir l'idée que notre scénario pédagogique convoque du jeu.

3.2. Apport des expérimentations précédentes

Le scénario ludopédagogique proposé s'appuie sur celui développé dans le cadre du projet *Chticode* avec des élèves de 8-10 ans (Peter *et al.,* 2019). Celui-ci a ensuite fait l'objet d'une première évolution dans le cadre d'une étude exploratoire (projet « MOTIF..MOTIF.. ») menée avec des élèves de 6-7 ans en cours d'apprentissage de la lecture (Léonard *et al.,* 2020).

Ainsi, l'expérimentation *Chticode* (Peter *et al.,* 2019) menée avec des élèves de 8-10 ans, propose un scénario ludopédagogique constitué de séquences alternant l'usage d'un jeu de plateau et d'activités en ligne réalisées sur tablette et ordinateur. Le jeu de plateau consiste à commander le déplacement d'un personnage vers une case cible (personnage non orienté puis orienté). La même situation est ensuite reprise et renforcée sur une plateforme où l'objectif est de contrôler le déplacement d'un robot virtuel avec un langage de programmation par blocs, puis avec un langage de programmation textuel (*Python*).

Suite à cette première expérimentation, un nouveau scénario ludopédagogique, « MOTIF..MOTIF.. 6-7 ans » (Léonard *et al.,* 2020), a été créé avec une approche mixte. Lors de cette étude exploratoire, la notion de motif a été introduite dans un contexte de reproduction de frises colorées avec des briques de construction, activité connue en amont des élèves qui l'avaient déjà vécue en classe dans d'autres contextes. Le changement fondamental, par rapport à l'activité avec les 8-10 ans, porte sur la suppression de la gestion explicite des déplacements du robot lors de l'activité connectée. Dans une deuxième étape, la reconnaissance et la synthèse de motifs redondants ont été reprises dans un contexte de déplacements.

Cette étude exploratoire a montré que la mobilisation de la notion de « répétition », qui passe par la reconnaissance et la synthèse de motifs redondants, est abordable avec des élèves de 6-7 ans en cours d'apprentissage de la lecture. Toutefois le contexte joue un rôle déterminant. Pour cette tranche d'âge, la notion de « répétition » est accessible dans un environnement où le motif peut être identifié visuellement. L'expérimentation a montré qu'en revanche, dans un contexte de déplacement de personnage, la gestion du repérage dans l'espace (gauche/droite) associé au fait de ne pas pouvoir visualiser le motif constituent des obstacles majeurs à l'identification de motifs redondants.

Comme l'objet de cette étude est de déterminer si la reconnaissance de motifs redondants et leur synthèse sous forme de répétition est accessible à un public encore plus jeune, nous abandonnons complètement, et c'est sans doute l'une des originalités de cette étude, le contexte de déplacement d'un personnage, évacuant ainsi les difficultés de repérage spatial. Nous nous concentrons sur le contexte de reproduction de frises colorées qui nous semble plus accessible pour aborder la notion de « répétition » avec des élèves de 5-6 ans non lecteurs. Plus précisément, l'expérimentation vise à amener la notion de « répétition » progressivement, en distinguant différentes étapes : passer du *faire* au *faire faire,* passer du langage naturel à un langage formel très simple, identifier des régularités que nous nommons motifs, dénombrer les motifs identifiés et utiliser l'élément de langage formel qui code la répétition.

3.3. Conception du dispositif expérimental

Une première version du scénario ludopédagogique a été expérimentée en classe, permettant d'identifier les difficultés des élèves. La figure 1 présente le scénario effectivement réalisé suite aux observations de la première séance et des retours des professeurs des écoles lors des sessions de formation. On note l'alternance d'activités débranchées préparatoires à l'activité numérique ainsi qu'une séance, effectuée en différé plusieurs semaines après les précédentes, qui mobilise l'environnement numérique en mode créatif. Cette dernière séance avait pour but d'observer la capacité des élèves à remobiliser les concepts abordés précédemment.

À noter qu'un groupe d'élèves, dans une première version du scénario, a utilisé l'environnement à la fin de la première séance, avant l'introduction du jeu de cartes, ce qui apparaît en grisé dans la figure 1 ci-dessous. Nous nous sommes toutefois assurés que les résultats sur le reste des séances ne sont pas affectés pour ce groupe, ce qui est bien le cas.

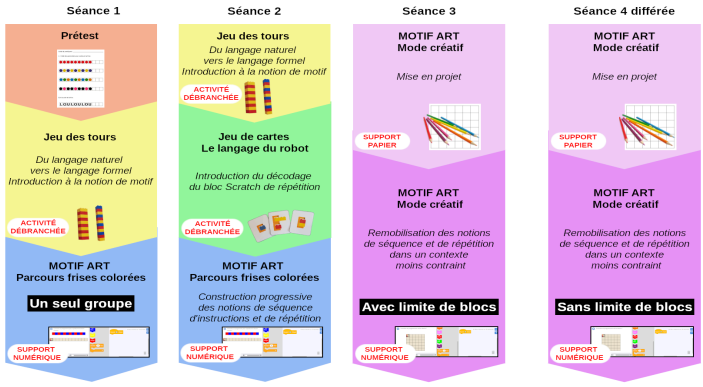


Figure 1 · Scénario ludopédagogique

Les trois premières séances du scénario ludopédagogique ont lieu à quelques jours d'intervalle (lundi et vendredi d'une semaine, et lundi de la semaine suivante). Les quatre groupes sont pris successivement en charge par l'expérimentatrice au cours de la journée. Il s'agit donc d'une approche recherche action. En effet, l'expérimentatrice a elle-même exercé à ce niveau de classe pendant plusieurs années, et a été présentée aux jeunes élèves comme une « *maîtresse qui vient nous aider* ». Présenter l'expérimentation de cette manière vise à mettre les jeunes élèves en confiance en attribuant à la nouvelle personne un rôle qui leur est familier et à optimiser de ce fait, le caractère écologique de la situation.

La durée des séances, initialement fixée à 45 minutes, a fluctué, de 35 minutes à 1 h 15, pour s'adapter aux contraintes logistiques de l'école. En particulier, la séance avec le premier groupe, avant la récréation du matin, a duré plus longtemps que les autres, 1 h 15, alors que celle avec le deuxième groupe, après la récréation a été plus courte, environ 35 minutes. Les deux séances de l'après-midi ont, quant à elles, duré 45 minutes comme prévu initialement. Cet écart a été en partie compensé par le roulement des groupes, trois des quatre groupes ayant bénéficié d'une séance plus longue.

Cette expérimentation s'est déroulée de janvier à mars 2020 dans une école maternelle de la métropole lilloise dans le nord de la France. Elle a concerné 36 élèves de 5-6 ans, scolarisés en grande section. Dans cette école, les élèves sont répartis dans des groupes de travail stables et hétérogènes. Ces groupes ont été conservés pour l'expérimentation. Trois groupes, constitués de 9 ou 10 élèves sont issus de la même classe. Un quatrième groupe de 9 élèves est issu d'une autre classe.

3.4. Activités du scénario ludopédagogique

Dans cette section, nous décrivons les quatre séquences du scénario ludopédagogique proposées aux élèves. Ces séquences alternent l'usage d'activités débranchées pour mobiliser les concepts dans un cadre non technique, le renforcement de ces concepts dans un environnement de programmation par blocs et la mobilisation de ces concepts au cours d'une activité créative dans ce même environnement :

– l'activité débranchée « Le jeu des tours » (briques de construction),

– l'activité débranchée « Le langage du robot » (jeu de cartes),

– le parcours frises colorées MOTIF ART dans une approche de type résolution de problèmes,

– le parcours créatif MOTIF ART dans une approche de mini-projet.

3.4.1. Séquence 1 : activité débranchée « Le jeu des tours »

La première séquence du scénario ludopédagogique est un jeu sérieux intitulé « Le jeu des tours ». Il est de type collaboratif et son objectif est d'amener les élèves à identifier un motif redondant sur une séquence linéaire de couleurs et à exprimer verbalement la répétition de ce motif.

Le but du jeu est de reproduire une tour constituée de briques de couleur. Le jeu se déroule en binôme avec un messager et un constructeur :

– le messager est le seul à pouvoir aller voir la tour modèle. Son rôle consiste à observer ce modèle, à en identifier ses caractéristiques, puis à guider verbalement son partenaire ;

– le constructeur est le seul à avoir le droit de manipuler les briques afin de reconstruire la tour sur la base des informations transmises par le messager.

Ce type de jeu de transport d'informations est utilisé de manière fréquente à l'école maternelle, notamment pour la construction du nombre. Placer « Le jeu des tours » comme première séquence de notre scénario ludopédagogique nous assure ainsi de nous situer dans un contexte familier pour les élèves et leur enseignant ou enseignante. C'est aussi dans cette optique que nous mobilisons des briques de construction, jouets très répandus dans les classes.

Toutes les tours proposées comme modèles sont structurées suivant une suite logique : motifs de longueur 2 ou 3 (figure 2). Les caractéristiques des tours à reproduire constituent les variables didactiques de la situation. Outre la taille et la visibilité du motif, les couleurs, le type de brique et le

nombre de briques utilisées varient et permettent de complexifier plus ou moins la situation.

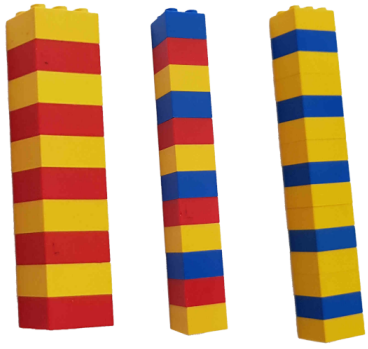


Figure 2 · Exemples de modèles de tours à décrire

Après l'explication des règles et la mise en place de la situation de jeu, plusieurs parties sont jouées, entrecoupées par des phases collectives de retour d'expérience et d'élaboration de stratégies. Ces phases collectives, animées par l'adulte, visent à faire prendre conscience de la nécessité de la précision et de la non-ambiguïté des indications données par le messager au constructeur.

La prise de conscience du besoin d'un langage précis et non ambigu constitue un premier pas vers le passage au langage formel nécessaire pour transmettre des ordres à une machine.

Les notions de motif et de répétition sont introduites par une manipulation. Chaque élève disposant d'une tour à la fin du jeu (la tour modèle ou la tour reproduite), l'adulte demande de casser cette tour en morceaux identiques. Il précise qu'il faut obtenir le plus de morceaux identiques possibles. Chaque morceau obtenu correspond donc à un motif. Il suffit de compter le nombre de morceaux pour obtenir le nombre de répétitions du motif.

Ainsi, la séquence du jeu des tours a pour fonction d'introduire très concrètement, par un jeu de rôle et la manipulation de matériel tangible, des concepts et des pratiques au cœur de la pensée informatique : transmission d'informations, traitement d'informations, langage formel, repérage et dénombrement de régularités que nous appelons « motifs » dans notre contexte.

3.4.2. Séquence 2 : activité débranchée « Le langage du robot »

La séquence 2 est une activité de jeu sérieux se présentant sous la forme d'un jeu de cartes intitulé « Le langage du robot ». Cette séquence a pour objectif de faciliter la transition du langage naturel mobilisé dans la première activité vers un langage formel permettant de réaliser une programmation par blocs. Cette activité permet d'amener les élèves à construire le sens du bloc de répétition (le bloc orange sur la figure 3) présent dans le langage de programmation Scratch, préalablement à l'activité numérique.

La présentation de la séquence comporte des éléments narratifs qui visent à faire le lien avec le jeu des tours et à introduire le personnage du robot qui sera utilisé dans la séquence numérique ainsi que les règles du jeu.

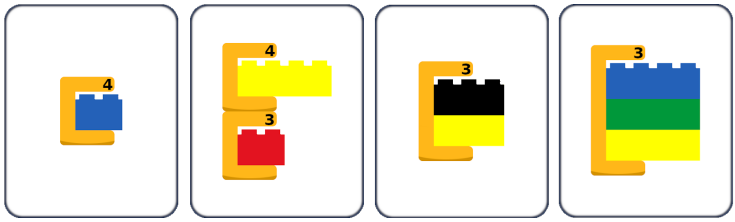


Figure 3 · Exemples de cartes du jeu « Le langage du robot »

Il s'agit d'un jeu collaboratif par équipe de trois à cinq dont le but est de construire des tours qui correspondent au message contenu sur une carte qui est fournie à l'équipe. Il est demandé que chaque élève construise une tour, mais pour gagner, il est nécessaire que toutes les tours construites par l'équipe soient identiques. Cette règle, qui rend le jeu collaboratif, invite les élèves à échanger entre eux sur la signification du message et à s'aider mutuellement. La validation est faite par l'équipe, en présence de l'expérimentatrice, en comparant les productions de l'équipe avec un modèle. Pour ce jeu, la phase de bilan a donc lieu au sein de l'équipe. L'objectif de celle-ci est de construire progressivement la signification du bloc de répétition à partir des réussites et erreurs des élèves.

Chaque carte est composée de deux types d'éléments graphiques, dont l'univers d'origine est différent. Les briques de construction sont représentées schématiquement. Pour cette représentation, de « *type iconique* » (Pierce, 1965, p. 143), une correspondance visuelle directe avec l'objet représenté est conservée. Chaque carte porte aussi une

représentation de « *type symbolique* » (Pierce, 1965, p. 167-168) du processus de répétition. Cette représentation ressemble à celle de l'élément de langage Scratch : la couleur orange et la forme ont été conservées, ainsi que l'écriture chiffrée du nombre d'itérations. En revanche, tous les éléments textuels ont été supprimés, ce qui rend cette représentation plus épurée et plus adaptée à des non-lecteurs.

3.4.3. Séquence 3 : Activité connectée « MOTIF ART – frises colorées »

Après deux activités débranchées, les élèves basculent sur une activité connectée sur tablette numérique avec l'environnement MOTIF ART. C'est un environnement de programmation épuré, dans un langage simplifié issu de Scratch (figure 4). Seuls des blocs codant les couleurs et le bloc de répétition sont disponibles. L'exécution d'une instruction (blocs d'action de coloriage) provoque le coloriage de la case courante et le déplacement automatique du robot d'une case vers la droite ou un retour à la ligne. Les cases sont donc automatiquement traitées dans un ordre qui correspond au sens de la lecture. Le fait de lier l'action de coloriage de la case où se trouve le robot, avec un déplacement vers la droite permet d'occulter les difficultés liées au repérage dans l'espace qui peuvent survenir avec un robot ou le déplacement d'un élément sur un quadrillage. Pour des raisons techniques, il n'était pas possible de présenter le modèle de coloriage et le résultat sous une forme verticale qui aurait rappelé la tour verticale et l'empilement des blocs de la partie droite. Si cela peut potentiellement poser problème à certains élèves, cette configuration présente néanmoins l'avantage d'éviter que les élèves ne reproduisent le modèle par une simple correspondance terme à terme.

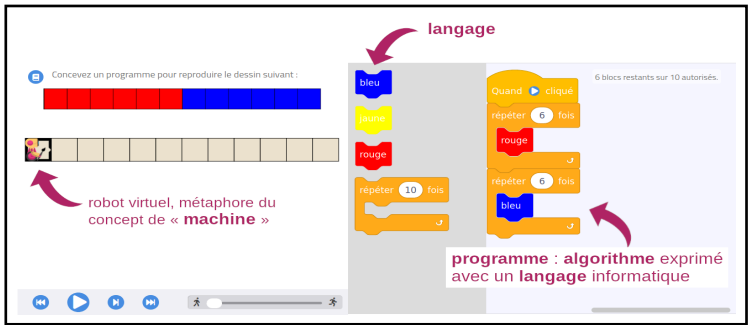


Figure 4 · Description des composantes de l'environnement MOTIF ART

Un bouton « lecture » permet à l'utilisateur de lancer l'exécution de son programme. L'exécution est jouée sur l'interface, avec une correspondance entre la case traitée et l'instruction exécutée qui est placée en surbrillance. Un mode pas à pas, qui permet d'exécuter une instruction à la fois, apporte un retour à l'élève. Il vise à faciliter le repérage et la correction des erreurs en autonomie (débogage).

L'interface a déjà été utilisée lors de l'étude exploratoire mais a été adaptée pour la rendre accessible aux non lecteurs. Ainsi, chaque bloc est de la couleur qu'il code, facilitant l'utilisation des éléments de ce langage formel simplifié.

L'objectif de la séquence est d'amener les élèves à identifier un motif redondant sur une séquence linéaire de couleurs et d'exprimer cette répétition de manière synthétique en utilisant le bloc *répéter*. Pour chaque problème, appelé « puzzle » dans ce contexte, la tâche de l'élève est donc de concevoir un programme afin de reproduire une frise colorée présente à l'écran.

Cette séquence s'appuie sur la progression pédagogique expérimentée lors de l'étude exploratoire, organisée en paliers matérialisés par les fonds colorés (figure 5). Le nombre de blocs pour concevoir le programme est limité, ce qui contraint l'utilisation du bloc *répéter* afin de réussir chaque puzzle.

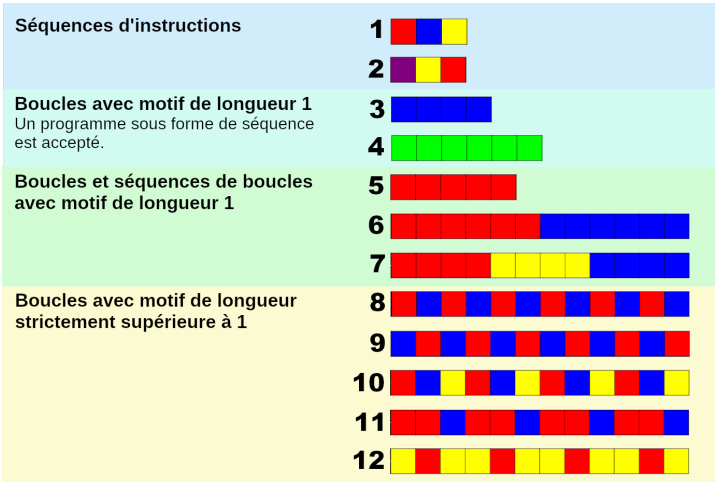


Figure 5 · Étapes conceptuelles de la progression pédagogique : séquence d'instructions simples, répétition avec une seule puis plusieurs instructions dans le corps de la boucle

La frise colorée peut être vue comme une tour couchée et est présentée comme telle aux élèves pour faire le lien avec l'activité débranchée. De plus, les briques de construction restent disponibles pour le cas où l'élève a besoin de refaire la manipulation de déconstruction de la tour pour identifier le motif redondant.

3.4.4. Séquence 4 : Activité connectée « MOTIF ART - mode créatif »

Dans cette activité, l'élève est libre de concevoir un programme dont l'exécution produit son propre dessin (figure 6). Dans un premier temps l'élève colorie complètement une grille de papier avec les couleurs qu'il souhaite. Dans un second temps, l'objectif consiste à reproduire le coloriage réalisé à l'écran en concevant un programme adéquat. Comme pour les frises de la section précédente, le déplacement du robot virtuel sur la grille, et en particulier le retour à la ligne, est géré automatiquement par le système. La grille est parcourue ligne par ligne, comme pour la lecture d'un texte.

Cette activité est mobilisée dans deux contextes différents : soit en limitant le nombre de blocs du programme afin de forcer l'usage de la répétition, soit sans imposer cette limite. Des grilles de différentes tailles sont proposées aux élèves (4x4, puis 6x6 et finalement 12x12). Par exemple, dans le cas de la grille 6x6 pour le mode avec nombre limité de blocs, seuls 20 blocs sont disponibles pour remplir les 36 cases.

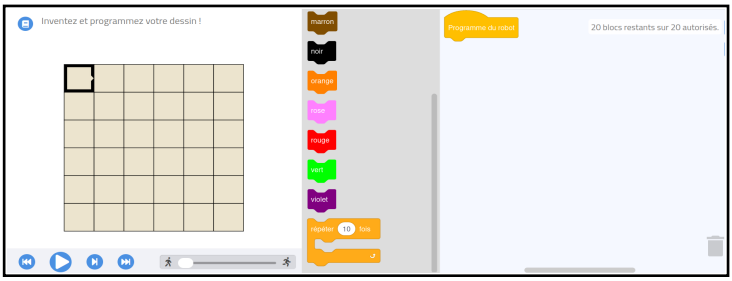


Figure 6 · L'interface de l'activité créative de MOTIF ART

4. Résultats de l'expérimentation

4.1. Activité du jeu des tours

La séquence du jeu des tours est jouée avec les quatre groupes, soit les 32 élèves présents lors de la première séance. La mise en place du jeu est

aisée, les consignes sont bien comprises et respectées, les élèves s'investissent dans l'activité.

Lors de la première itération de la phase de jeu, sur l'ensemble des 16 binômes (répartis sur les quatre groupes d'élèves), un seul réussit à reproduire la tour modèle. Dans chacun des quatre groupes, on couvre quasiment l'ensemble des types d'erreurs possibles (répertoriés sur la figure 7 avec le nombre d'occurrences). Certaines tours contiennent plusieurs types d'erreurs. Tous les binômes produisent une tour avec une alternance de couleurs, sauf dans un cas où le constructeur a joué avec les briques sans construire de tour.

	MODÈLE correctement reproduit	ERREURS			
		Différence de hauteur	Type de brique différent	Ordre du motif inversé	Couleurs différentes du modèle
Productions des élèves					
Nombre d'occurrences	1	9	7	6	3

Figure 7 · Analyse des erreurs observées lors du « jeu des tours »

La phase collective qui suit, après confrontation de la tour produite avec la tour modèle, permet l'explicitation des erreurs et fait émerger la notion « d'information utile ». Il ressort de cette phase collective une liste d'informations utiles à identifier par le messager et à transmettre au constructeur : modèle de briques, couleurs utilisées, nombre de briques de la tour, ordre des briques. Une deuxième itération du jeu se déroule avec des tours qui comportent des motifs de longueur 3, sauf pour un des groupes, plus en difficulté dans le repérage des informations utiles lors de la phase collective, et pour lequel nous sommes restés sur un motif de longueur 2. Pour les trois groupes pour lesquels le temps de séance restant

a permis une seconde itération du jeu, huit binômes sur les treize réussissent à reproduire la tour modèle.

La plupart des messages contiennent les informations précises qui permettent aux élèves de réussir la tâche. Voici un exemple de message relevé, qui correspond à la description de la tour du milieu (figure 3) : « *Tu prends des briques avec 4 "pitons"... des rouges, des bleues et des jaunes. Tu mets une jaune, rouge, une bleue, jaune, une rouge, une bleue. Tu fais ça jusqu'à 12.* ». Cependant, on remarque dans ce message que l'élève n'isole pas le motif. Le constructeur reproduit une frise où il s'agit de continuer une suite logique. De ce fait, c'est le nombre total de briques qui est indiqué et non le nombre de motifs.

Amener à isoler et dénombrer les motifs par la manipulation décrite à la fin de la section 3.4.1 est l'objectif de la phase collective qui suit immédiatement le jeu. Dix-neuf élèves sur les 26 qui ont participé à l'activité réussissent la manipulation (tableau 1). Les autres ont cassé leur tour pour retrouver les briques initiales. Le terme « motif » est introduit pour désigner un des morceaux obtenus. Les élèves sont alors invités à compter le nombre de motifs, ce qui est très concret puisque ce nombre de motifs correspond au nombre de morceaux. Ainsi, cette manipulation de déconstruction de la tour permet d'introduire de manière débranchée les notions de motifs et de répétition, avant de les aborder dans un environnement de programmation simplifié.

Tableau 1 · Synthèse des résultats lors des différentes phases de la séquence du « jeu des tours »

	Nombre d'élèves	**Mode de groupement**	**Réussite**
Jeu – Itération 1	32	Binôme	**1**/16
Jeu – itération 2	32	Binôme	**8**/13
Manipulation – Identification du motif	26	Individuel	**19**/26

L'analyse nous permet de conclure que la séquence du jeu des tours est pertinente chez ces élèves de 5-6 ans pour introduire des éléments que nous identifions comme relevant de la pensée informatique. Ils ont été initiés à la notion « d'information utile », à la nécessaire précision et à la non-ambiguïté de ces informations. Cette étape prépare l'introduction d'un

premier langage formel afin de communiquer avec une machine. Finalement, la séquence introduit l'identification de motifs redondants et leur dénombrement, ce qui constitue une première étape pour l'introduction de la notion de « répétition ».

4.2. Activité du jeu de cartes « *Le langage du robot* »

Pour l'ensemble des groupes, neuf équipes de 3 à 4 élèves ont été constituées. L'ensemble des élèves entre dans le jeu et tente de décoder le message qui est sur la carte pour construire une tour. Quelques élèves le font de manière individuelle sans tenir compte de la règle de collaboration. À quelques exceptions près, les élèves échangent sur la signification du message et se coordonnent pour la construction de la tour. Le tableau 2 reprend les observations et quelques commentaires entendus pendant l'activité.

Tableau 2 · Observations réalisées lors de la phase de décodage

 Répétition avec motif de longueur 1	Les élèves s'inscrivent rapidement dans l'activité. Pas de remarques particulières. Toutes les équipes réussissent facilement.
 Séquence de répétition avec motif de longueur 1	Quelques élèves produisent une alternance de couleurs (jaune, rouge, jaune, rouge…), interprétant le message en se référant au « jeu des tours ». La régulation se fait au sein de l'équipe : « *Ça ne peut pas, il n'y a pas pareil de briques* » ; « *Tu vois le premier crochet c'est que pour les rouges et l'autre c'est que pour les jaunes* ». Finalement, pour ce type de carte encore, l'ensemble des équipes réussit.
 Répétition avec un motif de longueur 2	Dans ce cas, la signification du message a été plus difficile à appréhender. Environ la moitié des élèves ne tient pas compte de la différence avec la carte précédente et produit une tour avec deux zones de couleur. Pour trois équipes, c'est la confrontation avec la tour modèle et l'étayage de l'adulte qui conduisent à comprendre l'erreur. Dans les autres équipes, les échanges entre élèves aboutissent à la solution correcte. « *Ça, on l'a déjà fait, c'était l'autre carte, là c'est pas pareil, les Lego ils sont collés* », « *C'est comme le morceau, quand on a cassé les tours, il faut mettre 3 morceaux comme ça.* » sont des arguments qui ont entraîné une évolution de l'équipe vers la construction de la tour attendue. Seule la moitié des équipes réussit la reconstruction de la tour. On retrouve de nouveau le palier identifié dans les expériences sur des élèves plus âgés (5-6 ans et 8-10 ans) se produisant lors du passage d'un motif de longueur 1 à un motif de longueur 2.

Les élèves passent d'une description verbale lors du jeu des tours au décodage de quelques éléments visuels d'un langage formel très simple lors de ce jeu de cartes. Seul le sens du décodage est abordé. C'est la mobilisation de ce bloc « répéter » au sein de l'environnement MOTIF ART lors de l'activité suivante qui nous renseignera sur le sens de l'encodage, c'est-à-dire sur l'identification et la synthèse de motifs redondants.

4.3. Problèmes rencontrés lors de l'introduction de l'environnement MOTIF ART et évolutions du scénario ludopédagogique

Pour un problème de logistique, seul un groupe de 9 élèves a bénéficié de la séance longue et s'est vu proposer la séquence MOTIF ART sur tablette lors de la première séance, comme prévu dans le scénario ludopédagogique initial.

Le déroulement de cette séquence nous a permis de relever un ensemble de problèmes. Pour pallier les difficultés recensées, nous avons revu plusieurs éléments du scénario ludopédagogique avant d'aborder la deuxième séance.

– Lors de la présentation du parcours MOTIF ART, il semble que l'ensemble des élèves du groupe comprend que la frise colorée à l'écran correspond à la tour de l'activité précédente, mais le curseur qui marque la position de la case à peindre sur la frise (figure 6) n'est pas mis en relation avec le rôle de constructeur. Il manque une évocation de la machine plus explicite. Une modification a consisté à remplacer ce curseur par un robot peintre. Le but de l'élève devient « *faire peindre au robot la même frise que le modèle* ». Ce robot virtuel anthropomorphe est une représentation métaphorique du concept de « machine » (figure 4) au sens de Dowek (2011), qui en facilite une première approche par les élèves de 5-6 ans.

– L'introduction du bloc répéter concomitante à la prise en main de l'interface est trop difficile. Même les élèves, qui ont réussi à identifier un motif dans le jeu des tours, ne semblent pas être en mesure de réinvestir cette compétence. Le jeu de carte « Le langage du robot » a donc été introduit en préalable à la séquence du parcours frises colorées de MOTIF ART (voir section 4.2).

– Le premier puzzle, dont la frise à reproduire comportait 12 cases de couleur identiques, nécessitait de dénombrer jusque 12 et de connaître l'écriture chiffrée de ce nombre, compétences qui se situent en limite du domaine numérique maîtrisé par la plupart des élèves de cet âge. Seuls quelques élèves ont réussi au moyen d'un étayage important. Ce qui n'était pas probant. Une série de puzzles plus faciles a donc été ajoutée au début du

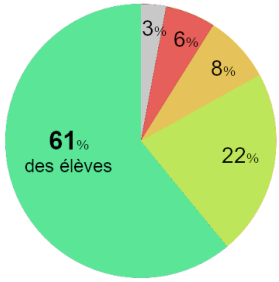
parcours, pour assurer la prise en main de l'environnement avant d'aborder la répétition, objet principal de notre étude.

L'analyse des traces numériques collectées a confirmé ces observations *in situ*. Le temps de prise en main de l'EIAH est relativement long lors de cette introduction de MOTIF ART avec la première version du scénario pédagogique : le temps moyen passé sur les deux premiers puzzles est respectivement de 8,17 min et 8,09 min, ce qui indique une entrée dans l'activité trop difficile.

4.4. Analyse des traces d'activités des élèves sur MOTIF ART

Pour évaluer si les modifications apportées rendent l'environnement plus accessible aux élèves de 5-6 ans, nous cherchons à estimer le temps de prise en main de l'environnement MOTIF ART par les 36 élèves lors de la séance 2 (figure 8). Pour cela, nous prenons en considération le temps passé sur le puzzle 1 (t1), le puzzle 2 (t2) et la différence entre les temps passés sur les puzzles 1 et 2 (d).

Lors de cette séance, 83 % des élèves prennent en main l'interface en moins de 5 minutes. On en déduit que les adaptations apportées à l'environnement MOTIF ART le rendent accessible aux élèves de 5-6 ans.



Prise en main aisée	t1 ≤ 1'30 d < 1'
Nécessité de quelques minutes sur le puzzle 1	t1 > 1'30 et t2 ≤ 1'30 1' < d ≤ 5'
Prise en main difficile mais effective après le puzzle 2	t2 ≤ 1'30 d > 5'
Prise en main non effective à l'issue du puzzle 2	t2 >1'30
Pas de soumissions sur les puzzles 1 et 2	NA

Figure 8 · Prise en main de l'environnement MOTIF ART lors de la séance 2

Nous considérons maintenant l'activité des élèves sur l'ensemble du parcours frises colorées de MOTIF ART pendant la séance 2. Le temps passé sur le parcours est d'environ 20 minutes, avec une variation de plus ou moins 5 minutes suivant les groupes.

Nous analysons les traces numériques de l'activité afin d'appréhender si le scénario ludopédagogique proposé permet aux élèves de s'initier aux notions de séquence d'instructions et de répétition.

La figure 9 montre le nombre d'élèves qui ont abordé chaque puzzle (en bleu) et parmi eux le nombre d'élèves qui ont réussi à résoudre le problème (en vert), et enfin le temps moyen passé sur chaque puzzle (en gris).

Les résultats pour les puzzles 1 à 4 montrent que la séquence de quelques instructions est accessible à quasiment l'ensemble des élèves de ce groupe.

À partir du puzzle 5, le nombre de blocs disponibles est limité, pour contraindre l'utilisation du bloc « répéter ». En cas de dépassement du nombre de blocs autorisé, un message clignote en rouge sur l'écran. Comme les élèves ne savent pas lire, l'expérimentatrice a explicité la signification de ce message. Ce passage à la boucle simple (une instruction dans le corps de la boucle), constitue un premier palier de difficulté. Vingt-quatre élèves, soit deux tiers d'entre-eux, valident le puzzle. Le temps moyen passé sur ce puzzle 5 augmente significativement. Le puzzle 6, qui introduit la séquence de boucles simples, n'est réussi que par un tiers des élèves. Sept élèves abordent le puzzle sans le valider. Nous sommes en présence d'un deuxième palier de difficulté. Peu d'élèves abordent la boucle avec un motif de longueur supérieure à deux (à partir du puzzle 8) lors de cette séance. Pour ces élèves, on observe un troisième palier de difficulté, avec un temps moyen passé sur le puzzle qui augmente à nouveau.

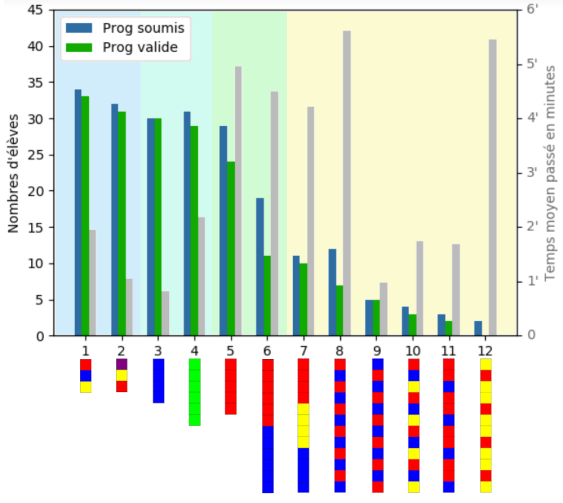


Figure 9 · Nombre d'élèves, parmi les 36, ayant soumis (bleu) et validé (vert) un programme pour chaque puzzle et temps moyen passé (gris)

Nous retrouvons les paliers de difficulté déjà identifiés lors des expérimentations Chticode (Peter *et al.,* 2019) et MOTIF..MOTIF.. 6-7 ans (Léonard *et al.,* 2020) quel que soit le parcours utilisé.

4.5. Analyse des productions réalisées lors de l'activité créative

La même activité créative a été proposée deux fois aux élèves, lors de la dernière séance d'apprentissage, puis un mois plus tard. Chaque séance a duré entre 35 et 45 minutes, suivant les groupes, et a été divisée en deux phases : le coloriage d'une grille papier dans le but de faire programmer son propre dessin au robot, puis la programmation de ce dessin dans l'environnement MOTIF ART.

Lors de la séance 3, le nombre de blocs est limité, ce qui contraint l'utilisation du bloc « répéter » pour réussir à remplir la grille. Une première grille de taille 4x4 est fournie aux élèves. Ceux qui ont fini rapidement ont réitéré l'activité sur une grille de taille 6x6. Lors de la séance différée, il s'agit d'appréhender si les élèves remobilisent la répétition. L'activité a lieu avec une grille de taille 6x6 et la limitation du nombre de blocs a été enlevée, seule différence entre les deux séances.

Le tableau 3 ci-dessous analyse finement le type de remobilisation des concepts entre la séance 3 et la séance 4 portant sur l'activité créative. Il se lit de la manière suivante : parmi les onze élèves ayant utilisé, en séance 3, un motif de longueur >1, trois l'ont remobilisé en séance 4 (nombre en vert), un a tenté sans y parvenir (nombre en rouge), quatre ont remobilisé un motif de longueur =1, etc.

Tableau 3 · Type de remobilisation spontanée des notions entre la séance 3 (en ligne) et la séance 4 (en colonne)

	Validé en séance 3	motif>1 séance 4	motif=1 séance 4	séquence séance 4
motif>1	11	3 / 1	4 / -	1 / 2
motif=1	13	- / 2	6 / 2	2 / 1
séquence d'instructions	1		- / 1	
	25	3 / 3	10 / 3	3 / 3

On observe la répartition suivante de remobilisation des concepts :

– 19 élèves remobilisent la notion de « répétition », soit 73 % d'entre eux.

– 13 élèves qui ont remobilisé la répétition ont réussi à reproduire le coloriage qu'ils avaient réalisé, soit 52 % des participants. Parmi ceux-ci, 3 ont utilisé une répétition avec un motif de longueur supérieure à 1, soit 12 % des participants.

– Fait intéressant : 2 élèves qui n'avaient pas atteint les motifs de longueur supérieure à 1 ont tenté leur usage lors de la séance différée.

Ces informations permettent d'affirmer qu'une remobilisation de la notion de « répétition » a été réalisée pour la moitié de l'effectif, ce qui est un résultat notoire. On constate de nouveau le palier de difficulté qui a été identifié lors des précédentes expérimentations lors du passage d'une à plusieurs instructions dans le corps de la boucle. Néanmoins, à l'issue de cette progression pédagogique, environ 1 élève de 5-6 ans sur 10 a franchi ce palier de difficulté dans ce contexte sans repérage spatial.

5. Appropriation des activités par des enseignants dans le cadre d'une action de formation continue

Le scénario ludopédagogique décrit dans les sections précédentes a été adapté dans le cadre d'une formation continue à destination d'enseignants et enseignantes d'école maternelle. Ce module a été réalisé pour le compte de la Maison pour la Science Nord Pas-de-Calais entre septembre 2019 et juin 2020 dans le cadre d'une action territoriale impliquant la circonscription de Condé-sur-Escaut. Cela représente une soixantaine de professeurs des écoles répartis sur les cycles 1, 2 et 3.

L'objectif de la formation était avant tout de faire évoluer les représentations biaisées de l'informatique, d'introduire les concepts structurants de l'informatique et de proposer des activités déjà testées avec des élèves de leur niveau afin de leur en faciliter l'appropriation.

Pour ce module, nous mettons en œuvre la démarche d'investigation portée par cette structure affiliée à la Fondation *La main à la pâte*. Après un recueil et un classement des représentations initiales sur l'informatique qui visent à démystifier le domaine, nous mettons les enseignants en situation d'expérimenter les séquences destinées à leurs élèves, soit à l'identique, soit avec une autre modalité. Par exemple, le passage d'informations pour le jeu des tours se fait à l'écrit, afin d'analyser a posteriori la pertinence des informations transmises. Cette investigation des activités est enrichie par un éclairage sur les concepts en jeu dans l'activité en cours et dans le scénario ludopédagogique à destination des élèves.

Nous pensons que l'introduction des concepts piliers de l'informatique (Dowek, 2011) d'abord en situation, de manière contextualisée, puis l'invitation à se replacer ensuite dans une posture réflexive qui prend en compte la transposition vers une pratique de classe, est de nature à favoriser l'appropriation des séquences proposées.

Ce module de formation continue de 9 heures a été déployé à l'échelle d'une circonscription. Il comporte trois phases. Lors d'une première séance de trois heures de formation en présentiel, les enseignants et enseignantes expérimentent le scénario *MOTIF..MOTIF.. 5-6 ans* dans le cadre de leur propre formation (figure 10).

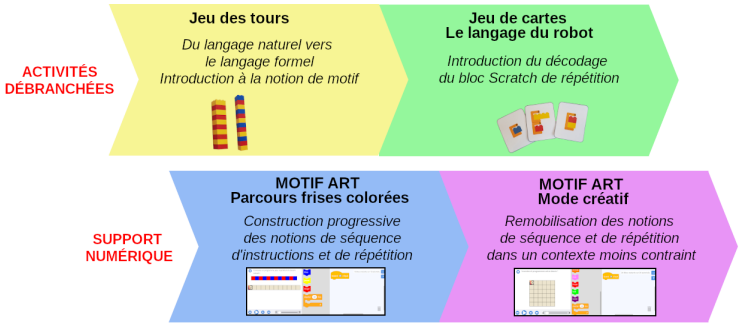


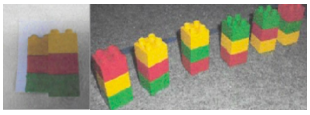
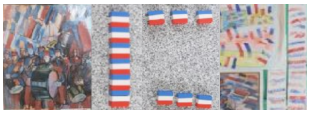
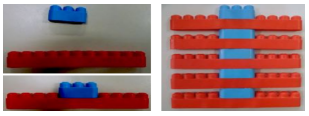
Figure 10 · Scénario *MOTIF..MOTIF..*

Trois autres heures sont réservées pour la préparation, la mise en place et un retour d'expérience d'une séquence d'initiation à la pensée informatique dans leur classe. Les trois dernières heures, à nouveau en présentiel, sont consacrées à la présentation des retours d'expériences et à leur discussion entre collègues et avec les formateurs et formatrices. Dans le même esprit que la première, cette séance est complétée par une investigation des séquences proposées aux élèves d'école élémentaire (expérimentation *Chticode*).

Lors de la phase de retour d'expériences, nous avons recueilli des traces d'activités : fiches de préparation, photos d'élèves en activité. Celles collectées auprès de la vingtaine d'enseignants et enseignantes de cycle 1 (tableau 4) portent sur la reconstruction d'un objet ou la recherche de motifs différents. Du matériel courant dans les classes de maternelle a été mobilisé : jeux de construction, autres jeux pédagogiques détournés de leur usage prescrit. Ces traces montrent une appropriation des séquences d'activités débranchées sur la notion de « motif », même avec des élèves encore plus jeunes que ceux qui

ont participé à l'expérimentation analysée précédemment. En revanche, les séquences avec support numérique, n'ont pas été remobilisées. Questionnés à ce sujet, les enseignants et enseignantes nous ont rapporté un équipement insuffisant ou obsolète et une connexion à internet de mauvaise qualité.

Tableau 4 · Traces des activités réalisées par les professeurs des écoles avec leur classe entre les deux séances en présentiel de la formation

Nature de l'activité	Âge (classe)	Matériel utilisé
Reproduire un objet sans voir le modèle. Rechercher des motifs différents à partir d'éléments de départ identiques	3-4 ans (petite section)	
	3-4 ans (petite section)	
	4-5 ans (moyenne section)	
Reproduire un modèle à partir de motifs déjà construits à choisir parmi un ensemble	4-5 ans (moyenne section)	
Identifier un motif dans une œuvre d'art, l'isoler et le reproduire	4-6 ans (moyenne et grande section)	
Rechercher des motifs différents à partir des mêmes éléments, répéter un motif pour reproduire une structure à l'identique	5-6 ans (grande section)	
Produire un motif, puis le répéter à l'identique	5-6 ans (grande section)	

6. Conclusion et perspectives

La réintroduction d'éléments d'informatique dans l'ensemble des cycles scolaires en France présente d'importants défis sur la formation des enseignants et la création de séquences pédagogiques pour les élèves des différents niveaux. Cet article présente les travaux que nous menons afin de répondre en partie à ces problématiques en proposant une démarche conjointe de création de scénario ludopédagogique d'initiation à l'informatique pour de jeunes élèves et de formation de leurs enseignants et enseignantes.

Nous nous interrogeons particulièrement sur la capacité des élèves de 5-6 ans à s'approprier la notion de « répétition » et sur les activités, débranchées et numériques, susceptibles de supporter efficacement cet apprentissage. Nous nous sommes également intéressés à l'appropriation d'une telle séquence pédagogique par les enseignants et enseignantes.

Le scénario ludopédagogique présenté s'adressait à des élèves non-lecteurs âgés de 5 à 6 ans et avait pour objectif de les initier au raisonnement algorithmique de base : les notions de langage formel, de séquence d'instructions et un focus particulier sur la notion de « répétition ». Lors de cette initiation, nous avons occulté les aspects spatiaux, difficulté identifiée dans des travaux précédents, afin de concentrer l'activité sur le développement de la capacité de reconnaissance de motifs visuels redondants. Pour cela, nous avons proposé un scénario ludopédagogique débutant par des activités débranchées, pour que les élèves découvrent les notions dans un contexte non-technique et avec un matériel connu. Puis, nous les avons exposés à un renforcement de ces notions dans un environnement simplifié de programmation en ligne. Le même type d'activité que celles réalisées en débranché y est repris mais nécessite l'encodage d'un algorithme dans un langage de programmation par blocs. La progression dans les problèmes à résoudre a débuté par de simples séquences d'instructions et s'est achevée sur des répétitions avec plusieurs instructions dans le corps de la boucle. Finalement, pour estimer l'appropriation du concept de répétition, les élèves ont réalisé une dernière séance, différée dans le temps, qui portait sur une activité créative qui nous a permis de quantifier précisément les concepts qu'ils ont remobilisés spontanément.

En parallèle de cette expérimentation, une action de formation continue impliquant une soixantaine d'enseignants et enseignantes du premier degré a été réalisée afin de les initier aux mêmes bases et de leur donner la possibilité de mettre en œuvre les séquences proposées. Leurs

retours d'expérience ont conduit à faire évoluer les activités et les supports. Cette démarche est nécessaire afin d'optimiser l'appropriation des concepts fondamentaux par les enseignants et enseignantes et faciliter la mise en œuvre des activités dans les classes. Ainsi, la formation ancre les concepts fondamentaux proposés par Dowek, en insistant sur l'importance du langage et les met en confiance avec un scénario ludopédagogique éprouvé et évoluant en fonction de leurs retours.

Les résultats d'analyse des observations réalisées en milieu écologique et des traces d'activité des élèves sur la plateforme sont significatifs et indiquent une appropriation du concept de répétition attestée par la remobilisation spontanée de répétitions lors de l'activité créative pour la moitié de l'effectif. Au niveau des enseignants et enseignantes formés, les retours attestent d'une réelle appropriation et de mises en œuvre originales et pertinentes, ce qui illustre leur compréhension des apprentissages fondamentaux en jeu lors des séquences proposées.

La prochaine étape consisterait à observer et collecter les résultats obtenus par les élèves guidés par leurs enseignants au regard de leurs propres séquences ludopédagogiques. Ce travail, entravé en 2020 par la crise sanitaire, sera mené prochainement.

Aussi, un travail important reste à réaliser au niveau de la mesure plus précise en amont et en aval des compétences algorithmiques des élèves. Les expérimentations menées avec les pré-tests et post-test n'ont pas été concluantes, mais ont permis d'aboutir à l'ajout de la séance différée avec l'activité créative qui fournit, malgré tout, des résultats intéressants. Le faible effectif et l'absence de groupe témoin est aussi une préoccupation qui nous incite à nuancer les résultats obtenus. Cependant, nous disposons maintenant d'une séquence éprouvée nous permettant de préparer une expérimentation à plus large échelle afin de confirmer ou infirmer les premiers résultats détaillés dans cet article.

REMERCIEMENTS

Ce travail est soutenu par le projet Interreg Teach Transition (https://teachtransition.eu) et par l'INSPÉ Hauts de France. Nous tenons aussi à remercier l'association France-IOI pour la mise à disposition de la plate-forme, l'ensemble des acteurs et actrices institutionnelles (enseignants et enseignantes, ERUN, conseillères pédagogiques, inspecteurs et inspectrices), ainsi que l'ensemble des élèves ayant participé à ces activités.

RÉFÉRENCES

Aggarwal, A., Gardner-McCune, C. et Touretzky, D. S. (2017). Evaluating the effect of using physical manipulatives to foster computational thinking in elementary school. Dans *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education - SIGCSE'17* (p. 9-14). https://doi.org/10.1145/3017680.3017791

Alvarez, J. (2018). La ludopédagogie. *Lectures.Cultures, 10*, 29-31.

Alvarez, J., Bellegarde, K., Flahaut, J.-J. et Lafouge, T. (2018, juillet). *Blue Bot Project Experiment* [communication orale]. 8th International Toy Research Association World Conference. https://hal-univ-paris13.archives-ouvertes.fr/hal-02090864

Bellegarde, K., Boyaval, J. et Alvarez, J. (2019). S'initier à la robotique/informatique en classe de grande section de maternelle. Une expérimentation autour de l'utilisation du robot Blue Bot comme jeux sérieux. *Review of Science, Mathematics and ICT Education, 13*(1). https://pasithee.library.upatras.gr/review/article/view/3105

Boytchev, P. (2014). *Logo Tree Project*. Studylib.Net. https://studylib.net/doc/5691847/logo-tree-project

Brackmann, C. P., Román-González, M., Robles, G., Moreno-León, J., Casali, A. et Barone, D. (2017). Development of computational thinking skills through unplugged activities in primary school. Dans *Proceedings of the 12th Workshop on Primary and Secondary Computing Education. WiPSCE'17* (p. 65-72). https://doi.org/10.1145/3137065.3137069

Catlin, D. et Blamires, M. (2019). Designing Robots for Special Needs Education. *Technology, Knowledge and Learning, 24*(2), 291-313. https://doi.org/10.1007/s10758-018-9378-8

Catlin, D. et Woollard, J. (2014). Educational robots and computational thinking. Dans Proceedings of the *Robotics in Education (RIE) 2014 Conference*. https://eprints.soton.ac.uk/365505/

Denis, B. (1987). Technologie de contrôle et LOGO : la robotique, ses enjeux, ses modalités. *Éducation : Tribune Libre, 208*, 61-67.

Dowek, G. (2011). Les quatre concepts de l'informatique. Dans *Actes du colloque Sciences et technologies de l'information et de la communication en milieu éducatif : Analyse de pratiques et enjeux didactiques* (p. 21-29). https://edutice.archives-ouvertes.fr/edutice-00676169/document

Drot-Delange, B., Pellet, J. P., Delmas-Rigoutsos, Y. et Bruillard, É. (2019). Pensée informatique : points de vue contrastés. *Sciences et Technologies de l'Information et de la Communication pour l'Éducation et la Formation, 26*(1), 39-61. http://sticef.org/num/vol2019/26.1.1.drot-delange/26.1.1.drot-delange.htm

Hochet, Y. (2013). *Evaluer le Serious Gaming : L'expérience autour de Sim City*, e-virtuoses 2013, Valenciennes, France.

Jarvinen, E.-M. (1998). The Lego/Logo Learning Environment in Technology Education: An Experiment in a Finnish Context. *Journal of Technology Education, 9*(2), 47-59.

Keymeulen, R. (2016). La ludopédagogie : que se cache t-il derrière ce terme ? *Blogue Intelligences multiples.*

Komis, V. et Misirli, A. (2011). Robotique pédagogique et concepts préliminaires de la programmation à l'école maternelle : Une étude de cas basée sur le jouet programmable Bee-Bot. Dans *Actes du colloque Sciences et technologies de l'information et de la communication en milieu éducatif : Analyse de pratiques et enjeux didactiques* (p. 271-281). https://edutice.archives-ouvertes.fr/edutice-00676143

Komis, V., Touloupaki, S. et Baron, G.-L. (2017). *Une analyse cognitive et didactique du langage de programmation ScratchJr.* Presses Universitaires de Namur.

Krumholtz, N. (1998). Simulating Technology Process to Foster Learning. *Journal of Technology Studies, 24*(1), 6-11.

Léonard, M., Peter, Y. et Secq, Y. (2020). Reconnaissance et synthèse de motifs redondants avec des élèves de 6-7 ans MOTIFS.MOTIFS.MOTIFS. 3 x MOTIFS. 3 x MOTIFS. *Dans Actes du colloque DIDAPRO 8-DIDASTIC– L'informatique, objets d'enseignements enjeux épistémologiques, didactiques et de formation.* https://hal.univ-lille.fr/hal-02971775

Lurçat, L. (1979). L'enfant et l'espace : le rôle du corps. PUF.

Nogry, S. (2019). Robotique pédagogique à l'école primaire : Quelle activité des élèves de Classe Préparatoire (6-7 ans) et quels apprentissages dans une séquence conçue par l'enseignant ? *Review of Science, Mathematics and ICT Education, 13*(1), 93-110. https://doi.org/10.26220/rev.3121

Ocko, S. et Resnick, M. (1987). *Integrating LEGO with LOGO. Making connections with computers and children.* The Media Laboratory, MIT.

Olmo-Muñoz, J., Cózar-Gutiérrez, R. et González-Calero, J. A. (2020). Computational thinking through unplugged activities in early years of Primary Education. *Computers & Education, 150,* 103832. https://doi.org/10.1016/j.compedu.2020.103832

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas.* Basic Books, Inc.

Peter, Y., Léonard, M. et Secq, Y. (2019). Reconnaissance de motifs et répétitions : Introduction à la pensée informatique. Dans *Actes du colloque Environnements Informatiques pour l'Apprentissage Humain.* https://hal.archives-ouvertes.fr/hal-02151035

Pierce, C. S. (1965). *Collected Papers II.* Harvard University Press.

Romero, M., Duflot-Kremer, M. et Viéville, T. (2018). Le jeu du robot : Analyse d'une activité d'informatique débranchée sous la perspective de la cognition incarnée. *Review of science, mathematics and ICT education, 13*(1), 35-49. https://hal.inria.fr/hal-01950335

Romero, M., Lille, B., Viéville, T., Duflot-Kremer, M., de Smet, C. et Belhassein, D. (2018). Analyse comparative d'une activité d'apprentissage de la programmation en mode branché et débranché. Dans *Actes de Educode - Conférence internationale sur l'enseignement au numérique et par le numérique.* https://hal.inria.fr/hal-01861732

Saxena, A., Lo, C. K., Hew, K. F. et Wong, G. K. W. (2020). Designing unplugged and plugged activities to cultivate computational thinking: An Exploratory study in early childhood education. *The Asia-Pacific Education Researcher, 29*(1), 55-66. https://doi.org/10.1007/s40299-019-00478-w

Touloupaki, S. et Baron, G.-L. (2019). *Apprendre à programmer à l'école primaire ? Une approche exploratoire en cycle 2*. Presses Univ. Septentrion.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35. https://doi.org/10.1145/1118178.1118215

Manipulations de robots programmables en classe par des élèves de 9-10 ans. Éducation au numérique et culture technique

▶ **Olivier GRUGIER** (Université Paris-Cité, Laboratoire EDA)

■ **RÉSUMÉ** • L'apprentissage de l'informatique a pris place dans les programmes d'enseignement de l'école primaire et avec lui, des robots programmables sont introduits dans les classes. Ces artefacts permettent également de construire une culture technique. Dans cet article, les activités de nature technique des élèves d'une classe de CM1 sont analysées. Cette analyse permet de révéler les schèmes mis en œuvre par les élèves pour s'approprier le robot BeeBot® puis un robot plus complexe, le ProBot®.

■ **MOTS-CLÉS** • éducation, apprentissage, robotique, activité, enfant, technologie.

■ ***ABSTRACT*** • *Digital education has taken place in elementary school curricula and with it, programmable robots are being introduced into school. These artefacts also allow acquiring a technical culture. In this article, the technological activities of 10 year olds in a class are analyzed. This analysis makes it possible to reveal the processes and approaches implemented by the children to appropriate the BeeBot® robot then a more complex robot, the ProBot®.*

■ ***KEYWORDS*** • *education, learning, robotic, activitie, children, technology.*

1. Introduction

Depuis 2016, l'apprentissage de l'informatique a pris place dans les programmes d'enseignement de l'école primaire (Vandevelde et Fluckiger, 2020). Devenu enjeu sociétal, cet enseignement vise à éduquer les citoyens en leur donnant les moyens de comprendre les traitements réalisés par les systèmes qu'ils utilisent et à initier chacun à des notions en informatique (Baron et Drot-Delange, 2016).

Cette recherche a été réalisée dans le cadre du projet ANR « IEcare » (http://iecare.lip6.fr/), portant sur l'étude de la conception et de la mise en œuvre de scénarios pédagogiques par des enseignants. En France, quelques professeurs des écoles choisissent d'utiliser des robots programmables en vue de faire acquérir des connaissances à la fois en technologie et en informatique aux élèves. Nous présentons dans cet article une étude de cas portant sur l'analyse d'un scénario pédagogique conçu par un enseignant avec deux robots programmables dans une classe de CM1.

Après un bref rappel historique concernant l'introduction de robots programmables dans des classes pour des visées d'apprentissage de l'informatique, une comparaison des deux robots BeeBot® et ProBot® utilisés dans cette classe de CM1 permettra d'identifier à la fois les points communs et les différences d'un point de vue technologique. La quatrième partie présente le cadre théorique à partir duquel les questions de recherche seront posées. Une fois la méthodologie développée, les résultats seront exposés avant de terminer sur des discussions et des perspectives de recherche.

2. Des robots programmables dans les classes

Depuis 1985 avec la présentation du Plan Informatique pour Tous (IPT) et en faisant suite aux travaux de Papert (1980) et d'Harel et Papert (1991), des robots programmables de sol pilotés ont été introduits dans des classes. Ce plan politico-scientifique amorcé dès les années 1970 avec des recherches exploratoires en éducation (Baron et Bruillard, 1996) traduit une volonté institutionnelle d'équiper de nombreux établissements scolaires par l'État.

Après l'expérience LOGO plusieurs fournisseurs de matériel (Jeulin, Valiant, MB, Swallow...) vont développer des robots de plancher avec des présentations et des caractéristiques variées, comme le montre Greff (1999). Certains seront distribués dans les écoles françaises, notamment les tortues T2, T3 et le robot Bigtrak.

Les premiers robots diffusés se présentent sous la forme d'un dôme transparent (figure 1). Ils sont programmables soit par l'intermédiaire d'un ordinateur raccordé à ce robot, soit par l'intermédiaire d'un lecteur de cartes perforées. Cependant, l'approche pédagogique n'est pas toujours intuitive. En effet, comme le soulignent plusieurs chercheurs (Grugier et Villemonteix, 2017 ; Komis et Misirli, 2015), la mise en œuvre des dispositifs expérimentaux en classe n'était pas toujours évidente à cause de la complexité du matériel informatique.

En classe, par des actions manipulatoires, les élèves arrivaient à générer des déplacements du robot. Cependant, étaient-ils capables d'analyser le fonctionnement du robot pour faire un lien entre les actions sur ce dernier et les déplacements engendrés ? Difficile à dire puisque les travaux de recherche ne se sont pas intéressés aux questions de compréhension du fonctionnement de ce système constitué du robot, de cartes de programmation et du boîtier de transfert.

Le robot Bigtrak, représentant un véhicule d'exploration lunaire, disposait d'un pupitre de programmation sur son dos. Par rapport aux robots précédents, il semble plus aisé de générer des déplacements après avoir programmé des instructions. Des expérimentations ont été menées dans des classes avec le Bigtrak comme en atteste un article relatant un projet informatique dans une école maternelle (Allari, 1986). Combes-Trithard (1984) relate que, suite à l'utilisation de ce robot, les élèves ont développé le repérage dans l'espace, un registre de langage, la rigueur et la capacité à ordonner des informations. Aucune mention n'est faite concernant des difficultés rencontrées par les élèves avec le Bigtrak. Cependant, au regard de la solution retenue pour le programmer, qui est proche des robots contemporains notamment le BeeBot®, il est probable que de jeunes élèves ont pu avoir des difficultés à comprendre la notion de stockage des instructions dans une mémoire difficilement perceptible (Grugier, 2021).

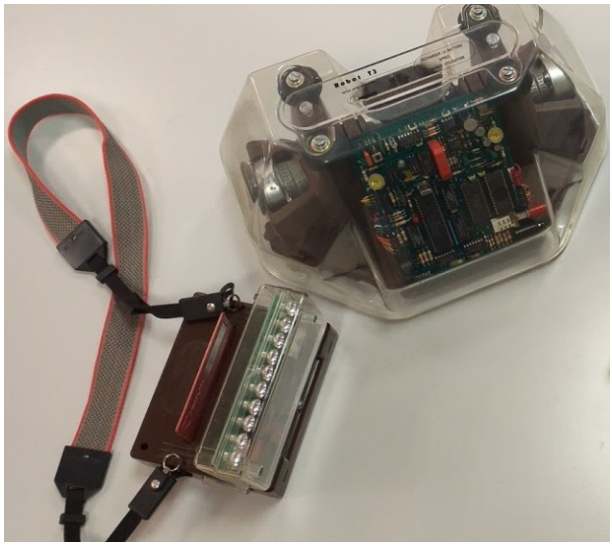


Figure 1 · Le robot T3 (tortue) et son boîtier de lecteur de carte et de transfert

Trois décennies plus tard, la miniaturisation des composants électroniques a permis de réduire les dimensions des robots tout en proposant un habillage propice à une utilisation scolaire. La robustesse de ces nouveaux robots autorise une manipulation hasardeuse propice à une utilisation dans des classes. De plus, « *l'évolution des langages et le développement d'environnements de programmation* » (Nogry, 2020) favorise la mise en œuvre de scénarios pédagogiques où les élèves apprennent à contrôler des robots à travers un langage spécifique (Bers *et al.,* 2014 ; Komis et Misirli, 2015 ; Komis *et al.*, 2017). Ces nouveaux robots laissent supposer une prise en main aisée et une plus grande compréhension du fonctionnement par les élèves.

3. Deux robots : des points communs et des divergences

Dans la suite de cet article, nous nous intéresserons à la culture technique pouvant être développée par l'utilisation des deux robots BeeBot® et ProBot® dans une classe de CM1. L'idée de culture technique à l'école obligatoire est « *celle d'appropriation de techniques comme composante de toute éducation générale* » (Martinand, 2014). L'utilisation de robots programmables nécessite l'appropriation de techniques.

Nous identifierons dans un premier temps les points communs et les divergences entre ces deux robots. Pour cela, nous distinguons plusieurs niveaux de connaissances qui peuvent être construites avec ces robots. Nous retenons les quatre niveaux de connaissance de Combarnous :

– « *la connaissance coutumière - perception est la familiarité avec l'objet ;*

– *la connaissance fonctionnelle est la compréhension de la fonction de l'objet ;*

– *la connaissance technologique est la connaissance du dessinateur, du monteur, du réparateur qui connaissent avec exactitude tous les éléments de l'objet ;*

– *la connaissance raisonnée correspond à une connaissance rationnelle et globale ; elle réunit donc des informations sur la fonction et le but de l'objet, des compréhensions des technicités qu'il renferme dans sa conception comme dans sa réalisation, des observations sur la place de l'objet dans le monde matériel, des réflexions sur son impact sur la société et sur les individus.* » (Combarnous, 1984, p. 234-235).

La connaissance coutumière et la connaissance fonctionnelle s'inscrivent dans les attentes des programmes scolaires pour un enseignement obligatoire à l'école en France. Par contre, la connaissance technologique et la connaissance raisonnée peuvent être attendues dans le cadre de formations professionnelles spécifiques. Pour identifier les connaissances fonctionnelles, il convient d'abord de clarifier le fonctionnement des robots (Denhière et Baudet, 1992). Ainsi, l'analyse fonctionnelle permet d'identifier les fonctions techniques retenues dans la conception des objets.

Les robots BeeBot® et ProBot® se caractérisent par une apparence robuste et ludique dont les déplacements peuvent se programmer par des actions successives sur des boutons situés au niveau d'un pupitre de commande qui est intégré. La fonction principale de ces robots est de pouvoir générer une succession de déplacements à partir d'une séquence composée de plusieurs instructions.

Figure 2 · Le robot BeeBot® à gauche et le robot ProBot® à droite

Différentes fonctions secondaires sont présentes. Pour chaque fonction, les concepteurs ont choisi différentes solutions techniques. Elles sont parfois identiques pour les deux robots et parfois différentes. Pour ces robots, deux procédures sont à mettre en œuvre avant de pouvoir générer un déplacement. La première consiste à actionner les interrupteurs positionnés sous le robot. L'un permet d'agir sur le circuit électrique pour l'alimentation en énergie des organes de commande et le second commande le circuit électrique alimentant le buzzer pour l'émission des sons. La seconde procédure consiste à programmer avec l'interface une séquence d'instructions en appuyant successivement et dans un ordre défini sur les boutons du clavier.

La comparaison des solutions techniques retenues pour répondre aux fonctions montre de nombreuses similitudes, notamment les modes de déplacements des robots sur le sol et les procédures de programmation. Ainsi, de nombreux points communs sont présents entre ces deux robots, ce qui permet d'émettre l'hypothèse suivante : après avoir découvert et généré un programme de déplacement avec un BeeBot®, un élève peut appréhender le ProBot® pour ensuite générer un programme de déplacement.

Cependant, des solutions techniques différentes sont présentes entre le robot BeeBot® et le ProBot® pouvant soulever des difficultés dans l'appropriation. Le ProBot® offre la possibilité de définir la longueur du pas de déplacement ainsi que l'angle de rotation. De plus, les lignes d'instructions du programme sont affichées sur un écran offrant la possibilité de les modifier (tableau 1). Il est également possible de programmer des boucles pour répéter des séquences d'instructions.

Tableau 1 · Différentes solutions techniques choisies pour répondre aux fonctions techniques entre les deux robots

Fonctions	Solutions choisies pour le BeeBot®	Solutions choisies pour le ProBot®
Programmer des actions	Pupitre composé de **7 boutons**	Pupitre composé de **23 boutons**
Visualiser les actions saisies		Un écran à cristaux liquides
Effacer un programme ou une ligne	Un bouton « X » permet **d'effacer l'ensemble des instructions**	Un bouton « clear » permet **d'effacer au choix une ligne d'instruction ou l'ensemble du programme**
Détecter un obstacle		Deux capteurs de contacts : un à l'avant et un à l'arrière
Détecter de la lumière		Un capteur de lumière
Détecter du bruit		Un capteur sonore
Se déplacer en ligne droite	En appuyant une fois sur le bouton flèche (devant ou derrière) du pupitre puis sur Go, **le robot se déplace d'un pas de 15 cm**.	En appuyant une fois sur le bouton flèche (devant ou derrière) du pupitre puis sur Go, **le robot se déplace d'un pas de 25 cm**. En appuyant une fois sur le bouton flèche puis sur les touches du clavier numérique et ensuite Go, **le robot se déplace de la longueur saisie**.
Effectuer une rotation	En appuyant une fois sur le bouton flèche (droite ou gauche) du pupitre puis sur Go, **le robot pivote d'un angle de 90°**.	En appuyant une fois sur le bouton flèche (droite ou gauche) du pupitre puis sur Go, **le robot pivote d'un angle de 90°**. En appuyant une fois sur le bouton flèche (droite ou gauche) puis sur les touches du clavier numérique du pupitre et ensuite Go, **le robot pivote de l'angle saisi (en degré)**.
Programmer une boucle		Le bouton « Rep » permet de répéter des commandes de déplacement.

Contrairement à de nombreux travaux (De Michele *et al.*, 2008 ; Highfield *et al.*, 2008 ; Komis et Misirli, 2012), nous ne nous intéressons pas à analyser le rôle de ces robots dans le développement, chez les élèves, de capacités de résolution de problèmes, de raisonnement logique ou encore de numération, mais à analyser les schèmes élaborés par les élèves pour appréhender le fonctionnement du robot BeeBot® puis du ProBot®. Ainsi, quelles sont les capacités d'action et les connaissances fonctionnelles développées par des élèves de début de cycle 3 concernant le fonctionnement du robot BeeBot® ? Ces capacités et connaissances fonctionnelles sont-elles transposables pour prendre en charge un autre robot programmable plus complexe du point de vue des solutions techniques proposées ?

4. Cadre théorique pour analyser les activités des élèves

L'introduction de robots programmables dans des classes du primaire favorise la construction d'une éducation au numérique (Baron, 2018) et à la technologie. Cette éducation permet, par la présence de technologie, l'acquisition d'une expérience informatique avec pour enjeu, comme le précise Bruillard (2016), « *de commencer à maîtriser une technologie de travail : développer également des capacités d'action sur le monde et de compréhension de ce monde, des valeurs autour du travail individuel et collectif* ». Ainsi, la construction d'une éducation au numérique ne peut se réduire à de la programmation. L'éducation au numérique est une structure composée de trois ensembles, science de l'informatique, culture technique, culture citoyenne, qui selon Baron (2018) forment différentes interactions. C'est la culture technique et les capacités d'action développées dans un milieu spécialement aménagé pour la rencontre avec les robots qui sont questionnées dans cet article.

L'analyse de séquences pédagogiques mises en place avec les robots, par des enseignants peu ou pas formés à un enseignement de l'informatique met en jeu à la fois des contenus en informatique (séquence d'instructions, programmation) et en technologie (comprendre le fonctionnement de l'objet, identifier les actionneurs) comme le montre Grugier et Nogry (Grugier et Nogry, 2021). Ces objets ont un double statut (Lebeaume, 2019) : objet pour apprendre des notions de programmation et objet à apprendre (c'est-à-dire apprendre son fonctionnement pour agir efficacement).

Grugier et Nogry (Grugier et Nogry, 2021) ont montré que les séquences pédagogiques mises en place en classe sont constituées de deux phases. La première phase est centrée sur la découverte du fonctionnement du robot et la seconde sur l'utilisation et la programmation. Lors de la première phase, les élèves déploient des aptitudes pour comprendre le fonctionnement et résoudre les problèmes rencontrés avec les robots. Ce caractère technique est selon Combarnous, la technicité. « *La technicité résulte de la réunion et de l'interaction permanente de trois composantes :*

– *une composante d'apparence philosophique, la rationalité dans sa forme particulière de réflexion technique ;*

– *une composante d'apparence matérielle, l'emploi d'engins, comme intermédiaire entre des volontés et des actions ;*

– *une composante d'apparence sociologique, les spécialisations des individus et des groupes d'exécution de tâches coordonnées* » (1984, p. 22-23).

Attardons-nous sur la notion d'engin utilisée par Combarnous. C'est une notion qui englobe à la fois des outils, des machines et des équipements. Elle fait référence au monde industriel lié à la production. Pour notre part, nous emploierons parfois le terme « d'objet technique », en référant à des solutions techniques retenues dans la conception, et également le terme d'artefact avec ses contraintes et ses spécificités qui sont à comprendre et à appréhender.

La rencontre, en classe, avec un robot programmable/un artefact, conduit l'enfant à développer un raisonnement empirique pour prendre en charge et mener des actions sur ce dernier en fonction du scénario pédagogique élaboré par l'enseignant. L'enfant s'adapte aux spécificités du robot, pour générer un programme. Le robot influence les activités de l'enfant de par les contraintes qu'il impose. En classe, les moments scolaires proposés autour de la découverte du fonctionnement des robots peuvent être collectifs ou individuels. Comme le précise Hatano (Hatano, 1990), les apprentissages se font au quotidien par des pratiques, par de l'observation de pratiques et par le langage. Une des spécificités de la manipulation et de la découverte d'un robot en classe est l'organisation sociale mise en œuvre par l'enseignant. Au cours des activités de raisonnement, de manipulation, d'échanges, l'enfant élabore des représentations sur le fonctionnement de l'artefact mis à disposition.

Selon Nogry *et al.* (2013), l'approche instrumentale développée par Rabardel (1995) offre un cadre d'analyse pour décrire le schème d'utilisation d'un artefact, comme les robots programmables BeeBot® et ProBot® dans la

durée. L'approche instrumentale « *offre un cadre conceptuel pertinent pour étudier la façon dont l'introduction d'une technologie induit [...] une transformation de l'activité de l'utilisateur* » (Rabardel et Bourmaud, 2003, p. 11). En classe, le robot programmable devient un instrument une fois associé à des schèmes d'utilisation. Un schème se définit selon Nogry comme une « *organisation invariante de l'action réalisée dans un objectif visé* » (Nogry, 2020, p. 4). Les schèmes constituent des outils qui permettent au chercheur de modéliser les relations entre le geste et la pensée élaborée par les élèves (Rabardel, 1995).

Les schèmes d'utilisation se mettent en place progressivement à travers les moments scolaires proposés par l'enseignant. En référence aux deux premières composantes de la technicité de Combarnous (1984), ces moments scolaires sont de nature technique lorsque les élèves cherchent à identifier la chaîne de fonctionnement d'un artefact à travers des tâches nécessitant une réflexion technique pour atteindre un objectif visé par l'enseignant. Nous supposons que l'enfant modifie son activité et que des schèmes d'utilisation se construisent par la manipulation de l'artefact et la découverte progressive des propriétés de ce dernier. Ainsi, l'approche instrumentale est retenue pour appréhender le schème d'utilisation d'une technologie en situation et la technicité permet d'identifier des moments scolaires de nature technique par la présence d'artefacts en classe et la rationalité déployée pour les utiliser.

Quels sont les indicateurs observables pouvant être retenus pour définir un schème d'utilisation ?

En reprenant nos travaux pour l'analyse de l'activité des élèves de maternelle (Grugier, 2021) et une partie des travaux de Spach (2017), l'analyse de l'activité des élèves se fait au travers de deux axes. L'un centré sur les processus élaborés par les élèves avec le robot (les gestes, les actions sur les boutons, les traces écrites des élèves, les discours) et l'autre, sur l'observation des mouvements des robots programmables suite aux actions des élèves. La réflexion et la succession des actions dans un ordre défini sur les boutons d'un robot programmable caractérisent le processus mis en œuvre. Le schème d'utilisation est considéré comme en place lorsqu'un élève met systématiquement en œuvre un processus permettant d'atteindre un même objectif visé.

Le cadre théorique ainsi développé, nous posons les questions de recherche suivantes :

– Quels sont les schèmes d'utilisation qui se construisent, pendant les moments scolaires de nature technique et restent disponibles dans la durée ?

– Quels sont les processus mis en œuvre par les enfants avec ces robots programmables, qui leur sont peu familiers ?

– Quels sont les schèmes transférables pour la mise en œuvre d'un robot programmable plus complexe de par les solutions techniques choisies ?

Dans les lignes suivantes est présentée la méthodologie élaborée pour apporter des réponses aux questions posées.

5. Méthodologie mise en œuvre

Cinq séances d'une heure et trente minutes étalées sur cinq semaines ont été consacrées à la découverte et à la manipulation du robot BeeBot® et du robot ProBot® dans une classe de CM1 de l'académie de Paris. La séquence pédagogique, décrite ci-après, a été mise en place pendant la troisième période de l'année.

L'école de cette classe de CM1 est intégrée à un réseau d'éducation prioritaire (REP). Les élèves sont majoritairement issus de catégories socioprofessionnelles défavorisées et ne disposent pas chez eux de robots programmables. Néanmoins, durant les deux premières périodes de l'année scolaire, les élèves ont découvert des notions de programmation à partir de tablettes tactiles et de l'application Scratch Junior. Ainsi, les notions de programme, de déplacement, de codage ont été explorées.

5.1. Séquence pédagogique proposée aux élèves

Cette séquence pédagogique s'est déroulée en trois phases. La première, découvrir le fonctionnement du robot BeeBot® (séances 1 et 2), la seconde programmer le BeeBot® (séances 3 et 4) et la dernière découvrir le fonctionnement du robot ProBot® (séance 5). L'objectif de la première phase était l'identification des fonctions des organes de commande. L'objectif de la seconde était d'amener les élèves à définir puis programmer une séquence d'instructions afin que le robot se déplace sur un parcours linéaire puis sur un parcours constitué d'obstacles à contourner. Lors de la dernière phase, l'objectif consistait à identifier les fonctions des organes de commande du robot ProBot® et à programmer un déplacement.

L'enseignant est intervenu auprès des différents groupes afin de s'assurer que chacun puisse découvrir les robots. La démarche pédagogique de l'enseignant laisse une place forte à l'investigation à travers la manipulation et l'expérimentation effectuées par les élèves. Pour cela, des groupes de 3 à 6 élèves ont été constitués. Pour chaque groupe, un robot était mis à disposition. Pendant les moments en classe, le chercheur n'est pas intervenu auprès des élèves.

5.2. Protocole de recueil des données

Le chercheur a filmé les élèves pendant les moments de manipulation du robot dans les groupes et pendant les échanges avec l'enseignant en classe entière. Cette classe n'étant pas équipée de robots, ces derniers ont été introduits par le chercheur. Ainsi, les élèves ne les avaient pas rencontrés auparavant.

Les groupes d'élèves ont été constitués par l'enseignant, mais pas spécifiquement pour ces moments scolaires sur les robots. Il s'agissait de ne pas introduire des variables supplémentaires notamment au niveau des rituels.

L'analyse des vidéos permet de relever les actions et les échanges pendant les moments où les élèves manipulent et échangent entre eux sur le fonctionnement des robots. Dans ce corpus vidéo, nous avons cherché à identifier des traces traduisant la construction de schèmes d'utilisation à travers les processus élaborés et développés par des élèves, en nous focalisant sur leurs gestes tout au long des différentes séances.

5.3. Composition du corpus

Seuls les extraits des vidéos contenant des moments scolaires de nature technique avec la présence d'un robot dans les groupes d'élèves ont été retenus. Les vidéos ont été découpées à l'aide du logiciel *Movie Maker* pour se concentrer sur les gestes des élèves et la chronologie de ces derniers dans la manipulation des robots. Par une collecte inductive à partir du visionnage des traces de constructions, des processus mis en œuvre ont été recherchés. Le corpus (tableau 2) est ainsi composé de capsules vidéo contenant des moments scolaires de nature technique avec des élèves qui manipulent les robots ou échangent sur les robots en classe. Afin de répondre à la problématique de ce travail de recherche, la durée des capsules vidéo peut varier entre quelques secondes (17 s) et plusieurs minutes (9 min).

Tableau 2 · Capsules vidéo constituant le corpus

Séance	Nombre de capsules vidéo	Durée totale
1	2	11 min 36 s
2	8	7 min 30 s
3	5	2 min 44 s
4	13	32 min 41 s
5	8	20 min 30 s

5.4. Protocole d'analyse des données

Les composantes de la technicité permettent de caractériser les moments scolaires de nature technique et l'approche instrumentale d'analyser les activités des élèves dans ces moments. Partant de ce cadre, une grille d'analyse, permettant de relever les processus mis en œuvre pour atteindre les objectifs visés dans les moments scolaires de nature technique, a été élaborée (tableau 3).

Tableau 3 · Exemple d'une grille d'analyse de l'activité des élèves pendant les moments scolaires de nature technique

<table>
<tr><th colspan="2">Numéro de séance :</th><th></th><th>1/5</th></tr>
<tr><th colspan="2">Durée de la capsule vidéo :</th><th></th><th>6 min 22 s</th></tr>
<tr><th colspan="2">Activités de nature technique</th><th>Processus mis en œuvre</th><th>Objectif visé</th></tr>
<tr><td>Artefact(s) utilisé(s)</td><td>BeeBot®, notice d'utilisation et câble d'alimentation</td><td rowspan="2">Dans les équipes, les élèves ouvrent la boîte contenant le BeeBot®, la notice de fonctionnement et le cordon d'alimentation. Après 20 s, les deux interrupteurs sont actionnés. Les élèves appuient sur les boutons orange puis sur le bouton GO.</td><td rowspan="2">Mettre en fonctionnement le robot et comprendre comment il fonctionne.</td></tr>
<tr><td>Réflexion technique</td><td>« Il faut le brancher. Je sais comment ça marche ! En fait, il faut d'abord l'arrêter et après appuyer sur les boutons orange puis sur GO. Il faut d'abord appuyer sur ce bouton (Effacer) ».</td></tr>
</table>

La section qui suit examine les schèmes d'utilisation des élèves qui se construisent dans un environnement aménagé (la classe). Cette approche se fonde sur une analyse d'une succession de séances durant lesquelles les processus manipulatoires se mettent en place et peuvent se modifier. Pour cela, la première partie consistera à présenter l'analyse du corpus par une approche chronologique des séances vécues par les élèves. Dans une seconde partie, une analyse comparative des processus manipulatoires développés avec les deux robots sera effectuée. Cette analyse est présentée par un texte en italique afin de mieux appréhender le contenu.

6. Schèmes d'utilisation élaborés par les élèves

6.1. Construction d'un schème d'utilisation pour allumer et programmer le robot BeeBot®

L'analyse de l'activité des élèves lors de la première phase met en évidence la mise en place d'une réflexion pour permettre de mettre en fonctionnement le robot BeeBot®. Les élèves doivent découvrir comment fonctionne ce nouvel objet.

Dans les groupes, les élèves ouvrent l'emballage dans lequel se trouvent le robot, la notice d'utilisation et le cordon d'alimentation. Certains élèves regardent la notice pendant que d'autres manipulent le robot. Après une vingtaine de secondes, l'observation et la manipulation du robot ont permis d'identifier la présence de deux interrupteurs placés sous le BeeBot®. *La notice est rapidement abandonnée pour se concentrer sur le robot et les voyants lumineux.* Dans les secondes qui suivent (pour certains groupes, 4 s après avoir actionné les interrupteurs), les élèves appuient sur les 4 boutons de direction puis sur le bouton « GO ». Le robot se déplace en roulant sur la table. Pour la suite de cette première séance, c'est principalement le bouton « GO » qui est utilisé pour générer le déplacement du robot.

Pour effacer la programmation, dans un premier temps, *les élèves pensent qu'il faut éteindre le robot. Après deux tentatives, cette hypothèse n'est pas retenue.* Dans un second temps, ils disent qu'il faut appuyer sur le bouton « X » (effacer).

L'objectif de cette première séance est rappelé par l'enseignant lors du moment de bilan : « qu'est-ce que vous avez compris du fonctionnement de ce robot *?* ». Immédiatement, un élève dit avoir remarqué que le robot à une mémoire. Une autre élève ajoute qu'il y a une mémoire car lorsque l'on appuie sur les boutons orange et puis sur GO, « il (le robot) répète, il se souvient de ce qu'il fait ».

Cette première rationalité technique sur le fonctionnement du robot BeeBot® est nécessaire pour pouvoir interpréter les déplacements et les associer aux actions sur le pupitre de commande.

La classe pense également que les boutons orange servent à « faire bouger » le robot. Comme le disent les élèves « quand on appuie sur le bouton vers la droite, il va à droite ». Pour un autre élève, « il faut aussi appuyer sur GO pour avancer ». Ces élèves ont également identifié la fonction du bouton pause. Pour cette classe, ce « bouton bleu permet d'arrêter le déplacement pendant 1 s ».

Un schème d'utilisation se met en place avec la mise en œuvre d'un processus pour d'une part mettre en fonctionnement le robot et d'autre part programmer un déplacement du robot. À la fin de cette première séance, les fonctions des différents boutons semblent identifiées cependant, la construction d'un processus pour programmer le robot ne sera stabilisée qu'à la fin de la seconde séance. Le processus mis en œuvre par les élèves consiste à agir sur les deux interrupteurs placés sous le robot puis à mettre en action la séquence d'instructions suivante : bouton effacer, boutons de direction orange, bouton « GO ». Les élèves vont ainsi générer des déplacements et identifier par la suite la longueur d'un pas de déplacement du robot. Pour cela, un autre artefact est introduit par l'enseignant. Il s'agit de la règle scolaire de mesure. Cet étayage proposé par l'enseignant guide (Bruner, 1983) la pratique des élèves non pas dans la compréhension du fonctionnement du robot, mais dans l'activité des élèves, en introduisant une procédure à suivre avant de programmer. Ainsi, le crayon et la feuille de papier sont utilisés pour représenter une séquence d'instructions (figure 3) reprenant une représentation graphique des boutons du pupitre de commande.

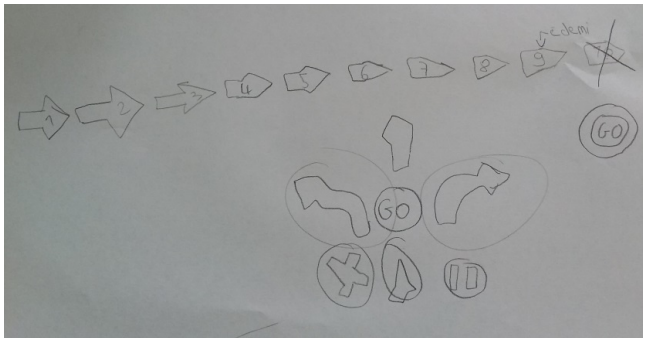


Figure 3 · Trace d'une séquence d'instructions permettant de programmer un BeeBot® pour un déplacement rectiligne

Cette représentation d'une séquence d'instructions, sur la figure 3, permet d'observer des traces d'une réflexion technique dans la programmation du BeeBot®. En effet, les élèves devaient programmer le déplacement du robot afin qu'il se déplace sur toute la longueur d'une table. Après avoir relevé qu'un pas du robot était de 15 cm, les élèves ont tracé sur le plateau de la table des repères tous les 15 cm. *Ils en ont déduit qu'il fallait que le robot se déplace de 9,5 pas.* Cependant, la conception du BeeBot® n'autorise pas la programmation d'un demi-pas. L'observation du fonctionnement du robot a donc conduit les élèves à programmer le robot de 10 pas puis de 9 pas.

La phase de découverte par la manipulation a permis aux élèves d'élaborer une procédure pour allumer le robot. Elle a également permis une amorce de construction d'une procédure permettant de générer des déplacements. Le passage par une représentation graphique de la séquence d'instructions a permis aux élèves d'apporter des corrections au programme suite à la découverte des caractéristiques de déplacement du robot.

6.2. Programmer efficacement le robot BeeBot®

Lors de la troisième séance, les élèves sont conviés à programmer le robot afin qu'il se déplace d'un point de départ jusqu'à un point d'arrivée. Sur les tables, deux repères papier (*post-it*) sont positionnés, un pour matérialiser le point de départ et l'autre, le point d'arrivée. Les élèves utilisent un crayon et du papier pour planifier le programme à implémenter. Des mesures de distances sont effectuées pour définir le nombre de pas nécessaires.

Les Post-its n'étant pas positionnés à des longueurs liées à un multiple de pas du robot BeeBot®, plusieurs stratégies se mettent en place, de manière à faire en sorte que ce dernier arrive soit à côté du point d'arrivée soit dessus. La réflexion menée consiste à atteindre l'objectif visé tout en prenant en compte les contraintes de fonctionnement du robot. Ceci conduit à des ajustements dans l'écriture d'une séquence d'instructions et à des modifications dans la programmation. Plusieurs itinéraires étant possibles, des stratégies différentes se mettent en place : se déplacer en ligne droite ou effectuer plusieurs virages (figure 4).

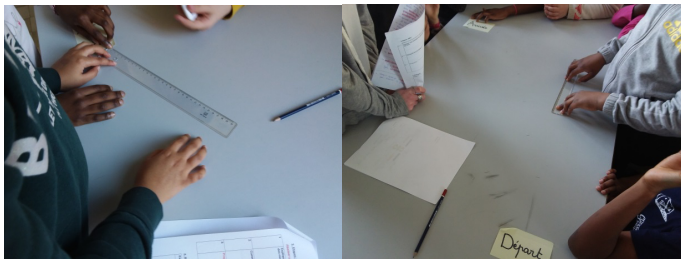


Figure 4 · Mesurer des distances pour définir un nombre de pas

Lors de la quatrième séance, l'objectif est de programmer le robot afin qu'il se déplace d'un point à un autre tout en évidant des obstacles (figure 5).

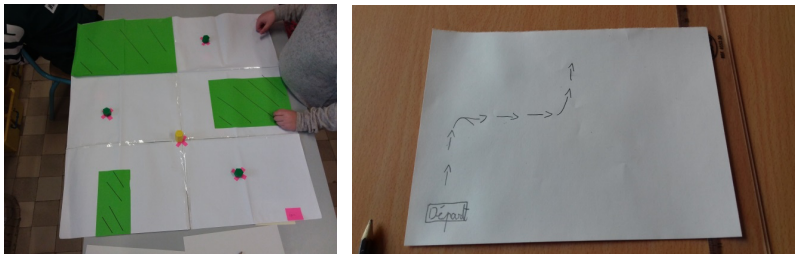


Figure 5 · Identification d'un itinéraire et représentation d'une séquence d'instructions de déplacement

Le schème d'utilisation élaboré lors des séances précédentes reste identique avec le processus : effacer, flèches de direction et « GO ».

Pour générer un déplacement, différentes stratégies sont mises en œuvre : identifier le nombre de pas à l'aide d'une règle, puis programmer et écrire la séquence d'instructions sur une feuille. Ainsi, pour certains, il s'agit d'une programmation pas à pas avec un déplacement du robot sur la table au fur et à mesure, pour d'autres des sous-programmes sont testés puis validés avant de passer au sous-programme suivant et enfin, certains élaborent la séquence d'instructions complète. Ces différentes stratégies ont déjà été observées, notamment par Komis et Misirli (Komis et Misirli, 2015), lors de la mise en œuvre d'un scénario pédagogique auprès d'enfants de 4 à 6 ans. Ces différentes stratégies sont liées, également, aux caractéristiques techniques du robot BeeBot®. En effet, l'impossibilité de visualiser les lignes de programmation du robot et de modifier un programme existant conduit les élèves à essayer plusieurs fois la

programmation d'une même section de parcours. De plus, chaque erreur sur les boutons du pupitre nécessite une nouvelle programmation. Ceci est d'autant plus vrai que le parcours est complexe.

6.3. Construction d'un schème d'utilisation pour le robot ProBot®

Lors de la dernière séance, chaque enfant devait allumer et programmer le robot ProBot® pour que ce dernier se déplace en formant des figures rectangulaires.

Les élèves utilisent le même processus qu'avec le BeeBot® pour prendre en charge ce nouveau robot. Ils retournent le robot pour agir sur les deux interrupteurs.

Par la suite, le processus de programmation qui consiste à appuyer sur la succession des touches « effacer-flèches-GO » est utilisé, comme en attestent les échanges verbaux dans un groupe d'élèves :

– Élève 1 : « C'est où effacer ? »

– Élève 2 : « Il faut peut-être appuyer sur menu ? »

– Élève 3 : « Maîtresse, c'est en anglais » (La touche effacer est baptisée « clear » par les concepteurs du robot.)

Le processus doit être adapté à ce nouvel artefact. Il apparaît par la suite une réflexion technique pour découvrir le fonctionnement du ProBot®. Ainsi, les enfants agissent sur les boutons représentant des flèches de déplacement puis sur « GO ».

Le robot se déplace. Après ces premières découvertes, les investigations se poursuivent vers les fonctions des autres boutons du pupitre de commande. Après de 2 à 3 minutes de manipulation, les élèves découvrent qu'il y a des instructions sur l'écran. Certains pensent qu'il s'agit du « programme qui est déjà fait ». L'investigation du ProBot® se poursuit, jusqu'à la fin de la séance en s'attardant sur les fonctions des boutons de commande et sur les effets qu'ils produisent sur le robot. Cependant, les fonctions de chaque bouton ne pourront pas être explorées pendant cette séance.

L'analyse fonctionnelle mise en place par les élèves consiste à agir sur un bouton et observer l'effet produit sur le robot. Les boutons qui sont similaires au BeeBot® sont explorés en premier. L'affichage des lignes d'instructions sur l'écran du ProBot® apporte une lisibilité et permet aux élèves de mieux associer les actions sur les boutons et les effets sur le robot.

L'analyse de l'activité des élèves durant les deux premières séances de manipulation et de découverte du robot BeeBot® met en évidence un schème d'utilisation avec la mise en place de deux procédures permettant de mettre en fonctionnement le robot et de le programmer. De plus, l'investigation menée par les élèves permet de développer une réflexion technique sur l'objet qui est transférée pour comprendre le fonctionnement du second robot, le ProBot®. Certaines similitudes de conception entre les deux robots (pupitre de commande avec des boutons remplissant les mêmes fonctions, mode de déplacement sur le sol...) facilitent la prise en main du second par les élèves. De ce fait, il persiste l'idée que la programmation d'un déplacement du robot ProBot® se fait uniquement à partir des flèches de direction comme sur le BeeBot®. Les élèves ne prennent pas en compte la possibilité de modifier la longueur d'un pas de déplacement. Ils ne sont pas dans une stratégie de programmation en explorant plusieurs possibilités, mais dans une stratégie de mise en mouvement et de déplacement du robot. Par contre, le changement de dénomination de la touche « effacer » désoriente les élèves.

7. Discussions et perspectives

Cette dernière partie de l'article est l'occasion de revenir sur les questions de recherche posées et de discuter des résultats présentés précédemment. L'analyse des vidéos de moments scolaires avec la présence de robots programmables permet de repérer la présence d'une élaboration de schème d'utilisation par des enfants de 9 et 10 ans. Les processus d'actions mis en œuvre par les élèves pour utiliser les robots programmables sont des traces d'une construction mentale développée pour une utilisation rationnelle.

Le repérage des gestes des élèves et des échanges dans des moments scolaires ont mis en évidence l'apparition d'un schème d'utilisation qui a été transféré d'un objet technique vers l'appropriation d'un nouveau.

7.1. Développement d'un schème d'utilisation avec le robot BeeBot®

Les élèves de cette classe de CM1 découvrent après quelques secondes d'investigation la présence de deux interrupteurs, placés sous le robot BeeBot®. L'action sur les deux organes de commande va leur permettre de prendre en charge le robot. Ce processus devient immédiatement une pratique systématique qui ne sera pas remise en question par la suite. Pousser les interrupteurs pour ouvrir ou fermer le circuit électrique ne

soulève aucune difficulté pour une utilisation du robot même si les fonctions de ces deux organes de commande ne sont pas discutées en classe. De même, la présence de deux interrupteurs, et non d'un seul, pour allumer le robot ne soulève pas de débat dans cette classe. En reprenant Combarnous (1984), il s'agit d'une connaissance coutumière qui s'est construite et non d'une connaissance fonctionnelle.

Dès la fin de la deuxième séance, le processus consistant à appuyer sur les touches « effacer-flèches-GO » se met en place pour programmer le robot BeeBot®. Ce processus ne peut être dissocié des contraintes techniques de ce robot. L'appropriation du robot s'est effectuée après une analyse fonctionnelle de chaque bouton du pupitre de commande. Le repérage des fonctions a permis d'identifier la présence d'une mémoire ne permettant pas de modifier un programme et nécessitant d'appuyer sur le bouton « effacer » pour en réaliser un nouveau.

Ainsi, la manipulation, l'observation et les échanges dans la classe à la fin des séances ont permis aux élèves de développer une capacité d'analyse en identifiant les boutons et les interrupteurs de commande. De plus, les actions sur les boutons du pupitre dans un ordre précis démontrent la construction du concept de séquence d'instructions.

7.2. Développement d'un schème d'utilisation avec le robot ProBot®

Les processus manipulatoires développés par ces élèves avec le robot BeeBot® ont été déployés pour une appropriation du robot ProBot®. Ainsi, les deux interrupteurs positionnés sous ce dernier sont actionnés rapidement, permettant de le mettre en fonctionnement. Le processus consistant à actionner successivement les boutons « effacer », « flèches » puis « GO » est mis en œuvre. Cependant, la présentation du pupitre de commande du robot ProBot® diffère du BeeBot®. Les élèves recherchent donc le bouton « Effacer » qui est baptisé « *Clear* ». En recherchant ce bouton permettant d'effacer un programme de la mémoire, l'investigation conduit les élèves à identifier la présence de lignes de programme qui apparaissent sur l'écran. Une analyse fonctionnelle se met alors en place pour faire un lien entre les actions sur les boutons et les instructions qui apparaissent sur l'écran.

7.3. Culture technique avec des robots programmables

Ces résultats tendent à montrer que l'introduction d'un artefact nouveau dans une classe conduit les élèves à acquérir une culture technique. Cette culture ne se réduit pas à des actions sur des boutons de commande d'un artefact, elle est également intellectuelle. L'élève se familiarise avec l'artefact pour se l'approprier, l'intégrer, le comprendre. L'artefact guide et oriente la réflexion technique nécessaire à son appropriation de par ses spécificités et de par sa conception. La rationalité technique s'inscrit sur deux niveaux. Le premier concerne la construction et l'installation d'un schème d'utilisation pour l'appropriation de l'artefact et le second dans la démarche mise en place pour rechercher et caractériser les fonctions des différents organes de commande de l'artefact. Le premier conduit à définir une utilisation principalement centrée sur l'artefact lui-même, avec ses spécificités, alors que le second conduit au développement d'une démarche d'appropriation adaptable pour la compréhension d'artefacts non familiers. Cette réflexion sur l'artefact, sur son usage, sur son fonctionnement, sur son appropriation, constitue les prémisses de l'acquisition d'une culture technique. Néanmoins, la question de la construction d'une culture technique, en reprenant Séris (1994), n'est pas que technique. Elle ne se réduit pas à l'acquisition de procédures. Car le rôle de l'école est de permettre un accès à une rationalité technique permettant d'« *optimiser l'accession aux techniques actuelles, voire aux techniques imminentes et futures* » (1994, p. 147).

La découverte d'objets techniques, par la manipulation libre dans un contexte scolaire avec une organisation en équipe, contribue à l'élaboration d'un schème d'utilisation visant à s'intéresser au fonctionnement des différents organes qui constituent l'objet. Les résultats montrent que l'analyse fonctionnelle permet aux élèves de s'approprier ensuite des objets plus complexes sans dépendre d'une démonstration de l'enseignant. Ceci est un résultat important qui montre la capacité des élèves à s'interroger sur le fonctionnement d'artefacts nouveaux et ensuite à se les approprier. L'analyse fonctionnelle mise en place, dans cette classe, consiste à agir sur les boutons de commande et à observer les actions engendrées sur les actionneurs. Cependant, la construction d'un schème d'utilisation avec le premier robot a limité l'investigation des élèves pour la découverte de nouvelles fonctions disponibles avec le second artefact. Les élèves se limitent à utiliser les boutons en commun sur les deux robots sans approfondir les autres boutons.

7.4. Perspectives

D'autres études seraient à mener pour analyser l'appropriation des robots programmables par les élèves. Les robots programmables permettent aux élèves de s'interroger sur leur fonctionnement et leur utilisation. La manipulation collective contribue à la construction de schèmes d'utilisation, mais pas seulement. Elle permet également la construction d'une culture technique comme composante d'une culture générale.

Deux axes de recherche sont à développer. Le premier concerne l'appropriation des artefacts par les élèves dans une démarche collective. Il serait intéressant d'explorer l'impact des rôles occupés (troisième composante de la technicité selon Combarnous (1984) par les élèves dans les moments scolaires sur la construction d'une culture technique. Dans les moments scolaires de groupe, où certains élèves manipulent et d'autres observent, les activités ne sont pas toujours les mêmes. Ces différences dans les activités conduisent-elles à la construction d'une culture technique identique ?

Le deuxième axe est en lien avec le choix des artefacts mis à disposition des élèves. Différents fournisseurs de matériel pédagogique proposent des robots de plancher programmables. Le choix, pour un enseignant, d'un robot n'est pas neutre. Au-delà des couleurs, des formes et de la constitution (avec la présence de capteurs ou non et les solutions techniques retenues pour le programmer, etc.), les robots vont permettre aux élèves de se construire des schèmes d'utilisation différents. Ainsi, quel est l'impact du choix des artefacts sur les apprentissages ?

Le travail est à poursuivre tant sur l'analyse des activités des élèves dans la prise en charge d'artefacts que sur celle des enseignants dans le choix de ces derniers.

RÉFÉRENCES

Allari, V. (1986). L'école de la tortue. *Revue de l'EPI, 42,* 136-146.

Baron, G.-L. (2018). Informatique et numérique comme objets d'enseignement scolaire en France : entre concepts, techniques, outils et culture. *Adjectif.* http://www.adjectif.net/spip/spip.php?article456

Baron, G.-L. et Drot-Delange, B. (2016). L'informatique comme objet d'enseignement à l'école primaire française ? Mise en perspective historique. *Revue française de pédagogie, 2,* 51-62.

Baron, G.-L. et Bruillard, É. (1996). *L'informatique et ses usages dans l'éducation*. PUF.

Bers, M., Flannery, L., Kazakoff, E. et Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers and Education, 72*, 145-157.

Bruillard, É. (2016). Quelle informatique à repenser et à construire pour les élèves de l'école primaire ? Dans F. Villemonteix, J. Béziat et G.-L. Baron (dir), *L'école primaire et les technologies informatisées. Des enseignants face aux TICE* (p. 29-38). Presses universitaires du Septentrion.

Bruner, J. (1983). *Le développement de l'enfant : savoir-faire, savoir dire*. PUF.

Combarnous., M. (1984). *Comprendre les techniques et la technicité*. Éditions sociales.

Combes-Trithard, F. (1984). *Enregistrer, lire, programmer à l'école maternelle*. Armand Colin-Bourrelier.

De Michele, S., Demo, B. et Siega, S. (2008). A Piedmont SchoolNet for a K-12 mini-robots programming project: Experience in primary schools. *Proceedings of SIMPAR 2008, Modeling and Programming for Autonomous Robots* (p. 90– 99).

Denhière, G. et Baudet, S. (1992). *Lecture compréhension de texte et science cognitive*. PUF.

Greff, E. (1999). En quoi le robot Algor constitue-t-il un objet didactique original. *Revue de l'EPI, 93,* 127-150.

Grugier, O. (2021). Éducation technologique dans des classes de maternelle. Apprentissages premiers dans l'utilisation et la compréhension d'un artefact robotisé. *RDST, 22*, 61-92.

Grugier, O. et Nogry, S. (2021). *Des objets programmables pour apprendre l'informatique à l'école* [communication]. Colloque Objets pour apprendre, objets à apprendre. Quelles pratiques enseignantes pour quels enjeux ?, Amiens, France.

Grugier, O. et Villemonteix, F. (2017). *Apprentissage de la programmation à l'école par l'intermédiaire de robots éducatifs. Des environnements technologiques à intégrer* [communication]. Colloque EIAH, Strasbourg, France.

Harel, I.-E. et Papert, S. (1991). *Constructionism*. Ablex Publishing.

Highfield, K., Mulligan, J. et Hedberg, J. (2008). Early mathematics learning through exploration with programmable toys. Dans O. Figueras, J.L. Cortina, S. Alatorre, T. Rojano et A. Sepulveda (dir.), *Proceedings of the Joint Meeting of PME 32 and PME-NA* XXX. (vol 3, p. 169–176). International Group for the Psychology of Mathematics Education Psychology of Mathematics Education.

Hatano, G. (1990). The nature of everyday science: a brief introduction. *British Journal of Developmental Psychology, 8*, 245-250.

Komis, V. et Misirli, A. (2012). L'usage des jouets programmables à l'école maternelle : concevoir et utiliser des scenarios pédagogiques de robotique éducative. *Skholê, 17,* 143-154.

Komis, V. et Misirli, A. (2015). Étude des processus de construction d'algorithmes et de programmes par les petits enfants à l'aide de jouets programmables. Dans. B. Drot-Delange, G.-L. Baron et E. Bruillard (dir.), *Informatiques en éducation : Perpectives curriculaires et didactiques* (p. 143-154). Presses universitaires Blaise-Pascal.

Komis, V., Romero, M. et Misirli, A. (2017). A Scenario-Based Approach for Designing Educational Robotics Activities for Co-creative Problem Solving. Dans D. Alimisis, M. Moro et E. Menegatti (dir.), *Educational robotics in the makers era (p. 158-169)*. Springer.

Lebeaume, J. (2019). Objets puis systèmes techniques au programme : éclairages pour une discussion de leurs statuts et de leurs fonctions dans l'enseignement. *Recherche en didactiques, 27,* 13-24.

Martinand, J.-L. (2014). Penser la culture technique pour une école obligatoire ? *Carnets rouges, 1*, 27-29.

Nogry, S. (2020). *Des objets pour apprendre. Articulation entre dynamiques d'appropriation en situation d'apprentissage et développement* [mémoire d'habilitation à diriger des recherches, Université Paris 8, France].

Nogry, S., Decortis, F., Sort, C. et Heurtier, S. (2013). Apports de la théorie instrumentale à l'étude des usages et de l'appropriation des artefacts mobiles tactiles à l'école. *STICEF, 20,* 1-34.

Papert, S. (1980). *Mindstorms : Children, computers and powerful ideas*. Basic Books.

Rabardel, P. et Bourmaud, G. (2003). From computer to instrument system: a developmental perspective. *Interacting with computers, 15*(5), 665-691.

Rabardel, P. (1995). *Les hommes et les technologies. Approche cognitive des instruments contemporains*. Armand Colin.

Spach, M. (2017). *Activités robotiques à l'école primaire et apprentissage de concepts informatiques Quelle place du scénario pédagogique ? Les limites du co-apprentissage* [thèse de doctorat, Université Paris Cité, France].

Séris, J.-P. (1994). *La technique*. PUF.

Vandevelde, I. et Fluckiger, C. (2020). *L'informatique prescrite à l'école primaire. Analyse de programmes, ouvrages d'enseignement et discours institutionnels* [communication]. Colloque Didapro-Didastic 8, Lille, France.

Apprentissage de la programmation informatique à la transition collège-lycée

▶ **Matthieu BRANTHÔME** (Cread, Univ Bretagne occidentale)

■ **RÉSUMÉ** • Cet article s'intéresse au passage d'une programmation graphique par blocs (Scratch) au collège à une programmation en lignes de code (Python) au lycée. Il s'agit, dans les termes de la théorie des situations didactiques (Brousseau, 1998) d'identifier les discontinuités propres à cette transition, puis de concevoir et de tester une ingénierie didactique permettant de la soutenir. Nous avons pu tester une séquence proposant la résolution de défis à travers la programmation d'une carte micro:bit auprès d'élèves de 3e lors d'un atelier sur un temps extrascolaire. Nous mettons en évidence : les apprentissages réalisés, la constitution d'une étape intermédiaire concernant les paradigmes de programmation et les contraintes techniques en jeu, le fort engagement des élèves supporté par le triptyque défi-badge-carte.

■ **MOTS-CLÉS** • transition Scratch-Python, didactique informatique, ingénierie didactique, programmation tangible, défis ludiques.

■ ***ABSTRACT*** • *This article focuses on the transition from block-based programming (Scratch) in middle school to text-based programming (Python) in high school. Using theory of didactic situations (Brousseau, 1998) concepts, we aim to identify the specific discontinuities to this transition and to design and test a didactic engineering to support it. We were able to test a sequence which propose the resolution of challenges through the programming of a micro:bit card with volunteer students during a workshop on extra-curricular time. We highlight: the students' learnings, the constitution of an intermediate stage concerning the programming paradigms and the technical constraints at stake, the strong commitment of the students.*

■ ***KEYWORDS*** • *Scratch-Python transition, computer sciences didactics, didactic engineering, physical programming, playful challenges.*

1. Introduction

Dans un contexte sociétal dans lequel l'informatique et le « numérique » prennent une place toujours plus importante (Abiteboul et Dowek, 2017), les programmes scolaires français du primaire et du secondaire ont connu, ces dernières années, de fortes évolutions. Ainsi, au primaire et au collège, la programmation informatique fait désormais l'objet d'une initiation. Au lycée, de nouvelles matières dédiées ont vu le jour récemment : sciences numériques et technologie (SNT) en seconde et numérique et sciences informatiques (NSI) en première et terminale. L'université évolue également en proposant de nouveaux parcours permettant de former les enseignants dans ces nouvelles disciplines : création d'un nouveau CAPES et de masters MEEF associés.

Face à cet état de fait, nous pouvons faire deux constats. D'abord, nous faisons l'hypothèse d'une carence en ressources pédagogiques à destination des élèves et en contenus pour la formation des enseignants. Ensuite, sur le plan scientifique, les travaux portant sur l'informatique scolaire sont encore rares, notamment ceux mettant en œuvre des concepts théoriques de didactique (Caron, 2020 ; Fluckiger, 2019).

Nous avons choisi, dans cet article, de nous focaliser sur un moment spécifique : la transition du collège au lycée, et plus particulièrement le passage de la programmation graphique par blocs au cycle 4 (logiciel Scratch) à la programmation, plus classique, en lignes de code en seconde (langage Python). Cette transition est susceptible d'engendrer des difficultés spécifiques que nous allons identifier puis chercher à comprendre dans l'objectif de proposer des moyens de les surmonter à travers la conception et la mise en œuvre d'une séquence d'enseignement dans une démarche d'ingénierie didactique.

Nous présentons d'abord une revue de travaux puis notre cadre théorique suivi de notre problématique et de notre méthodologie. Nous détaillons ensuite les résultats établis en suivant les étapes usuelles d'une ingénierie didactique : les analyses préalables, la conception et l'analyse *a priori* de la séquence qui en découle puis son analyse *a posteriori*. Enfin, nous concluons par quelques prolongements et ouvertures.

2. État de l'art

Le champ de recherche s'intéressant à l'enseignement-apprentissage de l'informatique en milieu scolaire est renaissant (Rogalski, 2015). En France, il se concentre principalement sur l'école primaire en portant son intérêt

sur trois types de mises en œuvre (Baron et Drot-Delange, 2016) : la programmation graphique, la robotique pédagogique et l'informatique débranchée. Notons, tout de même, des travaux récents concernant le secondaire : Delmas-Rigoutsos (2020) propose une étude épistémologique de la notion de « variable » et formule des recommandations curriculaires permettant aux élèves de mieux appréhender cette notion ; Libert et Vanhoof (2020) ont conçu puis évalué des activités pour initier des élèves aux enjeux de la programmation concurrente et de la synchronisation ; Journault *et al.* (2020) proposent une séquence d'enseignement sous la forme d'activités débranchées autour de la notion de « décidabilité algorithmique des problèmes ». L'enseignement secondaire et plus particulièrement le lycée constituent ainsi un nouveau terrain pour les travaux sur l'informatique scolaire. Néanmoins, à notre connaissance aucune recherche ne s'est, jusqu'à présent, penchée sur la transition collège-lycée.

Envisageant, lors de notre expérimentation, l'utilisation d'un artefact programmable, nous avons mené une revue de travaux internationaux sur la programmation tangible. La programmation tangible (*Physical Computing* en anglais) est un concept défini par Przybylla et Romeike comme : « *[It] covers the design and realization of interactive objects and installations and allows students to develop concrete, tangible products of the real world that arise from the learners' imagination* » (Przybylla et Romeike, 2014, p. 351). Elle fait l'objet de travaux dont l'arrière-plan théorique est celui du constructionnisme (Papert, 1980). Des articles transversaux (Blickstein, 2013 ; Hodges *et al.,* 2020 ; Przybylla et Romeike, 2014) en présentent les vertus : elle améliore la créativité, l'engagement et la compréhension des apprenants, favorise la collaboration et ouvre l'informatique à un plus large public. Ces travaux proposent également une revue et une classification des nombreux matériels disponibles sur le marché : jouets programmables (ex. : Bee-Bot), briques de construction programmables (ex. : Lego Mindstorm), cartes avec microcontrôleurs (ex. : Arduino) ou ordinateurs compacts (ex. : Raspberry Pi). Il existe également de nombreuses études empiriques autour de la programmation tangible (Merkousis *et al.,* 2017 ; Rubio *et al.,* 2013). Nous pouvons, en particulier, évoquer une très large expérimentation qui a été menée au Royaume-Uni en 2016 au cours de laquelle un million de cartes programmables BBC micro:bit ont été distribuées gratuitement à tous les élèves de 11-12 ans. L'objectif gouvernemental était le développement massif de l'usage de la programmation et des technologies numériques. Plusieurs articles

proposent une évaluation du dispositif du côté des élèves (Sentance *et al.,* 2017a) et des professeurs (Sentance *et al.,* 2017b) après la première année de fonctionnement. Les résultats montrent que la carte micro:bit favorise la créativité et que la nature physique du dispositif agit comme un facteur de motivation. Enfin, le caractère tangible de la carte est un élément clé qui stimule l'intérêt et améliore la compréhension des notions. Nous retenons donc pour la suite que la programmation tangible est susceptible d'améliorer : l'engagement dans les activités, la motivation et la compréhension des notions.

3. Cadre théorique

Nous exposons dans cette partie les concepts théoriques sur lesquels nous allons nous appuyer. Cette étude ayant pour but de concevoir puis de mettre en œuvre une expérimentation, elle s'inscrit dans la démarche méthodologique classique de l'ingénierie didactique s'appuyant sur la théorie des situations didactiques (TSD) (Brousseau, 1998) dont nous exposons les principaux concepts. Nous définissons ensuite les notions qui nous permettront d'étudier les différences intrinsèques entre les langages Scratch et Python. Il s'agit des registres sémiotiques (Duval, 1993) et de quelques paradigmes et méthodes de programmation informatique.

3.1. Théorie des situations didactiques et ingénierie didactique

La TSD s'articule autour de deux notions centrales, celle de situation et celle de milieu didactique. Brousseau définit une situation comme : « *une situation problème qui nécessite une adaptation, une réponse de l'élève.* » (1981, p. 112). Le milieu didactique est établi comme étant « *constitué des objets (physique, culturels, sociaux, humains) avec lesquels le sujet interagit dans une situation [...] C'est le système antagoniste de l'actant [...] tout ce qui agit sur l'élève et ce sur quoi l'élève agit.* » (2010, p. 2).

Brousseau considère que les activités proposées aux élèves doivent tendre vers le modèle de la situation adidactique. Ce type de situation a pour objectif de provoquer chez l'élève des adaptations à travers le choix judicieux de problèmes par le professeur. Ces problèmes doivent permettre à l'élève d'agir de son propre mouvement, guidé uniquement par la logique interne de la situation, sans s'appuyer sur les intentions didactiques de l'enseignant qui se refuse à intervenir comme proposeur de la connaissance qu'il cible (1998). Cet élève doit, de plus, pouvoir envisager une réponse initiale au problème posé sous la forme d'une procédure de base peu

efficace s'appuyant sur ses connaissances antérieures qui n'est pas celle visée dans la situation. Ce savoir cible doit permettre, lorsqu'il est mis en œuvre, de passer de cette stratégie de base à une stratégie « *optimale* » (Bessot, 2003). Le « *coût* » des stratégies doit faire sens pour l'élève au regard des contraintes qui pèsent sur lui. Dans ces conditions, l'élève peut construire le savoir en jeu en lui octroyant véritablement du sens.

La TSD est le cadre privilégié sur lequel s'appuie la méthodologie de l'ingénierie didactique. Artigue (1988) en propose une définition opérationnelle en détaillant ses phases dans un découpage temporel que nous développons ci-dessous.

D'abord, les analyses préalables. Il s'agit d'une étape préliminaire nécessitant l'évaluation des contenus épistémiques, des enseignements usuels et de leurs effets, des conceptions des élèves, des difficultés rencontrées et des diverses contraintes dans lesquelles va se situer l'expérimentation.

Ensuite, la conception d'une séquence et son analyse *a priori*. Lors de la conception de la séquence, le chercheur prend la décision d'agir sur un certain nombre de variables du système. Les variables globales fixent l'organisation générale de l'expérimentation et l'aménagement du milieu didactique. Les variables locales permettent la conception de la situation. L'analyse *a priori* permet ensuite de « *déterminer en quoi les choix effectués permettent de contrôler les comportements des élèves et leur sens* » (Artigue, 1988, p. 258). Elle comporte donc deux aspects. Une partie descriptive qui consiste à : décrire et justifier les options choisies concernant la constitution et l'aménagement du milieu didactique ; motiver les choix de conception et expliciter les savoirs en jeu pour chaque situation. Suit une partie prédictive, il s'agit alors d'imaginer et d'anticiper les différentes stratégies de résolution possibles et vérifier que celles comportant le meilleur coût pour l'élève résultent bien de la mise en œuvre de la connaissance ciblée.

Vient ensuite l'expérimentation qui consiste en la mise en œuvre effective de la séquence conçue et au recueil de données empiriques.

Enfin, l'analyse *a posteriori*. C'est lors de cette dernière étape de mise en regard des faits avec les prédictions qu'a lieu la validation ou l'invalidation des hypothèses engagées dans la recherche. Il est ensuite possible de réitérer le processus, à l'aune des résultats obtenus.

3.2. Systèmes sémiotiques

La distinction entre un concept et ses représentations est un élément important pour la compréhension et la conceptualisation des objets théoriques. Duval qualifie ces productions de représentations sémiotiques et les décrit comme « *des productions constituées par l'emploi de signes appartenant à un système de représentation qui a ses contraintes propres de signifiances et de fonctionnement* » (1993, p. 39).

Un système de représentation peut constituer un registre sémiotique à la condition qu'il permette trois activités cognitives fondamentales : la formation d'une représentation identifiable comme une représentation dans un registre donné ; la transformation interne au registre ; la conversion, c'est-à-dire une transformation externe au registre de départ.

Duval constate un cloisonnement des registres sémiotiques chez les élèves qui ne reconnaissent pas les mêmes objets à travers des représentations exprimées dans des registres différents. Une des raisons de ce cloisonnement est l'hétérogénéité des différents registres qu'il mesure à leur degré de congruence. Les trois critères de congruence sont :

– la correspondance sémantique entre les différentes unités signifiantes de chaque représentation : à chaque unité signifiante d'une représentation, on peut associer un élément signifiant de l'autre ;

– l'univocité sémantique entre les représentations : à chaque unité signifiante de la représentation de départ doit correspondre une unique unité signifiante dans l'autre représentation ;

– l'organisation des unités signifiantes doit être la même : ces unités doivent être appréhendées dans le même ordre dans les deux présentations.

Nous analyserons donc, par la suite, à l'aide de ces concepts, le degré de congruence des différents registres sémiotiques manipulés par les élèves afin d'y déceler d'éventuelles sources de difficultés.

3.3. Paradigmes et méthodes de programmation

Les aspects multiples que peut prendre l'agencement des instructions dans un programme informatique se distinguent en différents paradigmes de programmation (Floyd, 1978). Nous nous limitons, dans le cadre de ce travail, à différencier deux des quatre paradigmes classiques : la programmation impérative et la programmation orientée objet.

Le paradigme de la programmation impérative est caractérisé par un état implicite de la mémoire, désigné par des variables. Cet état peut être

modifié par des commandes (instructions) selon un principe de séquencement permettant un contrôle précis et déterministe des états (Hudak, 1989, p. 361). Ce paradigme de programmation s'attache donc à décrire comment les opérations doivent être effectuées séquentiellement pour résoudre un problème donné, en s'appuyant sur des espaces mémoires indexés par des variables.

Le paradigme de la programmation orientée objet consiste à concevoir et à manipuler uniquement des « objets » en tant qu'abstractions représentant des concepts où des entités du monde physique. Ces objets encapsulent en leur sein, à la fois les données (attributs) et les fonctionnalités qu'ils peuvent offrir (méthodes) (Stefik et Bobrow, 1985). Chaque type d'objet est défini, une fois pour toutes, dans une « classe » qui spécifie ses attributs et décrit ses méthodes. Il peut, ensuite, être instancié, à l'exécution, en fournissant des valeurs particulières pour ses attributs. Son état peut, par la suite, être modifié par le seul accès à ses méthodes. En programmation orientée objet, pour traiter un problème, il s'agit donc de le modéliser à l'aide d'objets proposant diverses fonctionnalités, puis d'articuler les instanciations et les appels aux méthodes de ces différents objets pour fournir une solution.

Outre ces aspects paradigmatiques, d'autres paramètres peuvent influencer la façon de résoudre un problème à l'aide d'un ordinateur. Nous parlerons alors de méthodes de programmation. Certaines architectures matérielles et logicielles permettent l'exécution parallèle, c'est-à-dire en même temps, de pièces de code différentes. On parle alors de programmation parallèle ou concurrente. Il s'agit cependant de prendre certaines précautions relativement aux accès concurrents aux mêmes données. D'autres environnements autorisent, quant à eux, l'exécution d'un seul programme à la fois.

En ce qui concerne le déclenchement de l'exécution de ces pièces de code, elles peuvent être lancées en ligne de commande dans la console d'un ordinateur. Une alternative consiste à ce que ces programmes soient « en attente » d'évènements (appui sur une touche du clavier, mouvement de la souris, réception d'un message) pour s'exécuter. On parle alors de programmation événementielle.

Nous mobiliserons, dans la suite, ces différents paradigmes et méthodes de programmation dans l'objectif de caractériser puis comparer les langages Scratch et Python.

4. Problématique

Nous partons de la question pratique suivante : comment accompagner et soutenir la transition du collège vers le lycée dans l'apprentissage de la programmation ?

Nous déclinons cette problématique principale en plusieurs sous-questions :

– Quels sont les changements à l'œuvre lors de la transition collège-lycée ? Sont-ils susceptibles de causer des difficultés d'apprentissage ? Et si oui, quelles sont ces difficultés ?

– Quelle ingénierie didactique concevoir qui soit susceptible d'amoindrir ces potentielles difficultés ? Comment opèrent ces soutiens aux élèves ?

5. Méthodologie

Lors de cette recherche, nous avons suivi la démarche méthodologique de l'ingénierie didactique dont nous avons présenté les grandes phases dans la section 3. Nous donnons ici les détails méthodologiques inhérents à chacune de ces phrases.

5.1. Analyses préalables

Lors des analyses préalables, nous nous sommes appuyés sur les instructions officielles du cycle 4 et de la classe de seconde (programmes de mathématiques et de SNT, documents d'accompagnements relatifs à l'algorithmique et à la programmation) ainsi que sur les propriétés intrinsèques des langages et des environnements techniques dans l'objectif d'analyser les changements sous-tendus par le passage de la programmation en Scratch à la programmation en Python.

5.2. Conception de la séquence

Pour cette phase, notre objectif était de concevoir une séquence d'enseignement à destination d'élèves en début de classe de seconde. Nous nous sommes d'abord reposés sur les éléments issus de notre revue de littérature et sur nos analyses préalables afin d'éclairer nos choix matériels et logiciels dans l'aménagement du milieu. Concernant la conception des activités proprement dites, nous avons construit une première version de notre séquence, de façon assez intuitive, en nous appuyant sur les savoirs ciblés. Nous avons ensuite réalisé l'analyse *a priori* des situations conçues, en repérant précisément les savoirs en jeu puis en anticipant toutes les stratégies répondant aux attendus. Ce travail a déclenché un cycle itératif d'analyses-modifications dont le produit final a été formalisé dans un support pédagogique numérique.

5.3. Expérimentation : terrain et données

Étant entré dans le dernier tiers de l'année scolaire nous n'avons pas eu accès à des élèves débutants de seconde, nous avons donc testé notre séquence auprès d'élèves de troisième. Cette expérimentation a pris la forme d'un atelier ayant eu lieu sur un temps extrascolaire : un mercredi après-midi dans les locaux d'un collège rural de taille moyenne. Cela offrait l'avantage de disposer d'une large plage horaire permettant de mettre en œuvre notre séquence d'un seul tenant et de conclure l'atelier par un entretien-bilan collectif « à chaud ». Notre méthodologie nécessitant la mise en place d'un dispositif individuel assez lourd et engendrant un grand volume de données, nous avons limité le nombre de participants à six élèves. Ces élèves étaient volontaires et ont été recrutés suite à une rapide présentation de la carte micro:bit à la fin d'un cours de mathématiques. Précisons que ces six volontaires ont un bon niveau scolaire en particulier en mathématiques. Il est également important de noter que, lors de cet atelier, nous avons joué le double rôle d'enseignant et de chercheur-observateur.

Pendant la première phase de l'expérimentation (trois heures) les élèves disposaient individuellement d'un ordinateur portable proposant un support pédagogique numérique, un environnement de développement ainsi que d'une carte micro:bit. Notre dispositif de collecte de données, consistait, pour chacun des six élèves, à :

– enregistrer les activités à l'écran de son ordinateur au moyen du logiciel OBS Studio ;

– filmer ses interactions avec la carte programmable à l'aide d'une caméra fixe sur trépied située dernière lui ;

– enregistrer le son de la webcam de l'ordinateur afin de capturer ses interactions avec le professeur et les autres élèves.

La figure 1, constituée de prises de vue effectuées juste avant et au cours de l'expérimentation, permet de se faire une idée de la mise en œuvre effective de ce dispositif.

Figure 1 · Prises de vue avant et pendant l'expérimentation

Lors de la seconde phase, celle du bilan, nous avons réalisé un entretien collectif de type « focus-group » alors que tous les élèves s'étaient regroupés sur l'ilot central pour partager une collation. L'objectif de cet entretien était de recueillir les impressions et les commentaires des élèves concernant leur vécu au cours de cette expérimentation. Les questions ont été regroupées selon cinq thèmes : engagement et motivation ; connaissances préalables ; formes des activités et ressources d'appui ; difficultés rencontrées ; évaluation des apprentissages.

Le traitement des données ainsi recueillies a d'abord consisté en un travail technique. Pour chaque élève, il a fallu convertir, synchroniser puis fusionner les fichiers issus des différents dispositifs de captation. Nous avons ainsi réuni, sur un même support numérique toutes les interactions de l'élève avec le milieu didactique, c'est-à-dire : l'utilisation de l'environnement de développement, la consultation du support pédagogique, la manipulation de la carte programmable et les échanges avec le professeur. Nous proposons dans la figure 2 une capture d'écran issue du montage vidéo final d'un élève. L'entretien a été retranscrit intégralement en gardant les thèmes dégagés lors de la rédaction du guide.

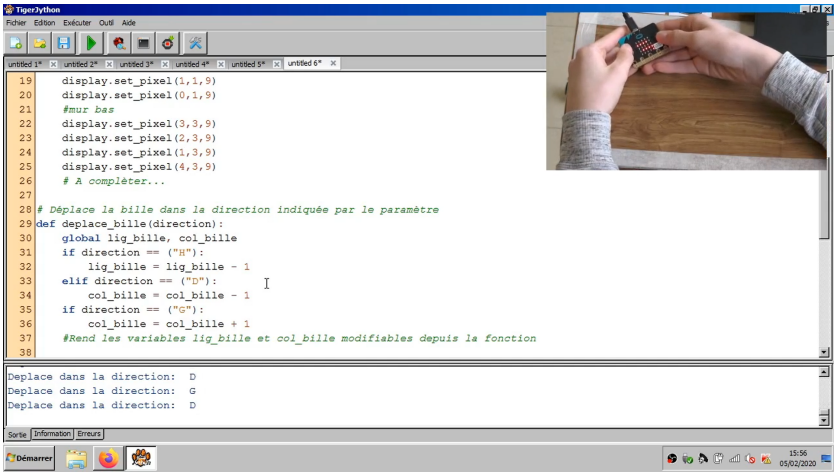


Figure 2 · Capture d'écran tirée d'un montage vidéo

Nous avons ensuite procédé à une analyse qualitative de ces données qui a consisté à consigner et à retranscrire : toutes les interactions des élèves avec le milieu didactique, toutes les difficultés rencontrées ainsi que toutes les stratégies mises en œuvre par les élèves. Dans un second temps, nous avons procédé à la réduction quantitative du volume des données en entreprenant un comptage des différents éléments.

5.4. Analyses *a posteriori*

Notre analyse *a posteriori* s'appuie donc principalement sur ces éléments qualitatifs et quantitatifs issus de nos captations vidéo ainsi que sur la transcription de l'entretien. Nous avons essayé d'estimer en quoi notre proposition pouvait aider les élèves à surmonter les difficultés inhérentes à la transition Scratch-Python que nous avons décelées lors de nos analyses préalables. Nous présentons dans les sections suivantes les résultats que nous avons obtenus à chaque phase de l'ingénierie didactique.

5.5. Analyses préalables

Nous exposons dans cette section une analyse des différents changements sous-tendus par la transition collège-lycée dans l'apprentissage de la programmation, et en particulier le passage de la programmation en Scratch à la programmation en Python. Ces écarts ont été catégorisés dans quatre domaines que nous détaillons ci-après : les contenus épistémiques, les paradigmes et méthodes de programmation, les registres sémiotiques, et les types d'activités.

5.6. Contenus épistémiques

En consultant le programme et les documents d'accompagnement de mathématiques du cycle 4 (MEN, 2018, 2016), nous pouvons affirmer qu'à ce niveau, les enseignants sont encouragés à adopter une approche très ludique à travers l'utilisation du logiciel Scratch qui est préconisé pour sa simplicité et sa robustesse. Les concepts à étudier sont principalement : les notions d'algorithme et de programme, la notion de variable, les boucles et la structure conditionnelle.

La lecture des programmes et des documents d'accompagnement de mathématiques (MEN, 2017, 2019b, 2019a) et de SNT (MEN, 2019c) de seconde, indique que, pour cette classe, les objectifs en termes de savoirs sont d'approfondir les notions introduites au cycle 4 (variables, structure conditionnelle et boucles), tout en ajoutant trois nouveautés : le typage des variables, la notion de « fonction » et la production d'un programme sous forme textuelle. Concernant les activités, il est clairement précisé qu'elles doivent être au service des autres parties du programme. Un langage de programmation est préconisé sans détour : il s'agit de Python. Il a été choisi pour sa simplicité et sa popularité (large écosystème dans la sphère éducative).

Au-delà des contenus épistémiques prescrits par l'institution à travers ses documents officiels, la programmation en Python conduit les apprenants à se confronter à une série de nouveaux savoirs de nature technique. En effet, l'environnement Scratch permet de se concentrer uniquement sur la partie algorithmique en proposant une programmation de très haut niveau cachant toute la partie technique liée à la machine. Les blocs opèrent un contrôle syntaxique total et sémantique partiel par leurs formes et couleurs. Les programmes se lancent par déclenchements sur des évènements, les sorties se font par l'expression ou les actions des « lutins ».

Le langage Python n'a pas été conçu dans un but pédagogique. Son utilisation nécessite l'installation d'un éditeur de code et d'un interpréteur. Selon l'environnement de développement choisi, il peut être nécessaire d'enregistrer le code source du programme dans l'arborescence du système de fichier de l'ordinateur puis de lancer l'interprétation du programme à travers la saisie d'une ligne de commande dans la console système de l'ordinateur. Dans le cadre de l'apprentissage du langage Python, il devient donc nécessaire pour les apprenants d'acquérir des connaissances, même partielles et imprécises, à propos de l'architecture des machines et des systèmes d'exploitations. De nouvelles contraintes apparaissent également : la nécessité

de correction syntaxique et sémantique du texte du programme, le blocage en cas d'erreur, la gestion dans la console système des entrées-sorties permettant les interactions avec le programme. Tout ceci constitue, de fait, un ensemble de nouveaux savoirs techniques qui pourraient parasiter l'apprentissage de ceux prescrits par l'institution. En effet, le risque encouru par les élèves au cours de cette exposition abondante et soudaine à de nouveaux éléments est la survenue d'une surcharge cognitive (Sweller, 1988) potentiellement néfaste à l'appréhension des concepts algorithmiques visés par les programmes d'enseignements.

5.7. Paradigmes et méthodes de programmation

Le passage du logiciel Scratch à la programmation en Python opère un basculement dans le paradigme et la méthode de programmation. En effet, le logiciel Scratch permet une programmation orientée objet. Le code est porté par des objets : les « lutins » et agit en modifiant leurs attributs (position, apparence, sons ou messages émis). Chaque programme, nécessairement attaché à un de ces objets peut donc être considéré comme l'une de ses méthodes. Le langage Python est utilisé en classe de seconde de manière impérative, c'est-à-dire en exécutant une suite séquentielle d'instructions modifiant les valeurs de variables. Il s'avère que les changements de paradigmes de programmation peuvent représenter des obstacles lors des apprentissages, car ils nécessitent des adaptations cognitives de la part des apprenants (Khazaei et Jackson, 2002 ; White et Sivitanides, 2005).

D'autre part, le logiciel Scratch met en œuvre les méthodes de programmation parallèle et événementielle. Ainsi, l'exécution du code est déclenchée par des évènements : clic sur une zone, touche appuyée, message reçu, etc. Cela peut, par ailleurs, occasionner l'exécution concurrente de plusieurs pièces de code. En Python, en fonction de l'environnement de développement utilisé, le code est lancé à l'intention du programmeur, en ligne de commande, ou au mieux, en cliquant sur un bouton de l'environnement de développement. De surcroît, la programmation parallèle n'est pas envisageable de façon simple au niveau du lycée. Il apparaît que ces différences modifient la façon dont on peut envisager la réponse à un problème posé. Les élèves doivent s'exprimer différemment en langage Python, en s'adaptant à une palette de possibilités plus pauvre. L'expression se retrouve, d'une certaine manière, bridée par le carcan de la séquentialité, alors que les élèves ont pris l'habitude au collège avec Scratch, d'utiliser la programmation événementielle et parallèle.

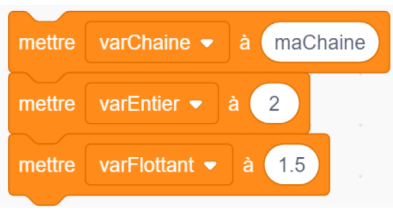
5.8. Registres sémiotiques

La programmation en Scratch et la programmation en Python font appel à des registres sémiotiques bien différents pour implémenter les concepts algorithmiques. Nous les nommerons, dans la suite, respectivement registre des blocs et registre des instructions.

Le registre des blocs est constitué de représentations sémiotiques sous la forme de blocs manipulables et assemblables de différentes formes et couleurs. Les couleurs définissent neuf catégories d'usages (orange : contrôle, violet : apparence, vert : opérateur, etc.) et les formes précisent à la fois l'organisation et la structure du programme (bloc de début, bloc de fin, bloc d'empilement, bloc d'imbrication), elles peuvent aussi indiquer le type du bloc (hexagonale : booléen ; rectangulaire arrondie : nombre ou chaîne de caractères).

Le registre des instructions comprend des représentations sémiotiques sous forme textuelle. Il s'agit des mots réservés du langage Python. Ce sont des mots-clés en anglais prenant la forme de prépositions (« *for* », « *in* ») de conjonctions (« *while* », « *if* », « *else* ») ou de verbes à l'impératif (« *return* », « *print* ») mélangés avec des symboles mathématiques (« = », « > ») ou typographiques (« : », « (», « " »).

Pour chaque notion algorithmique issue des programmes (variable, conditionnelle, boucle bornée et non bornée, fonction), nous avons reproduit les représentations sémiotiques dans le registre des blocs et des instructions puis nous avons évalué, à l'aune des critères énoncés par Duval, leur degré de congruence (faible, moyen ou fort). Rappelons que ces critères de congruence sont : la correspondance sémantique, l'univocité sémantique et l'organisation des unités signifiantes.




```
varChaine = "maChaine"
varEntier = 2
varFlottant = 1.5
varBooleen = True
```

Figure 3 · Variable : registre des blocs et des instructions

Variable : congruence moyenne. En observant les deux représentations sémiotiques reproduites dans la figure 3, on retrouve une correspondance

sémantique entre différentes unités signifiantes, à l'exception des guillemets caractérisant les chaînes de caractères en Python non présents dans le registre des blocs. L'univocité sémantique n'est cependant pas totalement respectée, car deux éléments « mettre » et « à » correspondent au signe « = » en Scratch. L'organisation des unités signifiantes est globalement la même.


```
var1 = 4
if var1 < 0:
    print("Négatif")
else:
    print("Positif")
```

Figure 4 · Conditionnelle : registre des blocs et des instructions

Conditionnelle : congruence forte/moyenne. On peut affirmer, après l'étude de la figure 4, que la congruence est très forte entre les deux représentations. La correspondance sémantique est quasi parfaite, mis à part les « : » suivant le « else » qui n'ont pas de pendant dans le registre des blocs. L'organisation et l'ordre étant elles aussi respectées. En revanche, si le nombre de branches de la conditionnelle est supérieur à deux, on perd l'univocité sémantique, car il faut imbriquer plusieurs blocs Scratch « si-sinon » pour produire l'équivalent du « elif » en Python.


```
for var1 in range(10):
    print(var1)
```

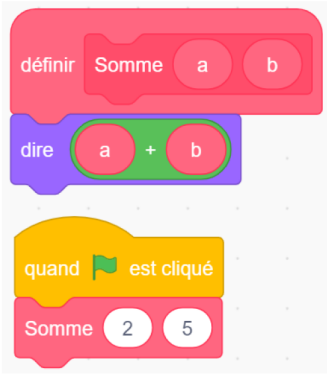
Figure 5 · Boucle bornée : registre des blocs et des instructions

Boucle bornée : congruence faible. L'examen de la figure 5 indique qu'il n'y a quasiment aucune correspondance sémantique entre les différentes unités signifiantes des deux représentations. Seul le « 10 » correspond. Remarque : Il n'y a pas de variable de boucle incrémentée automatiquement dans Scratch.


```
var1 = 10
while var1>0:
    print(var1)
    var1 = var1-1
```

Figure 6 · Boucle non bornée : registre des blocs et des instructions

Boucle non bornée : congruence faible. En observant la figure 6, nous pouvons affirmer qu'il n'y pas de correspondance sémantique entre les unités signifiantes. La condition d'arrêt étant inversée dans Scratch : « jusqu'à ce que » contre « while » (tant que) en Python.




```
def somme(a,b):
    print(a+b)

somme(2,5)
```

Figure 7 · Fonction : registre des blocs et des instructions

Fonction : congruence forte/moyenne. Après observation de la figure 7, nous pouvons avancer que le degré de congruence est très élevé pour les fonctions sans retour, car tous les critères sont respectés : correspondance sémantique, univocité sémantique et organisation des unités signifiantes. En revanche, si la fonction retourne une valeur, il n'est pas possible de

l'implémenter en l'état dans Scratch. Il faudra pour cela passer par la modification d'une variable globale ou une sortie (« dire »), ce qui diminue grandement le degré de congruence, car l'unité signifiante « return » ne correspond à rien dans Scratch.

Pour résumer, nous pouvons dire que le degré de congruence est assez disparate d'un concept à l'autre. Les variables, les conditionnelles et les fonctions permettent un changement de registre assez immédiat, là où les boucles ont des représentations assez éloignées dans les deux registres. Ces ressemblances et différences ont sans doute une influence sur l'apprentissage de la programmation en Python par les élèves. En effet, ils ont déjà rencontré certains concepts en cycle 4 dans leur représentation dans le registre des blocs. De tels connaissances antérieures peuvent constituer une aide pour les notions de variables, de conditionnelles et de fonctions (notons que, bien que disponibles dans Scratch, les fonctions ne font pas partie des attendus du collège) et plutôt un obstacle pour l'appréhension des notions de boucles (bornées et non bornées) en Python.

5.9. Type d'activité

Au cycle 4, les enseignants sont incités à proposer à leurs élèves des activités dans le registre ludique, en témoigne cet extrait du programme de mathématiques : « *Les élèves s'initient à la programmation, en développant dans une démarche de projet [...] Exemples d'activités possibles : jeux dans un labyrinthe, jeu de Pong, bataille navale, jeu de Nim, Tic-Tac-Toe, jeu du cadavre exquis.* » (MEN, 2018, p. 40). En revanche, les programmes d'enseignement de seconde invitent les enseignants à quitter le domaine ludique pour proposer aux élèves des activités en lien avec les autres parties des programmes : « *À la différence du programme de mathématiques du cycle 4 du collège, il s'agit donc d'adosser explicitement les activités de la partie algorithmique et programmation aux mathématiques.* » (MEN, 2017, p. 1). Pour illustrer, nous pouvons citer quelques exemples. Nous trouvons dans le programme de mathématiques (MEN, 2019b) parmi les activités de programmation : « *Déterminer si un entier naturel a est multiple d'un entier naturel b* » ; « *Déterminer une équation de droite passant par deux points donnés* ». Le programme de SNT (MEN, 2019c) propose lui aussi des activités en lien avec le reste du programme, citons par exemple : « *Calculer la popularité d'une page à l'aide d'un graphe simple puis programmer l'algorithme* ».

Au regard de l'âge des élèves de seconde et de leurs centres d'intérêt, il est possible que ce type d'activités, moins ludiques et plus scolaires, entraîne une diminution de leur engagement affectif (Hannula *et al.,* 2019) dans les travaux proposés. Ceci peut donc constituer un obstacle dans la transition collège-lycée.

6. Conception de la séquence

Nous exposons dans cette section les choix qui ont gouverné la conception de notre séquence d'enseignement.

6.1. Variables globales : aménagement du milieu

La séquence que nous avons conçue prend la forme d'un atelier, d'une durée de trois heures, destiné à des élèves en début de classe de seconde. Au cours de cette activité, nous mettons à la disposition des élèves, différents éléments : un dispositif programmable, une liste de défis, un environnement de développement et un support pédagogique numérique. Nous détaillons ces éléments ci-dessous.

6.1.1. Une carte programmable BBC micro:bit

Il s'agit ici de programmer un dispositif électronique pédagogique : la carte BBC micro:bit. Nous avons fait le choix d'un tel matériel dans l'objectif d'accroître l'engagement des élèves dans les activités (voir résultats de recherche en section 2). Cette carte est programmable en Python par l'intermédiaire d'une liaison USB. Elle dispose, entre autres, d'un écran constitué d'une matrice de 25 LED, de deux boutons utilisateurs et d'un capteur de luminosité.

6.1.2.Un ensemble de défis à relever

Nous avons conçu les activités sous la forme de défis à relever dans une volonté de ludification avec l'objectif d'encore renforcer l'implication des élèves (Dicheva *et al.,* 2015). Il s'agit, dans un souci de difficultés croissantes, de commencer par des affichages basiques sur la matrice de LED de la carte et de proposer ensuite la mise en œuvre de jeux simples. La résolution progressive, et dans l'ordre, de ces situations permet de gagner des « badges numériques » de développeur Python. Nous en avons créé trois, représentant trois niveaux : débutant, confirmé et expert. Ils permettent de valider, par étapes, les réussites des élèves. Le but est, ici aussi, de favoriser l'investissement des élèves dans les activités (Abramovich *et al.,* 2013 ; Gibson *et al.,* 2015). Ces badges sont des images numériques simples, ils ne respectent pas les formats standards partageables en ligne. Pour pallier ce

manque, nous avons décidé de les imprimer et de les plastifier afin de pouvoir les distribuer physiquement aux élèves au fur et à mesure de leurs succès.

6.1.3. Un EDI proposant des aides

L'environnement de développement que nous avons choisi est un environnement de développement intégré (EDI), c'est-à-dire un logiciel permettant à la fois l'édition du code et la gestion des interactions avec l'interpréteur Python (lancement des programmes, gestion des entrées-sorties, récupération des erreurs). Il s'agit du logiciel « TigerJython » issu des travaux du chercheur suisse Tobias Kohn. Au cours de sa thèse (2017), il a étudié les erreurs les plus courantes d'étudiants novices en programmation Python pour ensuite développer un EDI offrant des fonctionnalités de localisation et d'explications de ces erreurs courantes. Les messages sont affichés *in situ* et formulés en français dans un registre pratique et compréhensible par des débutants. Ce que ne permettent pas les EDI Python traditionnels qui se contentent, en général, d'afficher dans la console les erreurs détectées par l'interpréteur (messages en anglais et de nature plus technique). « TigerJython » offre également la possibilité de s'interfacer facilement avec la carte BBC micro:bit (un bouton permet, en un clic : la vérification syntaxique précoce, l'enregistrement, le téléversement et le lancement du programme depuis la carte). Les erreurs non levées par l'EDI sont classiquement détectées par l'interpréteur de la carte puis sont remontées dans la console de l'EDI.

6.1.4. Un support pédagogique numérique riche

Le support pédagogique de notre séquence se présente sous la forme d'une page Web. Nous avons opté pour un média numérique, car cela permet : de proposer des contenus plus riches et attractifs comme des animations ; d'offrir une meilleure agilité en facilitant la navigation entre les différentes parties à l'aide de liens hypertextes et d'un menu fixe sur la gauche de l'écran ; de favoriser les copier-coller et, par là même, d'éviter au maximum les erreurs syntaxiques ; de pouvoir profiter des fonctionnalités de recherche dans une page qu'offrent les navigateurs.

Ce support pédagogique est consultable en ligne (Branthôme, 2020). Il contient, dans l'ordre d'apparition, les éléments suivants :

– un guide de démarrage reprenant les grands principes de la programmation textuelle et les procédures de base permettant l'utilisation de l'environnement de développement ;

– un mémo récapitulant les principaux concepts de programmation que l'on peut mettre en œuvre à l'aide du langage Python. Nous en avons distingué cinq : les variables, les conditionnelles, les boucles bornées, les boucles non bornées et les fonctions. Nous y présentons chaque concept à travers un court texte, suivi de sa structure générique et d'un exemple illustratif ;

– une liste des 6 défis (certains comportant plusieurs actions) à relever. Pour chaque action, un texte sommaire décrit le fonctionnement attendu. Il est important de préciser que nous avons pris soin, dans cette formulation, de ne pas laisser transparaître les concepts et savoirs en jeu dans la situation dans le but de préserver au maximum leur adidacticité. Les défis sont, en majorité, illustrés d'une photo ou d'une animation permettant de lever toute ambiguïté concernant les attendus.

Enfin, il est important d'évoquer les fonctions du professeur dans cet atelier. Son rôle est de présenter succinctement le matériel aux élèves en début d'atelier puis de se mettre à leur disposition pour les aider à résoudre, en cas de nécessité, les problèmes qu'ils rencontrent. Les situations ayant été conçues en suivant le modèle adidactique, les élèves doivent cependant être autonomes vis-à-vis des savoirs en jeu, c'est à eux de trouver quels concepts, détaillés dans le mémo, doivent être mis en œuvre pour relever les défis et les actions. Nous venons de décrire les choix que nous avons faits au sujet des variables globales, nous allons maintenant entrer dans le détail des variables locales.

6.2. Variables locales : conceptions des activités

Rappelons au préalable que, sur le modèle des situations adidactiques, les savoirs en jeu ne sont pas explicités dans les situations proposées, mais sont rendus nécessaires par leur logique interne. Le tableau 1 reprend une partie des résultats de notre analyse *a priori*. Pour chaque défi et chaque action, nous exposons : une brève description de la situation (illustrée dans la figure 8), les fonctions à disposition des élèves pour interagir avec la carte, les notions en jeu en lien avec le programme ainsi que leur caractère obligatoire (O) ou facultative (F) dans la résolution du problème.

Tableau 1 · Descriptions des défis et notions en jeu

Description défis/actions	Fonctions à disposition	Notions en jeu
Défi 1 : afficher un smiley sur l'écran de la carte (figure 8-a).	**set_pixel(x,y,i)* – allume la LED à l'adresse *x,y* avec l'intensité *i*.	D1N1 : utilis. fonction – avec arg. sans retour. (O)
Défi 2 : simuler une pluie aléatoire à l'écran. Action 1 : faire tomber une goutte dans la 1re colonne (figure 8-b).	**set_pixel(x,y,i)* **sleep(t)* – met en pause l'exécution pendant *t* millisecondes.	D2A1N1 : utilis. fonction – avec arg. sans retour. (O)
Défi 2/Action 2 : faire tomber dix gouttes de suite dans la 1re colonne (figure 8-c).	**set_pixel(x,y,i)* **sleep(t)*	D2A2N1 : utilis. fonction – avec arg. sans retour. (O) D2A2N2 : boucle bornée – sans variable de boucle. (F)
Défi 2/Action 3 : faire tomber une goutte une fois dans chaque colonne (figure 8-d).	**set_pixel(x,y,i)* **sleep(t)*	D2A3N1 : utilis. fonction – avec arg. sans retour. (O) D2A3N2 : boucle bornée – avec variable de boucle. (F)
Défi 2/Action 4 : faire tomber dix fois une goutte de pluie dans une colonne à chaque fois choisie aléatoirement (figure 8- e).	**set_pixel(x,y,i)* **sleep(t)* **randint(x,y)* – retourne un entier aléatoire entre *x* et *y*.	D2A4N1 : utilis. fonction – avec arg. sans retour. (O) D2A4N2 : utilis. fonction – avec arg. avec retour. (O) D2A4N3 : affect. et utilisation d'une variable. (O) D2A4N4 : boucle bornée – sans variable de boucle. (F)
Défi 3 : jeu Pierre-feuille-ciseaux. Afficher de façon aléatoire à l'écran : une pierre, une feuille ou des ciseaux (figure 8-f).	**show(image)* – affiche des images prédéfinies à l'écran de la carte. **randint(x,y)*	D3N1 : utilis. fonction – avec arg. sans retour. (O) D3N2 : utilis. fonction – avec arg. avec retour. (O) D3N3 : affect. et utilisation d'une variable. (O) D3N4 : conditionnelle – branche « if ». (O) D3N5 : conditionnelle – branche « elif ». (F) D3N6 : conditionnelle – branche « else ». (F)
Déf 4 : jeu de rapidité. Le but du jeu est de réussir à appuyer plus de 30 fois sur le bouton A de la carte en	**scroll(mess)* – affiche un message sur l'écran de la carte. **sleep(t)*	D4N1 : utilis. fonction – avec arg. sans retour. (O) D4N2 : utilis. fonction – sans arg. avec retour. (O)

5 secondes. Tant que ce nombre n'est pas atteint, le jeu recommence.	** ba.get_presses()* – retourne le nombre d'appuis sur le bouton A	D4N3 : affect. et utilisation d'une variable. (O) D4N4 : boucle non bornée. (O) D4N5 : expression d'une chaîne de caractères. (O)
Défi 5 : station météo. Afficher à l'écran une image en fonction de la luminosité ambiante. Entre 0 et 10 : nuit étoilé ; entre 10 et 150 : un parapluie ; entre 150 et 255 : un soleil (figure 8-g).	** show(im)* ** print(val)* – permet de tracer les valeurs de luminosité dans la console. **read_light_level()* – renvoie le niveau de luminosité (0-255) qui arrive sur l'écran.	D5N1 : utilis. fonction – avec arg. sans retour. (O) D5N2 : utilis. fonction – sans arg. avec retour. (O) D5N3 : affect. et utilisation d'une variable. (O) D5N4 : conditionnelle – branche « if ». (O) D5N5 : conditionnelle – branche « elif ». (O) D5N6 : conditionnelle – branche « else ». (F)
Défi 6 : programmer un labyrinthe parcourable à l'aide d'une « bille » dirigeable par les boutons de la carte (figure 8-h). Action 1 : Compléter la fonction *dessine_mur()* qui dessine le labyrinthe.	** set_pixel(x,y,i)*	D6A1N1 : utilis. fonction – avec arg. sans retour. (O) D6A1N2 : déf. fonction – sans arg. sans retour. (O)
Défi 6/Action 2 : Compléter la fonction *depl_bille(dir)* qui déplace la « bille » en fonction de la chaîne de caractères qu'elle prend en entrée. (« H », « G » ou « D »)	** set_pixel(x,y,i)*	D6A2N1 : utilis. fonction – avec arg. sans retour. (O) D6A2N2 : déf. fonction – avec arg. sans retour. (O) D6A2N3 : affect. et utilisation d'une variable. (O) D6A2N4 : conditionnelle – branche « if ». (O) D6A2N5 : conditionnelle – branche « elif ». (F) D6A2N6 : comparaison de chaînes de caractères. (O)

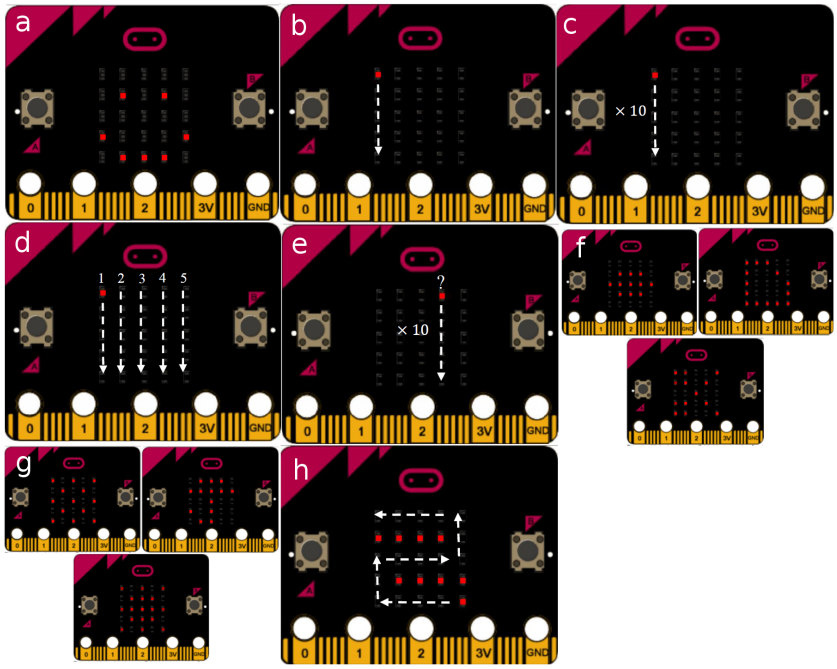


Figure 8 · Illustrations des attendus des différents défis

7. Analyses *a posteriori*

En nous appuyant sur les données récoltées lors de l'expérimentation, nous avons déterminé, dans chaque domaine, si les activités proposées ont permis de réduire les contraintes que nous avons repérées lors de l'analyse préalable.

7.1. Contenus épistémiques : apprentissages et réduction des contraintes techniques

Nous avons, en premier lieu, évalué les apprentissages des élèves dans les notions du cycle 4 à consolider (variable, conditionnelle, boucles bornées et non bornées) et dans celles à introduire en seconde (fonctions et typage). Pour cette analyse, nous nous sommes basés sur l'analyse *a priori* des défis et sur les stratégies utilisées par les élèves.

Nous pouvons distinguer deux cas, d'abord, certaines notions pouvaient être implémentées de façon facultative. Autrement dit, le problème pouvait être résolu sans les convoquer, leurs utilisations permettant cependant de fournir une solution optimale. Nous considérons

dans ce cas, à la suite de Brousseau, que si l'élève a réussi à mettre en œuvre la notion en jeu de manière autonome (en se saisissant des éléments présents dans le milieu didactique, sans l'aide explicite du professeur), c'est qu'il a pu acquérir des connaissances relatives à cette notion en lui assignant du sens. Dans le tableau 2, nous avons compilé pour chaque notion facultative contextualisée dans un défi : le nombre d'élèves ayant réussi le défi de manière autonome parmi les élèves l'ayant tenté ; le nombre d'élèves ayant implémenté cette notion parmi ceux ayant réussi le défi. Par exemple, pour la dernière ligne : la branche conditionnelle « else » était facultative dans le défi 5, ce défi a été réussi en autonomie par quatre élèves parmi les cinq l'ayant tenté, et un seul parmi eux a implémenté la branche conditionnelle « else ».

Tableau 2 · Implémentations des notions facultatives

Notions contextualisées	Réussites défi	Implé-ment.
D2A2N2 : boucle bornée.	5/6	5/5
D2A4N4 : boucle bornée.	6/6	6/6
D2A3N2 : boucle bornée + var boucle.	4/6	3/4
D3N5 : branche conditionnelle « elif ».	5/6	4/5
D6A2N5 : branche conditionnelle « elif ».	2/3	2/2
D3N6 : branche conditionnelle « else ».	5/6	1/5
D5N6 : branche conditionnelle « else ».	4/5	1/4

Nous pouvons remarquer que les boucles bornées, bien que non nécessaires, ont été quasi systématiquement utilisées lorsque le défi est réussi en autonomie (D2A2N2, D2A4N4, D2A3N2 : 14/15 - 93 %). Nous pouvons donc avancer que les élèves ont majoritairement intégré cette notion en lui attachant du sens quant à la répétition du code. En effet les stratégies mettant en œuvre cette boucle étaient optimales relativement à la longueur des programmes. La branche conditionnelle « elif » a été mise en œuvre pratiquement à chaque fois qu'elle pouvait l'être (D3N5, D6A2N5 : 6/7 – 86 %), ce qui n'est pas le cas de la branche « else » (D3N6, D5N6 : 2/9 – 22 %). Nous pouvons proposer une explication qui découle de l'observation de l'activité des élèves lors du défi 3 (première exposition à la conditionnelle). Ainsi, lors de ce défi, les élèves se sont tous appuyés sur l'exemple proposé dans le mémo sous la forme « if/elif/else », qu'ils ont essayé d'adapter dans l'objectif de distinguer trois cas. Quatre élèves ont ainsi ajouté une condition à la branche « else » (« *else var1==3* »). Lors de la tentative d'exécution qui suit, l'analyseur syntaxique de l'EDI remonte l'erreur suivante : « *'else' n'accepte pas de condition : utiliser 'elif'* ». Ces quatre élèves ont par la suite simplement modifié « else » en « elif ». Il semblerait donc que la distinction des cas que permet la structure

conditionnelle soit globalement comprise, en revanche le « court-circuit » qu'offre l'instruction « else » ne paraît acquis que par un seul élève.

Ensuite, d'autres notions devaient obligatoirement être mises en œuvre pour la réussite des défis, car rendues nécessaires par la logique interne de la situation. Dans ce cas, nous considérons également que si l'élève a réussi à mettre en œuvre la notion en jeu de manière autonome, c'est qu'il a pu acquérir des connaissances relatives à cette notion en lui donnant du sens. Nous avons rassemblé, dans le tableau 3, pour chaque notion obligatoire contextualisée dans un défi, le nombre d'élèves l'ayant implémentée de manière autonome parmi les élèves ayant tenté le défi. Par exemple, pour la première ligne : le défi 1 nécessitait l'utilisation d'une fonction avec argument et avec retour, six élèves ont réussi à implémenter cette notion en autonomie parmi les six ayant tenté le défi.

Tableau 3 · Implémentations des notions obligatoires

Notions contextualisées	Implément.
D1N1 : utilisation d'une fonction avec arg. et sans retour.	6/6
D2A1N1 : utilisation d'une fonction avec arg. et sans retour.	6/6
D2A2N1 : utilisation d'une fonction avec arg. et sans retour.	5/5
D2A3N1 : utilisation d'une fonction avec arg. et sans retour.	6/6
D2A4N1 : utilisation d'une fonction avec arg. et sans retour.	6/6
D3N1 : utilisation d'une fonction avec arg. et sans retour.	6/6
D4N1 : utilisation d'une fonction avec arg. et sans retour.	6/6
D5N1 : utilisation d'une fonction avec arg. et sans retour.	5/5
D5A1N1 : utilisation d'une fonction avec arg. et sans retour.	3/3
D6A1N1 : utilisation d'une fonction avec arg. et sans retour.	3/3
D2A4N2 : utilisation d'une fonction avec arg. et avec retour.	3/6
D3N2 : utilisation d'une fonction avec arg. et avec retour.	5/6
D4N2 : utilisation d'une fonction sans arg. avec retour.	1/6
D5N2 : utilisation d'une fonction sans arg. avec retour.	2/5
D6A1N2 : définition d'une fonction sans arg. et sans retour.	3/3
D6A2N2 : définition d'une fonction avec arg. et sans retour.	0/3
D2A4N3 : affection et utilisation d'une variable.	4/6
D3N3 : affectation et utilisation d'une variable.	5/6
D4N4 : affectation et utilisation d'une variable.	4/6
D5N3 : affectation et utilisation d'une variable.	4/5
D5A2N3 : affectation et utilisation d'une variable.	2/3
D4N4 : boucle non bornée.	5/6
D3N4 : branche conditionnelle « if ».	5/6
D5N4 : branche conditionnelle « if ».	4/5
D6A2N4 : branche conditionnelle « if ».	3/3
D5N5 : branche conditionnelle « elif ».	4/5
D4N5 : expression d'une chaîne de caractères	4/6
D6A2N7 : comparaison de chaînes de caractères	0/3

Notons d'abord que l'utilisation des fonctions ne semble pas poser de problème lorsqu'elles agissent sans retour (D1N1 -> D6A1N1 : 52/52 – 100 %). Dès lors qu'elles retournent une valeur, les élèves parviennent plus difficilement à les utiliser en autonomie (D2A4N2 -> D5N2 : 11/23 – 47 %). Lors de la définition des fonctions, c'est la gestion des arguments/paramètres qui entraîne des difficultés de mise en œuvre (D6A2N2 : 0/3 – 0 %). Nous pouvons avancer que les élèves savent utiliser une fonction, mais que leurs définitions et la gestion des entrées-sorties ne sont que partiellement acquises. Ensuite, l'utilisation de la boucle non bornée (D4N4 : 5/6 – 83 %), des branches conditionnelles « if » et « elif » (D3N4 -> D5N5 : 16/19 – 84 %) et des variables (D2A4N3 -> D5A2N3 : 19/26 – 73 %) s'effectue de façon relativement autonome. Nous pouvons arguer que les élèves ont réussi à intégrer l'utilité de ces notions, en les mobilisant à bon escient. Le typage des variables ne semble pas acquis en tant que tel, certains élèves ont été capables de délimiter des chaînes de caractères avec des guillemets (D4N5 : 4/6 – 67 %), mais ont rencontré des difficultés lors des comparaisons de chaînes (D6A2N7 : 0/3 – 0 %).

Au bilan, nous pouvons affirmer que notre séquence permet de réinvestir les notions introduites au cycle 4 (variables, boucles, conditionnelles) en leur conférant du sens, cependant elle ne permet pas la compréhension et l'intégration de tous les aspects des nouvelles notions prescrites en seconde (fonctions et typage).

L'enjeu consistait aussi à essayer de limiter la dilution des apprentissages algorithmiques prescrits dans quantité de nouveaux éléments de nature technique. En choisissant un environnement intégré dans lequel il suffit d'appuyer sur un bouton pour enregistrer le programme en cours d'édition, vérifier sa syntaxe, le téléverser sur la carte et lancer le programme à distance, nous dégageons l'élève de nombreuses considérations techniques. Cela semble d'abord avoir pour effet de limiter fortement les problèmes de nature technique qui font l'objet de seulement 5 % (6/113) des interventions du professeur.

Cet EDI propose, de plus, un système d'analyse précoce et explicite des erreurs les plus courantes. Cela constitue une aide précieuse pour les élèves qui découvrent la contrainte de correction syntaxique du code. Ce système joue, d'une certaine manière, le même rôle que les détrompeurs des blocs Scratch empêchant les erreurs de syntaxe. Même si les erreurs sont encore bien présentes (157 occurrences lors de l'atelier), ce système permet aux élèves de gagner en autonomie. En effet, seuls 13 % (15/117) des erreurs

repérées par l'EDI occasionnent une intervention du professeur contre 48 % (19/40) des erreurs issues de l'interpréteur.

Concernant les entrées-sorties, le choix de permettre aux élèves d'interagir avec un dispositif physique permet de déporter les interactions avec le programme de la console vers la carte. En effet, il n'est pas nécessaire de saisir les entrées du programme dans la console, cela se fait en agissant directement avec la carte, de même les sorties ne se manifestent pas uniquement par des messages dans la console, elles peuvent prendre toutes les formes permises par la matrice de LED de la carte. On se rapproche, en ce sens, du fonctionnement du logiciel Scratch.

Le support pédagogique numérique permet aux élèves de pratiquer très largement le copier-coller (on dénombre sur l'ensemble de l'atelier 39 copier-coller de code depuis le support pédagogique vers l'EDI). L'utilisation de cette fonctionnalité de duplication diminue mécaniquement le nombre d'erreurs syntaxiques.

En définitive, nous proposons à travers cette expérimentation une solution équidistante entre l'approche de très haut niveau offerte par Scratch, entièrement tournée vers les concepts algorithmiques et la programmation Python de plus bas niveau qui expose davantage les élèves aux éléments techniques sous-jacents.

7.2. Paradigmes et méthodes de programmation : une étape intermédiaire

Nous avons constaté un changement dans les paradigmes et méthodes de programmation mis en œuvre par les langages Scratch et Python. Notre but était donc d'essayer d'accompagner ce passage d'une programmation événementielle, orientée objet, et parallèle en Scratch à une programmation impérative et séquentielle en Python.

Lors de cet atelier, la majorité des défis avaient pour objectif de modifier l'état de l'objet « carte programmable » en accédant à ses méthodes : « *get_presses()* », « *read_light_level()* », « *set_pixel()* » ou encore « *scroll()* ». Ce genre d'approche, typique de la programmation orientée objet, se rapproche de celle de Scratch où il s'agit de modifier les propriétés d'objets « lutins » au moyen de ses méthodes : « *avance* », « *tourne* », « *pense* », « *distance de ?* », « *touche ?* » ou « *volume sonore ?* ».

Ensuite, les élèves sont amenés à lancer leur code à l'aide du bouton « *reset* » à l'arrière de la carte ou suite à l'actionnement des boutons à l'avant

de celle-ci. On peut y voir des similitudes avec la programmation événementielle mise en œuvre dans Scratch, où les programmes se lancent en réaction à des évènements.

Les multiples difficultés rencontrées par les élèves lors de la résolution du défi 4 (jeu de rapidité) proviennent pour partie du fait que cette situation n'était pas directement basée sur l'état de l'objet « carte », mais sur un compteur interne d'appuis. Ce type de problèmes relève d'une approche de la programmation plus séquentielle et impérative à laquelle les élèves ne sont pas accoutumés.

Nous pouvons, en résumé, affirmer, au regard des éléments avancés ci-dessus, que la séquence expérimentée au cours de cette ingénierie didactique, se veut être, ici aussi, une étape intermédiaire à mi-chemin entre deux paradigmes de programmation. Il faudra cependant, dans une deuxième phase, que les élèves se plient au paradigme impératif imposé par Python en classe de seconde.

7.3. Registres sémiotiques : une influence sur les apprentissages et une marge de manœuvre faible

Penchons-nous désormais sur les registres sémiotiques et l'influence sur les apprentissages des élèves du degré de congruence entre les différentes représentations des concepts en Scratch et Python. Notons qu'au moment de l'entretien, lorsque les élèves sont interrogés sur les liens qu'ils peuvent établir entre leurs activités habituelles en Scratch et ce qu'ils viennent de faire en Python, certains disent avoir remarqué des similitudes. D'abord concernant les concepts algorithmiques, quatre élèves font référence à la présence dans les deux langages de la structure conditionnelle et un à celle des boucles. D'autres éléments ressortent concernant les fonctions de service : le bloc « *dire* » est similaire à l'instruction « *print()* », le bloc « *attendre* » est équivalent à l'instruction « *sleep()* ». Trois élèves relèvent des concordances concernant la formulation impérative des instructions et la structuration des programmes : « *c'est un peu dans la même logique, si on veut qu'il fasse ça, il faut faire une ligne de code alors que c'est un bloc d'habitude* » et « *la position des blocs sur Scratch, ils étaient décalés au fur et à mesure, et là aussi* ». Les élèves sont donc en mesure d'établir des correspondances entre les deux langages. Cela permet, en particulier, de déceler chez eux des acquis notionnels antérieurs. Ces connaissances préalables constituent-elles une aide ou un obstacle dans l'apprentissage du langage Python ? Cela semble dépendre du degré de congruence entre les notions.

Ainsi, les acquis des élèves en Scratch doublés d'une congruence sémiotique moyenne forte du concept de « conditionnelle » ont sans doute contribué à la réussite rapide (temps moyen de 16,7 minutes) et générale du défi 3 (pierre-feuille-ciseaux). À l'inverse, le défi 4 (jeu de rapidité), qui met en jeu le concept de « boucle non bornée » ayant une très mauvaise congruence sémiotique, afficha un temps moyen de résolution de 47,5 minutes et donna lieu à de nombreuses difficultés. Nous avons, par exemple, pu relever des inversions de conditions (« jusqu'à »/« tant que ») symptomatiques dans les productions d'élèves.

Terminons en remarquant que notre marge de manœuvre était faible lors de la conception des activités. Au regard de notre analyse de congruence, les seuls leviers sur lesquels nous pouvions agir étaient de commencer par introduire la structure conditionnelle avec deux branches (congruence très forte dans ce cas contre moyenne avec trois branches) et de commencer avec des fonctions sans retour (congruence très forte contre moyenne pour les fonctions avec retour). Nous n'avons pas joué sur le premier paramètre ayant privilégié l'aspect ludique d'un jeu à « trois branches » (pierre-feuille-ciseaux), cela ne semble cependant pas avoir posé de problème aux élèves. Nous avons néanmoins agi sur le deuxième levier, et les résultats établis plus haut au sujet de l'utilisation autonome des fonctions sans retour semblent valider ce choix.

En résumé, les écarts sémiotiques entre les différentes représentations en Scratch et Python semblent avoir une influence sur la réussite des activités, mais la marge de manœuvre dont nous disposions lors de leur conception était très faible est ne permettait pas de soutenir véritablement l'entrée dans le registre sémiotique des instructions.

7.4. Type d'activités : un fort engagement

En dernier lieu, nous avons anticipé une baisse dans l'engagement des élèves dans les activités de programmation proposées en classe de seconde, les problèmes proposés semblant plus scolaires et moins ludiques comparativement aux activités soumises aux collégiens. L'enjeu résidait donc dans le fait de pouvoir proposer, par le biais de différents artifices, des activités motivantes mettant en œuvre le langage Python.

Plusieurs indices nous font plaider en faveur d'un fort engagement des actants dans les activités proposées. À la fin des trois heures d'ateliers, plusieurs élèves expriment leur envie de continuer à programmer la carte chez eux. D'abord un élève qui, avant la fin, lance à un autre, installé une table devant

lui : « *Je vais faire ça chez moi, direct, je vais demander à mon frère de m'en acheter un* ». Un troisième conclut, après s'être renseigné sur le prix de la carte sur Internet : « *Eh bien, je m'achète ça quand j'arrive chez moi.* ». Pendant l'entretien, lorsqu'il est demandé aux élèves s'ils seraient prêts à continuer chez eux si une carte leur était prêtée, ou s'ils conseilleraient l'atelier à d'autres personnes, ils répondent collectivement par l'affirmative.

Ensuite, en faisant remarquer aux élèves la durée de l'atelier, et en ajoutant que cela pouvait être long pour des collégiens, un élève réagit en disant : « *Ah nan, c'était tranquille* » et une autre ajoute : « *Nan, en vrai, c'est passé vite* ». Cette impression de temps qui « passe vite », indique, là aussi, un intérêt important pour le travail auquel ils venaient de prendre part. L'absence d'abandon en dépit des difficultés rencontrées par certains est aussi un élément allant dans ce sens.

Nous pouvons supposer que cet engagement dans les situations proposées est lié, pour partie, à la démarche de défi couplée à la présence de récompenses sous forme de badges. Certains éléments de l'entretien vont dans ce sens, une élève fait remarquer qu'« *il fallait qu'on réfléchisse et qu'on essaye de comprendre pas nous-même et du coup bah, quand on trouvait, c'était satisfaisant on va dire* », un autre complète en affirmant que « *des p'tits badges, ça encourage* ». Nous pouvons corroborer cette dernière déclaration par nos observations sur le terrain, ainsi, lors de l'atelier, les élèves ont réclamé leurs badges avec une certaine impatience lorsqu'ils leurs étaient dus et ils affichaient une certaine satisfaction lors de leurs remises.

Au final, nous pouvons avancer que la séquence que nous avons conçue favorise l'engagement dans les activités à la faveur du trio carte programmable-défis-badges.

8. Conclusion

En conclusion de cet article qui portait sur la programmation informatique à la transition collège-lycée, et en particulier sur le passage du logiciel Scratch au langage Python, nous allons d'abord en rappeler les principaux résultats. En suivant les principes méthodologiques de l'ingénierie didactique, nous avons débuté par une analyse préalable, qui nous a permis d'accréditer le fait que cette transition engendre des changements relatifs : aux contenus épistémiques, aux paradigmes de programmation, aux registres sémiotiques ainsi qu'aux types d'activités. Ces différences constituent, pour la plupart, des obstacles à franchir pour les élèves. Nous avons ensuite conçu une séquence d'enseignement dans

l'objectif d'accompagner les élèves dans le dépassement de ces difficultés. À l'épreuve du terrain, nous avons ensuite constaté que ces activités permettent d'agir sur les contenus épistémiques : en offrant de réinvestir les notions introduites au cycle 4 (variable, boucle et conditionnelle) en leur conférant du sens, mais aussi en constituant une première approche des savoirs ciblés en seconde (typage et fonctions) sans pour autant permettre leur acquisition et leur compréhension profonde, enfin en estompant la surcharge des apprentissages algorithmiques par certains savoirs techniques. Cette séquence propose également une étape intermédiaire dans le changement de paradigme de programmation, elle favorise un fort engagement des élèves à travers l'utilisation du trio carte programmable-défi-badge, mais ne parvient pas à soutenir véritablement l'entrée dans le registre sémiotique des instructions Python faute de leviers d'action suffisants.

Ces résultats doivent être considérés au regard des limites de notre méthodologie. Rappelons que les élèves qui ont participé à cette expérimentation sont volontaires, qu'ils montrent un intérêt préalable pour la programmation informatique, et qu'ils disposent déjà, pour certains, d'une petite expérience en la matière. Ils présentent tous un bon niveau scolaire général et de très bons résultats en mathématiques. Ce groupe de six élèves n'est donc pas représentatif d'une classe de trente-cinq élèves de seconde au niveau hétérogène et à l'intérêt variable pour l'informatique et la programmation. Notre méthodologie connaît d'autres imperfections, nous n'avons pas disposé d'assez de temps pour approfondir l'entretien avec les élèves, le double rôle professeur-chercheur est difficile à endosser pendant les expérimentations et pose la question de l'objectivité de l'analyse des activités du professeur.

Pour prolonger ce travail, il serait intéressant d'élargir le spectre de notre analyse préalable en la complétant par des analyses de terrain : ressources pédagogiques existantes (manuels scolaires, ressources numériques, etc.), observation de l'activité « ordinaire » des élèves en classe de troisième et de seconde. Il est également envisageable de concevoir une nouvelle itération de l'ingénierie basée sur les résultats du présent travail, mais également sur les nouveaux éléments émanant de cette analyse préalable élargie ainsi que d'une revue de littérature plus poussée notamment au sujet des enjeux psychologiques et cognitifs liés aux langages de programmation, à la programmation tangible ainsi qu'à la ludification des activités. Enfin, cette nouvelle ingénierie pourrait être testée sur des classes en début de seconde dans des conditions écologiques.

REMERCIEMENTS

Ce travail a été soutenu par la Région Bretagne et l'Université de Bretagne occidentale.

RÉFÉRENCES

Abiteboul, S. et Dowek, G. (2017). *Le temps des algorithmes*. Le Pommier.

Abramovich, S., Schunn, C. et Higashi, R. M. (2013). Are badges useful in education? It depends upon the type of badge and expertise of learner. *Educational Technology Research and Development, 61*(2), 217-232.

Artigue, M. (1988). Ingénierie didactique. *Recherches en didactique des mathématiques, 9*(3), 281-308.

Baron, G.-L. et Drot-Delange, B. (2016). L'informatique comme objet d'enseignement à l'école primaire française ? Mise en perspective historique. *Revue française de pédagogie, 2*, 51-62.

Bessot, A. (2003). Une introduction à la théorie des situations didactiques. *Les Cahiers du laboratoire Leibniz, 91*, 1-28.

Blikstein, P. (2013). Gears of our childhood: Constructionist toolkits, robotics, and physical computing, past and future. Dans J. Hourcade, E. Miller et A. Egeland (dir.), *Proceedings of the 12th International Conference on Interaction Design And Children* (p. 173-182). ACM.

Branthôme, M. (2020). *Atelier Python*. Github. https://matthieu-branthome.github.io/activite/index

Brousseau, G. (1981). Problèmes de didactiques des décimaux. *Recherches en didactique des mathématiques, 2*(1), 37-125.

Brousseau, G. (1998). *Théorie des situations didactiques : Didactique des mathématiques 1970-1990*. La Pensée sauvage.

Brousseau, G. (2010). *Glossaire de quelques concepts de la théorie des situations didactiques en mathématiques*. http://guy-brousseau.com/wp-content/uploads/2010/09/Glossaire_V5.pdf

Caron, P.-A., Fluckiger, C., Marquet, P., Peter, Y. et Secq Y. (2020). Éditorial des actes du colloque. Dans P.-A. Caron, C. Fluckiger, P. Marquet, Y. Peter et Y. Secq (dir.), *L'informatique, objets d'enseignements enjeux épistémologiques, didactiques et de formation. Actes du colloque DIDAPRO 8 – DIDASTIC* (p. 6-10). Université de Lille.

Delmas-Rigoutsos, Y. (2020) Variables, grandeurs et types. Dans P.-A. Caron, C. Fluckiger, P. Marquet, Y. Peter et Y. Secq (dir.), *L'informatique, objets d'enseignements enjeux épistémologiques, didactiques et de formation. Actes du colloque DIDAPRO 8 – DIDASTIC* (p. 38-51). Université de Lille.

Dicheva, D., Dichev, C., Agre, G. et Angelova, G. (2015). Gamification in education: a systematic mapping study. *Educational Technology & Society, 18*(3), 75–88.

Duval, R. (1993). Registres de représentation sémiotique et fonctionnement cognitif de la pensée. *Annales de didactique et de sciences cognitives, 5*, 37-65.

Floyd, R. W. (1978). The paradigms of programming. *Communications of the ACM, 22*(8), 455-460.

Fluckiger, C. (2019). *Une approche didactique de l'informatique scolaire*. Presses universitaires de Rennes.

Gibson, D., Ostashewski, N., Flintoff, K., Grant, S. et Knight, E. (2015). Digital badges in education. *Education and Information Technologies*, *20*(2), 403-410.

Hannula, M. S., Leder, G. C., Morselli, F., Vollstedt, M. et Zhang, Q. (2019). *Affect and Mathematics Education: Fresh Perspectives on Motivation, Engagement, and Identity*. Springer.

Hodges, S., Sentance, S., Finney, J. et Ball, T. (2020). Physical computing: A key element of modern computer science education. *Computer*, *53*(4), 20-30.

Hudak, P. (1989). Conception, evolution, and application of functional programming languages. *ACM Computing Surveys*, *21*(3), 359-411.

Journault, M., Lafourcade, P., Poulain, R. et More, M. (2020). Une preuve pour le lycée et l'indécidabilité du problème d'arrêt. Dans P.-A. Caron, C. Fluckiger, P. Marquet, Y. Peter et Y. Secq (dir.), *L'informatique, objets d'enseignements enjeux épistémologiques, didactiques et de formation. Actes du colloque DIDAPRO 8 – DIDASTIC* (p. 124-135). Université de Lille.

Khazaei, B. et Jackson, M. (2002). Is there any difference in novice comprehension of a small program written in the event-driven and object-oriented styles? Dans *Proceedings IEEE 2002 Symposia on Human Centric Computing Languages and Environments* (p. 19-26). IEEE.

Kohn, T. (2017). Teaching Python programming to novices: Addressing misconceptions and creating a development environment [Thèse de doctorat]. ETH Zurich.

Libert, C. et Vanhoof, W. (2020). Introduire la concurrence en début de seconde ? Dans P.-A. Caron, C. Fluckiger, P. Marquet, Y. Peter et Y. Secq (dir.), *L'informatique, objets d'enseignements enjeux épistémologiques, didactiques et de formation. Actes du colloque DIDAPRO 8 – DIDASTIC,* (p. 112-123). Université de Lille.

MEN. (2016). *Ressource d'accompagnement en algorithmique et programmation du programme de mathématiques (cycle 4).* Éduscol. https://cache.media.eduscol.education.fr/file/Algorithmique_et_programmation/67/9/RA16_C4_MATH_algorithmique_et_programmation_N.D_551679.pdf

MEN. (2017). *Ressource d'accompagnement en algorithmique et programmation du programme de mathématiques de seconde.* Éduscol. https://cache.media.eduscol.education.fr/file/Mathematiques/73/3/Algorithmique_et_programmation_787733.pdf

MEN. (2018). Programme d'enseignement cycle des approfondissements (Cycle 4). Dans *Bulletin officiel nº 30 du 26 juillet 2018.* MEN. https://cache.media.education.gouv.fr/file/30/62/8/ensel169_annexe3_985628.pdf

MEN. (2019a). *Préambule des ressources d'accompagnement en algorithmique et programmation du programme de mathématiques seconde et première.* Éduscol. https://cache.media.eduscol.education.fr/file/Mathematiques/34/0/RA19_Lycee_GT_2-1_MATH_preambule-algorithmique-programmation_1172340.pdf 127

MEN. (2019b). Programme de mathématiques de seconde générale et technologique. Dans *Bulletin officiel spécial n° 1 du 22 janvier 2019*. MEN. https://cache.media.education.gouv.fr/file/SP1-MEN-22-1-2019/95/7/spe631_annexe_1062957.pdf

MEN. (2019c). Programme de sciences numériques et technologie de seconde générale et technologique. Dans *Bulletin officiel spécial n° 1 du 22 janvier 2019*. MEN. https://cache.media.education.gouv.fr/file/SP1-MEN-22-1-2019/08/5/spe641_annexe_1063085.pdf

Merkouris, A., Chorianopoulos, K. et Kameas, A. (2017). Teaching programming in secondary education through embodied computing platforms: Robotics and wearables. *ACM Transactions on Computing Éducation, 17*(2), 1-22

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic Books.

Przybylla, M. et Romeike, R. (2014). Key competences with physical computing. Dans T. Brinda, N. Reynolds, R. Romeike et A. Schwill (dir.) *KEYCIT 2014-Key Competencies in Informatics and ICT* (p. 351-361). University of Potsdam.

Rogalski, J. (2015). Psychologie de la programmation, didactique de l'informatique : déjà une histoire... Dans G.-L. Baron, E. Bruillard et B. Drot-Delange (dir.), *L'informatique en éducation : perspectives curriculaires et didactiques* (p. 279-305). Presses universitaires Blaise-Pascal.

Rubio, M. A., Hierro, C. M. et Pablo, A. (2013). Using arduino to enhance computer programming courses in science and engineering. Dans L. Gómez Chova, A. López Martínez et I. Candel Torres (dir.), *Proceedings of EDULEARN13 conference* (p. 5127-5133). IATED.

Sentance, S., Waite, J., Hodges, S., MacLeod, E. et Yeomans, L. (2017). Creating cool stuff : Pupils' experience of the BBC micro:bit. Dans *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (p. 531-536). ACM.

Sentance, S., Waite, J., Yeomans, L. et MacLeod, E. (2017). Teaching with physical computing devices: The BBC micro:bit initiative. Dans E. Barendsen et P. Hubwieser (dir.), *Proceedings of the 12th Workshop on Primary and Secondary Computing Education* (p. 87-96). ACM.

Stefik, M. et Bobrow, D. G. (1985). Object-oriented programming: Themes and variations. *AI magazine, 6*(4), 40-62.

Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. *Cognitive science, 12*(2), 257-285.

White, G. et Sivitanides, M. (2005). Cognitive differences between procedural programming and object oriented programming. *Information Technology and management, 6*(4), 333-350.

Les métaphores conceptuelles : un premier pas vers une éducation critique à la robotique

▶ **Fanny BORAITA** (IRDENA, Université de Namur), **Anne-Sophie COLLARD** (NADI, Université de Namur), **Julie HENRY** (NADI, Université de Namur)[1]

■ **RÉSUMÉ** • Deux activités d'éducation à la robotique font l'objet d'une étude exploratoire : la première confronte des robots pédagogiques à 140 enfants (3-10 ans) et la deuxième permet à 13 adolescents (8-15 ans) de construire leur robot. Le focus est mis sur les métaphores utilisées spontanément par les formateurs et apprenants. Leur exploration vise à approfondir la compréhension des interactions entre formateurs, apprenants et robots afin de réfléchir à des problématiques à aborder dans une éducation critique à la technologie.

■ **MOTS-CLÉS** • éducation critique à la technologie, éducation à la robotique, représentations, analyse du discours, étude exploratoire.

■ ***ABSTRACT*** • *Two robotics education activities are the subject of an exploratory study: the first one confronts 140 children (3-10 years old) with educational robots and the second one allows 13 teenagers (8-15 years old) to build their robot. The focus is on the metaphors used spontaneously by the trainers and learners. Their exploration aims at deepening the understanding of the interactions between trainers, learners and robots in order to reflect on issues to be addressed in a critical education to technology*

■ ***KEYWORDS*** • *critical technology education, robot literacy, robot education, representations, speech analysis, exploratory study.*

[1] Les trois autrices ont contribué à parts égales à cette étude et sont donc mentionnées par ordre alphabétique.

1. Introduction

L'enseignement du code et l'éducation à la robotique dès le plus jeune âge sont intégrés, ou en cours d'intégration, dans les systèmes éducatifs partout dans le monde (Bers *et al.,* 2014 ; Malvezzi, 2021). Cependant, en Belgique francophone, l'informatique, y compris la pensée informatique, est jusqu'ici presque absente de l'enseignement obligatoire, notamment pour les élèves de 5 à 18 ans (Henry et Joris, 2016 ; Joris et Henry, 2014). Le manque de formation des enseignants (Henry et Joris, 2013) est souvent cité comme raison à cette absence. Pourtant, dans un avenir proche, ils seront amenés non seulement à intégrer des outils numériques dans leurs classes, mais aussi à enseigner des compétences numériques (Henry et Smal, 2018).

La promotion de cet enseignement est largement véhiculée par les médias et soutenue au niveau politique, à travers différents plans de développement des secteurs Technologies de l'information et de la communication (TIC) et des plans de financement d'outils numériques dans les écoles. En outre, les ressources se multiplient sur Internet : témoignages, vidéos, activités clés-sur-porte (testées ou non en classe), conseils en tous genres, comparaisons techniques de robots, etc. Des activités éducatives intégrant la robotique sont mises en place dans certains établissements scolaires, ainsi que dans des lieux d'éducation non formelle. Cependant, les aspects didactiques, l'évaluation des compétences acquises grâce à ce type d'activités et les articulations interdisciplinaires (Malinverni *et al.,* 2021) ont été peu questionnés et problématisés. L'introduction à la robotique dans l'enseignement est souvent limitée à ses aspects purement techniques (Alimisis et Kynigos, 2009 ; Ferrada-Ferrada *et al.,* 2020 ; Kubilinskiené *et al.,* 2017 ; Rusk *et al.,* 2008 ; Stone et Farkhatdinov, 2017) sans que soit pris en compte ses liens étroits avec la société (Riek et Howard, 2014 ; Sullins, 2015 ; Zawieska, 2020).

L'étude exploratoire décrite dans cet article vise à participer à la construction d'un modèle d'éducation critique à la robotique, incluant les dimensions techniques, sémiotiques et sociales de l'objet technologique/informatique (Henry *et al.,* 2018). La dimension sociale réfère notamment aux moyens de nouer une relation appropriée avec les robots intelligents (Suto, 2013). Ces relations peuvent être profondes, les enfants considérant les robots comme des entités sociales (Fior *et al.,* 2010 ; Fridin et Belokopytov, 2014 ; Kahn *et al.,* 2012 ; Scopelliti *et al.,* 2004).

La compréhension des concepts numériques et de leurs enjeux sociétaux est particulièrement dépendante des discours tenus dans les

activités d'éducation au numérique. Des métaphores y sont utilisées de manière spontanée, révélatrices du mode de pensée et des conceptions (Levin et Wagner, 2006 ; Moser, 2000). Ces métaphores forgent les représentations des environnements numériques (Manches *et al.*, 2021). En explorant les métaphores spontanées présentes dans les discours tenus lors d'activités de formation à la robotique, l'étude cherche à approfondir la compréhension des interactions entre les apprenants, les formateurs et le robot et à en appréhender les enjeux en termes éducatifs. Deux activités de robotique respectivement destinées à des enfants et à des adolescents ont constitué les cas d'étude. Les discours des élèves ont été analysés pour identifier les métaphores spontanément mobilisées à la fois par les formateurs et par les apprenants, mais également pour déterminer le rôle qu'elles jouent au sein des activités.

La principale contribution de cette étude est, à travers l'exploration des métaphores utilisées spontanément et la définition de leurs rôles, l'identification et la formulation de problématiques à aborder dans le cadre d'une éducation critique à la technologie, et plus spécifiquement à la robotique.

2. Travaux antérieurs

2.1. Les robots

Selon Lambert, un robot est « *un système formé par un réseau complexe et interactif de capteurs, de processeurs et d'actionneurs, agissant dans un environnement de manière partiellement ou totalement indépendante de l'humain* » qui en fait l'usage (2019, p. 10). Il distingue deux types de robots :

– les **robots mécatroniques** sont des constructions complexes d'éléments mécaniques et électroniques ;

– les **robots électroniques**, appelés « bots », agissent sur les réseaux sociaux, les e-mails ou les bases de données.

De plus, le degré d'autonomie des robots est variable :

– les robots peuvent être des **systèmes automatiques**. Par exemple, des robots dont les instructions sont ordonnées *a priori* par le programmeur ;

– les robots peuvent être des **robots autonomes**. Par exemple, des robots dont l'ensemble des comportements possibles n'est pas entièrement ordonné par le programmeur.

L'étude présentée ici se concentre sur des activités impliquant des robots qui sont à la fois des robots mécatroniques, tangibles, dont les actions peuvent être directement observées à travers leurs mouvements, et des robots automatiques « *opérant dans des environnements parfaitement déterminés et connus, dont les actions, totalement prévisibles, sont régies par les instructions strictes de leur logiciel, mettant en œuvre les intentions des programmeurs* » (Lambert, 2019, p. 12).

Introduits dans l'enseignement, les robots sont souvent associés à des technologies qui favorisent l'apprentissage (Gaudiello et Zibetti, 2013, 2016). On parle de robotique éducative (Alimisis, 2013 ; Romero et Sanabria, 2017). Les robots sont utilisés comme outils numériques à des fins pédagogiques dans diverses matières, notamment en mathématiques, en sciences et en sciences de l'ingénieur, mais aussi dans des matières plus éloignées (Benitti, 2012 ; Felicia et Sharif, 2014). Ils favorisent également le développement de certaines compétences métacognitives (*soft skills*) (Eguchi, 2014 ; Gaudiello et Zibetti, 2013 ; Romero et Sanabria, 2017). De plus, différentes études montrent qu'ils encouragent la motivation des enfants à apprendre, notamment par le biais de la narration d'histoires (Benitti, 2012 ; Kory et Breazeal, 2014) ou l'utilisation de la compétition (Bazyley *et al.,* 2014 ; Ma et Williams, 2013 ; Sklar *et al.,* 2002).

Mais le robot n'est pas seulement un moyen d'apprentissage, il peut aussi être l'objet d'un apprentissage. On parle alors d'éducation à la robotique. Trois orientations peuvent être données à cette éducation (Henry *et al.,* 2018) : les robots peuvent être utilisés (1) pour initier les apprenants aux concepts de l'informatique et de la pensée informatique dans l'optique d'une formation fondamentale, (2) pour développer les secteurs des TIC/STEM[2] par une formation spécifique qui vise à développer une expertise ou (3) pour former tous citoyens à la « culture numérique ». L'approche proposée dans cet article s'inscrit dans cette troisième orientation qui considère l'éducation à la robotique comme faisant partie d'une éducation critique à la technologique, discutant le « *rôle de la technologie dans les sociétés et la vie quotidienne des gens* » (Saariketo, 2014).

Gaudiello et Zibetti (2013) distinguent deux types de robots utilisés dans l'éducation.

[2] STEM : sciences, technologies, mathématiques et ingénierie

– Les « **robots à utiliser** », qui sont souvent de type humanoïde (ou animal). Ce sont des « boîtes noires ». Leurs composants ne sont ni manipulables ni observables. Leur fonctionnement technique interne n'est pas directement compréhensible. Ces robots favorisent des perceptions et des interactions avec la machine qui semblent proches de celles qu'on peut avoir avec des êtres vivants. Toutefois, ils peuvent conduire à certaines frustrations dues aux capacités d'interaction en réalité limitées des robots avec les êtres vivants (Kerepesi *et al.,* 2006 ; Melson *et al.,* 2009 ; Robinson *et al.,* 2013).

– Les « **robots en kit à construire** » sont des « robots avec lesquels penser ». Leurs composants sont observables et manipulables. Leur fonctionnement technique interne est plus facilement appréhendable. Ces robots sont plus favorables à l'éducation des enfants, leur donnant « *la possibilité de devenir un auteur plutôt qu'un consommateur de technologie* » (Gaudiello et Zibetti, 2013).

L'étude décrite ici se concentre sur des activités impliquant les deux types de robots.

2.2. Les métaphores conceptuelles

Selon la théorie de la métaphore conceptuelle (Lakoff et Johnson, 1980), les métaphores utilisées naturellement dans le langage ne sont pas des artifices linguistiques. Elles sont des indicateurs de nos représentations mentales. Elles révèlent des processus cognitifs qui permettent de se représenter le monde. « *Notre système conceptuel ordinaire, qui nous sert à penser et à agir, est de nature fondamentalement métaphorique* » (Lakoff et Johnson, 1980). Les projections métaphoriques permettent de comprendre et de vivre quelque chose, souvent plus abstrait, en termes de quelque chose d'autre, plus concret ou physique. La projection métaphorique n'est pas complète, en ce sens que le concept visé ne devient pas tout à fait le concept métaphorique. Elle conduit cependant à se concentrer sur certains aspects du concept visé, mis en évidence par le concept métaphorique mobilisé, tout en masquant les autres aspects. Par exemple, dans la phrase « vos arguments sont indéfendables », la discussion est considérée en partie comme une guerre. La métaphore utilisée met en évidence des aspects de la discussion qui sont similaires à la guerre, comme le fait de devoir se défendre, tout en masquant d'autres aspects présents dans la discussion comme la coopération, entre autres.

Les métaphores spatiales sont des processus cognitifs fondamentaux qui permettent de développer une compréhension fondée sur l'expérience

physique et directe de l'environnement ou des objets. Par exemple, le bonheur est évoqué en termes de verticalité : plus de bonheur est un sommet. Les métaphores spatiales sont utilisées sans s'en rendre compte et c'est pourquoi, en tant qu'observateur, il est difficile de les identifier au premier coup d'œil. À côté de ces métaphores qui sont fondamentales mais peu riches, Lakoff et Johnson (1980) identifient les métaphores structurelles. Elles mobilisent un concept métaphorique structurellement plus complexe et conduisent à des compréhensions particulières des domaines conceptuels visés, comme la guerre dans la métaphore « la discussion est la guerre ».

Plusieurs travaux ont mis en évidence l'utilisation spontanée de métaphores pour représenter les environnements numériques, en particulier les métaphores spatiales (Collard et Fastrez, 2010 ; Collard *et al.,* 2012), mais aussi des métaphores structurelles qui rendent ces environnements concrets ou physiques (Barr *et al.,* 2003 ; Madsen, 2000). Ces métaphores ne sont pas neutres au niveau des représentations et des interactions avec les environnements numériques qu'elles impliquent. Collard (2012) a notamment montré leur influence sur la compréhension et les comportements de navigation des utilisateurs.

3. Méthodologie, contextes et objectif de recherche

Les études de cas multiples permettent de comprendre un phénomène contemporain en profondeur et dans son contexte réel (Alexandre, 2013 ; Yin, 2009). Dans le cas de cette étude, deux cas constituent des lieux d'observation différents : un premier cas mobilisant trois « robots à utiliser » dans un contexte scolaire (étude de cas 1) ; le second se basant sur un « robot en kit » à construire lors d'une activité extrascolaire (étude de cas 2). Tous les robots sont mécatroniques et automatiques.

3.1. Contexte 1 : étude de cas 1

Dans la première étude de cas, une séquence de quatre animations de 40 minutes et une évaluation formative ont été mises en place auprès d'enfants âgés de 3 à 10 ans, issus de la même école. Au total, 140 enfants de sept classes différentes (une par niveau) ont participé à l'étude. Les formateurs étaient deux chercheuses, une informaticienne et une pédagogue. Les enseignants habituellement en charge des enfants participaient en tant qu'observateurs.

La première animation consistait à familiariser les enfants avec le matériel informatique et à déshumaniser le robot. Les enfants ont d'abord dessiné leurs représentations d'un robot et d'un ordinateur. Ensuite, ils ont découvert le matériel informatique en construisant leur propre ordinateur, en papier (matériel disponible sur Hello Ruby : http://www.helloruby.com/play/2) (figure 1). Enfin, ils ont manipulé de vrais composants électroniques.

Figure 1 · Construire son propre ordinateur

Dans les trois animations suivantes, les robots Bee-bot, Blue-bot, et/ou Ozobot ont été utilisés, selon l'âge des enfants. Ces robots ont été sélectionnés pour leurs différents modes d'interaction (boutons, barre de programmation ou application mobile sur une tablette) et leur popularité dans l'enseignement.

La deuxième animation visait à apprendre le langage de programmation des trois robots, sachant que Bee-bot et Blue-Bot réagissent au même langage. Il s'agit d'une activité débranchée. Les enfants disposaient d'un jeu de cartes (figure 2) qui reproduisait les instructions associées aux différents robots. Un enfant jouait le rôle d'un robot dans un labyrinthe géant. Les autres ont écrit, avec les cartes, un programme permettant de faire sortir le robot du labyrinthe. Dans un premier temps, la solution a été codée avec le langage du robot Bee-Bot. Une fois vérifiée, elle a été traduite dans la langue du robot Ozobot.

Figure 2 · Un jeu de cartes pour programmer

L'objectif des troisièmes et quatrièmes animations était d'amener les enfants à écrire un programme, à le tester en programmant le robot et à le corriger et/ou l'optimiser (en utilisant les concepts de programmation de variable et de boucle). Cet exercice a été reproduit deux fois (2 x 40 minutes) afin que les enfants puissent manipuler deux robots possédant des langages de programmation différents.

Plusieurs labyrinthes ont été proposés (figure 3). Les enfants ont écrit des programmes pour faire bouger les robots dans ces labyrinthes.

La séquence des animations s'est terminée par une évaluation formative. Individuellement, les enfants ont fait des dessins d'un robot et d'un ordinateur pour les comparer avec leurs représentations initiales. Ils ont également dû corriger sur papier un programme permettant au robot de sortir d'un labyrinthe donné. Ensuite, par groupe de trois, les enfants ont discuté de leurs productions, mais aussi de leur compréhension des concepts de base de la programmation (variable et boucle) avec un formateur. Enfin, un débriefing avec les enseignants a également été organisé pour réfléchir sur le matériel utilisé, les activités mises en place et pour qu'ils expriment leur ressenti par rapport aux enfants.

Figure 3 · Programmer les robots

3.2. Contexte 2 : étude de cas 2

La deuxième étude de cas consiste en une activité d'initiation à la robotique réalisée dans le cadre d'un stage de vacances pour enfants et adolescents. Cinq enfants âgés de 8 à 12 ans et six adolescents de 12 à 15 ans ont participé à cette activité sur cinq demi-journées (complétées par un apprentissage des langues). Le formateur était un ingénieur en robotique.

Les objectifs de l'activité étaient de concevoir, construire et programmer un « robot en kit » (conçu par le formateur). Deux niveaux ont été proposés en fonction de l'âge des participants :

– les enfants devaient programmer le robot pour qu'il se déplace dans un circuit, en interaction avec son environnement ;

– les adolescents devaient programmer le robot pour qu'il forme un dessin de leur choix sur une feuille de papier à l'aide d'un feutre.

D'abord, les participants ont dessiné les éléments du robot (figure 4) sur le logiciel de conception 3D Tinkercad (https://www.tinkercad.com/). Ils ont ensuite imprimé ces éléments *via* une imprimante 3D.

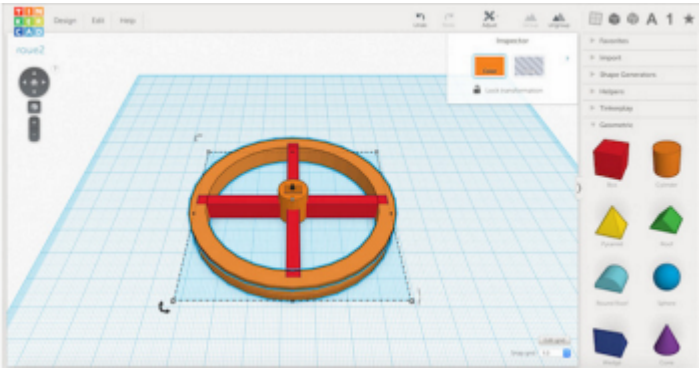


Figure 4 · Concevoir le robot

Ensuite, les participants ont assemblé les différentes pièces imprimées, une carte Arduino et divers capteurs pour construire le robot (figure 5). Enfin, ils ont programmé le robot conçu *via* l'application Blockly@rduino (http://www.techmania.fr/BlocklyDuino/).

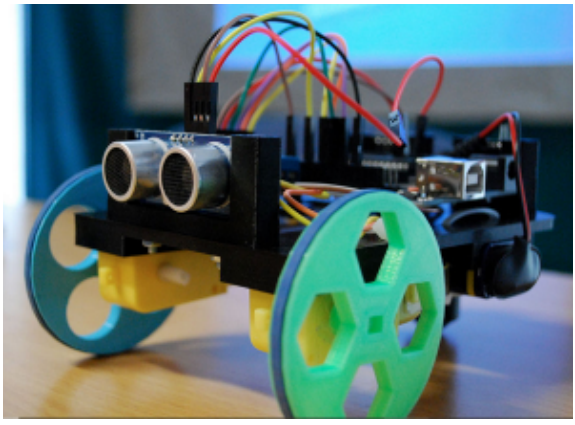


Figure 5 · Concevoir le robot

3.3. Objectif de recherche

Cette étude vise à identifier, suivant une démarche qualitative, les différentes représentations du robot, et plus largement de la machine et de ses composants, que les apprenants et les formateurs mobilisent dans ces deux activités d'éducation à la robotique. Plus précisément, elle vise à identifier les métaphores conceptuelles, spatiales et structurelles, qui sont observables dans le langage des apprenants et des formateurs et à analyser

le rôle de ces métaphores au cours de l'activité. En explorant ces métaphores, l'étude cherche à approfondir la compréhension des interactions entre les apprenants, les formateurs et le robot et à en appréhender les enjeux en termes éducatifs.

3.4. Collection des données

Dans l'étude de cas 1, toutes les activités, ainsi que les discussions menées lors de l'évaluation formative, ont été filmées pour capturer les interactions des enfants (entre eux, avec le formateur, avec le matériel mis à leur disposition et/ou le robot) et permettre aux chercheurs d'écouter leurs verbalisations.

Pour l'étude de cas 2, des données ont été recueillies auprès des deux groupes de participants (enfants et adolescents) pendant deux jours par observation ethnographique. L'ensemble des interactions ont été consignées dans un carnet de notes par le chercheur dans sa position d'observateur externe.

3.5. Analyse des données

Les métaphores utilisées dans les discours des formateurs et des apprenants, ainsi que dans leurs interactions entre eux et avec la machine, ont été identifiées en s'inspirant de la méthode « metaphor identification procedure » (MIP) (Pragglejaz Group, 2007). Il s'agit, pour commencer, de cerner le sens général du discours soumis à l'analyse, à savoir ici, le contexte des activités éducatives dans lesquelles les discours prennent place. Ensuite, la méthode vise à identifier les unités lexicales qui feront l'objet de l'analyse. Nous nous sommes centrées sur les unités lexicales portant sur les technologies manipulées, c'est-à-dire l'ordinateur, ses composants, le robot et l'imprimante 3D (pour le second cas d'étude). La dernière étape consiste à appliquer une grille en trois points à chaque unité lexicale pour identifier la présence d'un sens métaphorique. Premièrement, le sens de l'unité lexicale est identifié dans le contexte du discours, c'est-à-dire, pour le cas qui nous concerne, relativement aux technologies utilisées dans les deux activités éducatives. Deuxièmement, il s'agit de déterminer si l'unité lexicale peut prendre un sens plus fondamental dans d'autres contextes que celui du discours analysé. Les sens plus fondamentaux sont plus concrets, relèvent d'une action physique, sont plus précis, ou plus anciens. Par exemple, les capteurs à l'avant du robot sont considérés comme « des yeux ». Pour les enfants, les « yeux » sont plus concrets et possèdent un sens plus ancien et bien plus connu que celui de capteur. Troisièmement, pour

identifier la présence d'une métaphore, il faut que le sens en contexte analysé contraste avec ce sens plus fondamental mais peut être compris en le comparant à celui-ci. Si c'est le cas, il s'agit d'une métaphore. Dans l'exemple précédent, les capteurs ne sont pas des yeux à proprement parler, mais leur fonctionnement peut être comparé à celui des yeux sous certains aspects. Pour terminer et afin de consolider les analyses, les métaphores identifiées ont fait l'objet de discussions au sein de l'équipe de recherche.

4. Résultats

4.1. Les rôles

L'analyse du discours des formateurs et des apprenants a permis, dans un premier temps, de classer les métaphores utilisées selon le rôle qu'elles jouent dans le discours. Trois rôles ont été identifiés :

– « la métaphore qui aide à comprendre » : ce premier rôle consiste à vulgariser certains concepts informatiques ou électroniques et à comprendre le fonctionnement de l'ordinateur, du robot ou de ses composants ;

– « la métaphore qui rend tangible » : ce deuxième rôle consiste à rendre concrets certains concepts abstraits et à mettre en relation certaines fonctions ou certaines interactions de la machine avec l'expérience que nous avons de notre environnement physique ou social ;

– « la métaphore qui sert d'accroche » : ce troisième rôle participe à la construction d'une atmosphère ludique, affective et imaginaire, et intervient au niveau des relations « sociales » entre les apprenants, les formateurs et le robot.

Ces rôles ne sont pas exclusifs les uns des autres : une même métaphore peut assumer plusieurs rôles à la fois. Cependant, dans un souci de clarté et de lisibilité des résultats, les métaphores utilisées dans le langage des formateurs et des apprenants sont classées en fonction des trois rôles identifiés.

Dans les extraits des discours, le style gras met l'accent sur les unités lexicales relatives à la métaphore, celles qui justifient sa catégorisation dans un rôle.

4.1.1. La métaphore qui aide à comprendre

4.1.1.1. Dans le discours du formateur

Dans l'étude de cas 1, les formateurs ont utilisé la métaphore pour aider les enfants à comprendre les composants d'un ordinateur. Par exemple, pour expliquer ce qu'est la mémoire vive, ils utilisent la métaphore du « *livreur qui relève l'information du disque dur et l'apporte au processeur* ». La métaphore de l'armoire est utilisée pour expliquer ce qu'est le disque dur : « *c'est comme une armoire avec beaucoup de tiroirs* ».

Dans les activités de programmation, les métaphores sont également utilisées par les formateurs pour vulgariser les instructions. Par exemple, ils expliquent qu'il faut « *utiliser un langage précis, avec des flèches, pour expliquer au robot ce qu'il doit faire pour se déplacer dans les labyrinthes* ».

Dans l'étude de cas 2, avec des enfants plus âgés et des adolescents, le formateur utilise également des métaphores pour expliquer le matériel utilisé. Par exemple, il explique que le sonar fonctionne « *comme une chauve-souris qui envoie des ultrasons – ceux-ci détectent les obstacles et reviennent* ». Quant au fonctionnement de l'impression 3D des composants, il explique que « *le plastique avec lequel vous imprimez, c'est l'encre de l'imprimante* ». Comme dans l'étude de cas précédente, les métaphores utilisées font également référence à des comportements, des actions qui sont connus pour aider à comprendre comment la machine fonctionne : « *Nous avons programmé l'ordinateur et tout ce qu'il a à faire, c'est lire sur la carte où il doit aller* », « *Nous allons programmer cette lampe, nous allons lui dire quoi faire* ».

4.1.1.2. Dans le discours des apprenants

Les résultats montrent que les apprenants utilisent également des métaphores qui aident leur compréhension. Dans l'étude de cas 1, les enfants s'interrogent sur les différents composants, utilisant parfois des métaphores autres que celles utilisées par les formateurs. Par exemple, ils s'interrogent sur la mémoire RAM en utilisant la métaphore d'un **bus** : « *mais si la RAM n'a pas de roue, comment va-t-elle se déplacer ?* » Les apprenants utilisent également des métaphores liées à des comportements et des actions connues : « E*st-ce qu'elle* (la mémoire RAM) *court comme nous ? À la même vitesse ?* », « *Il* (le robot) *ne comprend pas ce que tu dis, il comprend juste "avance d'un pas"* ».

Dans l'étude de cas 2, les résultats sont similaires. D'une part, les apprenants utilisent d'autres métaphores que celles des formateurs : « *c'est*

comme un puzzle ». D'autre part, ils utilisent également des verbes pour exprimer des actions et des comportements qui leur sont spécifiques : « *mon robot, quand je lui demande de bouger, il ne bouge pas* », « *tu dois lui dire quoi faire quand il rencontre des obstacles* », « *tourne, tourne* ! », « j*e dois le programmer pour qu'il écoute* ».

4.1.2.La métaphore qui rend tangible

4.1.2.1. Dans le discours du formateur

Les résultats montrent que les formateurs utilisent des métaphores dans leur discours pour concrétiser des concepts et opérations liés au robot. Dans le cas d'étude 1 où les robots Bee-bot et Blue-bot ont été utilisés, les formateurs utilisent directement la métaphore de l'abeille dans leurs instructions pour rendre les robots concrets et manipulables : « *les robots devant vous sont des petites abeilles, une jaune et une transparente. Elles vont devoir se frayer un chemin dans le labyrinthe* ».

Dans l'étude de cas 2, le formateur utilise des métaphores pour guider les enfants et les adolescents dans leur assemblage et leur découverte du robot en leur permettant d'avoir une image concrète du fonctionnement ou du matériel. Par exemple, le formateur explique : « *un des deux yeux envoie des vibrations* », « *c'est la table à dessin, la table où vous venez déposer tous vos objets* », « *voici plusieurs familles de blocs, plusieurs bibliothèques de blocs* ». Le formateur **mime** avec son corps la façon dont le robot agit pour montrer aux enfants et aux adolescents comment le robot va tourner : « *vous voyez, voici comment il se comporte* ».

4.1.2.2. Dans le discours des apprenants

Les résultats montrent que les apprenants utilisent des métaphores qui font référence à des mots et des objets qu'ils connaissent et rencontrent dans leur vie quotidienne. Par exemple, dans l'étude de cas 1, si les formateurs ont utilisé la métaphore du livreur pour désigner la mémoire RAM, les enfants ont dit : « *elle ressemble plus à un facteur, c'est comme si elle avait des petites jambes pour aller vite porter des cartes, des enveloppes* ».

Dans l'étude de cas 2, les résultats vont dans le même sens. Les enfants et les adolescents parlent d'un **puzzle** lorsqu'ils assemblent leur robot.

4.1.3.La métaphore qui sert d'accroche

4.1.3.1. Dans le discours du formateur

Dans les deux études de cas, les résultats montrent la construction d'un cadre ludique, affectif et imaginaire à travers l'utilisation de métaphores dans les discours des formateurs.

Dans l'étude de cas 1, ils expliquent aux enfants l'activité de résolution de labyrinthe en les faisant participer à des histoires. Par exemple, pour un labyrinthe, les formateurs expliquent : « *La petite abeille doit faire la chasse aux couleurs dans le labyrinthe, en cherchant la couleur verte, puis bleue (...) et finir par la couleur violette pour sortir* », « *c'est un labyrinthe avec une forêt, un loup et le petit chaperon rouge (...) tu dois aider le loup à courir jusqu'au Petit Chaperon rouge (...) et aider le Petit Chaperon rouge à arriver chez sa grand-mère plus vite que le loup* ».

Dans l'étude de cas 2, le formateur utilise également des métaphores dans leur rôle d'accroche pour guider les enfants et les adolescents dans le montage et la programmation de leur robot en kit. « *Quand vous aurez fini de le programmer, c'est là que ça sera magique...* », « *il fait ce qu'il doit faire. On l'a programmé pour qu'il s'arrête à l'obstacle. Il aimerait continuer, mais il s'arrête. Si vous ne voulez pas qu'il le fasse, vous devez le programmer. Vous pouvez lui dire de faire demi-tour* ».

Ce type de métaphore se retrouve également dans le discours du formateur lorsqu'il fait référence au fonctionnement des robots. Par exemple, il dit : « *je suis attaqué* » (en référence à un robot qui tourne autour de lui), « *il est nerveux* » (à propos d'un robot qui va vite et tourne sur lui-même), ou « *nous allons faire un chœur de robots si tous les robots chantent en même temps* ».

4.1.3.2. Dans le discours des apprenants

Dans le discours des apprenants de l'étude de cas 1, les résultats montrent que les métaphores qui suscitent un cadre relationnel et une émotion sont fortement présentes. Les enfants s'accrochent à la famille qui entoure le « *petit robot Chaperon Rouge* » en expliquant, par exemple : « *elle va chez sa mère et elle se promène dans la forêt* ». Ils utilisent également les sentiments pour exprimer les actions des robots : « *il est fou* », « *c'est un petit coquin* » (en parlant du « *robot-loup* » qui se déplace vers le mur).

Chez les enfants plus âgés et les adolescents de l'étude de cas 2, on trouve également dans les discours de nombreuses métaphores faisant

référence aux émotions et aux sentiments ainsi qu'au cadre ludique : « *j'ai tout mis là, il est content* », « *il n'est pas sage... Pourquoi s'arrête-t-il ?* », « *il apprend vite* » (admiration), « *il n'arrête pas d'avancer, c'est bien. Il va faire le tour du monde. (...) Je te jure qu'il fait vroum vroum vroum vroum* ».

Nous assistons également à la création d'une relation émotionnelle, en particulier lorsque les apprenants parlent à leur robot ou de leur robot : « *le mien aime danser* », « *attention, tu marches sur mon robot... le pauvre petit* », « *je ne suis pas content* (en parlant au robot)... *tu dois m'écouter, je suis ton papa et tu n'as pas de maman* », « *je suis content de toi, petit robot, tu as assez travaillé pour aujourd'hui* », « *regarde mon beau petit robot, il tourne sur lui-même dans sa petite cage* », « *ah, il dessine maintenant... c'est mon petit* ».

4.2. Les types

En plus d'être classés selon les rôles endossés dans le discours, les concepts métaphoriques ont été classés, dans un deuxième temps, selon leur type. Trois types de concepts métaphoriques mobilisés à travers les différents rôles ont été identifiés : les métaphores spatiales fondamentales, les métaphores « non vivantes » et les métaphores « vivantes ».

Les métaphores spatiales sont couramment utilisées pour représenter les environnements numériques et révèlent la place particulière des concepts spatiaux dans notre fonctionnement cognitif fondamental, comme le montrent Lakoff et Johnson (1980). Elles rendent la technologie numérique concrète et compréhensible, comme le fait de parler de lieux, de trous ou de boîtes dans les applications. Par exemple, le formateur de l'étude de cas 2 fait référence à un environnement spatial lorsque les apprenants ouvrent l'application de modélisation 3D : « *On arrive sur l'environnement, l'endroit où on va faire des dessins en 3D* ». Ou encore : « *Maintenant vous effacez tout et revenez vers une table complètement vide* ».

Les métaphores « non vivantes » sont des analogies avec des objets similaires. Par exemple, l'assemblage est « *comme un puzzle* », les éléments sont « *stockés dans une bibliothèque* », un composant est « *un bus* », etc. Ces métaphores ou comparaisons permettent souvent de comprendre un fonctionnement, un environnement ou un élément. En général, dans les données analysées, elles ne sont pas globales, ne désignent pas le robot dans son ensemble, mais sont plutôt axées sur des éléments spécifiques de celui-ci.

Les métaphores « vivantes » sont utilisées pour rendre certains concepts à la fois compréhensibles, tangibles et accrocheurs. Dans les données analysées, elles peuvent être limitées à certains aspects du robot (par exemple, le sonar fonctionne comme une chauve-souris, la carte mère est un cerveau) ou être globales (par exemple, le robot est perçu comme un petit compagnon/animal domestique, ou comme une abeille).

5. Discussion

Dans la lignée des travaux de Lakoff et Johnson (1980), les résultats de cette étude exploratoire montrent sans surprise que les métaphores sont utilisées par les formateurs pour aider les apprenants à comprendre les composants d'un ordinateur ou d'un robot et leur fonctionnement. Du côté des apprenants, les métaphores qu'ils utilisent sont un indicateur de la manière dont ils les comprennent. Les métaphores permettent aussi aux formateurs de rendre plus concrets les concepts informatiques et aux apprenants de mieux les appréhender en les rendant tangibles.

Les résultats les plus intéressants proviennent du rôle d'accroche et des types de métaphores identifiés. Les résultats montrent que la machine est perçue comme vivante *a priori*, qu'il s'agisse de l'ordinateur ou des différents composants du robot. Les formateurs et les participants lui ont attribué un fonctionnement autonome (par ex. : « *il peut faire quelque chose par lui-même* ») ou certains sentiments (par ex. : « *il est heureux* »). Ces résultats vont dans le sens de l'animisme décrit par Piaget (1929), à savoir la propension à attribuer la vie et la conscience aux objets inanimés. L'intuition animiste des enfants les conduit à considérer tout objet rencontré comme ayant une intelligence, une fonction biologique, une intention et une personnalité (Carey, 1987 ; Okita *et al.,* 2005), sans pour autant se demander si cet objet est vivant (Turkle, 1995). Le moment où le robot commence à bouger exacerbe les métaphores « vivantes », il « prend vraiment vie » (par ex. : il « *court, danse, répond, refuse de faire quelque chose, agit mal,* », etc.). Il devient un personnage avec lequel les individus interagissent. Une empathie se développe envers la machine en mouvement. On peut observer le rôle que joue la narration (dans les métaphores utilisées comme accroche) pour renforcer la perception du robot comme un personnage qui « vit » une histoire. Ce phénomène peut être observé aussi bien pour les robots « boîte noire »/humanoïdes que pour les « robots en kit ». C'est donc plus le mouvement, et, derrière lui, l'idée d'une autonomie propre, que l'apparence du robot qui semble favoriser la représentation « vivante » de la machine.

Le risque des métaphores « vivantes » est une méconnaissance de la nature des interactions avec le robot et des enjeux qui y sont liés. Comme le souligne Tisseron (2015), au-delà d'un attachement aux objets qui se développe naturellement, les objets en mouvement tels que les robots génèrent une empathie émotionnelle et cognitive. Elle conduit à un transfert entre l'homme et la machine : « *le robot est comme moi et je suis comme le robot* ». Cette représentation « vivante » du robot masque le fait que la machine n'a pas d'intention propre. Un double déplacement de l'emplacement de l'intention s'opère alors. Le premier déplacement fait oublier que le robot et le cadre de l'activité ont été conçus et mis en place par les formateurs. L'intention initiale est du côté des formateurs, mais occultée. Les apprenants sont mis dans un processus créatif et apparaissent en effet comme les initiateurs (par ex. : « *vous allez créer un robot vous-même* », « *vous allez aider l'abeille* »). Le deuxième transfert déplace l'intention des apprenants vers la machine. Le robot acquiert une autonomie propre et passe du statut d'exécutant des instructions des apprenants à celui d'agent qui exécute une tâche selon sa propre volonté. Ce qui est en jeu ici, c'est le fait que ce double déplacement occulte entièrement les intentions humaines qui se traduisent dans le robot et son fonctionnement, pour mettre en avant une machine « vivante » qui opère selon ses propres intentions. Le risque est dès lors de considérer les intentions du robot comme « objectives » (allant de soi) et de ne pas pouvoir questionner la « subjectivité » (les normes et les valeurs humaines) embarquée dans la machine.

6. Conclusion et travaux futurs

Cet article présente une analyse des discours tenus par les formateurs et les apprenants lors d'activités d'initiation à la robotique pour les enfants et les adolescents. À travers l'exploration des métaphores mobilisées spontanément, il vise à observer les représentations des apprenants et leurs interactions avec les robots impliqués dans des activités éducatives afin de contribuer au développement de problématiques à aborder dans le cadre d'une éducation critique à la technologie. Les analyses sont effectuées à partir de la théorie de la métaphore conceptuelle (Lakoff et Johnson, 1980). Si les métaphores reflètent la compréhension du fonctionnement de la machine par les utilisateurs, elles peuvent en cacher certains aspects (Tisseron, 2015). Elles créent alors des attentes, des frustrations et des risques. C'est à ces aspects que cette étude accorde une attention particulière.

Elle prend forme à travers deux activités d'éducation à la robotique. La première activité a impliqué les robots BeeBot, BlueBot et Ozobot dans sept classes d'enfants de 3 à 10 ans. La seconde activité consistait en une formation de cinq demi-journées pour 13 participants de 8 à 15 ans. Ils ont conçu, construit et programmé un robot.

Outre l'identification des types de métaphores utilisées spontanément et la définition de leurs rôles au sein des discours, trois points sont plus particulièrement à souligner parmi les résultats de cette étude. Le premier concerne l'importance du mouvement et du déplacement considérés comme « autonomes », soutenus par le récit qui place le robot dans le rôle d'un personnage (métaphores utilisées comme accroche). Le mouvement renforce une relation empathique avec le robot. Deuxièmement, la métaphore vivante masque les aspects spécifiques de la machine. Elle conduit à des représentations erronées de son fonctionnement et à des attentes du « vivant » qui ne sont pas réalisées. Troisièmement, la métaphore « vivante » déplace l'intention de l'apprenant, où le robot est exécutant, vers la machine, où le robot est un agent, en occultant, d'une part, l'intention du formateur en tant que concepteur de l'activité ou du modèle de robot et, d'autre part, l'intention de l'apprenant en tant que producteur et/ou programmeur du robot.

Ces trois points constituent des problématiques à aborder dans le cadre d'une éducation critique à la technologie. Ainsi, il est nécessaire de prendre conscience de la métaphore « vivante » dans les activités d'éducation à la robotique. Si la compréhension du fonctionnement du robot est essentielle, elle ne peut se faire sans déconstruire les représentations de la machine. L'idée n'est pas d'éliminer l'utilisation des métaphores, qui ont leur rôle dans l'éducation et les processus cognitifs de compréhension, mais d'être précis sur la localisation de l'intention et la nature de l'autonomie de la machine. Une piste serait de présenter les robots comme des constructions sociales qui reflètent des intentions humaines. Des recommandations pourraient être faites aux formateurs pour les aider à formuler des discours éclairés sur l'informatique et les concepts liés à la robotique auprès des enfants.

Pour ce faire, cette étude exploratoire devrait être approfondie non seulement par une analyse plus précise des données collectées (confrontation des deux cas, prise en compte de l'âge des apprenants, etc.), mais également par une augmentation des cas d'étude.

RÉFÉRENCES

Alexandre, M. (2013). La rigueur scientifique du dispositif méthodologique d'une étude de cas multiple. *Recherches qualitatives, 32*(1), 26-56.

Alimisis, D. (2013). Educational robotics: Open questions and new challenges. *Themes in Science and Technology Education, 6*(1), 63-71.

Alimisis, A. D. et Kynigos, C. (2009). Constructionism and robotics in education. Dans Dimitris Alimisis (dir.), *Teacher education on robotic-enhanced constructivist pedagogical method,* (p. 11–26). ASPETE.

Barr, P., Biddle, R. et Noble, J. (2003). Interface Ontology: Creating a Physical World for Computer Interfaces. Dans *Proceedings of the 8th European Conference on Pattern Languages of Programms (EuroPLoP '2003),* p. 1-18.

Bazylev, D., Margun, A., Zimenko, K., Kremlev, A. et Rukujzha, E. (2014). Participation in robotics competition as motivation for learning. *Procedia-Social and Behavioral Sciences, 152*, 835-840.

Benitti, F. B. V. (2012). Exploring the educational potential of robotics in schools: A systematic review. *Computers & Education, 58*(3), 978-988.

Bers, M. U., Flannery, L., Kazakoff, E. R. et Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers & Education, 72*, 145-157.

Carey, S. (1987). *Conceptual change in childhood.* Cambridge, MA: MIT Press.

Collard, A. S. (2012) Apprendre dans un monde virtuel. *Document numérique*, 71-93.

Collard, A. S. et Fastrez, P. (2010). *A model of the role of conceptual metaphors in hypermedia comprehension* [communication]. Cognition and Media (CICOM'09), Santo Domingo.

Collard, A. S., Fastrez, P. et Brouwers, A. (2012). Convivialité et métaphores dans les interfaces de systèmes interactifs. *Interfaces numériques, 1*(3), 471.

Eguchi, A. (2014). Educational robotics for promoting 21st century skills. *Journal of Automation Mobile Robotics and Intelligent Systems, 8*(1), 5-11.

Felicia, A. et Sharif, S. (2014). A review on educational robotics as assistive tools for learning mathematics and science. *Int. J. Comput. Sci. Trends Technol., 2*(2), 62-84.

Ferrada-Ferrada, C., Carrillo-Rosúa, J., Díaz-Levicoy, D. et Silva Díaz, F. (2020). Robotics from STEM areas in primary school: A systematic review. *Education in the Knowledge Society, 22*, 1-18

Fior, M., Nugent, S., Beran, T. N., Ramirez-Serrano, A. et Kuzyk, R. (2010). Children's relationships with robots: Robot is child's new friend. *Journal of Physical Agents, 4*(3), 9–17.

Fridin, M. et Belokopytov, M. (2014). Embodied robot versus virtual agent: Involvement of preschool children in motor task performance. International *Journal of Human–Computer Interaction, 30*(6), 459–469.

Gaudiello, I. et Zibetti, E. (2013). La robotique éducationnelle : état des lieux et perspectives. *Psychologie française, 58*(1), 17-40.

Gaudiello, I. et Zibetti, E. (2016). Learning robotics, with robotics, by robotics: *Educational robotics.* John Wiley & Sons.

Henry, J., Hernalesteen, A., Dumas, A. et Collard, A.-S. (2018). Que signifie éduquer au numérique ? Pour une approche interdisciplinaire Dans *Actes du colloque Didapro 7 – DidaSTIC. De 0 à 1 ou l'heure de l'informatique à l'école*.

Henry, J. et Joris, N. (2013). Maîtrise et usage des TIC : la situation des enseignants en Belgique francophone. Dans B. Drot-Delange, E. Bruillard et G.-L. Baron (dir.) *Sciences et technologies de l'information et de la communication (STIC) en milieu éducatif*. https://edutice.archives-ouvertes.fr/edutice-00875643

Henry, J. et Joris, N. (2016). Informatics at secondary schools in the French-speaking region of Belgium: myth or reality? Dans *Actes du colloque The International Conference on Informatics in Schools: Situation, Evolution and Perspectives* (p. 13-24).

Henry, J. et Smal, A. (2018). « Et si demain je devais enseigner l'informatique ? » Le cas des enseignants de Belgique francophone. Dans *Actes du colloque Didapro 7 – DidaSTIC. De 0 à 1 ou l'heure de l'informatique à l'école*.

Joris, N. et Henry, J. (2014). L'enseignement de l'informatique en Belgique francophone : état des lieux. *Bulletin de la société informatique de France, 2*, 107–116.

Kerepesi, A., Kubinyi, E., Jonsson, G. K., Magnússon, M. S. et Miklósi, Á. (2006). Behavioural comparison of human–animal (dog) and human–robot (AIBO) interactions. *Behavioural processes, 73*(1), 92-99.

Kory, J. et Breazeal, C. (2014, August). Storytelling with robots: Learning companions for preschool children's language development. Dans *Proceeding of the 23rd IEEE international symposium on robot and human interactive communication* (p. 643-648).

Kubilinskienė, S., Žilinskienė, I., Dagienė, V. et Sinkevičius, V. (2017). Applying robotics in school education: A systematic review. *Baltic journal of modern computing, 5*(1), 50-69.

Lambert, D. (2019). *La robotique et l'intelligence artificielle*. Fidélité Eds.

Lakoff, G. et Johnson, M. (1980). *Metaphors we live by*. University of Chicago press.

Levin, T. et Wagner, T. (2006). In their own words: Understanding student conceptions of writing through their spontaneous metaphors in the science classroom. *Instructional Science, 34*(3), 227.

Ma, Y. et Williams, D. C. (2013). The potential of a First LEGO League robotics program in teaching 21st century skills: An exploratory study. *Journal of Educational Technology Development and Exchange (JETDE), 6*(2), 2.

Madsen, K. H. (2000). Magic by metaphors. Dans *Proceedings of DARE 2000 on Designing augmented reality environments* (p. 167-169).

Malinverni, L., Valero, C., Schaper, M. M. et de la Cruz, I. G. (2021). Educational Robotics as a boundary object: Towards a research agenda. *International Journal of Child-Computer Interaction, 29*, 100305.

Malvezzi, M. (2021). Education in & with Robotics to Foster 21st-Century Skills. Dans *Proceedings of EDUROBOTICS 2020*. Springer Nature.

Manches, A., McKenna, P. E., Rajendran, G. et Robertson, J. (2020). Identifying embodied metaphors for computing education. *Computers in Human Behavior, 105*, 105859.

Melson, G. F., Kahn Jr, P. H., Beck, A., Friedman, B., Roberts, T., Garrett, E. et Gill, B. T. (2009). Children's behavior toward and understanding of robotic and living dogs. *Journal of Applied Developmental Psychology, 30*(2), 92-102.

Moser, K. S. (2000). Metaphor analysis in psychology—Method, theory, and fields of application. *Forum Qualitative Sozialforschung/Forum: Qualitative Social Research, 1*(2).

Okita, S., Schwartz, D., Shibata, T., Nakamura, O. et Tokuda, H. (2005). Exploring young children's attributions through entertainment robots. Dans *Proceedings of the 14th IEEE International Workshop on Robot and Human Interactive Communication* (p. 390–395). IEEE.

Piaget, J. (1929). *The child's conception of the world*. Savage, MD: Littlefield Adams.

Pragglejaz Group (2007). MIP: A Method for Identifying Metaphorically Used Words in *Discourse, Metaphor and Symbol, 22*(1), 1–39.

Riek, L. D. et Howard, D. (2014). A code of ethics for the human-robot interaction profession. Dans *Proceedings of We Robot, 2014*. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2757805

Robinson, H., MacDonald, B., Kerse, N. et Broadbent, E. (2013). The psychosocial effects of a companion robot: a randomized controlled trial. *Journal of the American Medical Directors Association, 14*(9), 661-667.

Romero, M. et Sanabria, J. (2017). Des projets de robotique pédagogique pour le développement des compétences du XXIe siècle. Dans M. Romero, B. Lille et A. Patiño (dir.), *Usages créatifs du numérique pour l'apprentissage au XXIe siècle*. Presses de l'Université du Québec.

Rusk, N., Resnick M., Berg R. et Pezalla-Granlund M. (2008) New pathways into robotics: Strategies for broadening participation. *Journal of Science Education and Technology, 17*(1), 59-69.

Saariketo, M. (2014). Imagining alternative agency in techno-society: outlining the basis of critical technology education (EN). *Media practice and everyday agency in Europe*, 129-138.

Scopelliti, M., Giuliani, M. V., D'Amico, A. M. et Fornara, F. (2004). If I had a robot at home: Peoples' representation of domestic robots. Dans S. Keates, J. Clarkson, P. Langdon et P. Robinson (dir.), *Designing a More Inclusive World* (p. 257–266). Springer.

Sklar, E., Eguchi, A. et Johnson, J. (2002). RoboCup Junior: learning with educational robotics. Dans *Robot Soccer World Cup* (p. 238-253). Springer.

Stone, A. et Farkhatdinov, I. (2017). Robotics education for children at secondary school level and above. Dans *Proceeding of Annual Conference Towards Autonomous Robotic Systems* (p. 576-585). Springer, Cham.

Sullins, J. P. (2015). Applied professional ethics for the reluctant roboticist. Dans *Proceedings of the 10th ACM/IEEE Conference on Human-Robot Interaction (HRI2015): The Emerging Policy and Ethics of Human-Robot Interaction Workshop.*

Suto, H. (2013). Robot Literacy: An Approach For Sharing Society With Intelligent Robots. *International Journal of Cyber Society and Education, 6*(2), 139-144.

Tisseron, S. (2015). *Le jour où mon robot m'aimera : Vers l'empathie artificielle*. Albin Michel.

Turkle, S. (1995). *Life on the screen: Identity in the age of the Internet*. Simon and. Schuster.

Yin, R. K. (2009). *Case study research: Design and methods, 5*. Sage.

Zawieska, K. (2020). Roboethics as a research puzzle. Dans *Proceeding of the14th ACM/IEEE international conference on human-robot interaction, 2019 (p.* 612–613).

Analyse d'une formation des futurs enseignants relative à l'enseignement de la programmation, *via* l'outil Minecraft: Education Edition

▶ **Charline CARLOT** (Cocof), **Audrey KUMPS** (Umons), **Bruno DE LIEVRE** (Umons)

■ **RÉSUMÉ** • Dans cet article, nous nous intéressons à l'impact des profils des futurs enseignants du primaire et de mathématiques sur l'évolution de leurs perceptions et leur intention pédagogique dans l'enseignement de la programmation. Pour cela, une formation utilisant l'outil Minecraft : Éducation Edition a été proposée. Les résultats, interprétés à partir de différents modèles issus du modèle de TAM (Davis et Davis, 1989), permettent de fournir quelques pistes de réflexion pour la formation initiale et continue des enseignants.

■ **MOTS-CLÉS** • formation, programmation, perception, pédagogie du jeu vidéo, Minecraft.

■ ***ABSTRACT*** • *In this article, we are interested in the impact of the profiles of future primary and mathematics teachers on the development of their perceptions and educational intent in teaching programming. For this, training using the Minecraft tool: Education Edition has been proposed. The results, interpreted from different models derived from the TAM model (Davis et Davis, 1989), make it possible to provide some avenues of reflection for the initial and continuing training of teachers.*

■ ***KEYWORDS*** • *training, programming, perception, video game pedagogy, Minecraft.*

1. Introduction

À une époque où l'enseignement en Belgique francophone connaît un grand bouleversement avec la mise en application de la réforme éducative, intitulée « Pacte pour un enseignement d'Excellence », et l'instauration d'un nouveau référentiel numérique (FWB, 2018), il est essentiel de se demander si les enseignants se sentent compétents face à ce nouveau contenu à enseigner. En effet, malgré les actions menées par le système éducatif pour satisfaire aux attentes de la société, comme la mise en place de formations pour les enseignants et les élèves ainsi que le financement de divers projets éducatifs, un décalage entre les besoins de la société et la formation des enseignants est constaté (Singh Rajput, 2000). De plus, le milieu éducatif exprime ses difficultés face à la manipulation des nouvelles technologies et à leur incorporation pédagogique (Digital Wallonia, 2018). Pourtant, l'enseignant étant essentiel à la mise en place des pratiques numériques, il est important « *d'intégrer plus massivement le numérique dès la formation initiale des enseignants et de proposer des cursus où la composante numérique est présente de manière disciplinaire et transversale* » (Digital Wallonia, 2018, p. 1).

Parmi les compétences du programme relatives à l'éducation au numérique figure la programmation. Ce contenu d'enseignement, déjà intégré au programme de nombreux pays (Karsenti, 2019), est autant une nécessité économique pour le pays, qu'une nécessité sociétale pour les jeunes élèves et les futurs citoyens (Archambault, 2015). Il constitue également un moyen de développer des capacités de résolution de problèmes, de créativité, de pensée critique, de construction, de comptage et de lecture (Karsenti, 2019 ; Romero, 2016). L'enseignement de la programmation aux élèves contribue donc à ce que ces derniers puissent devenir des citoyens responsables dans la société actuelle (Karsenti et Bugmann, 2017a). Il est du rôle de l'école « *de dispenser les connaissances scientifiques et techniques qui permettront aux futurs citoyens d'être en phase avec la société dans laquelle ils vivent* » (Archambault, 2015, p. 12).

Il faut donc veiller à former les (futurs) enseignants à l'enseignement de cette nouvelle compétence. Pour cela, les formations doivent prendre en compte les perceptions initiales des enseignants qui peuvent influer sur l'ensemble du parcours de formation et sur leur comportement post-formation (Crahay *et al.*, 2010). De plus, au vu des usages pédagogiques mis en œuvre par de nombreux enseignants en Amérique ou en Europe, ainsi que des bénéfices transdisciplinaires et disciplinaires sur les élèves (Karsenti

et Bugmann, 2017b), il semble intéressant que la formation se réalise *via* une modalité ludique : le célèbre jeu vidéo Minecraft par exemple.

Concernant le public visé par ces formations, selon Henry et Smal (2018), les étudiants des cursus « instituteur primaire » et « AESI mathématiques » sont les plus susceptibles de se retrouver, lors de leur carrière professionnelle, à enseigner cette compétence numérique.

Dès lors, nous en sommes arrivés à l'hypothèse que l'administration d'une formation relative à l'enseignement de la programmation, *via* l'outil « Minecraft », aux futurs enseignants de primaire et aux futurs enseignants de mathématiques du secondaire inférieur, impacterait positivement leur intention ainsi que leur perception de l'utilité et de l'utilisabilité de son instruction.

2. Cadre théorique

2.1. Enseignement de la programmation

L'enseignement de l'informatique dans les écoles existe depuis les années 70, mais ce n'est que très récemment que le système éducatif belge a mis en œuvre une politique de grande ampleur pour faciliter son introduction (Guardiola, 2014). Ainsi, depuis 2015, *via* son opérateur stratégique concernant le numérique, à savoir Digital Wallonia et son programme École Numérique, la Wallonie a distribué plus de 32 000 équipements à travers 1 200 établissements scolaires et a entrepris de nombreux projets pour former les apprenants et le corps enseignant au numérique. Par exemple, le projet #Wallcode s'est focalisé sur l'initiation et la sensibilisation à la programmation et a organisé, entre 2015 et 2018, plus de 800 séances auprès de 40 000 apprenants, enfants et adultes.

De plus, en 2017, la réforme du système éducatif de la FWB a intégré officiellement l'éducation au numérique comme discipline à enseigner. Bien que les programmes ne soient pas encore publiés, le rapport intitulé « Stratégie numérique pour l'éducation » (FWB, 2019) confirme la présence du numérique, et notamment de la programmation au point « *Axe 1. Définir les contenus et ressources numériques au service des apprentissages – AP 1.1. Définir les savoirs, savoir-faire et compétences de la "société numérique" dans le cadre du nouveau tronc commun renforcé* » (p. 9). Cependant, malgré l'ensemble de ces démarches, seuls 21 % des 2 066 établissements de la FWB ayant participé à l'enquête (Digital Wallonia, 2018), affirment que l'éducation aux compétences numériques est incluse dans leur projet d'établissement. Ainsi, l'enseignement de la programmation n'est présent,

à travers une intégration transversale dans des cours obligatoires ou optionnels, que dans 10 % des établissements des Régions wallonne et bruxelloise. Plus précisément, 76 % des enseignants interrogés affirment ne posséder aucune connaissance du codage.

2.2. Impacts de l'enseignement de la programmation

Pour créer du changement dans le système éducatif et, plus précisément, auprès des enseignants, il est nécessaire de faire connaître les bénéfices de l'usage du numérique en éducation qui développerait l'impartialité, l'objectivité, l'attractivité et l'efficacité de l'éducation (De Poortere, 2017 ; Karsenti, 2019 ; Terosier, 2017, cité par Merchin, 2017).

La réduction de l'écart entre l'offre et la demande dans le marché du travail est un premier argument important dans l'introduction de la programmation dans le système éducatif. Cependant, comme le précise Karsenti, l'inclusion de la programmation dans les parcours scolaires va au-delà de la nécessité de « *former un bassin de programmeurs compétents en vue de répondre aux besoins du marché du travail* » (2019, p. 1). En effet, l'apprentissage de cette compétence numérique couvre d'innombrables avantages sociaux et éducatifs pour les élèves. L'aspect social se rattache au désir d'amener les apprenants au statut de citoyen autonome et de créateur dans notre communauté technologique. Karsenti et Bugmann (2017a) évoquent l'importance de former les élèves à la compréhension des technologies car ils seront amenés à les rencontrer dans leur vie quotidienne et professionnelle. Or, les outils numériques n'ont pas d'effets « magiques », ils sont conçus par des femmes et des hommes à travers des codes, invisibles, mais pourtant bien présents, qui vont les guider dans leurs actions et leurs comportements (Terosier, 2017, cité par Merchin, 2017). Le risque est que la fracture numérique, le fossé entre « *ceux qui savent et ceux qui ne savent pas* », se creuse et que la société favorise les premiers (De Poortere, 2017). Pour cela, il est essentiel d'intégrer tôt dans le cursus scolaire, l'apprentissage de la culture numérique et de la programmation. Comme l'ont démontré de nombreux pays (Nouvelle-Écosse, France, Royaume-Uni, États-Unis, Suède…) ceci est réalisable dès la maternelle grâce à des outils adaptés aux objectifs et à l'âge tels que : Scratch, ScratchJr, Bee-Bot, Dash, le robot humanoïde NAO, ou encore Minecraft Éducation (Karsenti et Bugmann, 2017 b).

Concernant les impacts éducatifs, la programmation permet la visualisation immédiate des résultats aux élèves ce qui augmente leur motivation et leur engagement (Desjardins *et al.,* 2018) ainsi que leur estime

de soi et leur sentiment de compétence (Karsenti, 2019). De plus, la littérature permet d'établir un premier lien entre les avantages de l'enseignement de la programmation et les « *5 compétences du XXIe siècle* » : la pensée critique, la collaboration, la résolution de problèmes, la créativité et la pensée informatique (Romero, 2016) :

– l'esprit critique est développé lors des activités de construction de programmes où les élèves sont amenés à commettre des erreurs et à les rectifier pour trouver ensuite la solution. Tout en développant leur autonomie, ils apprennent à mener une réflexion critique sur leur production et les démarches utilisées pour rechercher et gérer ces blocages en utilisant, sans crainte du jugement, une démarche d'essai-erreur (Desjardins *et al.,* 2018) ;

– la collaboration est développée à travers des projets actifs et collaboratifs où, pour arriver à une solution commune qui répondra au contexte et à la tâche donnée, les élèves sont amenés à développer et utiliser leurs compétences de communication, d'échange et d'argumentation avec leurs camarades (Karsenti, 2019) ;

– la créativité s'avère d'une grande utilité car, la programmation n'étant pas une science exacte, il existe de nombreuses démarches et solutions différentes pour répondre à une tâche ou à une situation problème ;

– la résolution de problèmes et la pensée informatique sont mises en œuvre quand les élèves doivent analyser une situation, rechercher les informations nécessaires à l'émission d'hypothèses, déterminer la solution, la rédiger sous une forme algorithmique et enfin, la traduire en un programme informatique à mettre en œuvre afin de vérifier son efficacité. Toutes ces étapes, demandant rigueur, structuration et organisation (Terosier, 2017, cité par Merchin, 2017), sont exploitées dans de nombreuses disciplines autres que l'informatique, telles que les sciences, les mathématiques ou le français (Karsenti, 2019).

2.3. Formation des enseignants

L'absence de cours d'informatique, et plus spécifiquement de programmation, auprès des élèves est majoritairement due à un manque de formation en sciences informatiques des enseignants durant leur cursus en Haute-École (Henry et Smal, 2018). En effet, plus de 60 % des enseignants de FWB interrogés affirment ne pas maîtriser, ne pas connaître ou ne pas comprendre les compétences propres à l'algorithmique et à la programmation.

Suite à la réforme éducative belge, la programmation est apparue dans le référentiel couvrant le primaire et le secondaire du degré inférieur (FWB, 2018). Pour l'enseignement de cette compétence au niveau primaire, il semble évident qu'une formation spécifique doit être dispensée dans le cursus des futurs enseignants. En ce qui concerne le niveau du secondaire inférieur, Henry et Smal, chercheuses dans le groupe de travail SI2 (sciences informatiques pour le secondaire inférieur) et dans le centre de recherche de l'université de Namur, stipulent que « *seuls les enseignants en mathématiques pourraient, à ce titre, prétendre posséder des compétences suffisantes pour assurer un cours d'initiation à l'informatique* » (2018, p. 133). De plus, les auteures affirment que, dans le monde de l'enseignement secondaire, les « enseignants "*en devenir* [...] *constituent sans doute le public le plus susceptible de se retrouver en charge d'un cours d'« éducation au numérique* » (p. 136-137). Il semble donc primordial de se centrer sur l'amélioration de la formation initiale de ces futurs enseignants.

2.4. Améliorer la formation des enseignants

De nombreuses études menées à travers le monde ont démontré que la corrélation entre la formation initiale des enseignants et les résultats des élèves était faible. Ces résultats ne mettent pas en cause l'enseignement supérieur mais plutôt l'incohérence entre la formation et les besoins du terrain (Bernard *et al.,* 2004) ainsi que le manque de liens entre les savoirs théoriques et les savoirs pratiques (Caron et Portelance, 2017). Il est donc important d'analyser les besoins professionnels des futurs enseignants avant de repenser leur formation.

De plus, les futurs enseignants, avant même leur entrée dans la formation initiale, sont soumis à des doxas qui vont fortement influencer leur formation ainsi que leur future carrière professionnelle (Crahay *et al.,* 2010). En sachant que « *plus une croyance est ancienne, plus elle est tenace* » (Kagan, 1992, cité par Vause, 2009, p. 39), les futurs enseignants ont eu le temps de se forger leurs propres croyances depuis l'école primaire. Enfin, le fossé entre le type d'enseignement attendu par les enseignants du terrain et celui qu'ils ont connu depuis le début de leurs études primaires peut être un frein pour la formation (Vause, 2009). Plus spécifiquement concernant les formations propres à l'intégration des technologies dans le monde éducatif, l'intégration de stratégies technopédagogiques peut provoquer une certaine résistance de la part des enseignants. Celle-ci serait causée par la peur du changement et la non-maîtrise des outils technologiques (Rey et Coen, 2012).

2.5. Modalités d'une « bonne » formation

Pour juger de l'efficacité d'une formation enseignante, il est nécessaire d'évaluer cette dernière. La question est de savoir sur quels critères elle doit être évaluée. Hattie, après l'étude de plus de 800 méta-analyses, en ressort quatre principaux (2012, p. 119) :

– « *la modification de la perception de l'activité par l'enseignant ;*
– *l'accumulation de connaissances ;*
– *l'évolution du comportement de l'enseignant ;*
– *l'impact sur les résultats des élèves.* »

L'étude de Hattie démontre que de nombreux auteurs ont mis en évidence l'impact de la motivation et des perceptions des enseignants sur leur engagement et sur leur investissement dans la profession. Ainsi, il est intéressant que les formateurs prennent en compte ces deux variables dans leurs formations.

Notre étude se centre sur l'évolution de la perception des futurs enseignants au cours de sa formation.

Tout d'abord, le sentiment de compétence des enseignants, correspondant à leur sentiment de capacité à exécuter des missions spécifiques (Bandura, 1997), est corrélé avec leur engagement professionnel (Barroso da Costa et Loye, 2016) ainsi qu'avec leur capacité à le concrétiser (Friedman et Kass, 2002). Ensuite, le fait qu'un enseignant possède une intention d'usage envers un outil numérique peut conduire à une utilisation (plus adéquate) de ce dernier dans sa pratique professionnelle. Le modèle de TAM (Davis et Davis, 1989), amélioré par Heerink (2010), met en avant ces liens entre perceptions, intentions et utilisations.

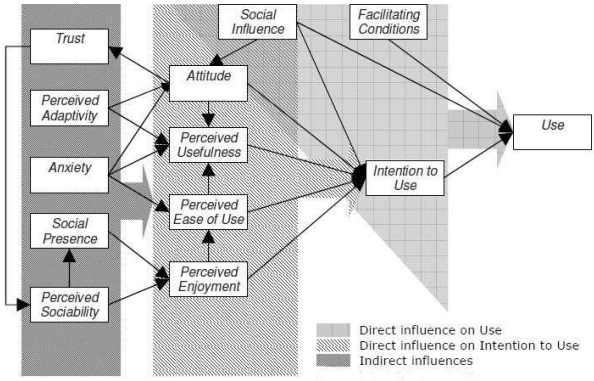


Figure 1 · Modèle de l'accessibilité d'un outil technologique (Heerink, 2010)

Alors que, dans le modèle de TAM, seule l'appréciation, positive ou négative, de l'utilité, de l'utilisabilité et de l'acceptabilité d'un outil peut avoir une influence sur l'intention d'usage d'une technologie, Heerink fait aussi intervenir des facteurs propres au profil de l'utilisateur (figure 1). Alors que le facteur d'utilité est directement relié à l'intention, celui de l'utilisabilité ne l'influe qu'indirectement. En effet, ce dernier agit à travers la vision d'utilité de l'outil en l'augmentant ou en la diminuant selon que l'on perçoit son usage comme plus ou moins aisé (Davis et Davis, 1989). Enfin, Nair et Mukunda Das (2012) stipulent que la facilité d'utilisation perçue par les enseignants est un facteur essentiel expliquant leur attitude à l'égard de l'utilisation des technologies. Une perception positive de l'utilité ne suffit pas à les utiliser dans les classes, il faut aussi qu'ils en perçoivent l'utilisation aisée.

En FWB, 76 % des 2585 enseignants ayant répondu à l'enquête affirment n'avoir aucune connaissance en programmation (figure 2).

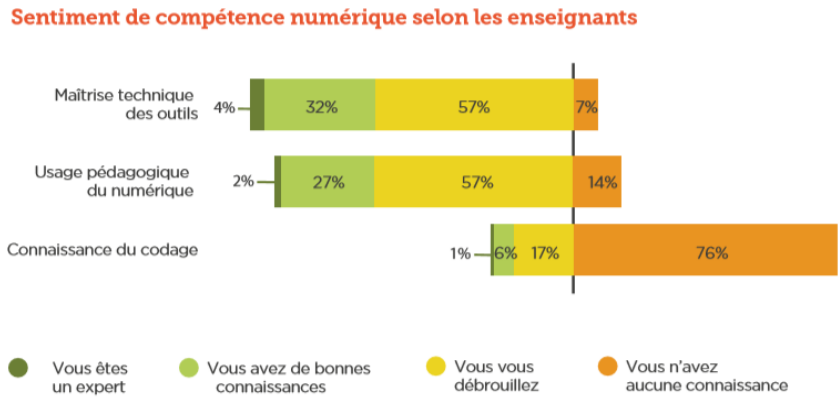


Figure 2 · Sentiment de compétence numérique des enseignants selon le rapport de Digital Wallonia (2018)

De plus, seuls 17 % des enseignants interrogés disent avoir reçu au moins une formation liée à la programmation, lors de leur formation initiale ou continue. Il semblerait qu'aucun rapport relatif à la perception des enseignants face à l'intérêt pédagogique et à la facilité d'enseignement de cette compétence numérique n'ait été réalisé en Belgique.

En France, Roche *et al.* (2018) ont réalisé une enquête auprès de 578 enseignants et enseignantes de primaires de la région nantaise. Il en ressort que 24 % d'entre eux disaient avoir bénéficié d'une formation relative à la programmation (ce qui est supérieur de 7 % aux résultats belges). De plus, 60 % des répondants étaient convaincus de l'utilité de l'enseignement de la programmation pour les apprentissages interdisciplinaires, le développement des capacités de résolution de problèmes et l'insertion professionnelle des apprenants. Puisque l'intérêt pour un enseignement s'évalue au travers de la perception de son utilité (Roche *et al.,* 2018), il est possible de conclure que ces 60 % enseignants appréhendaient bien l'intérêt de l'enseignement de la programmation pour les élèves. Cependant, en ce qui concerne la facilité, 75 % soutenaient que les objectifs d'apprentissage n'étaient pas clairs, 83 % confiaient ne pas être à l'aise avec les notions relatives à la programmation qu'ils étaient susceptibles de devoir enseigner et, enfin, 78 % considéraient que cette compétence numérique était difficile à mettre en œuvre sur le terrain. En somme, les enseignants interrogés exprimaient une meilleure perception de l'utilité que celle de l'utilisabilité et étaient plutôt positifs concernant l'intégration des sciences informatiques dans les écoles.

Herry et Mougeot (2007), quant à eux, constatent une différence notable entre les enseignants du primaire et ceux du secondaire dans les écoles francophones d'Ontario : les enseignants du secondaire ont une vision de leur maîtrise du numérique (d'ordre technique, social, informationnel et épistémologique) plus élevée. Ceci peut s'expliquer par le fait que le niveau de compétence technologique des enseignants impacte leur perception des difficultés relatives à leur usage en contexte scolaire (Duguet et Morlaix, 2017).

Le rôle de la formation des enseignants est donc de modifier les *a priori* de ces derniers concernant l'utilité et l'efficacité de l'enseignement de la programmation. Pour cela, il est important de revoir les programmes de formation (Roche *et al.,* 2018).

2.6. Comment adapter la formation aux perceptions ?

Selon Fleitz (2004), seul un rapport « pratique » entre l'enseignant et la formation peut potentiellement amener ce dernier à une innovation de ses pratiques pédagogiques. Pour cela, il est essentiel qu'une formation applique le principe d'isomorphisme pédagogique, c'est-à-dire un enseignement « pratique » du contenu accompagné d'un aspect réflexif et

interactif de cette dernière (Muller, 2018). Le contenu abordé doit être lié à la discipline de l'enseignant et être en adéquation avec les demandes et les besoins de la société et, ainsi, des élèves. En somme, cela signifie que le contenu de la formation (savoir et/ou pratique) ne doit pas sembler trop abstrait ou trop éloigné du vécu professionnel. Il doit faire sens et sembler utile à l'enseignant. Cependant, cela n'amène l'enseignant que vers une « éventuelle » innovation. Pour qu'il y ait une modification de son comportement professionnel, c'est-à-dire pour qu'il innove en situation de travail, le contenu de la formation doit « *prendre un sens dans le contexte de l'enseignant* » (Fleitz, 2004, p. 87). En effet, c'est lorsque le contenu est illustré *via* une pratique existante et dans une situation particulière, proche de celles rencontrées par l'enseignant sur le terrain, que ce dernier va lui donner du sens. Bien qu'un certain degré d'écart soit admissible et franchissable pour quelques personnes, un trop grand fossé entre le vécu pédagogique de l'enseignant et la formation peut être insurmontable. Il est donc préconisé de réaliser des formations « prudentes », c'est-à-dire des formations qui s'appuient sur l'expérience et les compétences des enseignants et qui proposent des projets simples avec des objectifs peu élevés qui engendrent de petits changements atteignables progressivement (Muller, 2018). Ainsi, parce qu'ils sont connus pour leur forte ténacité face au changement, les enseignants ne doivent pas subir une transformation trop violente de leurs pratiques habituelles. Fleitz (2004) illustre cette incapacité qu'ont certains enseignants à innover, en prenant l'exemple de l'activité « danse » : malgré une perception positive de son utilité pédagogique, certains enseignants ne se résolvent pas à réaliser cette activité artistique en classe car elle va à l'encontre de leur personnalité.

Laird (2018) ajoute que, pour qu'un enseignant s'approprie une démarche pédagogique nouvelle, il est nécessaire qu'il vive cette dernière en tant qu'élève. En effet, selon lui, l'utilisation des principes d'isomorphisme lors d'une formation permettrait aux enseignants de mieux façonner les différents outils et stratégies pour que leurs élèves puissent, à leur tour, vivre cette expérience d'apprentissage.

2.7. La pédagogie vidéoludique

Outre ses capacités à motiver extrinsèquement les apprenants et à leur procurer des situations d'apprentissage mobilisant de manière cohérente la notion, ou le concept, d'« apprendre à apprendre », la pédagogie vidéoludique offre d'autres avantages interdisciplinaires qui sont aujourd'hui reconnus et mis en avant : la collaboration, l'autonomie,

l'esprit critique et la capacité à comprendre, mémoriser et maîtriser des savoirs, grâce notamment à la pédagogique active du « faire » (*Learning by doing*) et à l'essai-erreur sans la peur de la défaite et du jugement (Annart, 2019). Parmi la quantité de jeux vidéo disponibles, Minecraft a réussi à se créer une place parmi les stratégies éducatives de nombreux enseignants grâce à ses bienfaits pédagogiques (Clôatre, 2018 ; Karsenti et Bugmann, 2017 b).

3. Méthodologie

3.1. Contexte

Cette recherche a eu lieu au cours de l'année scolaire 2019-2020 auprès des sections « instituteurs primaires » et « AESI mathématiques » dans deux hautes écoles de la FWB : Condorcet Mons et Henallux Champion. Pour la constitution de l'échantillon, nous nous sommes basés sur la disponibilité des sujets. Il fut ainsi composé de 82 sujets répartis en deux groupes en fonction de leur section d'étude, « instituteurs primaires » (G1 = 43) et « AESI mathématiques » (G2 = 39).

3.2. Prise des données

L'objectif de notre recherche étant de mesurer l'effet de l'appartenance à la section d'étude des participants sur l'évolution de trois variables dépendantes (l'intention d'enseignement de la programmation, la perception de l'utilité et celle de l'utilisabilité de cette compétence numérique), nous avons mis en place un plan expérimental qui implique deux temps de prise de mesure. Nous l'exprimons sous la forme de deux observations pré et post-expérimentales :

– l'observation préexpérimentation consiste en un questionnaire qui permet d'évaluer le niveau d'intention et de perception des futurs enseignants avant leur participation à la formation ;

– l'expérimentation est constituée de la formation relative à la programmation ;

– l'observation post-expérimentation consiste en un questionnaire qui permet d'évaluer le niveau d'intention et de perception des enseignants après la participation à la formation en vue d'analyser l'effet de cette dernière.

Pour créer et articuler les différentes questions du préquestionnaire, nous nous sommes inspirés des items présents dans l'enquête de Roche *et al.* (2018). Son étude est composée de neuf questions fermées (selon une échelle de Likert) regroupées en trois catégories : l'utilité de l'enseignement de la

programmation, la facilité perçue de sa mise en œuvre et la capacité à rédiger des programmes informatiques. Afin d'augmenter la validité de contenu de notre préquestionnaire, nous avons détaillé les trois catégories grâce à notre revue de la littérature relative à la programmation et au numérique (FWB, 2018 ; Karsenti, 2019). Ainsi, dans un premier temps, nous avons complété les questions de Roche *et al.* pour qu'elles correspondent aux recherches et aux rapports ayant révélé des avantages supplémentaires à l'enseignement de la programmation. Dans un deuxième temps, le rapport de Digital Wallonia (2018), relatif au numérique et à l'éducation, nous a permis d'avancer les freins didactiques, techniques ou matériels qui pourraient empêcher les futurs enseignants de réaliser, par la suite, des activités de programmation dans leur classe. Enfin, afin de pallier le biais relatif à la maîtrise des jeux vidéo en pédagogie, nous avons ajouté deux questions dans la catégorie « facilité d'enseignement ». La première se rapporte à la connaissance et à l'aisance des participants envers Minecraft. La seconde s'intéresse à la capacité à faire confiance à ses élèves au niveau de la maîtrise technique d'un outil numérique.

Afin de faciliter le traitement des résultats, nous avons conçu des questions fermées à choix multiples ou avec une échelle de Likert à six niveaux (0 à 5). Sur la base des catégories de Roche *et al.* (2018) et avec l'ajout de questions propres aux caractéristiques initiales des participants, notre questionnaire s'organise en quatre parties relatives à la programmation : les caractéristiques initiales, l'utilité perçue de son enseignement, l'utilisabilité perçue de sa mise en œuvre ainsi que l'intention de comportement. Nous avons privilégié le terme « utilisabilité » à « facilité », car il s'agit du terme employé dans les modèles propres aux perceptions liées à aux numérique (Davis et Davis, 1989 ; Heerink, 2010).

En ce qui concerne le post-questionnaire, celui-ci est organisé de la même manière que le préquestionnaire, c'est-à-dire avec des questions fermées. Cependant, il ne comporte plus que trois catégories relatives à la programmation : l'intention d'enseignement, l'utilité et l'utilisabilité perçues dans sa mise en œuvre. En effet, les questions relatives aux trois caractéristiques initiales ont été retirées. Enfin, nous avons ajouté deux questions ouvertes afin de recueillir les avis positifs et négatifs des enseignants face à la formation et aux activités réalisées.

Afin d'analyser les données récoltées *via* les questionnaires, nous avons attribué une valeur à chaque réponse. Ensuite nous avons additionné les valeurs obtenues aux réponses de chaque catégorie (les caractéristiques initiales, la perception de l'utilité, celle de l'utilisabilité et l'intention

d'enseignement) pour calculer des scores, des moyennes, des écarts-types et des corrélations.

3.3. Choix de l'outil numérique

Le choix de l'univers Minecraft nous a semblé être une évidence au vu de sa popularité auprès des jeunes ainsi que des nombreux avantages pédagogiques qu'il peut engendrer (Clôatre, 2018 ; Karsenti et Bugmann, 2017b). Nous avons choisi d'exploiter la version éducative de Microsoft car cette dernière, créée dans un but exclusivement pédagogique, présente quatre avantages :

– ce « jeu sérieux » propose un paramétrage simplifié et adapté à une utilisation en classe ;

– il compte un ensemble de matériaux pédagogiques spécifiques par rapport à la version originale tels les éléments du tableau Mendeleïev et le portfolio ;

– la version éducative est composée de la plateforme MakeCode qui permet aux joueurs de manipuler les matériaux et un robot (l'agent) *via* un logiciel de codage en langage par blocs (identique au langage Scratch) et en langage JavaScript. Les élèves peuvent ainsi programmer des constructions et simuler de la robotique dans un univers 3D ;

– Microsoft a aménagé un site (https://education.minecraft.net/) pour fédérer une communauté d'enseignants qui utilisent Minecraft Éducation et pour ainsi rassembler les activités et questionnements de chacun.

3.4. Organisation de la formation

Pour l'élaboration de notre formation, nous avons veillé à respecter un équilibre entre les aspects théoriques, pratiques ainsi que les connaissances pédagogiques et didactiques de l'enseignement de la programmation (Timperley *et al.,* 2007). Pour ce faire, nous avons placé les participants en position d'apprenants actifs et nous avons divisé la formation en 3 parties : une introduction théorique, des ateliers pratiques et un débriefing.

Tout d'abord, la première partie a pour objectif de retracer le contexte éducatif de l'apprentissage à la programmation ainsi que ses avantages pédagogiques et ses stratégies d'enseignement, puis de décrire l'outil numérique Minecraft et ses possibilités pédagogiques. De manière à mettre en avant les pratiques professionnelles, nous avons diffusé une vidéo réalisée et montée dans le cadre de notre recherche. Celle-ci diffuse l'interview de six enseignants qui utilisent Minecraft avec leurs élèves. Ils expliquent de quelle manière ils l'utilisent et donnent leur opinion sur cet

outil. L'objectif est que les participants à la formation prennent conscience des besoins de la société et du rôle qu'ils auront à jouer en tant que futurs acteurs du système éducatif.

Lors de la seconde partie, nous avons mis les étudiants en situation d'apprentissage. Cette expérience isomorphique permet aux participants d'oublier leur statut de pédagogue en prenant la place de l'apprenant, d'éprouver et vivre les mêmes émotions que ce dernier et de visualiser le comportement et le rôle du formateur pour s'en inspirer dans sa future carrière professionnelle (Caena, 2011). De plus, cette pratique aide les participants à modifier leurs croyances négatives quant à l'enseignement de la programmation (Laird, 2018).

Figure 3 · Activité « Labyrinthe. Défi : qui sera le premier ? »

Pour ce faire, quatre activités[1] ont été créées sous forme de défis avec des objectifs simples afin de ne pas décourager les participants (Muller, 2018), ont été proposées.

– Activité 01 : « L'œuf ou la poule. Défi : les poules pondent-elles plus d'œufs si elles sont dans un enclos plus grand ? ». Dans cette leçon, les élèves sont amenés à construire des enclos carrés et rectangulaires et y faire apparaître des poules afin de répondre à la question de départ. Pour ce faire, ils doivent manipuler les blocs en mode débranché avec des exercices d'ordonnance avant de passer dans le monde numérique de Minecraft.

– Activité 02 : « Volcan et éruption. Défi : construisons notre volcan ». Dans cette leçon, les élèves sont amenés à construire un volcan en éruption,

[1] L'ensemble des activités est accessible sur le Pearltrees de Charline44442.

étape par étape, afin de simuler une éruption explosive, dans le cadre d'un cours sur le thème de la découverte des volcans. Pour ce faire, les élèves commencent par schématiser un volcan (la forme extérieure ainsi que le réservoir et la cheminée) en veillant à ce qu'il soit réalisable dans le monde cubique de Minecraft. Ensuite, ils sont amenés à se questionner sur la démarche à suivre pour construire la cheminée et le réservoir grâce à l'aide du robot programmable. Seulement après cette période de réflexion, les élèves ont accès à des parties de code qu'ils vont devoir, en débranché, rassembler et ordonner en un unique programme et, ensuite, en branché, recopier dans le MakeCode de Minecraft pour vérifier qu'il fonctionne. La dernière étape consiste en la mise en place des fusées explosives et de la lave grâce à un exercice de modelage qui va leur permettre de réaliser pas à pas la simulation de l'éruption explosive.

– Activité 03 : « Labyrinthe. Défi : qui sera le premier ? ». Dans cette leçon, les élèves doivent manipuler leur robot afin de lui faire parcourir divers itinéraires (figure 3). Premièrement, les élèves découvrent les blocs de déplacement du robot grâce à un tutoriel fourni par MakeCode. Ensuite, les élèves doivent réécrire un programme identique à celui du tutoriel en l'adaptant au parcours auquel leur robot fait face. La leçon propose trois niveaux de parcours ainsi que deux niveaux de labyrinthes parfaits. De cette manière, les élèves peuvent avancer à leur rythme et à leur niveau.

– Activité 04 : « Labyrinthe 2. Défi : sortons d'ici ! ». Dans cette leçon, les élèves apprennent à écrire des programmes permettant au robot de trouver, seul, la sortie de n'importe quel labyrinthe. Pour commencer, les élèves réalisent un exercice d'autorégulation dans lequel ils doivent corriger et déterminer le fonctionnement d'un programme mystère. Ensuite, à travers des exercices de création et de réflexion (stratégie intitulée « Voler de ses propres ailes »), les élèves sont amenés à modifier le programme initial et à créer de nouveaux programmes pour répondre à diverses questions émises par l'enseignant.

Chaque leçon provient de projets testés sur le terrain qui ont donc été expérimentés dans un contexte de classe réel avec des élèves de primaire ou de secondaire du degré inférieur. Ensuite, les activités ont été mises en œuvre en utilisant les stratégies pédagogiques de l'enseignement de la programmation (Desjardins *et al.,* 2018) en veillant à associer des moments branchés et débranchés (Vincent, 2018). Les étudiants, regroupés en duo sur un PC, travaillent de manière autonome et le formateur est présent pour répondre aux éventuelles difficultés. Chaque activité est donc propice à la

réalité du terrain et peut faire sens pour les futurs enseignants (Fleitz, 2004 ; Rey et Coen, 2012).

La troisième partie a pour objectif de faire ressortir, lors d'un débriefing, les expériences d'apprentissage vécues durant la journée ainsi que les éventuels manques et freins, les interactions avec les pairs étant une étape primordiale dans la transformation des croyances initiales et des perceptions (Muller, 2018). Pour ce faire, à travers un tour de table, les étudiants ont exprimé oralement leur ressenti face à la journée de formation.

3.5. Questions de recherche

Face à l'importance des perceptions de l'utilité et de l'utilisabilité dans l'engagement professionnel (Davis et Davis, 1989), l'objectif de notre recherche est :

– de confirmer, en préformation, une différence de profil entre nos deux groupes concernant les deux niveaux de perception et l'intention d'enseignement (Q1) ;

– d'évaluer si l'évolution de ces trois variables est présente et similaire dans nos deux groupes lorsqu'ils sont face à une formation identique (Q2).

Heerink (2010) stipulant que le profil des participants peut avoir une influence sur l'évolution des perceptions et de l'intention d'enseignement, nous veillerons à analyser les corrélations entre les trois variables et les caractéristiques initiales des participants (voir partie 4.2).

4. Présentation et analyse des résultats

4.1. Profils de nos deux groupes avant la formation (Q1)

Pour rappel, Herry et Mougeot (2007) constatent une différence notable entre les perceptions relatives aux technologiques des enseignants du primaire et du secondaire dans les écoles francophones d'Ontario. Pour la FWB, dans la même optique, Henry et Smal (2018) affirment que « *seuls les enseignants en mathématiques pourraient, à ce titre, prétendre posséder des compétences suffisantes pour assurer un cours d'initiation à l'informatique* » (p. 133).

Nous pouvions donc imaginer que nos deux groupes se distingueraient, dans le sens où, le groupe 2, qui suivait, en plus du cours de TIC, un cours relatif à l'algorithmique, aurait une meilleure perception de l'enseignement de la programmation.

Tableau 1 · Scores d'utilisabilité, d'utilité et d'intention d'enseignement pour les futurs enseignants du primaire (G1) et du secondaire inférieur (G2), avant la formation

	Utilisabilité		Utilité		Intention d'enseignement	
	G1	**G2**	**G1**	**G2**	**G1**	**G2**
Taille	43	39	43	39	43	39
Moyenne	28,50	35,30	48,30	53,40	37,20	53,80
Écart-type	5,40	5,09	25,44	22,39	0,49	0,51
Coefficient de variation	75,94	57,74	43,92	34,91	131,45	93,87

Les résultats du groupe 2 (voir tableau 1) démontrent des scores de perception et d'intention d'usage plus élevés que ceux du groupe 1. Néanmoins, cette différence entre les moyennes n'est pas statistiquement significative ($p = 0{,}175$; $p = 0{,}140$; $p = 0{,}251$; $p = 0{,}134$). Les perceptions et l'intention d'enseignement des deux groupes ne divergent donc pas en début de formation et nous pouvons donc nous attendre à une évolution homogène.

D'après Roche *et al.* (2018) 60 % des professeurs du primaire trouvent l'enseignement de la programmation utile, alors que seulement 30 % la trouvent facile à enseigner. Les résultats que nous avons obtenus vont dans ce sens : les scores de perception de l'utilité sont plus élevés que ceux de l'utilisabilité. Toutefois, ils se trouvent à la limite des 50 % pour les 2 groupes (G1 = 48,3 % ; G2 = 53,4 %). Nous pouvons donc suggérer que les répondants n'ont pas une vision très positive de l'enseignement de la programmation.

Alors que nos deux groupes présentent un profil similaire concernant les trois variables étudiées, leurs caractéristiques initiales s'avèrent différentes puisqu'ils divergent au niveau de leur formation préalable en programmation (G1 = 17,4 % ; G2 = 37,9 % ; $p < 0{,}001$). Une telle différence dans le niveau de formation en faveur du groupe 2 peut se justifier par la présence du cours d'algorithmique dans leur formation initiale.

Par contre, les deux groupes déclarent une faible maîtrise de l'outil Minecraft ($p = 0{,}056$) ce qui contredit les propos de Herry et Mougeot (2007) qui affirment que les compétences techniques des enseignants influencent leur vision de la facilité d'enseignement. Les deux groupes déclarent aussi un intérêt élevé pour l'enseignement de la programmation ($p = 0{,}963$), ce qui peut paraître étonnant si on considère que les enseignants en

mathématiques sont les plus susceptibles d'enseigner la programmation (Henry et Smal, 2018).

Tableau 2 · Analyse descriptive des caractéristiques initiales des futurs enseignants du primaire (G1) et du secondaire (G2)

	Intérêt		Formation		Maîtrise outil	
	G1	G2	G1	G2	G1	G2
Taille	43	39	43	39	43	39
Moyenne	68,80	69,20	17,44	37,95	22,09	29,81
Écart-type	0,47	0,47	2,40	3,09	2 415	1,90
Coefficient de variation	66,60	67,60	137,67	81,37	121,50	79,66

En ce qui concerne les moyennes des trois caractéristiques initiales, les premières données, relatives à l'intérêt (G1 = 69,8 % ; G2 = 69,2 %), correspondent aux résultats de l'étude de Roche *et al.* (2018) qui affirment que 60 % des enseignants du primaire interrogés portent un intérêt à l'enseignement de cette compétence numérique. Les résultats des deux autres caractéristiques sont nettement sous la moyenne. Concernant la formation, les résultats du groupe 1 (17 %), se retrouvent totalement dans la proportion d'enseignants de la FWB (Digital Wallonia, 2018), tout en restant inférieurs au score (24 %) obtenu par Roche (2018). Le groupe 2, quant à lui, surpasse toutes ces données pour se différencier significativement du G1. Ici aussi, le cours d'algorithmique, présent dans le cursus des AESI Mathématiques, justifie ce constat de divergence. En ce qui concerne la maîtrise de l'outil, les moyennes sont surprenantes compte tenu du fait que Minecraft est reconnu mondialement comme le jeu le plus joué et qu'il appartient à la génération des participants de notre étude (Pateau, 2018). Dès lors, nous nous attendions à ce que les scores, relatifs à la maîtrise de l'outil, soient nettement supérieurs.

Les 83 étudiants présentant un profil de départ similaire au niveau des deux variables de perception et de leur intention d'enseignement, nous avons analysé les relations entre ces trois variables dépendantes pour l'ensemble de notre échantillon.

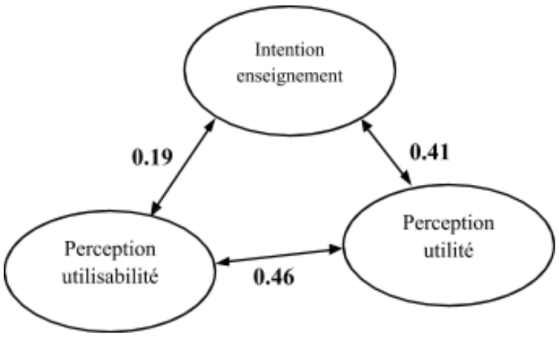


Figure 4 · Corrélations entre les variables dépendantes en préformation

Au vu des résultats (figure 4), nous pouvons confirmer une tendance de variation entre chacun des niveaux de perception avec l'intention d'enseignement : d'intensité moyenne pour la perception de l'utilité et d'intensité faible pour celle de l'utilisabilité. Précisons que, lorsque les groupes sont scindés, les mêmes tendances pour le groupe 2 se révèlent d'intensité supérieure : la relation entre l'intention d'usage et l'utilité est d'intensité « forte[2] » ($r = 0{,}41$) ; et celle avec l'utilisabilité est « faible » ($r = 0{,}19$). Le concept théorique de TAM expliquerait nos résultats, qui indiquent que la perception de l'utilité a un impact plus important sur la volonté d'inclusion du numérique que celle de l'utilisabilité. Dès lors, bien qu'il concerne initialement l'utilisation d'outils numériques dans les pratiques professionnelles, le modèle de TAM (Davis et Davis, 1989) peut être élargi à notre étude relative à l'enseignement d'une compétence numérique (la programmation) *via* un outil numérique.

De ce fait, nos résultats, peu élevés (G1 = 37,2 % ; G2 = 53,8 %), propres à l'intention d'enseignement de la programmation de nos deux groupes sont appuyés par le schéma de TAM (Davis, 1989) qui stipule que les perceptions de l'utilité et l'utilisabilité impactent l'intention pédagogique des enseignants. En parallèle, ces données rejoignent l'étude menée par le groupe SI2 (Henry et Smal, 2018), révélant que plus de 60 % des enseignants interrogés déclarent avoir un faible sentiment de compétence propre à l'algorithmique et à la programmation. Cette déclaration entraîne dès lors

[2] Par convention, il existe différentes balises pour qualifier l'intensité d'une relation entre deux variables (Cohen, 1988).

une diminution de leur engagement professionnel et plus spécifiquement de leur intention d'enseignement (Barroso da Costa et Loye, 2016). Notons que le groupe 2, contrairement au groupe 1, affiche une intention d'usage moyennement positive (53,8 %). Ceci sous-entend qu'ils ont pour projet d'enseigner la programmation lors de leur future carrière professionnelle. Ce constat coïncide avec le fait que l'algorithmique n'apparaît que dans les programmes d'étude des AESI mathématiques.

En résumé, toutes ces observations, laissant entendre qu'il existe une grande marge de progression possible, pourraient justifier l'existence de notre formation et qu'il serait plus important de se centrer sur l'évolution de l'utilité que sur celle de la facilité, pour obtenir un plus grand impact sur le comportement futur des enseignants.

4.2. Évolution des résultats après la formation (Q2)

L'objectif de notre étude étant d'évaluer l'intérêt de notre formation sur les perceptions et sur l'intention d'enseignement des participants, nous nous interrogeons sur les changements opérés suite à la formation.

Tableau 3 · Scores d'utilisabilité, d'utilité et d'intention d'enseignement déclarés par les futurs enseignants du primaire (G1) et du secondaire (G2) après la formation

	Utilisabilité		Utilité		Intention d'enseignement	
	G1	G2	G1	G2	G1	G2
Taille	43	39	43	39	43	39
Moyenne	36,93	53,23	49,38	63,44	48,80	76,90
Écart-type	20,52	18,29	21,95	18,07	0,51	0,43
Coefficient de variation	55,57	34,36	44,46	28,48	103,6	53,64

En post-formation, les différences de moyenne des scores d'intention d'enseignement ($p = 0{,}009$) et de perception ($p = 0{,}006$; $p < 0{,}001$) entre nos deux groupes indiquent que, dorénavant, ils se distinguent clairement. En analysant les deux niveaux de perception, nous constatons toujours que le score moyen de l'utilité est plus élevé que celui de l'utilisabilité et que le groupe 2 est plus homogène que le groupe 1. À ce stade, nous ne pouvons pas affirmer que, grâce à la formation, les deux niveaux de perception sont au même niveau. En effet, selon Nair et Mukunda Das (2012), une haute perception de l'utilité seule ne suffit pas pour que l'enseignant modifie ses

pratiques pédagogiques. La perception d'utilisabilité des enseignants domine leur engagement professionnel. En ce qui concerne l'intention d'enseignement de la programmation, les résultats post-formation du groupe 1 (48,8 %) semblent refléter une vision professionnelle dont l'enseignement de la programmation ne fait pas partie. A contrario, dans la vision du groupe 2 (76,9 %), cette compétence numérique fait bien partie de leurs missions.

De manière générale, les deux groupes montrent une évolution positive. Bien que les gains de G1 et ceux de G2 n'atteignent pas les 25 %, seuil de significativité d'un gain d'ordre comportemental (Gérard, 2003), ils s'en rapprochent fortement. À ce stade de la réflexion, nous pouvons donc affirmer que la formation a atteint son objectif. Cependant, seules la perception de l'utilisabilité et l'intention d'enseignement ont évolué de manière identique pour les deux groupes ($p = 0{,}067$; $p = 0{,}742$). La perception de l'utilité évolue plus significativement pour le groupe 2 ($p = 0{,}019$).

Les 83 étudiants présentant des profils différents en post-formation, nous avons analysé les relations entre ces trois variables pour chacun des groupes (figure 5).

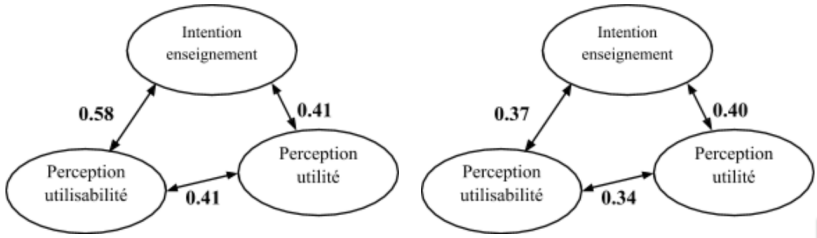


Figure 5 · Corrélations entre les variables en post-formation (G1–G2)

En ce qui concerne les liens entre l'intention d'instruction et les niveaux de perception (figure 5), nous observons des résultats significatifs mais avec des intensités différentes en fonction des groupes. Pour le groupe 1, bien que le modèle de TAM démontre l'influence directe de l'utilité sur l'intention d'enseignement, c'est la vision de l'utilisabilité qui prédomine (Nair et Mukunda Das, 2012). Pour le groupe 2, les résultats laissent supposer qu'il y a une même influence pour les deux niveaux de perception. Ainsi, l'utilité et la facilité d'enseignement sont deux arguments équivalents pour pouvoir enseigner cette compétence numérique.

Pour répondre à ce constat, notre revue de la littérature nous suggérait que les trois caractéristiques (maîtrise de l'outil, l'intérêt et la formation) pourraient avoir un impact sur nos variables (Heerink, 2010). Premièrement, selon Duguet et Morlaix (2017), le niveau de compétence technologique des enseignants impacte leur perception, et ainsi leur intention d'usage (Davis et Davis, 1989). Deuxièmement, l'intérêt est le reflet de la perception de l'utilité (Roche *et al.,* 2018). Enfin, le niveau de formation à la programmation préalable a un impact sur les niveaux de perception et l'engagement professionnel, car celle-ci permet aux participants de se retrouver dans un contexte familier (Fleitz, 2004). Or, en tenant compte du fait que les groupes sont considérés comme deux groupes distincts en post-formation, les caractéristiques initiales s'avèrent impacter différemment l'évolution de nos deux groupes. D'un côté, pour le groupe 1, les gains de perception et d'intention d'enseignement de ces mêmes variables révèlent une relation inversement proportionnelle avec la caractéristique « intérêt ». Cela signifie que les participants de ce groupe, qui avaient un fort intérêt avant la formation, ont tendance à afficher une évolution des perceptions et d'intentions d'usage moins importante. Inversement, ceux qui possédaient un faible intérêt ont vu cette évolution s'accroître de manière plus conséquente. Nous pouvons expliquer ce contact par les croyances du participant (Heerink, 2010). Ainsi, une personne, qui au départ a un fort intérêt et donc des croyances vis-à-vis de la formation attendue, peut être déçue car ses attentes ne sont pas atteintes. À l'inverse, une personne, qui n'a aucun intérêt de départ et aucune attente particulière, affiche une plus grande progression. D'un autre côté, pour le groupe 2, tout comme pour le groupe 1, l'intérêt a une relation inversement proportionnelle avec l'intention d'usage. Quant au niveau de formation et celui de la maîtrise de l'outil, ils ont un impact positif sur le gain de la perception d'utilisabilité. Ainsi ceux qui ont une meilleure formation préalable et une plus grande maîtrise de l'outil, affichent une meilleure progression dans leur perception de la facilité de l'enseignement de la programmation. Donc, pour ce groupe, il semble essentiel d'insister davantage sur ces deux caractéristiques lors de la partie théorique de la formation.

5. Discussion

Les analyses relatives au profil de départ (caractéristiques initiales, intentions d'enseignement et niveaux de perception) montrent que nos deux groupes ont un même profil à l'exception de leur formation préalable à la programmation. Effectivement, lorsque nous consultons les contenus

des deux cursus, nous constatons une dissemblance au niveau de l'éducation au numérique. Cependant, du point de vue technique, malgré les différences de niveau, aucune difficulté de ce type n'a été remarquée. Le travail de groupe, l'apprentissage collaboratif ainsi que le suivi du formateur ont certainement permis d'aboutir à ce résultat (Karsenti et Bugmann, 2017a).

Un participant « hostile » aura une perception négative face à l'enseignement numérique, alors que « l'enthousiaste » manifestera de l'intérêt et considérera son utilité dans l'apprentissage des élèves (Roche *et al.,* 2018). Dans notre expérimentation, les résultats au niveau des perceptions et de l'intention d'enseignement étant moyens, ils ne nous permettent pas de classer les participants dans une de ces catégories mais bien dans une autre, à mi-chemin. Dès lors, notre objectif est que tous nos candidats, quel que soit leur profil initial, deviennent « enthousiastes ».

En ce qui concerne l'évolution des résultats, à savoir le gain perçu au travers des variables « intention d'enseignement », « perception de l'utilité » et « perception de l'utilisabilité », de manière générale, nous observons un impact positif pour nos deux groupes. En d'autres mots, grâce à la formation, les deux groupes voient l'évolution de leur intention d'enseigner la programmation au cours de leur future carrière et celle de leur perception quant à sa facilité d'enseignement, augmenter de manière similaire. Effectivement, la construction de la formation, tenant compte de l'équilibre théorie-pratique (Timperley *et al.,* 2007), des stratégies d'enseignement de la programmation (Desjardins *et al.,* 2018) et de la méthodologie de la formation (Caena, 2011), est un atout central pour cette évolution. Cette dernière, bien que positive pour nos deux groupes, affiche néanmoins une différence d'intensité au niveau des perceptions : le groupe 2 progresse davantage. La différence de progression constatée entre le groupe 1 et le groupe 2 peut être expliquée par le fossé entre leurs croyances (issues de leur propre vécu scolaire depuis la maternelle jusqu'en Haute École) et le contenu de la formation (Borko et Putnam, 1996 ; Crahay *et al.,* 2010 ; Hollingsworth, 1989 ; Kagan, 1992 ; Richardson, 1996 ; Richardson et Placier, 2001 ; Vause, 2009). En effet, à ce jour, dans les programmes de l'enseignement primaire la programmation n'apparaît pas encore, elle n'est présente qu'au niveau secondaire sous l'intitulé « cours d'informatique ». Dès lors, nous supposons que les futurs enseignants du primaire ne s'attendent pas à devoir enseigner cette compétence ce qui peut fausser l'impact de la formation. D'un autre côté, pour que les participants d'une formation voient leur perception évoluer positivement

il faut que le contenu ne soit pas trop éloigné des pratiques acquises en formation initiale (Roche *et al.,* 2018). Pour les enseignants du primaire, il semble que la formation proposée ne se rapproche pas suffisamment de leur formation initiale ; alors que pour les enseignants de mathématiques du secondaire, le fossé semble moins important. Ces éléments expliqueraient la différence entre l'évolution des perceptions de nos deux groupes.

6. Conclusion

À l'heure où l'enseignement en Fédération Wallonie-Bruxelles (FWB) se situe à un tournant majeur de son histoire avec l'entrée en vigueur du Pacte pour un enseignement d'Excellence (FWB, 2017) et l'apparition, notamment, d'un référentiel numérique (FWB, 2019), Digital Wallonia (2018) insiste fortement sur une intégration primordiale de l'éducation au/par le numérique dès la formation initiale des enseignants. Parmi ces compétences numériques, compte tenu de l'impact positif de son intégration pédagogique (Karsenti, 2019 ; Romero, 2016), figure la programmation (Carretero *et al.,* 2017 ; Henry et Smal, 2018). Il est donc primordial de repenser la formation initiale des enseignants, et notamment celle des enseignants du primaire et des AESI mathématiques (Henry et Smal, 2018). Partant du constat que plus de la moitié de la population belge joue à des jeux vidéo et sachant que de nombreux chercheurs (Annart, 2019 ; Tresse, 2012) mettent l'accent sur les bienfaits de l'utilisation de la pédagogie vidéoludique en classe, nous avons choisi d'utiliser, dans cette expérimentation, l'univers du célèbre jeu Minecraft (Pateau, 2018).

À l'image de certains pays qui ont déjà intégré cette compétence dans leurs programmes (Karsenti et Bugmann, 2017a), nous soutenons qu'il est primordial de sensibiliser, initier et former les différentes générations à la programmation. Ainsi, tenant compte de leurs perceptions et leurs croyances, nous avons dispensé une formation de six heures à 83 étudiants de deux hautes écoles wallonnes. Celle-ci respectait un équilibre entre la théorie, la pratique, les connaissances pédagogiques et didactiques de l'enseignement de la programmation (Timperley *et al.,* 2007) et tournait autour de trois parties. La première retraçait le contexte pédagogique relatif à la programmation, décrivait l'outil Minecraft et ses possibilités pédagogiques. La deuxième correspondait à la partie pratique, dans laquelle les futurs enseignants vivaient les activités à la place des apprenants. Enfin, la troisième et dernière partie constituait en un débriefing au cours duquel

chacun verbalisait son vécu à travers la formation. Les perceptions jouant un rôle majeur dans l'engagement professionnel (Heerink, 2010), l'utilisation de deux questionnaires en, pré- et post-formation, nous a permis de visualiser celles relatives à l'enseignement de la programmation en vue de répondre à notre hypothèse : l'administration d'une formation active et concrète des futurs enseignants du primaire et de mathématiques du secondaire inférieur, relative à l'enseignement de la programmation, *via* l'outil Minecraft : Éducation Édition, a un impact similairement positif, sans tenir compte de leur profil de départ, sur leur intention ainsi que sur leur perception de l'utilité et de l'utilisabilité de son instruction.

Alors qu'en début de formation, nos deux groupes présentaient le même profil concernant les deux niveaux de perception et l'intention d'enseignement de la programmation, au terme de la formation, l'analyse des nouveaux résultats indique que nos deux groupes n'ont pas progressé de la même manière. En tenant compte des résultats et des limites (relatives au contexte de participation, à la formation et aux moyens personnels), certaines adaptations devraient être faites. La formation devrait s'organiser non pas sur une journée mais bien sûr une plus longue période (un quadrimestre ou deux) dans le cadre du cours de TIC. De plus, elle devrait se focaliser davantage sur les croyances des enseignants du primaire qui s'imaginent difficilement dispenser une telle matière. Or, avec l'arrivée du nouveau référentiel numérique, il est nécessaire de changer cette croyance. Enfin, au niveau du contenu de la formation, nous varierions les outils numériques (Minecraft, Scratch, Robot...) ce qui permettrait d'installer un meilleur équilibre théorique-pratique. Nous pourrions aussi inclure plus de liens avec les pratiques existantes (collaborations avec intervenants externes, observation) : nous maintiendrions les situations où les participants expérimentent le jeu en se mettant à la place de leurs élèves et nous y ajouterions une partie où ils expérimenteraient cette activité en tant qu'enseignants, lors de leur stage. Ces différentes expériences nourriraient la réflexion demandée lors du débriefing, leur permettant ainsi d'échanger avec le groupe.

Enfin, bien que les résultats ne puissent être généralisés, ce retour d'expérience nous permet de suggérer quelques pistes pour une meilleure évolution des perceptions et de l'intention d'enseignement de la programmation des futurs enseignants du primaire et de mathématiques. Une première serait d'étaler davantage cette formation sur l'ensemble du cursus en y intercalant une période d'expérimentation réelle (en stage ou lors de collaborations entre école secondaire et le cours d'AFP). Une

seconde perspective serait de dispenser cette même formation, mais dans le cadre des formations continues, c'est-à-dire avec un public d'enseignants expérimentés.

RÉFÉRENCES

Annart, J. (2019). *Jeux vidéo et éducation. Ateliers de pédagogie (vidéo)ludique.* Centre Ressources Illettrisme (CRI).

Archambault, J.-P. (2015). Enseigner la discipline informatique avec un Capes et/ou une agrégation d'informatique. *Revue de l'association EPI, 179*, 1-3.

Bandura, A. (1997). *Auto-efficacité : Le sentiment d'efficacité personnelle.* De Boeck.

Barroso da Costa, C. et Loye, N. (2016). L'engagement professionnel affectif chez les nouveaux enseignants du primaire et du secondaire : une étude canadienne. *Revue des sciences de l'éducation, 42*(3), 1-35. https://doi.org/10.7202/1040084ar

Bernard, R. M., Abrami, P. C., Lou, Y., Borokhovski, E., Wade, A., Wozney, L. Wallet, P. A. Fiset, M. et Huang, B. (2004). How does distance education compare with classroom instruction? A meta-analysis of the empirical literature. *Review of Education Research, 74*(3), 379-439. https://doi.org/10.3102/00346543074003379

Borko, H. et Putnam, R. (1996). Learning to Teach. Dans D. Berliner et R. Calfee (dir.), *Handbook of Educational Psychology* (p. 673-708). MacMillan.

Caena, F. (2011). *Literature review. Quality in teachers' continuing professional development.* Commission européenne.

Caron, J. et Portelance, L. (2017). La collaboration entre chercheuse et praticiens dans un groupe de codéveloppement professionnel. *Éducation et socialisation, 45.*

Carretero, S., Vuorikari, R. et Punie, Y. (2017). *DigComp 2.1: The Digital Competence Framework for Citizens - With eight proficiency levels and examples of use.* Publication Office of the European Union. https://doi.org/10.2760/38842

Cloâtre, S. (2018, 2 janvier). *Minecraft in the classroom: conquering new educational territories ...,* http://budwhite72.legtux.org/?p=1212

Cohen, J. (1988). *Statistical Power Analysis for the Behavioral Sciences* (2nd ed.). Lawrence Erlbaum Associates.

Crahay, M., Wanlin, P., Issaieva, E. et Laduron, I. (2010). Fonctions, structuration et évolution des croyances (et connaissances) des enseignants. *Revue française de pédagogie, 172,* 85-129.

Davis, F. D. et Davis, F. (1989). Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. *Management Information Systems Quarterly, 13*(3), 319-340.

Desjardins, A., Tran, A et Girard, M-A. (2018, 22 mai). *Codage, programmation et robotique.* https://ecolebranchee.com/dossier-programmation-developpement-de-pensee-informatique/

De Poortere, C. (2017, 7 juillet). *Les enfants, le code et l'informatique : propagande et réalité.* https://www.pointculture.be/magazine/articles/focus/les-enfants-le-code-et-linformatique-propagande-et-realite/

Digital Wallonia (2018). *Baromètre digital Wallonia – Éducation et numérique 2018 : infrastructure, ressources et usages du numérique dans l'éducation en Wallonie et à Bruxelles*. Agence du numérique.

Duguet, A. et Morlaix, S. (2017). Perception des TIC par les enseignants universitaires : l'exemple d'une université française. *Revue internationale des technologies en pédagogie universitaire, 14*(3), 5-16.

Fleitz, T. (2004). Formation continue et transformation des pratiques enseignantes : le rapport à la formation. *Savoirs : La vie adulte en question, 1*(4), 79-97. https://doi.org/10.3917/savo.004.0079

Freidman, I et Kass, E. (2002). Teacher self-efficacy: A classroom-organization conceptualization. *Teaching and Teacher Education, 18*(6), 675-686. https://doi.org/10.1016/S0742-051X(02)00027-6

Fédération Wallonie-Bruxelles. (2017). *Pacte pour un enseignement d'Excellence : Avis nº 3 du groupe central*. Ministère de la Fédération Wallonie-Bruxelles.

Fédération Wallonie-Bruxelles (2018). *Stratégie numérique pour l'éducation*. Ministère de la Fédération Wallonie-Bruxelles.

Fédération Wallonie-Bruxelles (2019). *Pacte : Charte des référentiels*. Ministère de la Fédération Wallonie-Bruxelles.

Gérard, F.-M. (2003). L'évaluation de l'efficacité d'une formation. *Gestion 2000, 20*(3), 13-33.

Guardiola, C. (2014). Apprendre à coder, un effet de mode ou un enjeu de société ? *Revue de l'association EPI,* 168, 1-3.

Hattie, J. (2012). *Visible learning: a synthesis of over 800 meta-analyses relating to achievement*. Routledge.

Heerink, M. (2010). Assessing acceptance of assistive social agent technology by older adults: the almere model. *International journal of social robotics, 2*(4), 361-375.

Henry, J. et Smal, A. (2018). *Et si demain je devais enseigner l'informatique ? Le cas des enseignants de Belgique francophone* [communication]. Colloque Didapro 7 – DidaSTIC. De 0 à 1 ou l'heure de l'informatique à l'école, Lausanne.

Herry, Y. et Mougeot, C. (2007). *Recherche en éducation en milieu minoritaire francophone*. Presses de l'Université d'Ottawa.

Hollingsworth, S. (1989). Prior beliefs and cognitive change in learning to teach. *American Educational Research Journal, 26*(2), 160-189.

Karsenti, T. (2019, 29 mai). *12 raisons d'apprendre à coder à l'école*. https://www.edcan.ca/wp-content/uploads/FAITS-EN-ÉDUCATION_12-raisons-d'apprendre-à-coder-à-l'école_v2.pdf

Karsenti, T. et Bugmann, J. (2017). *Transformer l'école avec Minecraft ? Résultats d'une recherche menée auprès de 118 élèves du primaire*. Crifpe.

Karsenti, T. et Bugmann, J. (2017). Pourquoi apprendre à coder à l'école ? *Apprendre et enseigner aujourd'hui, 7*(1), 33-36.

Laird, B. (2019, 25 juin). *Translating professional development into practice*. [Vidéo]. Youtube. https://www.youtube.com/watch?v=A7K4Q0Hskbs

Roche, D., Higuera, C. et Michaut, C. (2018). Enseigner la programmation informatique : comment réagissent les professeurs des écoles ? *Notes du CREN, 27,* 1-7.

Merchin, A. (2017, 14 juin). *Le code informatique permet à l'enfant d'apprendre à apprendre*. https://www.lemonde.fr/femmes-a-part/article/2017/06/14/claude-terosier-le-code-informatique-permet-a-l-enfant-d-apprendre-a-apprendre_5144120_5102575.html

Muller, F. (2018). *Des enseignants qui apprennent, ce sont des élèves qui réussissent*. ESF.

Nair, I. et Mukunda Das, V. (2012). Using Technology Acceptance Model to assess teachers' attitude towards use of technology as teaching tool: a SEM Approach. *Internal Journal of Computer Applications, 42*(2), 1–4. https://doi.org/10.5120/5661-7691

Pateau, F. (2019, 15 juin). *Minecraft compte toujours plus de joueurs mensuels que Fornite*. https://gamewave.fr/minecraft/minecraft-compte-toujours-plus-de-joueurs-mensuels-que-fortnite/

Rey, J. et Coen, P-F. (2012). Évolutions des attitudes motivationnelles des enseignants pour l'intégration des technologies de l'information et de la communication. *Formation et profession, 20*(2), 19–32. https://doi.org/10.18162/fp.2012.177

Richardson, V. (1996). The role of attitudes and beliefs in learning to teach. Dans J. Sikula (dir.), *Handbook of research on teacher education* (p. 102-119). Macmillan.

Richardson, V. et Placier, P. (2001). Teacher change. Dans D.V. Richardson (dir.), *Handbook of research on teaching* (p. 905-950). American Educational Research Association.

Romero, M. (2016). De l'apprentissage procédural de la programmation à l'intégration interdisciplinaire de la programmation créative. *Formation et profession, 24*(1), 87–89. http://dx.doi.org/10.18162/fp.2016.a92

Singh Rajput, J. (2000). La formation des enseignants. *Revue internationale d'éducation de Sèvres, 25*, 41–51. https://doi.org/10.4000/ries.2564

Timperley, H., Wilson, A, Barrar, H. et Fung, I. (2007). *Teacher professional learning and development best evidence synthesis*. Ministry of Education. https://www.educationcounts.govt.nz/publications/series/2515/15341

Tresse, J. (2012, 15 juillet). *Enseigner avec les jeux vidéo : vers une pédagogie vidéoludique ?* https://www.educavox.fr/accueil/debats/enseigner-avec-les-jeux-video-vers-une-pedagogie-videoludique

Vause, A. (2009). Les croyances et connaissances des enseignants à propos de l'acte d'enseigner. Vers un cadre d'analyse. *Les Cahiers de recherche en éducation et formation, 66.*

Vincent, J.-M. (2018). *L'informatique débranchée : Le numérique sans ordinateur : activités de découverte*. Collection Tangente – Éducation.

Analyse des processus d'entraide dans le cadre d'un laboratoire virtuel et distant pour l'apprentissage de l'informatique

▶ **Pierre BELLET** (Lhumain), **Rémi VENANT** (LIUM), **Chrysta PÉLISSIER** (Lhumain), **Stéphanie MAILLES VIARD METZ** (Adef), **Julien BROISIN** (Irit)

■ **RÉSUMÉ** • Cette recherche transdisciplinaire vise à analyser les comportements des étudiants en situation d'entraide dans le cadre d'un laboratoire virtuel et distant pour l'apprentissage de l'informatique. Cette étude permet d'identifier différentes configurations d'entraide, et révèle que les étudiants contribuent de façon homogène aux sessions de *chat* supportées par un outil de consultation de terminal d'un pair. Elle montre également que la difficulté et le calibrage de la tâche impacteraient la qualité des sessions d'entraide.

■ **MOTS-CLÉS** • EIAH, entraide, laboratoire virtuel et distant, interaction, apprentissage de l'informatique.

■ ***ABSTRACT*** • *This transdisciplinary research seeks to analyze student behaviour in mutual aid situations, in the context of a virtual and remote laboratory for computer education. This study allows to identify different configurations of mutual aid, and reveals that students contribute homogeneously to chat sessions supported by a peer terminal consultation tool. It also shows that the task difficulty would impact the quality of the mutual aid sessions.*

■ ***KEYWORDS*** • *technology-enhanced learning, mutual aid, virtual and remote laboratory, interaction, computer education.*

1. Introduction

Le développement des technologies de l'information et de la communication a considérablement modifié, depuis une dizaine d'années, la conception de l'activité d'apprentissage. En effet, de nombreux environnements sont aujourd'hui conçus et développés dans la perspective de proposer de nouvelles modalités de formation comme l'apprentissage par les pairs. Par cette dynamique de développement technique, l'enjeu est de mettre à la disposition des apprenants des environnements mais également des modalités pédagogiques différentes, plus souples, laissant apparaître des possibilités de prise en compte de stratégies d'apprentissage diverses et propres à chacun.

Ces nouvelles modalités laissent entrevoir de nouvelles perspectives de recherche au sein d'équipes pluridisciplinaires. En effet, elle permet aux scientifiques d'explorer ou de convoquer, dans leurs cadres théoriques respectifs, des connaissances *a priori* éloignées de leurs champs de recherche d'origine. Il s'ensuit de cette approche une cristallisation autour d'une problématique commune et d'un décloisonnement des objets de recherche généralement abordés dans une perspective monodisciplinaire. La possibilité d'un décloisonnement entre les disciplines intervenant dans le processus de conception (Lonchamp, 2003) s'effectue dans le cadre d'une collaboration (Blessing, 1993 ; Jacobs *et al.,* 2002), d'une initiative volontaire des acteurs mettant en commun des idées convergentes pour atteindre des buts partagés de tâches complexes (Darses, 2002). Cependant, envisager la conception et l'analyse des usages de ces environnements dans une perspective pluridisciplinaire soulève de vrais défis (Bourdeloie, 2012) : définir ensemble une même méthode et/ou identifier la coordination de ces méthodes afin de produire des analyses pertinentes pour la communauté scientifique mais aussi pour les praticiens (c.-à-d. concepteurs de ces environnements, et enseignants proposant des dispositifs de formation).

Dans cet article, nous présentons les résultats d'une recherche exploratoire réalisée en équipe pluridisciplinaire. Ces résultats permettent de caractériser l'entraide telle qu'elle a été mise en place par les apprenants au cours de l'usage d'un laboratoire virtuel et distant expérimenté dans un dispositif de formation à l'université. Notre intérêt porte plus particulièrement sur la plateforme Lab4CE, un laboratoire virtuel et distant (*Virtual and Remote Laboratories*, VRL). Ces laboratoires sont des environnements dédiés à l'apprentissage exploratoire (van Jooligen *et al.,*

2005). Intégrés à des activités médiatisées par l'informatique, ils ont pour objectif de développer chez les apprenants, outre les compétences du domaine d'apprentissage, des compétences d'entraide, de collaboration. En effet, à travers certaines fonctionnalités offertes par le laboratoire telles que la messagerie instantanée ou le partage d'écrans/de terminaux, les utilisateurs sont susceptibles de formuler une demande d'aide, d'y répondre avec une aide destinée à un collaborateur, ou encore de notifier l'acceptation de l'aide reçue. La consultation des terminaux par des pairs impliqués dans l'activité demandée permet de s'informer sur les procédures suivies, comparer les actions menées par chacun, et ainsi progresser dans la démarche d'activité.

La recherche proposée articule, d'une part, des visées pragmatiques d'élaboration d'un dispositif d'apprentissage et d'un outil pour soutenir la démarche d'entraide entre apprenants engagés dans une même activité collaborative d'apprentissage, et d'autre part, des visées heuristiques de caractérisation de l'entraide en s'appuyant sur la compréhension des situations d'interaction entre apprenants. Par l'analyse de l'usage des fonctionnalités de Lab4CE, et en particulier des outils de messagerie instantanée (c.-à-d. *chat*) et de consultation du terminal d'un pair, nous avons accès à la partie explicite du comportement d'entraide des apprenants, offrant une certaine compréhension du processus qu'ils mettent en œuvre dans le cadre de la résolution de problèmes d'apprentissage. Si l'identification de la nature des messages présents dans le chat (demande d'aide, offre d'aide, acceptation/refus d'aide) constitue un premier niveau d'approche, leur coordination dans la mise en fils d'interaction vient compléter ces observations. En effet, les fils d'interactions permettent de caractériser la dynamique interactionnelle mise en jeu par les apprenants dans le laboratoire lors de l'expérimentation menée. L'analyse de l'utilisation de l'outil de consultation du terminal d'un pair constitue un autre niveau d'approche permettant d'identifier les artefacts numériques qui peuvent supporter les apprenants lors d'une activité d'entraide, mais également de préciser les interactions prenant place entre apprenants pendant une session d'entraide.

Pour soutenir cette articulation, nous prenons appui sur le paradigme de recherche *Design-Based Research* (DBR) (The Design-Based Research Collective, 2003 ; Wang et Hannafin, 2005). Cette méthode, qui permet d'instancier des modèles sous la forme d'applications numériques en vue d'une utilisation contextualisée, se caractérise par des principes méthodologiques particulièrement intéressants dans le contexte de la

conception d'un dispositif et d'une application numérique, comme le principe de reconcevoir de manière cyclique des dispositifs pédagogiques, la nécessité de collaborer étroitement entre différents acteurs, et le fait de conduire les recherches en milieu naturel. L'intrication de la recherche et de la conception d'applications numériques est alors bilatérale et itérative.

La section qui suit expose le cadre théorique nous permettant de positionner la problématique de l'entraide et de formuler les questions de recherche traitées dans cet article. La section 3 expose l'expérimentation sur laquelle s'appuie notre recherche et précise la méthodologie d'analyse telle qu'elle a pris forme dans une démarche de projet pluridisciplinaire. Nous décrivons les procédures d'analyse qualitative et quantitative que nous avons mises en œuvre à partir de l'observation des échanges qui ont eu lieu au sein du chat du laboratoire distant, et des interactions ayant pris place avec l'outil de consultation du terminal d'un pair. Les résultats de ces analyses sont ensuite présentés puis discutés.

2. *Background* : la notion d'« aide » et d'« entraide »

2.1. Aide et recherche d'aide

Si la recherche d'aide a longtemps été considérée comme une marque de dépendance de la part de l'apprenant, les travaux de Nelson-Le Gall (1981) ont montré qu'elle pouvait être appréhendée comme une stratégie témoignant de la capacité de l'apprenant à s'autoréguler. Karabenick et Gonida (2017) identifient plus particulièrement deux types de recherche d'aide, selon les objectifs visés :

– la « recherche d'aide exécutive » (*executive help seeking*) qui relève d'une stratégie d'évitement du travail à réaliser (par exemple, quand l'apprenant demande la solution à autrui) ;

– la « recherche d'aide instrumentale » (*instrumental help seeking*) qui correspond à la démarche dont l'intention est de progresser dans ses apprentissages et de comprendre les moyens de résolution des situations-problèmes qui lui sont présentées (Zorn et Puustinen, 2017).

Cette intention permet de caractériser la recherche d'aide en tant que stratégie de régulation. Toutefois, elle se distingue des autres stratégies par deux aspects notables (Karabenick et Gonida, 2017) :

– au contraire des stratégies cognitives (mémorisation, organisation...), elle comprend nécessairement une forme d'interaction sociale, que ce soit avec les autres apprenants et/ou l'enseignant ;

– c'est la seule stratégie qui est potentiellement stigmatisante en raison de son éventuelle non-pertinence, et qui induit un « coût » personnel : par exemple, le fait de se sentir redevable vis-à-vis d'un autre apprenant qui lui a précédemment apporté de l'aide.

Cette double particularité permet de comprendre que la recherche d'aide est liée à la capacité de l'apprenant à faire preuve de compétences sur le plan cognitif et métacognitif, mais aussi sur le plan émotionnel et sur le plan social (Karabenick et Berger, 2013) :

– les compétences cognitives et métacognitives renvoient au fait d'être capable d'identifier la difficulté (besoin d'aide), de prendre conscience qu'une aide peut lui permettre de dépasser cette difficulté, de savoir comment demander de l'aide ;

– les compétences émotionnelles correspondent à la capacité de l'apprenant à réguler ses émotions quand il rencontre une difficulté : l'idée de paraître incompétent peut menacer l'estime de soi de l'apprenant et susciter un sentiment de honte qui peut l'amener à renoncer à demander de l'aide ;

– les compétences sociales concernent la capacité à identifier les personnes pouvant apporter l'aide désirée : le fait de demander de l'aide peut générer chez l'individu un sentiment de faiblesse vis-à-vis de la personne à laquelle il s'adresse (enseignant ou pairs par exemple).

L'identification de ces différentes compétences ont permis de dissocier le processus de recherche d'aide, de la notion de « besoin d'aide » : l'étudiant n'est pas forcément conscient de son besoin, ce qui l'amène à ne pas rechercher/demander de l'aide. Il peut également éviter de la rechercher parce qu'il se sent mal à l'aise vis-à-vis de la personne destinataire de la demande, ou encore rencontrer des difficultés dans sa formulation. C'est ainsi que les apprenants qui sont objectivement ceux qui ont le plus besoin d'aide, sont également ceux qui vont le moins la rechercher (Aleven *et al.,* 2006 ; Karabenick et Knapp, 1988 ; Puustinen, 1998 ; Ryan *et al.,* 1998). Au contraire, les étudiants les plus performants, qui utilisent plus fréquemment d'autres stratégies, sont ceux qui vont le plus probablement demander de l'aide, à différentes personnes s'ils pensent qu'ils en ont réellement besoin (Karabenick et Gonida, 2017). Ces résultats confirment le constat de Zimmerman et Martinez-Pons (1990) : la demande d'aide est le plus souvent effectuée par des étudiants plus avancés. Elle peut être formulée en direction de l'enseignant et/ou des pairs. Newman (2000, 2008) souligne la différence entre une recherche d'aide effectuée auprès d'un enseignant et celle réalisée auprès de leurs pairs, en termes d'impact

sur leurs apprentissages et en termes de coût. Pour Ryan et Shim (2012), la recherche d'aide informelle (entre pairs) est jugée plus accessible que la recherche formelle (auprès de l'enseignant) qui fait autorité.

2.2. Entraide

Si l'entraide scolaire est souvent associée aux démarches de soutien, voire de « cours particulier » (Duthoit *et al.,* 2011), le *chat*, en tant qu'outil numérique dans un laboratoire virtuel et distant, offre une fonctionnalité technique favorisant l'interaction sur le court terme dans le but d'avancer sur le problème à résoudre proposé par un enseignant dans une situation donnée. Dans une approche explicite de l'aide, les messages au sein du *chat* constituent un ensemble de fils de discussion qui *a priori* dans le cas d'une entraide peut débuter par une requête/demande d'aide, se poursuivre par une/des réponses/aides formulées par d'autres participants, et se clôturer par une acception de l'aide donnée (remerciement, présentation du résultat obtenu grâce à l'aide offerte). Le début d'une aide peut par ailleurs se situer à plusieurs moments (Thobois-Jacob et Pélissier, 2020) :

– après une demande d'aide explicite formulée au cours du lancement d'une activité ou lors de son déroulement (le plus souvent) ;

– après une recherche d'information que l'apprenant réalise seul ou en groupe à partir de consignes formulées par l'enseignant pour réaliser l'activité ;

– avant toute activité de l'apprenant dans le cadre d'une anticipation de l'enseignant en lien avec une difficulté prévisible ;

– après une activité réalisée dans le cadre d'un accompagnement de l'apprenant dans sa démarche réflexive visant à relier les nouveaux apprentissages à ceux déjà construits.

2.3. Questions de recherche

Dans cette recherche exploratoire, nous avons d'une part étudié l'utilisation de l'outil de communication par les étudiants lors d'un épisode d'entraide, et d'autre part l'exploitation de l'outil de consultation du terminal d'un pair lors d'un épisode d'entraide.

En ce qui concerne l'outil de communication instantanée, nous nous sommes en particulier intéressés aux questions de recherche suivantes : comment les échanges présents dans le *chat* intégré au laboratoire virtuel et distant supportent-ils le processus d'entraide entre les apprenants pendant la tâche d'apprentissage ? Cette tâche d'apprentissage mise en jeu dans le laboratoire virtuel impacte-t-elle la qualité (en termes de

complétude) des épisodes d'entraide ? Les apprenants s'engagent-ils individuellement et/ou collectivement, de manière uniforme, dans les différents épisodes d'entraide ?

Au niveau de la consultation du terminal d'un pair dans le processus d'entraide entre les apprenants, les questions suivantes sont au cœur de nos préoccupations : comment la consultation de l'écran/du terminal d'un pair supporte-t-elle le processus d'entraide ? Dans quelle mesure la consultation du terminal d'un pair est-elle un élément déclencheur d'un processus d'entraide ?

Pour apporter des éléments de réponse à ces questions, nous nous sommes appuyés sur une analyse qualitative et quantitative de données récoltées dans le cadre d'une expérimentation menée en contexte écologique dans l'enseignement supérieur. Les détails de l'expérimentation et notre méthodologie de recherche sont exposés dans la section qui suit.

3. Méthodologie

3.1. Description de l'expérience menée

3.1.1. Participants

Trente-neuf participants ont été sollicités pour réaliser cette expérimentation. Ce sont des étudiants inscrits en première année d'une formation universitaire technologique (Diplôme universitaire de technologie français — DUT). Cette formation, dite professionnalisante (c.-à-d. qui intègre la réalisation de projets et de stages en entreprises), se déroule sur deux ans après l'enseignement secondaire. Les étudiants sont sélectionnés après le baccalauréat en fonction de leurs résultats lors des deux dernières années du cursus d'enseignement secondaire (lycée) et d'une lettre de motivation. Dans notre étude, la spécialité est l'informatique. Majoritairement de genre masculin (30 hommes pour 9 femmes), les étudiants sont âgés de 17 à 19 ans. L'expérimentation s'inscrit dans le cadre d'un exercice permettant de réviser des cours de programmation informatique en langage Shell. L'objectif pédagogique est de réaliser un exercice collaboratif dans le but de réviser un langage informatique. Le laboratoire virtuel utilisé pour l'expérimentation (voir section suivante) supporte le travail collaboratif et enregistre les interactions des apprenants avec le système. Dans ce cadre, un pseudonyme est affecté à chaque participant et l'ensemble des participants est ensuite réparti aléatoirement en 13 trinômes. Cinq groupes sont constitués de trois hommes, sept groupes comprennent une femme et deux

hommes, et un groupe est composé de deux femmes et un homme. Les membres de chaque groupe sont installés dans des salles différentes et interagissent à distance avec l'outil de messagerie instantanée (*chat*) intégré au laboratoire virtuel. Les étudiants savent utiliser des outils de communication (discussion instantanée écrite ou audiovisuelle). Ils travaillent régulièrement à distance *via* des outils collaboratifs (wiki, éditeurs de texte partagés, gestion de projets, de tâches). Dans cette formation, les étudiants ont l'habitude de travailler ensemble.

3.1.2. L'EIAH : la plate-forme Lab4CE

Dans cette expérimentation, nous utilisons la plateforme Lab4CE dédiée à l'apprentissage de l'informatique (Broisin *et al.,* 2017a ; Venant *et al.,* 2017). Elle permet d'instrumenter, sous forme de différentes visualisations et de consoles, l'ensemble des actions et interactions requises pour résoudre le problème posé. L'interface mise à disposition des apprenants lors de l'expérimentation est illustrée par la figure 1.

Cette plateforme possède aussi des fonctionnalités pour collecter les données reflétant les différentes actions (exécution de commandes, visualisation de terminaux du groupe) réalisées par chaque apprenant et les interactions verbales qu'ils ont entre eux. Toutes les actions et interactions sont positionnables temporellement, individuellement et collectivement, afin de procéder à leur étude en fonction du type d'analyse des interactions (Dyke *et al.,* 2009).

3.1.3. Dispositif pédagogique

Les étudiants doivent réaliser un exercice collaboratif dans leur discipline principale, l'informatique. Ils travaillent en trinôme au développement d'une application de gestion et de vente de produits. L'organisation et la coordination des membres du trinôme sont laissés libres. Les modes d'interaction au sein d'un trinôme sont limités aux fonctionnalités offertes par Lab4CE.

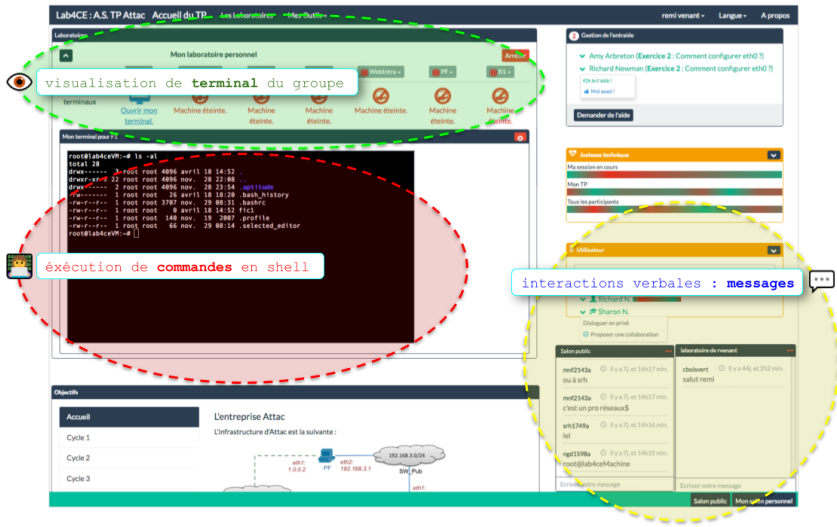


Figure 1 · Interface homme-machine du laboratoire virtuel et distant Lab4CE

L'application de gestion est découpée en trois parties : client, fournisseur et gestionnaire. Le module du client simule un processus de commande et d'achat relativement à des produits vendus par une entreprise. Le fournisseur approvisionne le stock en produits vendus par cette entreprise. Le gestionnaire établit le bilan des stocks de l'entreprise. La figure 2 présente une formalisation de la tâche à réaliser. Chaque module correspond à un script devant être développé en langage Shell. Ici, le Shell désigne à la fois le langage dans lesquelles les commandes sont écrites et l'interpréteur dans lesquelles elles sont exécutées. Les trois modules sont indépendants les uns des autres, c'est leur interdépendance fonctionnelle qui permet d'atteindre les objectifs de l'application.

Au sein d'un trinôme, un étudiant est chargé de réaliser un seul des trois modules de l'application. Chaque trinôme est donc constitué d'un étudiant client, d'un étudiant fournisseur et d'un étudiant gestionnaire. Les travaux conjoints de chaque membre permettent d'aboutir à l'application fonctionnelle finale (collaboration). Ces activités sont ainsi articulées autour d'une communication qui est instrumentée par la fonctionnalité de discussion instantanée intégrée à Lab4CE.

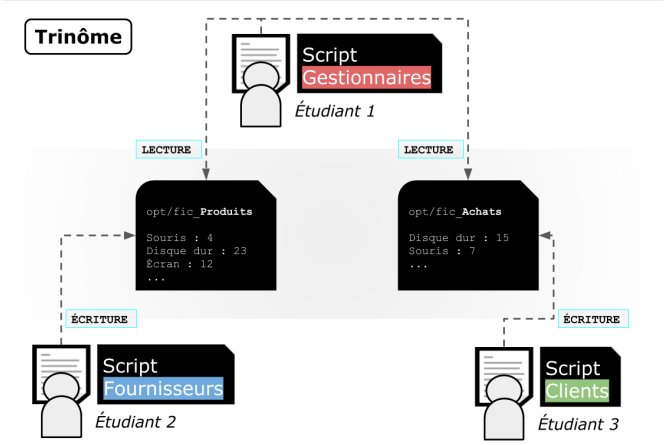


Figure 2 · Découpage fonctionnel de la tâche d'apprentissage

Le problème posé dans le cadre de l'expérimentation met donc en jeu des tâches interdépendantes à un objectif commun qui se résolvent par la conception collaborative nécessitant la reformulation du problème, le partage des représentations communes, la génération et gestion de contraintes, la proposition de solutions, l'évaluation itérative et la synchronisation cognitive tout au long de l'activité (Falzon *et al.*, 2004). L'organisation interne de chaque groupe pour réaliser la tâche étant spontanée, seuls la contrainte de temps et les algorithmes à développer sont imposés.

3.1.4. Déroulement/Expérimentation

L'expérimentation dure 2 h 30. Les premières minutes sont consacrées à la présentation des fonctionnalités du laboratoire distant Lab4CE par ses concepteurs. Ils expliquent le déroulement des exercices. Les étudiants sont ensuite appelés par trinôme et récupèrent chacun un dossier contenant :

– les renseignements nécessaires à leur identification (pseudonymes, espaces sur lesquels se connecter, etc.) ;

– des consignes sur le déroulement de la séance ;

– les énoncés des exercices (prise en main et exercice principal) comprenant les annexes algorithmiques.

Tous les étudiants ayant le même rôle sont affectés à la même salle informatique, comme le montre la figure 3. Une salle est donc réservée aux clients, une autre aux fournisseurs, et encore une autre aux gestionnaires.

À leur arrivée dans la salle, les étudiants doivent ouvrir une session sur leur machine et se connecter à Lab4CE avec les informations qui leur ont été transmises pour réaliser l'exercice. Ils peuvent communiquer avec les autres membres de leur trinôme grâce à la fonctionnalité de discussion instantanée. Ils réalisent alors la phase de prise en main, puis l'exercice principal.

Cinq observateurs sont mobilisés : deux d'entre eux sont les concepteurs de Lab4CE, ils constituent une ressource technique potentielle pour accompagner les étudiants ; les autres sont des chercheurs en sciences humaines s'intéressant au processus d'entraide et à sa mise en œuvre.

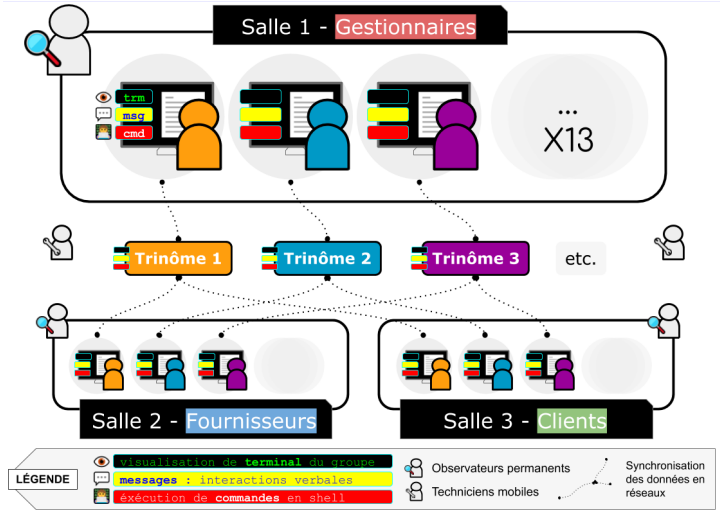


Figure 3 · Répartition des apprenants dans les salles selon leur rôle dans l'exercice à réaliser

3.2. Description de l'expérience menée

Pour répondre à nos questions de recherche, nous avons mis en place une méthode mixte, combinant une analyse qualitative des messages échangés dans l'outil de communication instantanée avec une analyse quantitative des traces d'interaction des apprenants avec le laboratoire Lab4CE. L'objectif principal de l'analyse qualitative est de structurer les messages échangés entre apprenants dans l'outil de communication instantanée afin d'identifier les épisodes d'entraide.

Dans le contexte des laboratoires virtuels et distants, l'analyse quantitative des traces permet de concevoir des tableaux de bord affichant des informations statistiques d'accès à un laboratoire (Orduña *et al.,* 2014), d'évaluer le progrès des apprenants dans l'acquisition de compétences (Romero *et al.,* 2015), ou de favoriser l'*awareness* et la réflexion des apprenants (Broisin *et al.,* 2017b). Dans la recherche présentée ici, l'analyse quantitative des données a pour but d'évaluer l'engagement des apprenants dans la tâche d'entraide ou le temps consacré aux épisodes d'entraide, mais également d'étudier les modalités mises en œuvre par les étudiants pour s'entraider.

3.2.1. Procédure d'analyse qualitative des données

Le travail de recherche s'est déroulé en deux temps. Premièrement, nous avons procédé à une analyse des données produites par les étudiants dans le *chat* de Lab4CE au cours de l'activité demandée. Le logiciel QDA Miner Lite a été utilisé. Son système de codage de segments de textes donne la possibilité d'étiqueter chaque composant d'un corpus de façon indépendante, d'après des catégories qui émergent au fur et à mesure du codage du corpus. Ces catégories (d'abord arrêtées isolément par les auteurs de cette étude) ont ensuite fait l'objet d'une mise en commun, débouchant sur un consensus interjuges global. Trois catégories ont plus particulièrement été identifiées :

– Des demandes d'aide qui soulignent des difficultés rencontrées par un participant. Ces demandes prennent la forme de questions ou d'objections posées aux autres participants, de manière explicite (question précise directe) ou implicite (expression d'un argument ou d'une difficulté à laquelle un membre du groupe se retrouve exposé et qu'il partage avec ses pairs dans le but éventuel d'obtenir une aide). Leur présentation à l'ensemble des acteurs constitue un élément déclencheur de l'interaction, et plus particulièrement de l'offre d'aide.

– Des propositions ou offres d'aide qui contiennent des informations susceptibles de solutionner le problème évoqué dans une demande d'aide, ou de permettre aux participants de progresser dans la réalisation de la tâche d'apprentissage. Ces propositions prennent la forme d'expressions, de phrases, de mots, d'éléments de ponctuation qui portent un message lié par exemple à l'utilisation des commandes Shell (pour attendre le résultat attendu), à l'organisation du travail collectif au sein du trinôme, ou encore à la diffusion d'une correction orthographique permettant aux autres participants de comprendre le message.

– Des acceptations/refus d'aide dans lesquels le demandeur d'une aide signifie qu'il a bien reçu l'aide offerte, qu'il remercie l'auteur de l'aide offerte, que celle-ci répond/ne répond pas aux difficultés qu'il rencontrait, ou encore qu'il occasionne une nouvelle difficulté.

Tableau 1 · Extrait des résultats de l'analyse qualitative pour le codage des messages échangés dans l'outil de communication instantanée

Heure	Rôle	Util.	Action	Message	Term. Util.	Cat. aide
16 h 46 min 3 s	Four.	etienne	message	role		Aide offerte
16 h 46 min 8 s	Four.	etienne	message	je regarde ton terminal la		Aide offerte
16 h 46 min 23 s	Client	ernest	terminal		etienne	
16 h 46 min 26 s	Four.	etienne	message	juste en dessous du echo met "read reponse"		Aide offerte
16 h 46 min 31 s	Four.	etienne	message	reponse est le nom de la variable		Aide offerte
16 h 46 min 59 s	Four.	etienne	message	non juste reponse		Aide offerte
16 h 47 min 6 s	Four.	etienne	message	"reponse" c'est le nom de ta variable		Aide offerte
16 h 47 min 20 s	Gest.	fabien	message	quelle variable ?		Demande d'aide
16 h 47 min 25 s	Four.	etienne	message	et ensuite en dessous tu fais un if qui teste si reponse c'est 1 ou 2		Aide offerte
16 h 47 min 33 s	Four.	etienne	message	la variable qui contient ce que l'utilisateur a rentré		Aide offerte
16 h 47 min 58 s	Gest.	fabien	message	oui mais a quel moment je dis que ce quil rentre ca va dans reponse ?		Demande d'aide
16 h 49 min 4 s	Four.	etienne	message	ca se fait tout seul		Aide offerte
16 h 49 min 11 s	Four.	etienne	message	c'est le "read"		Aide offerte
16 h 49 min 16 s	Four.	etienne	message	ca prend ce que l'utilisateur a entré		Aide offerte
16 h 49 min 16 s	Gest.	fabien	message	ok merci		Aide acceptée

Dans le tableau 1, les résultats de l'analyse des échanges réalisés *via* le *chat* apparaissent dans la dernière colonne : demande d'aide (en bleu cyan), proposition d'aide (en vert) et acceptation/refus d'aide (en orange).

Une fois cette phase d'analyse validée, les catégories ont pu être quantifiées et organisées de manière à nous permettre de suivre l'évolution de la place qu'elles occupent dans le discours des participants tout au long du projet étudié.

3.2.2. Procédure d'analyse quantitative des données

Nous avons par la suite procédé aux calculs d'indicateurs (explicités ci-dessous) à partir du jeu de données généré pour en faire l'analyse statistique. L'ensemble des manipulations a été réalisé en Python sous *Jupyter Notebook* avec la suite *scikit-learn*.

Outre l'analyse descriptive, une tentative d'analyse structurelle en Machine Learning a été réalisée, mais le jeu de données n'était pas assez conséquent pour en obtenir des résultats pertinents. Cette dernière partie a donc été retirée de l'étude.

4. Résultats

À travers les résultats obtenus, nous questionnons l'entraide en tant que phénomène spontané permettant à chaque apprenant de réaliser ses propres activités. Dans cette expérience, Lab4ce offrait deux fonctionnalités qui pouvaient être exploitées par les étudiants pour réaliser l'entraide : la messagerie instantanée et l'accès au terminal des autres participants. La théorie soutient ces deux formes d'interactions dans l'apprentissage exploratoire, et donc dans les travaux pratiques en sciences (Hofstein et Lunetta, 2004). Toutefois, le cadre théorique ne statue pas sur la modalité d'utilisation de ces interactions au sein de l'apprentissage. Les raisons évoquées sont d'un ordre plus pragmatique : il est nécessaire aux apprenants de pouvoir communiquer lors de la mise en confrontation de résultats divergents, et d'avoir accès au dispositif de l'autre pour comparer les situations et arriver à un consensus sur les connaissances acquises après expérimentation (Broisin *et al.*, 2017a).

Nous souhaitons donc déterminer quelles furent les fonctionnalités offertes par Lab4CE qui ont contribué à la mise en œuvre de l'entraide et, le cas échéant, comment elles ont été utilisées. Les analyses détaillées dans les sections ci-après mettent en avant les principaux résultats suivants :

– Le *chat* est principalement utilisé par les apprenants dans le cadre de sessions d'entraide (presque 70 % des messages échangés *via* le *chat* sont relatifs à l'entraide).

– Certaines corrélations tendent à montrer que plus de la moitié des sessions d'entraide sont « complètes » (c'est-à-dire incluant une demande, une offre et une acceptation d'aide) à travers les outils proposés dans le laboratoire virtuel.

– La situation d'apprentissage semble impacter la complétude des sessions d'entraide. Dans notre expérimentation, lorsqu'un apprenant prend un rôle spécifique lié à la tâche d'apprentissage, plus de 80 % de ses sessions d'entraide sont « complètes ».

– Certains indicateurs de répartition montrent une homogénéité très significative du nombre et du type de messages d'entraide (i) formulés par étudiant, et (ii) échangés au sein des différents groupes d'apprenants.

– L'outil d'accès au terminal d'un pair a été utilisé plus intensément par les étudiants impliqués dans les sessions d'entraide que par les autres. Ces premiers résultats semblent montrer l'utilité de cet outil pour soutenir les apprenants lors d'une session d'entraide.

– L'analyse quantitative des données montre que 55 % des sessions d'entraide sont précédées par au moins un accès au terminal d'un pair, et motive des études complémentaires pour valider l'hypothèse que la consultation de l'écran/terminal d'un pair est un des facteurs de motivation à l'origine d'une session d'entraide.

4.1. Utilisation de la messagerie instantanée pour l'entraide

Dans un premier temps, nous souhaitions déterminer la proportion de messages d'entraide parmi tous ceux échangés *via* le *chat*. Selon notre codification, un message est soit relatif à une entraide (demande, offre ou acceptation d'aide), soit non lié à l'entraide (c.-à-d. hors tâche), soit caractérisé par un résultat de codification ambigu (non consensuel).

Comme illustré sur la figure, 4,69 % des messages étaient relatifs à l'entraide. Presque un tiers concernaient des offres d'aide, tandis qu'un quart d'entre eux exprimait une demande. Un cinquième des messages n'était pas du ressort de l'entraide, et 10 % étaient trop ambigus pour être traités. Le *chat* semble donc être un outil essentiellement utilisé ici pour l'entraide, avec une faible proportion de messages ne traitant pas de la tâche et donc, non relative à l'entraide.

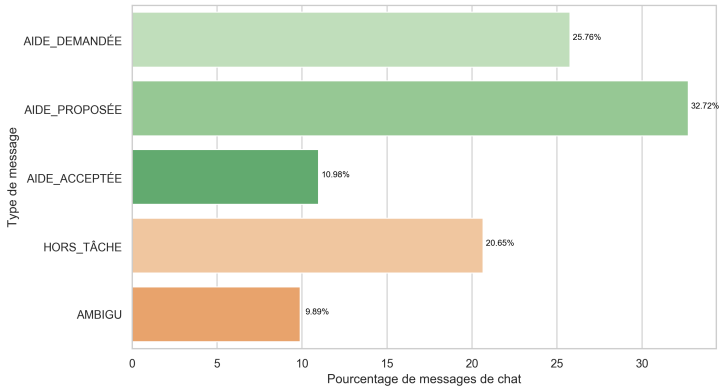


Figure 4 · Proportion des types de messages échangés *via* le *chat* selon notre codification

Du point de vue global, la distribution des messages par étudiant est représentée sur la figure 5. Ces données permettent d'observer qu'un apprenant a émis des messages d'entraide aussi bien pour demander de l'aide que pour en proposer.

À l'échelle de 6 classes, nous observons par ailleurs une corrélation (de Pearson) significative entre le nombre de messages de demande d'aide et le nombre de messages d'acceptation d'aide ($r = 0.53$, $p < 0.01$). Cette corrélation est d'autant plus forte pour les étudiants ayant les rôles « client » ($r = 0.64$, $p < 0.05$) et « gestionnaire » ($r = 0.83$, $p < 0.01$). En revanche, aucune corrélation significative n'a été identifiée pour les étudiants ayant le rôle « fournisseur ».

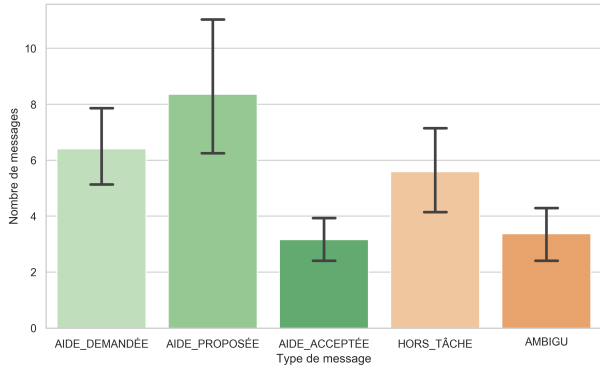


Figure 5 · Répartition des messages transmis *via* le *chat* par étudiant et catégorie de message

Ces résultats semblent montrer que les sessions d'entraide furent « complètes » pour plus de la moitié d'entre elles en général, et pour plus de 80 % d'entre elles lorsqu'un « gestionnaire » prend part à la session.

Toutefois, la différence de résultats entre le groupe des « fournisseurs » et les autres est à remarquer. L'absence de corrélation montre que les demandes d'aide des fournisseurs ne sont pas satisfaites, ce qui pourrait s'expliquer par la complexité des tâches d'apprentissage demandées aux étudiants « fournisseurs » dans cette expérimentation. Cette hypothèse reste toutefois à confirmer.

Aussi, nous n'identifions pas de corrélation, toujours à l'échelle de la classe, entre demande ou acceptation d'aide, et offre d'aide. Ces résultats ne permettent pas de mettre en avant, par exemple, un phénomène d'entraînement où les étudiants actifs en tant qu'aidés le seraient également en tant qu'aidants.

4.2. Analyse de l'utilisation du *chat* pour l'entraide

4.2.1.Dispersion et répartition globale

Afin de déterminer l'homogénéité de l'utilisation du *chat* pour l'entraide entre les apprenants, groupes et rôles, nous avons choisi d'étudier la distribution des données à travers deux classes d'indicateurs : un indicateur de dispersion et un indicateur de répartition. Nous utilisons les écarts interquartiles comme indicateur de dispersion puisque les données ne sont pas distribuées normalement pour la plupart. Nous mesurons la répartition avec l'indicateur de Gini (Thomas *et al.,* 1999) qui est un coefficient variant de 0 à 1 ; la, valeur « 0 » signifie une répartition uniforme parfaite, tandis que la valeur « 1 » signifie que l'ensemble des données est associé à un unique apprenant. Ainsi, un coefficient de Gini à 1 pour le nombre de messages de demande d'aide par étudiant signifierait, par exemple, que toutes les demandes ont été faites par un unique étudiant, tandis qu'une valeur de 0 signifierait que chaque étudiant a effectué le même nombre de demandes.

Nous avons appliqué les deux indicateurs sur les variables suivantes :

– nombre de messages d'aide (#helpMessages), c'est à dire de demandes, propositions acceptations et/ou refus émis par un apprenant ;

– nombre de sessions d'entraide auxquelles un étudiant a participé (#helpSessions) ;

– nombre de messages de type « aidé » (#helpeeMessages), c'est-à-dire désignant soit des demandes, soit des acceptations d'aide ;

– nombre de messages de type « aidant » (#helperMessages), c'est-à-dire désignant des offres d'aide) ;

– nombre de sessions d'entraide auxquelles un étudiant a participé en tant qu'aidé (#helpeeSessions) ;

– nombre de sessions d'entraide auxquelles un étudiant a participé en tant qu'aidant (#helperSessions).

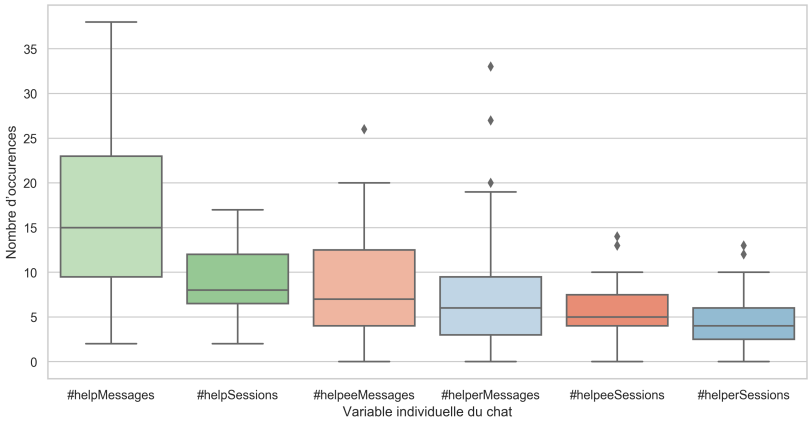


Figure 6 · Indicateurs de dispersion relatifs au nombre de messages et de sessions d'entraide par étudiant

Les indicateurs de dispersion sont représentés sur la figure 6 et les coefficients de Gini apparaissent dans le tableau 2.

Tableau 2 · Coefficients de Gini relatifs au nombre de messages et de sessions d'entraide par étudiant

Variable	Coefficient de Gini (par apprenant)
#helpMessages	0.17
#helpSessions	0.14
#helpeeMessages	0.19
#helperMessages	0.24
#helpeeSession	0.17
#helperSession	0.18

Les coefficients de Gini indiquent globalement une répartition assez homogène des différentes mesures entre étudiants. Toutefois les différences interquartiles Q1-Q3 pour les messages de type « aidé » (#helpeeMessages) et « aidant » (#helperMessages) vont du simple au double, alors qu'elles sont plus resserrées pour le nombre de sessions auxquelles a participé un apprenant en tant qu'aidé (#helpeeSession) ou aidant (#helperSession). On notera par ailleurs pour la variable #helperMessages la présence d'*outliers* représentant des étudiants « super-actifs », mais également la valeur la plus haute du coefficient de Gini (c.-à-d. 0.24). Cette variable est donc la moins homogène parmi celles étudiées.

Cette analyse peut toutefois être biaisée par l'effet de groupe. Si un étudiant « super-actif » au sein d'un groupe entraîne une hausse de la participation des autres apprenants, on pourrait alors avoir une hétérogénéité inter-groupe dans l'utilisation du *chat*.

4.2.2. Dispersion et répartition par groupe

Pour étudier cet hypothétique effet et analyser les différences d'utilisation du *chat* entre les 13 groupes, nous avons utilisé les mêmes indicateurs de dispersion et de répartition sur les variables suivantes :

– Nombre de messages d'aide émis au sein du groupe (#helpMessagesGP) ;

– Nombre de sessions d'entraide au sein du groupe (#helpSessionsGP) ;

– Nombre de messages de type « aidé » (#helpeeMessagesGP), c'est-à-dire désignant soit des demandes, soit des acceptations d'aide, émis au sein du groupe ;

– Nombre de messages de type « aidant » (#helperMessagesGP), c'est-à-dire désignant des offres d'aide, émis au sein du groupe ;

– Nombre de sessions d'entraide auxquelles au moins un des étudiants du groupe a participé en tant qu'aidé au sein du groupe (#helpeeSessionsGP) ;

– Nombre de sessions d'entraide auxquelles au moins un des étudiants du groupe a participé en tant qu'aidant au sein de son groupe (#helperSessionsGP).

Les indicateurs de dispersion sont représentés sur la figure 7 et les coefficients de Gini apparaissent dans le tableau 3.

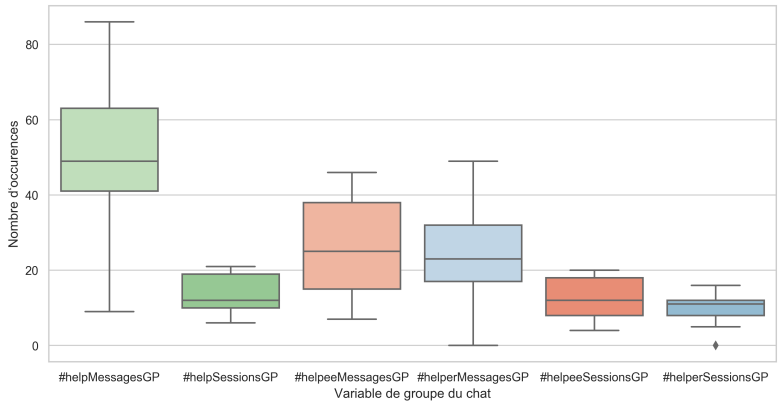


Figure 7 · Indicateurs de dispersion relatifs au nombre de messages et de sessions d'entraide par groupe

Les indicateurs montrent une répartition homogène de l'utilisation du *chat* pour l'entraide au sein des différents groupes, y compris pour le nombre de messages de type « aidant ». On remarque toutefois qu'ici encore, la plage de valeurs pour la variable #helperMessagesGP est la plus étendue, mais avec une faible valeur de l'indicateur de Gini associé, ainsi qu'une faible différence inter-quartiles Q1-Q3. En prenant en compte les résultats de répartition et de dispersion par étudiant, il est possible que si la majeure partie des groupes a un nombre de messages de type « aidant » similaire, la répartition de ces messages entre étudiants d'un même groupe n'est pas forcément homogène.

Tableau 3 · Coefficients de Gini relatifs au nombre de messages et de sessions d'entraide par groupe

Variable	Coefficient de Gini (par groupe)
#helpMessagesGP	0.15
#helpSessionsGP	0.11
#helpeeMessagesGP	0.16
#helperMessagesGP	0.17
#helpeeSessionsGP	0.12
#helperSessionGP	0.13

4.3. Analyse de l'accès au terminal de l'autre pour l'entraide

Comme mentionné dans la section 2, les étudiants étaient libres de consulter, à tout moment, le terminal d'un des membres de leur triade. Nous avons souhaité déterminer si cette fonctionnalité était utilisée pour l'entraide, mais également comme facteur de motivation à l'entraide. Pour permettre une analyse comparative selon le rôle de l'étudiant dans l'entraide, nous avons déterminé si pendant une session d'entraide, un étudiant qui y prenait part, tenait le rôle d'aidant ou d'aidé. Nous nous sommes appuyés sur la classification des messages émis *via* le *chat*. Toutefois, il s'avère que parfois, pendant une même session, un étudiant a transmis à la fois des messages de demandes et d'acceptation d'aide, et des messages d'offre d'aide. Pour déterminer le rôle principal de l'étudiant au sein d'une session d'entraide, nous avons donc choisi ce codage particulier :

– aidant : l'étudiant a émis plus de messages d'offre d'aide que de messages de demande et d'acceptation d'aide ;

– aidé : l'étudiant a émis plus de messages de demande et d'acceptation d'aide que de messages d'offre d'aide ;

– ambigu : l'étudiant a émis autant de messages de demande et d'acceptation d'aide que de messages d'offre d'aide ;

– externe : l'étudiant n'a pas émis de message de demande, d'acceptation ou d'offre d'aide pendant la session.

Dans le cas de la dernière catégorie « Externe », l'apprenant n'a pas pris part à la discussion relative à la session d'entraide, mais nous ne pouvons affirmer qu'il n'a fait partie en aucune manière de cette entraide. Il est possible qu'un étudiant consulte les échanges apparaissant dans le *chat* sans pour autant y contribuer, et qu'il soit incité à consulter le terminal d'un autre pour un problème que lui-même rencontre. Aussi, avant de proposer ou offrir une aide, il est possible que l'étudiant souhaite confirmer ses propres connaissances. Si les échanges du *chat* montrent qu'une session d'aide s'est mise en place pendant ce laps de temps, l'étudiant peut être amené à se raviser.

4.3.1. Utilisation du terminal pendant l'entraide

Nous avons analysé ici quelles étaient les principales utilisations du terminal selon le rôle des utilisateurs impliqués dans une action d'accès au terminal (c.-à-d. celui qui y accède et celui dont le terminal est consulté). Il est utile ici de rappeler que, parmi les 175 sessions d'entraide, seules 92 mobilisent au moins un aidant et un aidé.

Le nombre de sessions pour lesquelles au moins un étudiant d'un rôle X a accédé au terminal d'un autre étudiant de rôle Y est fourni dans le tableau 4. Ainsi, parmi les 175 sessions d'entraide observées, 134 ont fait intervenir l'accès au terminal d'un pair. Le tableau mentionne également le temps total de consultations du terminal d'un pair selon le rôle de l'étudiant.

Tableau 4 · Nombre et durée des sessions en fonction du rôle de l'étudiant dans les sessions d'entraide

Rôle de l'accesseur	Terminal d'un aidant	Terminal d'un aidé	Terminal d'un ambigu	Terminal d'un externe
Aidant	13 (9,7 %) 312s	**25 (18,7 %)** **1234s**	0 (0,0 %) -	12 (9,0 %) 140s
Aidé	**30 (22,4 %)** **1164s**	15 (11,2 %) **673s**	3 (2,2 %) 50s	**22 (16,4 %)** **612s**
Ambigu	4 (3 %) 60s	5 (3,7 %) 173s	0 (0,0 %) -	5 (3,7 %) 61s

Nous remarquons que la majorité des utilisations du terminal en situation d'entraide concernent l'accès au terminal d'un aidant par un aidé, d'un aidé par un aidant, et d'un externe par un aidé. Les temps totaux passés sur le terminal soulignent une utilisation importante pour les deux premiers cas. Nous ne pouvons toutefois pas négliger les situations où un aidé a observé le terminal d'un autre aidé, ni celles où un aidant a consulté le terminal d'un autre aidant ou d'un externe.

Les deux valeurs les plus fortes vont dans le sens de l'utilisation de l'outil lors d'une entraide convenue entre aidé et aidant, avec l'accès au terminal dans un but prescriptif (lorsque l'aidant consulte l'aidé) ou démonstratif (lorsque l'aidé consulte l'aidant). Concernant les autres valeurs, plusieurs hypothèses peuvent expliquer l'accès au terminal entre deux aidants, deux aidés, ou bien impliquant un externe.

Tout d'abord, puisque l'accès au terminal d'un pair dans Lab4CE ne nécessite qu'un clic de souris, un comportement « mécanique » de la part d'un apprenant cliquant rapidement sur les différents terminaux provoquerait l'augmentation artificielle de certaines variables par rapport à d'autres.

Pour tester cette hypothèse, nous avons analysé la durée passée sur le terminal d'un pair selon les rôles dans l'entraide, et observé leurs distributions. La distribution des durées de consultation du terminal d'un pair selon les rôles fait apparaître des éléments intéressants pour caractériser l'utilisation du terminal dans l'entraide. La figure 8 permet d'observer des médianes similaires lorsqu'un aidé consulte le terminal d'un autre aidé (8s.), d'un ambigu (9s.) ou d'un externe (7s.), ainsi que lorsqu'un aidant observe le terminal d'un aidé (10s.) ou d'un autre aidant (6s.). En revanche, la médiane du temps passé sur le terminal d'un aidant par un aidé est relativement faible (4s.), laissant présager une possible différence d'utilisation de l'outil selon les rôles impliqués dans l'accès au terminal. Aussi, il est à noter les cas particuliers, assez nombreux, caractérisés par des valeurs très élevées (presque 300s. pour un aidant ayant observé un aidé). Il y a donc eu une très forte hétérogénéité de l'utilisation du terminal dans l'entraide.

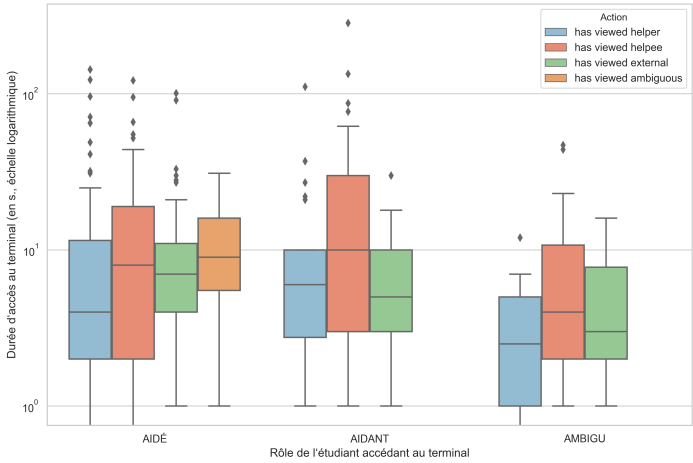


Figure 8 · Durée de consultation du terminal d'un pair selon les rôles des étudiants lors d'une session d'entraide

Pendant une session d'entraide, les étudiants du groupe ne sont pas forcément tous impliqués dans l'entraide « formelle » (c-à-d. discussion sur le *chat* dont les messages sont liés à la session d'entraide). Ainsi, il est possible d'analyser si le terminal a été, pendant les sessions d'entraide, utilisé plus intensément par les étudiants impliqués dans l'entraide que par les autres. Nous considérons ici un étudiant « impliqué » dans une session d'entraide de son groupe comme ayant un des rôles aidant, aidé ou ambigu

pendant cette session, tandis qu'un étudiant non impliqué est celui ayant le rôle externe pendant une session de son groupe (c-à-d. qu'il n'a pas pris part aux échanges du *chat* pendant cette session). Les étudiants impliqués ont accédé 134 fois au terminal d'un autre pour une durée totale de 4479 s., tandis que 60 accès au terminal d'un pair ont été effectués par des étudiants non impliqués pendant les sessions d'entraide, pour une durée totale de 1650s. Pour comparer ces valeurs, nous calculons la moyenne par rapport aux nombres de combinaisons étudiant-session d'entraide dont nous dénombrons 340 couples pour les étudiants impliqués et 185 pour les étudiants non impliqués. Ainsi, pendant une session, un étudiant impliqué a effectué en moyenne 0,39 accès au terminal d'un autre, pour une durée moyenne de 13,17s, contre 0,32 accès d'une durée de 8,92s pour un étudiant non impliqué.

Bien qu'il semble que l'outil d'accès au terminal d'un pair ait été utilisé plus intensément par les étudiants impliqués dans les sessions d'entraide que par les autres, nous avons souhaité déterminer si la durée passée sur les terminaux par des étudiants impliqués dans l'entraide différait de façon significative avec celle des étudiants non impliqués. Toutefois, une ANOVA de Kruskal-Wallis (la distribution des échantillons n'étant pas normale) ne permet pas de rejeter l'hypothèse nulle et de statuer sur la différence entre les deux échantillons, que l'on utilise la durée passée sur les terminaux ou son rapport à la durée de la session d'entraide.

Si l'on peut statuer sur la différence de quantité d'utilisation d'un point de vue global, nous n'observons pas de différence comportementale individuelle sur la seule base du temps passé pour distinguer les situations d'utilisation du terminal pour l'entraide. Toutefois, cette observation est potentiellement biaisée par le fait que nous ne puissions qualifier une participation à l'entraide qu'à travers l'utilisation du *chat*. Or, il se peut que certains étudiants aient participé à l'entraide de manière passive sur le plan de la communication, tout en ayant consulté le terminal d'autres étudiants.

4.3.2. Utilisation du terminal comme facteur de motivation pour l'entraide

En dehors des sessions d'entraide, l'accès au terminal d'un pair a également été utilisé. Le terminal permettant d'observer ce qu'un autre fait, il pourrait participer à l'étape de détection d'un besoin d'aide personnel (au sens de Nelson-Le Gall (1981)), ou à l'inverse, de détection d'un besoin d'aider un pair.

Certaines sessions d'entraide suivant immédiatement d'autres sessions d'entraide, voire même étant incluses dans des sessions plus longues, nous n'avons retenu dans notre analyse que les 83 sessions disposant de traces en amont qui ne soit pas liées à des sessions d'entraide. En effet, le moment de l'accès au terminal d'un pair en amont d'une session d'entraide varie énormément, comme le montre, sur la figure 9, la distribution temporelle des accès au terminal de pairs précédant une session d'entraide. Nous avons donc décidé de conserver les traces d'accès à un terminal n'appartenant pas à des sessions d'entraide, et précédant dans la limite de 150 secondes une session d'entraide (soit environ 43 % des traces de terminal avant entraide). Ce filtrage permet de conserver le maximum de traces dans le minimum de temps précédant la session d'entraide. En effet, plus le temps avant la session est long, plus le nombre de facteurs motivant l'entraide peut augmenter. Ce filtrage sert donc à comparer des mesures semblables.

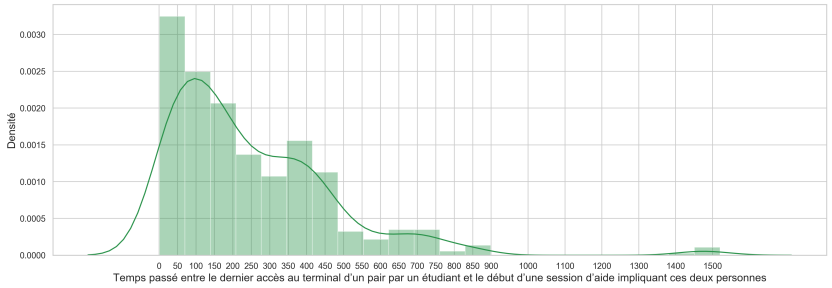


Figure 9 · Distribution des accès au terminal d'un pair avant le début d'une session d'entraide

Le tableau 5 expose les pourcentages de sessions impliquant, dans les traces en amont, l'usage du terminal. La « source » fait référence au rôle futur de l'utilisateur accédant au terminal lors de la session d'entraide qui va suivre, alors que la « cible » correspond au rôle futur de l'utilisateur dont le terminal est consulté. Les lignes et colonnes A* mentionnent les accès quel que soit le rôle parmi « aidé », « aidant » et « ambigu ». Ces valeurs mentionnent donc le pourcentage de sessions impliquant dans les traces en amont l'usage du terminal par, ou vers un étudiant ayant pris part à la session. Il y a donc plus de 60 % des sessions pour lesquels le terminal a été mobilisé en amont par un étudiant ayant pris part à la session vers un pair ayant lui aussi été actif dans cette session.

Tableau 5 · Pourcentage de sessions qui ont été précédées d'au moins un accès au terminal d'un pair

Cible → Source ↓	**Aidé**	**Aidant**	**Ambigu**	**Externe**	**A***
Aidé	**7,23 %**	**24,09 %**	**01,20 %**	**27,71 %**	**32,53 %**
Aidant	**18,07 %**	03,61 %	01,20 %	09,64 %	22,89 %
Ambigu	01,20 %	02,41 %	01,20 %	3,61 %	04,82 %
Externe	24,10 %	08,43 %	00,00 %	**15,66 %**	32,53 %
A*	30,12 %	26,51 %	03,61 %	40,96 %	**60,24 %**

5. Discussion des résultats

Les résultats des différentes analyses qualitatives et quantitatives ouvrent sur des interprétations que nous souhaitons repositionner dans la définition de la notion d'« entraide ».

Premièrement, les corrélations réalisées dans le cadre de cette étude semblent montrer que les sessions d'entraide sont majoritairement « complètes ». Un processus d'entraide pourrait donc être défini et caractérisé selon des étapes (demande, aide offerte, aide acceptée/refusée), des éléments déclencheurs (ex : difficultés rencontrées par un demandeur) ou encore des phrases de mises en œuvre de l'entraide qui ne sont pas forcément linéaires mais qui structurent ce processus (ex : stratégie de demande d'aide, procédure de formulation de l'aide dans le *chat*, compréhension de l'aide formulée). Des variables en lien avec le rôle de chaque acteur impliqué, et le niveau de complexité de l'activité collective menée, restent à être identifiées.

Deuxièmement, la non-corrélation entre demande ou acceptation d'aide et offre souligne l'opportunité de notifier les acteurs face à l'entraide, selon leur profil. Deux profils pourraient être identifiés dans le processus proposé : un étudiant plutôt aidé ou plutôt aidant. La question des origines de ces profils ainsi que des stratégies de développement restent à explorer.

Puis, dans le cadre de ce processus d'entraide, des variables d'échanges intergroupes restent à observer. En effet, même si la majeure partie des groupes a un nombre de messages d'aidants similaire, la répartition de ces

messages au cours de la tâche collective n'est pas forcément homogène. Une approche vers des facteurs de détermination de cette répartition de ces messages est à envisager.

Troisièmement, le processus d'entraide se construit à partir d'échanges (verbaux et de partages de terminaux) intégrés à des stratégies propres de chacun des acteurs impliqués : stratégie mise en place pour aider les autres participants, être aidé par eux, ou encore une stratégie visant à échanger socialement sur des points qui n'ont peut-être rien à voir avec la tâche (ex : prise de contact, sécurisation sur des modalités d'interactions offertes par le dispositif numérique). Ces dernières stratégies, même si elles ne sont pas directement reliées à un processus d'entraide, introduisent, participent peut-être à sa mise en place.

Quatrièmement, le processus d'entraide dans cette étude a été abordé à partir des traces, des informations observables, qualifiables et quantifiables. Cependant, celles-ci ne sont pas les seules à permettre de spécifier le processus d'entraide. Une approche des ressentis des apprenants (ex : sentiment de confiance aux personnes sollicitées dans ce processus, sentiments d'efficacités personnelle et collective) mais aussi des représentations de cette entraide pourrait venir compléter ces analyses de traces et ainsi donner du sens aux interactions déjà analysées et issues des échanges ayant eu lieu *via* le *chat* et la consultation des terminaux.

Cinquièmement, l'analyse de l'utilisation des terminaux comme facteur de motivation pour l'entraide ne permet pas de statuer sur cette hypothèse. Toutefois, la possibilité de voir le terminal d'un pair à tout moment était une fonctionnalité que les étudiants ne connaissaient pas avant de débuter l'expérimentation. Il serait donc utile de réitérer cette étude dans un contexte où ceux-ci y sont déjà habitués. Enfin, l'étude des causes de l'entraide est, la plupart du temps, réalisée du point de vue de l'aidé. Notre expérimentation montre que parfois, le futur aidant semble être proactif vis-à-vis du futur aidé. Il serait intéressant d'investiguer plus en profondeur cette observation.

6. Conclusion

Dans cet article, nous avons présenté les résultats d'une recherche transdisciplinaire exploratoire qui souligne et caractérise la nature des interactions entre les apprenants qui ont utilisé le laboratoire virtuel distant Lab4CE dans un cours d'informatique à l'Université. Deux problématiques ont été plus particulièrement abordées : celle de l'apport

du *chat* intégré au laboratoire dans le processus d'entraide, et celle de la participation de la consultation du terminal dans ce même processus.

Ce travail d'analyse inédit à notre connaissance a permis de produire de nouvelles connaissances sur le processus d'entraide et de manière générale, à propos de l'impact des technologies sur l'entraide dans un contexte de laboratoire distant. Les résultats montrent que :

– différentes configurations d'entraide sont observées en fonction de l'organisation des fils d'interactions entre les participants à l'entraide ;

– la tâche d'apprentissage semble impacter la qualité des sessions d'entraide ;

– les étudiants contribuent de façon homogène aux messages d'entraide transmis sur l'outil de communication instantanée ;

– l'outil de consultation du terminal d'un pair supporte les épisodes d'entraide ;

– cet outil pourrait également être un des facteurs de motivation à l'origine d'une session d'entraide.

Ces observations restent à mettre en perspective. Si les résultats présentés dans cet article montrent que les outils de messagerie instantanée et de consultation de l'écran d'un pair sont utiles pour soutenir le processus d'entraide, ils montrent également que ces outils ne semblent pas être suffisants. En effet, une part non négligeable de sessions d'entraide n'ont pas été « complètes » dans notre expérimentation, c'est-à-dire qu'elles n'ont pas reçu de proposition/offre satisfaisante. Des analyses plus fines des interactions prenant place entre les apprenants lors d'une session d'entraide, combinées à des observations in situ et à des questionnaires adressant les attentes des apprenants en termes de support à l'entraide, devraient permettre d'identifier certains leviers de motivation à la participation à l'entraide, et ainsi de proposer de nouveaux artefacts numériques soutenant ces leviers dans le contexte des laboratoires virtuels et distants.

La crise sanitaire internationale de 2020 et les modalités d'enseignement à distance ou hybrides qui ont été massivement mises en place ont fait apparaître de nouvelles pratiques et de nouveaux comportements d'entraide entre les apprenants. Notre expérience d'enseignant montre qu'aujourd'hui les étudiants de l'enseignement supérieur utilisent des outils de communication et de partage (tels que la plateforme Discord qui offre la possibilité de facilement créer des salons de communication vocaux et écrits, ou des espaces d'échanges de documents

en ligne, ou encore la messagerie électronique) externes aux environnements d'apprentissage afin de masquer les échanges qui sont mis œuvre pour réaliser les tâches d'apprentissage demandées. Afin de favoriser l'engagement des apprenants dans le processus de demande d'aide au sein même des environnements d'apprentissage, nous étudions actuellement la conception d'une fonctionnalité intégrée à Lab4CE permettant aux apprenants d'adresser une demande d'aide à tous les apprenants impliqués dans la même tâche d'apprentissage, ou à un apprenant en particulier, ou encore à un enseignant connecté à la plate-forme. Cette fonctionnalité vise également à offrir à l'étudiant à l'initiative de la demande d'aide la possibilité de masquer son identité, et ainsi de réaliser des demandes d'aide de manière anonyme afin d'éviter certains phénomènes affectifs bloquants. Toutefois, d'autres études relatives au positionnement de l'étudiant vis-à-vis de ses pairs ainsi que des enseignants, doivent être menées afin d'identifier certains facteurs (psychologiques) qui pourraient participer à l'augmentation de la motivation des apprenants à initier des épisodes d'entraide au sein même des systèmes d'apprentissage. Ainsi chaque apprenant pourrait réellement profiter des connaissances et savoir-faire des autres, et de nouvelles connaissances liées au processus d'entraide pourraient émerger.

RÉFÉRENCES

Aleven, V., McLaren, B., Roll, I. et Koedinger, K. (2006). Toward meta-cognitive tutoring: a model of help-seeking with a cognitive tutor. *International Journal of Artificial Intelligence in Education, 16*, 101–130.

Blessing, L. (1993). A process-based approach to computer supported engineering design. Dans *Proceedings of the 9th International Conference on Engineering Design* (p. 1393-1400). Heurista.

Bourdeloie, H. (2012). L'appropriation des dispositifs d'écriture numérique : translittératie et capitaux culturel et social. *Études de Communication, 38*, 23–36.

Broisin, J., Venant, R. et Vidal, P. (2017a). Lab4CE: A remote laboratory for computer education. *International Journal of Artificial Intelligence in Education, 27*(1), 154-180.

Broisin, J., Venant, R. et Vidal, P. (2017b). Awareness and reflection in virtual and remote laboratories: the case of computer education. *International Journal of Technology-Enhanced Learning, 9*(2-3), 254-276.

Darses, F. (2002). Trois conditions sociotechniques pour l'optimisation de la conception continue du système de production. *Revue française de gestion industrielle, 21*(1), 5-27.

Duthoit, E., Mailles-Viard Metz,S. Charnet, C. et Pélissier, C. (2011). Entraide en ligne : le cas d'un forum de discussion utilisé en tant que ressource externe au contexte d'apprentissage. Dans *Actes du colloque « Échanger pour apprendre en ligne »*. https://hal.archives-ouvertes.fr/hal-02010556/document

Dyke, G., Lund, K. et Girardot, J.-J. (2009). Tatiana: an environment to support the CSCL analysis process. Dans *Proceedings of the 9th International Conference on Computer Supported Collaborative Learning* (p. 58–67). International Society of the Learning Sciences.

Falzon, P., Darses, F., Détienne, F. et Visser, W. (2004). Les activités de conception et leur assistance. Dans P. Falzon (dir.), *Ergonomie* (p. 545–563). Presses universitaires de France « Hors collection ».

Hofstein, A. et Lunetta, V. (2004). The laboratory in science education: Foundations for the twenty-first century. *Science education, 88*(1), 28–54.

Jacobs, S. E., Sokol, J. et Ohlsson, A. (2002). The newborn individualized developmental care and assessment program is not supported by meta-analyses of the data. *The Journal of Pediatrics, 140*(6), 699–706.

Karabenick, S. A. et Berger, J.-L. (2013). Help seeking as a self-regulated learning strategy. Dans *Proceedings of the Applications of self-regulated learning across diverse disciplines: A tribute to Barry J. Zimmerman* (p. 237-261). IAP.

Karabenick, S. A. et Gonida, E.N. (2017). Academic help seeking as a self-regulated learning strategy. Dans P.A. Alexander, D.H. Schunk, J.A. Greene, (dir.) *Handbook of self-regulation of learning and performance* (p. 421–433). Routledge/Taylor & Francis Group.

Karabenick, S.A. et Knapp, J.R. (1988). Help seeking and the need for academic assistance. *Journal of Educational Psychology, 80*, 406–408.

Lonchamp, J. (2003). *Le travail coopératif et ses technologies*. Hermès.

Nelson-Le Gall, S. (1981). Help-seeking: An understudied problem-solving skill in children. *Developmental Review, 1*, 224-246.

Newman, R. S. (2000). Social influences on the development of children's adaptive help seeking: The role of parents, teachers, and peers. *Developmental Review, 20*, 350–404.

Newman, R. S. (2008). Adaptive and non-adaptive help seeking with peer harassment: An integrative perspective of coping and self-regulation. *Educational Psychologist, 43*, 1–15.

Orduña, P., Almeida, A., López-de-Ipiña, D. et Garcia-Zubia, J. (2014). Learning analytics on federated remote laboratories: Tips and techniques. Dans *Proceedings of the 5th IEEE Global Engineering Education Conference* (p. 299-305). IEEE.

Puustinen, M. (1998). Help-seeking behavior in a problem-solving situation: Development of self-regulation. *European Journal of Psychology of Education, 13*(2), 271–282.

Romero, S., Guenaga, M., García-Zubía, J. et Orduña, P. (2015). Automatic assessment of progress using remote laboratories. *International Journal of Online and Biomedical Engineering, 11*(2), 49-54.

Ryan, A.M., Gheen, M.H. et Midgley, C. (1998). Why do some students avoid asking for help? An examination of the interplay among students' academic efficacy, teachers' social-emotional role, and the classroom goal structure. *Journal of Educational Psychology, 90*(3), 528–535.

Ryan, A.M. et Shim, S.S. (2012). Changes in help-seeking from peers during early adolescence: Associations with changes in achievement and perceptions of teachers. *Journal of Educational Psychology, 104*, 1122–1134.

The Design-Based Research Collective (2003). Design-Based Research: An Emerging Paradigm for Educational Inquiry. *Educational Researcher, 32*(1), 5-8.

Thobois-Jacob, L. et Pélissier, C. (2020). Help-seeking at the Heart of an Interactive Space: the Case of a Flipped Classroom at University. Dans Pélissier C. (dir.), *Support in Education* (p. 213-232). John Wiley & Sons.

Thomas, V., Wang, Y. et Fan, X. (1999). *Measuring education inequality: Gini coefficients of education.* The World Bank.

van Joolingen, W. R., de Jong, T., Lazonder, A. W., Savelsbergh, E. R. et Manlove, S. (2005). Co-Lab: Research and development of an online learning environment for collaborative scientific discovery learning. *Computers in Human Behavior, 21*(4), 671-688.

Venant, R., Sharma, K., Vidal, P., Dillenbourg, P. et Broisin, J. (2017). Étude du comportement des apprenants en situation de travaux pratiques et de sa corrélation sur leur réussite académique. Dans *Actes de la 8e Conférence sur les environnements informatiques pour l'apprentissage humain* (p. 17-28). Atief.

Wang, F. et Hannafin, M. J. (2005). Design-based research and technology-enhanced learning environments. *Educational Technology Research and Development, 53*(4), 5-23.

Zimmerman, B.J. et Martinez-Pons, M. (1990). Student differences in self-regulated learning: Relating grade, sex, and giftedness to self-efficacy and strategy use. *Journal of Educational Psychology, 82*, 51–59.

Zorn, S. et Puustinen, N. (2017). L'aide aux apprentissages : le cas des collégiens avec un trouble du spectre de l'autisme et de leurs enseignants. *Recherches en éducation, 30*, 111–124.

Git4School : un tableau de bord pour assister la prise de décisions de l'enseignant lors des cours de génie logiciel

▶ **Jean-Baptiste RACLET** (Irit, Université Toulouse 3), **Franck SILVESTRE** (Irit, Université Toulouse 1), **Mika PONS** (Irit, Université Toulouse 3)

■ **RÉSUMÉ** • Cet article présente Git4School, un tableau de bord pour les enseignants offrant des visualisations s'appuyant sur des données extraites des dépôts Git des apprenants, combinées à des informations contextuelles temporelles. Sur la base de plusieurs expérimentations, nous montrons un bon sentiment général des apprenants sur l'utilisation de Git dans un contexte éducatif. De plus, nous montrons comment Git4School peut être utile pour identifier les faiblesses des conceptions des situations d'apprentissage et cibler leur amélioration.

■ **MOTS-CLÉS** • enseignement du génie logiciel, conception de situation d'apprentissage, intervention pédagogique, prise de décision, tableau de bord, Git.

■ ***ABSTRACT*** • *This article presents Git4School, a dashboard for teachers providing visualizations based on data extracted from learners' Git repositories, combined with temporal contextual information. Based on several experiments, we show a good general feeling of learners about using Git in an educational context. We show how Git4School can be useful for identifying weaknesses in learning designs and for targeting their needs for improvement.*

■ ***KEYWORDS*** • *software engineering education, learning design, pedagogical intervention, decision-making, dashboard, Git.*

1. Introduction

Une conception de situation d'apprentissage[1] décrit une séquence d'apprentissage à l'échelle d'une partie ou d'une unité d'enseignement entière. Elle identifie les principaux acteurs impliqués (enseignants et apprenants), les tâches d'enseignement et d'apprentissage à accomplir ainsi que les ressources pédagogiques utilisées pour soutenir les activités de la séquence (Lockyer *et al.,* 2013). Ce concept est apparu au début des années 2000 dans le but de favoriser la mutualisation de scénarios pédagogiques exploitant les technologies numériques (Emin *et al.,* 2011).

Si les conceptions de situation d'apprentissage permettent de décrire une intention pédagogique, elles n'identifient cependant pas comment les apprenants participent réellement à cette conception, que ce soit pendant son développement ou *a posteriori*. Pour obtenir de telles informations, il est nécessaire de s'appuyer sur la collecte et l'analyse de traces d'utilisation (Choquet *et al.,* 2007), ce que l'on désigne plus récemment sous l'appellation *learning analytics*. Les *learning analytics* constituent un concept clé pour obtenir une perspective globale de l'impact des activités d'apprentissage (Wise, 2014). En effet, les *learning analytics* s'appuient sur le recueil d'ensembles de données correspondant aux traces du travail des apprenants afin de mesurer leur engagement et, éventuellement, d'extraire des preuves d'apprentissage. Dans le contexte de conception de situation d'apprentissage, la visualisation des données d'apprentissage peut soutenir la prise de décision de l'enseignant au cours d'une séquence : synthétisée sous la forme d'un tableau de bord (Schwendimann *et al.,* 2017), elle permet aux enseignants d'obtenir des informations en temps réel sur l'avancement des élèves dans leurs activités et permet ainsi d'adapter l'intervention pour rendre leur situation d'apprentissage plus efficace.

Cet article s'inscrit dans cette vision et l'aborde dans le contexte particulier de l'enseignement du génie logiciel. L'enseignement dans cette discipline de l'informatique combine des aspects théoriques abstraits avec des considérations très pratiques souvent outillées. En particulier, les travaux pratiques impliquent généralement l'utilisation d'un environnement de développement intégré avec un éditeur de code source, un compilateur et un débogueur. En plus de ces outils, des outils de gestion de versions de code source comme Git (https://git-scm.com/) qui sont

[1] En se basant sur le TEL thesaurus, le terme anglophone « *learning design* » est traduit dans cet article par « conception de situation d'apprentissage ».

largement utilisés dans l'industrie du logiciel (Bourque et Fairley, 2014), sont parfois utilisés pour collecter les travaux des étudiants. Dans cet article, nous explorons une manière originale de construire des visualisations s'appuyant sur des données extraites de dépôts Git combinées à des informations contextuelles temporelles complémentaires (par exemple, le travail a été effectué à l'intérieur ou à l'extérieur de la classe, avant ou après une intervention ou une activité particulière). Cela nous a amené aux deux questions de recherche suivantes :

– (**RQ1**) Les étudiants sont-ils capables, quel que soit leur niveau d'études, d'utiliser des outils de gestion de versions comme Git à des fins éducatives ?

– (**RQ2**) Dans quelle mesure un tableau de bord construit à partir de données extraites de Git combinées à des données contextuelles temporelles complémentaires peut-il aider l'enseignant à prendre des décisions pendant un cours ?

Pour répondre à (**RQ1**), nous avons mené une étude quantitative en contexte écologique sur les activités des étudiants relatives à l'utilisation de Git dans 5 cours répartis dans 4 niveaux d'études différents. Nous avons abordé (**RQ2**) d'abord en concevant le tableau de bord appelé Git4School et ensuite par une étude qualitative sur l'utilisation de Git4school en contexte écologique.

La section 2 présente un état de l'art des travaux relatifs aux tableaux de bord, à la prise de décision dans le contexte de la conception de situation d'apprentissage ainsi qu'à l'exploitation de données d'apprentissage extraites des outils de gestion de versions. La section 3 présente l'étude quantitative relative à l'utilisation de Git par les étudiants. La section 4 est consacrée à la description du tableau de bord Git4School tandis que la section 5 présente deux études de cas sur l'utilisation de Git4School dans les cours en face à face. L'article se termine par une synthèse des réponses apportées à (**RQ1**) et (**RQ2**) et propose des pistes d'amélioration.

2. Travaux connexes

Faisant partie de la grande famille des outils d'analyse de données, les outils et solutions issus des *learning analytics* sont efficaces lorsqu'ils aident les personnes à prendre des décisions et à agir (Van Harmelen et Workman, 2012). Pour tenir cette promesse d'efficacité, nous suivons l'approche préconisée par Gašević *et al.* (2015) qui défendent que les *learning analytics* devraient être fortement ancrées dans les cadres pédagogiques issus de la recherche en éducation.

Comme défini par Lockyer *et al.* (2013) une conception de situation d'apprentissage décrit la séquence des tâches, des ressources et des supports d'apprentissage qu'un enseignant construit pour les cours des étudiants. Dans leurs travaux, les auteurs indiquent, de plus, que les objectifs d'apprentissage et le plan pédagogique peuvent être évalués au moyen de *learning analytics*. Celles-ci fournissent les informations manquantes sur la manière dont les étudiants participent au plan initial et peuvent ensuite aider les enseignants et les formations à évaluer l'impact des activités d'apprentissage.

Plus axée sur la prise de décision des enseignants pendant les séquences de cours, la communauté de recherche sur l'apprentissage collaboratif supporté par l'informatique a développé des idées similaires autour du concept de technologie d'orchestration (Tchounikine, 2013 ; Tissenbaum et Slotta, 2019) : « *des technologies qui fournissent aux enseignants des moyens de surveillance ou d'intervention* ». Les technologies d'orchestration prennent souvent la forme de tableaux de bord aidant les enseignants à obtenir un retour d'information pour savoir quand et où intervenir (Tissenbaum et Slotta, 2019).

Enfin, Wise (2014) a proposé le concept de conception d'intervention pédagogique fondée sur les *learning analytics*[2] en tant qu'« *efforts systématiques pour intégrer l'utilisation des learning analytics comme partie productive des pratiques d'enseignement et d'apprentissage dans un contexte éducatif donné* ». Notre proposition s'inscrit directement dans la lignée des travaux de Wise puisque nous utilisons les *learning analytics* et plus précisément les visualisations afin d'aider les enseignants à réaliser des interventions pendant leurs cours en face à face. Le tableau de bord que nous avons conçu et que nous décrivons plus loin s'inspire principalement des recherches menées par Bakharia *et al.* (2016, p. 330) qui visent à « créer des représentations plus significatives des données pour les enseignants ». Dans notre cas, nous avons combiné deux dimensions du cadre défini par Bakharia *et al.* : la dimension temporelle et la dimension dynamique de cohorte, les données provenant d'une part de l'outil de gestion de version du code source et, d'autre part, des données fournies par l'enseignant.

Il existe dans la littérature plusieurs travaux concernant l'extraction de métriques à partir des journaux d'un outil de gestion de version du code.

---

[2] En anglais, « *pedagogical learning analytics intervention design* ».

Contrairement à notre approche qui analyse les messages de *commits* porteurs d'informations sémantiques (étape de conception de l'apprentissage au moment de la réalisation d'un exercice identifié), ces travaux s'intéressent plutôt aux informations sur la volumétrie des *commits*, leur fréquence, leur temporalité par rapport à une échéance.

La plupart des travaux existants considèrent les outils de gestion de version centralisés appelés CVS (Liu *et al.,* 2004 ; Mierle *et al.,* 2005) ou bien alors SVN (Glassy, 2006 ; Jones, 2010 ; Kay *et al.,* 2006 ; Poncin *et al.,* 2011). À notre connaissance, la seule approche qui s'appuie sur Git est celle de Robles et Gonzalez-Barahona (2013). Les objectifs de ces travaux sont variés : le suivi des étudiants (Glassy, 2006), l'analyse de la collaboration entre les différents membres d'un groupe de travail (Jones, 2010 ; Kay *et al.,* 2006 ; Mittal et Sureka, 2014 ; Robles et Gonzalez-Barahona, 2013) avec éventuellement la corrélation avec les résultats obtenus (Liu *et al.,* 2004) ou une prédiction des performances (Mierle *et al.,* 2005). Aucun de ces travaux ne tente de combiner les données extraites des dépôts de code avec d'autres données contextuelles pour la prise de décision dans les interventions pédagogiques comme nous le présentons dans cet article.

3. Évaluation quantitative

L'objectif principal de l'étude quantitative qui est présentée dans cette section est de fournir des éléments de réponse à la question de recherche (**RQ1**).

3.1. Contexte des différentes expérimentations

Durant l'année universitaire 2019-2020, les concepteurs de Git4School ont demandé à leurs étudiants d'utiliser Git ainsi que le service d'hébergement GitHub (https://github.com), dans le cadre de 5 enseignements différents concernant le développement logiciel. Plus précisément, les étudiants devaient soumettre leurs travaux en effectuant des opérations de *commit* au sein d'un dépôt Git individuel. Ceci a entraîné la création de 180 dépôts Git par 156 étudiants, contenant 3 082 *commits*.

Le tableau 1 indique le nombre de dépôts créés par les étudiants en fonction du niveau du diplôme (selon la classification *International Standard Classification Of Education* (ISCED, 2012) qu'ils préparent et leur année d'études.

Tableau 1 · Nombre de dépôt en fonction du niveau d'études

Niveau ISCED	5 (DUT, licence pro)		7 (master)	
Année d'étude	2e	3e	1re	2e
Nombre de dépôts	73	16	55	36
Nombre d'étudiants	49	16	55	36

Parmi tous les *commits* effectués par les étudiants, on peut distinguer des *fix commits* qui comportent un message imposé par l'enseignant dans son énoncé afin d'indiquer la fin d'un exercice donné.

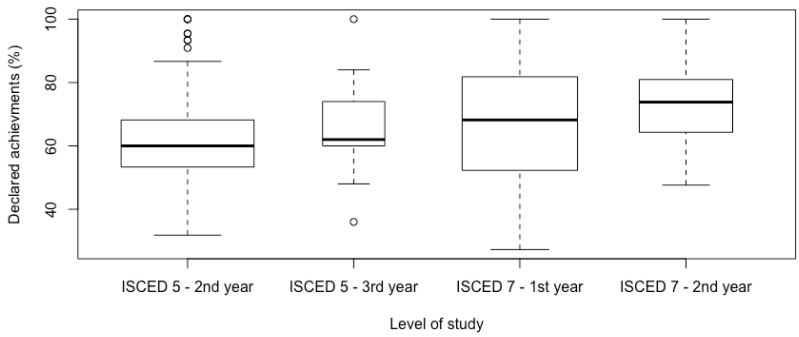


Figure 1 · Part de *fix commits* sur l'ensemble des *commits* en fonction de l'année d'étude

Dans les dépôts mentionnés ci-dessus, 2424 *fix commits* ont été identifiés. La figure 1 montre leur distribution selon le niveau du diplôme et l'année d'étude de l'étudiant.

Précisons que, même lorsque les étudiants n'ont pas effectué tous les exercices, tous ont réussi, quel que soit leur niveau d'études, à utiliser Git et GitHub en suivant la procédure imposée pour soumettre leurs travaux.

3.2. Analyse de l'enquête auprès des étudiants

À la fin de chaque cours, nous avons mené une enquête auprès des étudiants pour évaluer leur ressenti envers Git et GitHub. Le tableau 2 énumère différentes affirmations que les étudiants ont été invités à évaluer en leur donnant une note selon une échelle de Likert à 7 niveaux : « 1 » signifiant « Pas du tout d'accord » et « 7 » signifiant « Tout à fait d'accord ».

Tableau 2 · Affirmations évaluées par les étudiants

#	Affirmations évaluées
1	J'ai trouvé qu'il était facile d'utiliser Git et GitHub pour soumettre mon travail.
2	J'ai trouvé l'utilisation de Git et GitHub adaptée à la soumission de mon travail.
3	J'aimerais que mes travaux futurs soient également soumis en utilisant Git et GitHub.
4	J'ai trouvé que l'utilisation de Git et GitHub ajoutait de la complexité au travail.
5	J'ai trouvé intéressante l'utilisation de Git et GitHub lors du travail.
6	J'ai trouvé motivante l'utilisation de Git et GitHub lors du travail.

La figure 2 représente la répartition des 61 réponses (39 % des étudiants participant aux expérimentations) que nous avons obtenues. Les affirmations 1 et 4 ont été demandés dans le but d'évaluer la difficulté perçue relative à l'utilisation de Git et GitHub. Avec une médiane respectivement de 6 et 2 et un IQR resserré autour des médianes, l'enquête révèle que l'utilisation de Git et GitHub est largement considérée comme facile et n'augmente pas la complexité du travail à effectuer.

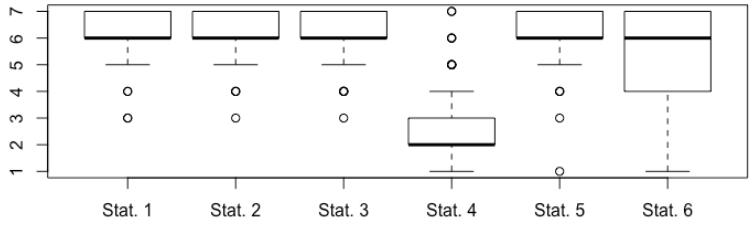


Figure 2 · Distribution des réponses des étudiants

Ensuite, les affirmations 2 et 3 permettent d'évaluer l'utilité perçue de Git et GitHub pour la soumission des travaux. Là encore, avec des médianes de 6 et un IQR resserré autour des médianes, l'enquête indique que les élèves considèrent Git et GitHub comme étant adaptés pour soumettre leurs travaux et sont prêts à utiliser ces outils pour de futurs travaux.

Enfin, les affirmations 5 et 6 visent à évaluer dans quelle mesure les étudiants considèrent Git et GitHub comme des outils intéressants et motivants. Les deux médianes se situent à 6. L'IQR pour la déclaration 6 indique une plus grande distribution des réponses par rapport à la motivation. On peut considérer que les étudiants trouvent très

majoritairement intéressant l'utilisation de Git et GitHub alors qu'une grande majorité la trouve motivante. La mobilisation d'outils utilisés dans le contexte professionnel s'intègre pleinement dans un des six leviers identifiés par Poumay (2014) pour donner du sens aux apprentissages et augmenter l'intérêt des étudiants. Les résultats obtenus sur les affirmations 5 et 6 confortent cette proposition.

En outre, il est possible de vérifier que les réponses données par les étudiants ne dépendent pas de leur niveau d'études. À cette fin, pour chaque affirmation A_i de l'enquête, nous effectuons le test exact de Fisher pour tester la dépendance entre la variable du niveau ISCED et la variable R_i^2 définie comme suit :

$$Soit\ R_i^7\ l'évaluation\ donnée\ par\ un\ étudiant\ à\ l'affirmation\ A_i.$$

$$Pour\ i\ \in \{1,2,3,5,6\}:$$

$$R_i^2 = \begin{cases} 1\ si\ R_i^7 \leq 4 \\ 2\ si\ R_i^7 > 4 \end{cases} \qquad R_4^2 = \begin{cases} 1\ si\ R_4^7 \leq 3 \\ 2\ si\ R_4^7 > 3 \end{cases}$$

Le tableau 3 indique la *p-valeur* calculée avec le test exact de Fisher pour évaluer la dépendance entre la variable du niveau ISCED et la variable R_i^2 pour chaque énoncé.

Tableau 3 · Résultats du test exact de Fisher

Affirmation	1	2	3	4	5	6
p-value	0,44	1	1	0,35	0,37	1

Toutes les *p-valeurs* sont très supérieures à 0,05, indiquant que le niveau d'études est indépendant de la réponse aux différentes questions posées.

3.3. Synthèse de l'évaluation quantitative

L'évaluation quantitative présentée fournit des réponses à la question de recherche (**RQ1**) concernant l'obligation pour les étudiants d'utiliser Git et GitHub à des fins pédagogiques quel que soit leur niveau d'études. En effet, elle révèle, d'une part, l'utilisation concrète de Git et GitHub par tous les étudiants inscrits dans les cinq cours concernés par les expérimentations et, d'autre part, un bon ressenti général des étudiants concernant la facilité, l'utilité, l'intérêt et la motivation à utiliser ces deux outils dans un contexte éducatif, quel que soit leur niveau d'études.

4. Le tableau de bord Git4School

Git4School (https://git4school.firebaseapp.com/) est un tableau de bord, destiné aux enseignants, permettant de visualiser les progrès des élèves.

Sur la page web dédiée à chaque dépôt, GitHub propose un onglet « Insights » qui fournit diverses mesures comme la temporalité, la volumétrie et la répartition des *commits* entre les différents contributeurs du dépôt. Dans cette section, nous verrons que Git4School permet de visualiser des informations davantage sémantiques sur le travail effectué par les étudiants.

4.1. Hypothèses sur le contexte d'utilisation

Les étudiants travaillent individuellement pour développer une partie logicielle dans laquelle les principales fonctionnalités peuvent être mises en œuvre avec des notions de programmation ciblées. Lorsque les étudiants ont produit leur réponse à un exercice, ils effectuent un *commit* avec un message spécifique indiquant que l'exercice a été achevé et un *push* dans un dépôt privé GitHub qui leur est attribué. L'attribution de tels dépôts est facilitée par l'initiative GitHub Classroom (https://classroom.github.com/) qui automatise la création de dépôts Git avec des droits d'accès configurables tout en incluant un code de démarrage. Dans sa conception de la situation d'apprentissage, l'enseignant peut avoir prévu des tâches à accomplir par l'étudiant, par exemple le travail de développement en autonomie ou la confrontation des solutions entre étudiants via une revue du code par les pairs. Une solution fournie par l'enseignant peut également être publiée. Dans ce qui suit, ces événements seront appelés « jalons ».

4.2. Les visualisations proposées par Git4School

La plateforme Git4School est une application web dont le code source (https://github.com/git4school/git4school-visu) est publié sous la licence Apache 2.0. Son utilisation nécessite que l'enseignant dispose d'un compte GitHub. L'API REST de GitHub (https://developer.github.com/v3/) est utilisée pour extraire des informations sur les *commit* effectués par les étudiants dans leurs dépôts.

Les deux seules hypothèses de travail qui conditionnent l'utilisation de Git4School sont d'une part, l'utilisation de GitHub par les étudiants et, d'autre part, le respect de la bonne pratique consistant à effectuer un

commit après chaque achèvement d'un exercice. Git4School est donc très peu invasif à la fois d'un point de vue méthodologique mais aussi en termes d'outils : la plateforme peut être utilisée quel que soit l'environnement de travail (système d'exploitation, IDE) et le langage de programmation.

L'enseignant, une fois connecté à Git4School, fournit un ensemble d'éléments de configuration à travers l'interface dédiée ou via un fichier au format JSON contenant les informations suivantes :

– la liste des dépôts d'étudiants hébergés sur GitHub ;

– une liste d'identifiants pour les différents exercices qui seront recherchés dans les messages de *commits*. La présence d'un identifiant est interprétée comme l'achèvement par l'étudiant de l'exercice correspondant ;

– la date et la durée des séances de travail en face à face ;

– pour chaque exercice, des jalons peuvent être mentionnés : revue par les pairs, publication d'une solution, remédiation, etc.

Ensuite, l'enseignant peut accéder à trois visualisations. Sur chacune d'elles, les couleurs portent des informations sémantiques : le vert indique un exercice réalisé en autonomie avant toute intervention déclenchée par l'enseignant ; l'orange indique un exercice réalisé après un jalon de type « évaluation par les pairs » ; enfin, le rouge indique un exercice réalisé après la publication d'une solution par l'enseignant.

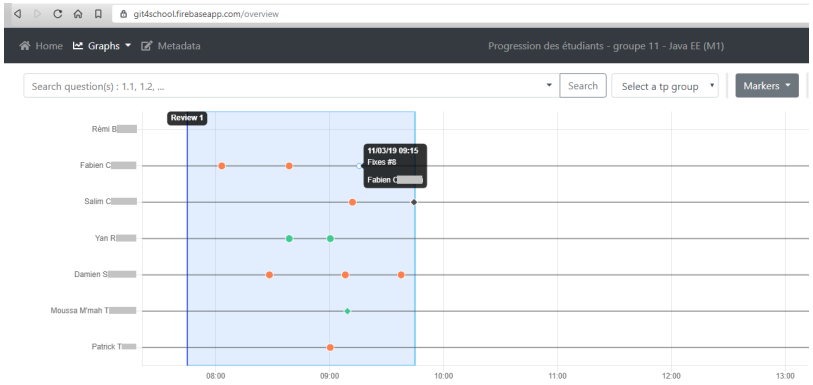


Figure 3 · Visualisation du progrès global dans Git4School

La première visualisation (figure 3) permet de visualiser pour chaque étudiant, en ordonnée, tous ses *commits* dans le temps, en abscisse. Les temps correspondant aux séances en face à face (fond bleu clair) et aux jalons (lignes

verticales) sont également identifiés. Les *commits* sont représentés avec des couleurs différentes selon qu'ils sont effectués avant ou après un jalon.

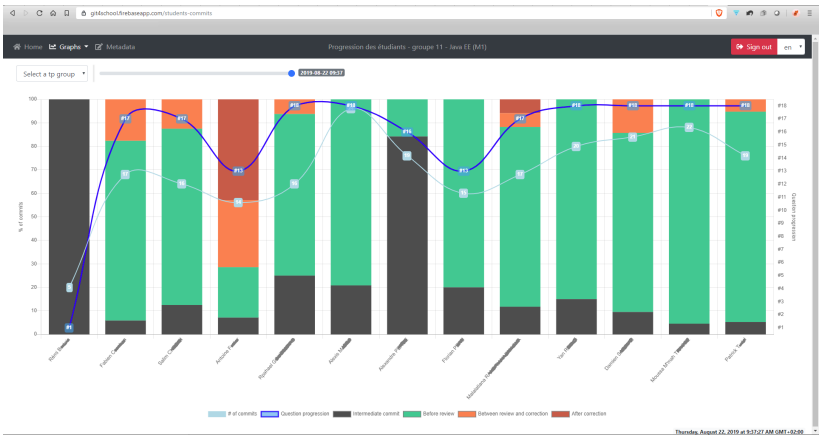


Figure 4 · Visualisation de la quantité de *commits* de chaque type pour chaque élève dans Git4School

La deuxième visualisation (figure 4) indique, sous forme d'un diagramme en bâtons, la quantité de *commits* de chaque étudiant colorée en fonction de leur position temporelle par rapport aux différents types de jalons. Pour chaque étudiant apparaissent aussi l'identifiant du dernier exercice achevé et le nombre de *commits* exécutés.

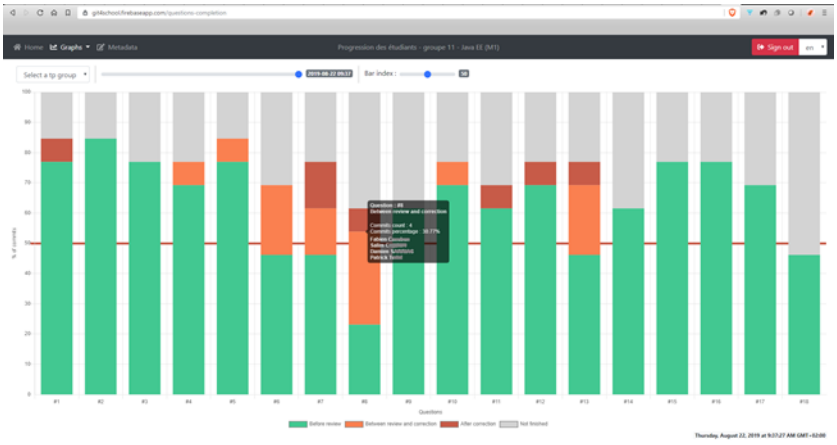


Figure 5 · Visualisation du niveau achèvement des exercices dans Git4School

La troisième visualisation (figure 5) permet de visualiser sous forme d'un diagramme en bâtons, pour chaque exercice, le nombre d'étudiants qui l'ont réalisé et dans quelles circonstances (*commits* lors d'un travail en autonomie, après revue par les pairs et après publication de la solution).

5. Évaluation qualitative des expériences

Nous nous concentrons maintenant sur 2 des 5 expériences mentionnées dans l'évaluation quantitative. Ces études de cas ont eu lieu dans le cadre de cours d'informatique dispensés à différents niveaux de l'enseignement supérieur et suivant différents modèles d'apprentissage. Les deux études visaient à explorer dans quelle mesure les visualisations fournies par Git4School peuvent effectivement aider les enseignants à prendre des décisions au cours des différentes séquences composant leur cours. Nous fournissons ainsi des éléments de réponses en relation avec (**RQ2**).

5.1. Présentation des deux études de cas sélectionnées

1.1.1. Étude de cas 1

La première étude a eu lieu en deuxième année de Master en génie logiciel dans le cadre d'un cours consacré à l'API de persistance de Java (JPA). Un groupe de 36 étudiants était inscrit à ce cours. Ils ont suivi 10 interventions planifiées de 2 heures chacune, au cours desquelles ils ont mis en pratique les connaissances théoriques qu'ils avaient reçues précédemment sur JPA. Ces interventions ont suivi un protocole pédagogique original développé par l'équipe pédagogique au cours des dernières années pour couvrir d'une part l'acquisition de compétences individuelles en programmation et d'autre part l'acquisition de compétences sociales liées à la programmation. Ainsi, les interventions se sont déroulées comme une succession de cycles composés de 3 phases ordonnées comme suit :

– phase de travail en autonomie : chaque élève travaille individuellement sur un même projet spécifié par un ensemble de tests unitaires et d'intégration. Les résultats et les messages fournis lors des tests automatisés donnent aux élèves un *feed-back* qui facilite leur progression vers chaque objectif d'apprentissage ;

– phase de revue par les pairs : les étudiants participent à une séquence de revue par les pairs durant laquelle ils sont invités à présenter et à argumenter en faveur de leur solution à un exercice donné ou à présenter

leur compréhension de l'utilisation de l'API. Cette phase de revue par les pairs est orchestrée à l'aide de la plateforme elaastic (https://elaastic.irit.fr/elaastic-questions) qui met en œuvre certains principes d'apprentissage actif présentés (Silvestre *et al.,* 2015) ;

– phase de restitution : l'enseignant présente les résultats et commente une ou plusieurs solutions ou explications. Il anime les discussions.

À la fin de chaque contribution significative au projet, il a été demandé à chaque étudiant de réaliser un *commit* et un *push* de son travail sur son dépôt GitHub dès que tous les tests passaient avec succès sur la partie concernée du projet. L'indication de soumettre le travail faisait partie de l'énoncé du sujet.

1.1.2. Étude de cas 2

La deuxième étude de cas a eu lieu en deuxième année d'un cours de diplôme universitaire technologique (DUT) informatique visant à acquérir des connaissances et des compétences en matière de développement d'applications web avec le langage de programmation PHP (https://www.php.net). Deux groupes d'environ 25 étudiants ont participé au cours. La plupart d'entre eux étaient novices sur le sujet enseigné. Pendant 5 semaines, l'enseignant a utilisé Git4School. Durant cette période, chaque semaine, les étudiants ont reçu un cours théorique d'une heure sur l'utilisation d'une bibliothèque pour connecter des applications *PHP* avec des bases de données relationnelles ou sur la pertinence du modèle de conception modèle-vue-contrôleur dans une application web. Ensuite, ils ont mis en pratique ce contenu théorique pendant une session d'une heure trente suivant un modèle d'apprentissage assez simple : ils devaient essayer de compléter les exercices jusqu'à ce que le professeur décide d'interrompre le travail individuel pour donner des conseils ou présenter la solution d'un exercice ou de l'ensemble de l'exercice. Les étudiants étaient encouragés à s'entraider, mais aucun protocole d'enseignement rigoureux n'a été établi pour parvenir à cette instruction par les pairs, contrairement à la première étude de cas.

Pour permettre l'utilisation de Git4School, il a été demandé aux étudiants de réaliser un *commit* et *push* de leur travail à la fin de chaque exercice. Les indications de *commit* et *push* du travail ont été explicitement énoncées dans les sujets des différents travaux pratiques. Les exercices n'étant pas accompagnés de tests automatiques, les étudiants décidaient eux-mêmes si l'exercice était terminé. Ils pouvaient éventuellement demander à l'enseignant de le vérifier.

5.2. Résultats et discussions

Nous discutons maintenant des observations qui ont émergées de l'utilisation de Git4School au cours des études de cas décrites précédemment.

Déclenchement des interventions planifiées

La visualisation du taux d'achèvement par exercice a été très utile pour déclencher les interventions prévues dans les situations d'apprentissage (phases de revue par les pairs pendant les séquences relatives à l'étude de cas 1 et phases de restitution dans les deux études de cas). En effet, il est possible de déclencher une intervention à partir d'un seuil configurable de taux d'achèvement des élèves. Comme le montre la figure 5, le seuil personnalisé apparaît sur la visualisation sous la forme d'une ligne horizontale rouge. Par exemple, pour un exercice donné, l'enseignant de la première étude de cas déclenchait la phase de revue par les pairs lorsque plus ou moins 60 % des élèves avait terminé l'exercice.

Obtenir un feed-back sur l'efficacité d'une intervention planifiée

Les trois visualisations proposées par Git4School affichent les *commits* des élèves avec une couleur fonction de la position dans le temps du *commit* par rapport aux interventions déclenchées par l'enseignant. En particulier, la visualisation du taux d'achèvement par exercice donne un aperçu précis de l'efficacité des interventions planifiées en montrant combien de *commits* sont réalisés par les étudiants après chaque type d'intervention. Un enseignant peut alors facilement prendre conscience de l'impact de ses interventions. Plus les interventions planifiées font partie de la conception de l'apprentissage, plus les informations obtenues sont riches. En effet, dans l'étude de cas n° 2 où les évaluations par les pairs n'étaient pas explicitement prévues, il n'a pas été possible de mesurer l'impact des revues par les pairs qui étaient improvisées car elles n'étaient pas prises en compte dans l'outil comme des jalons.

D'autre part, les informations fournies par Git4school étant faciles à partager, Git4School peut aider à l'adoption de techniques, de ressources et/ou d'outils pédagogiques à fort impact à plus grande échelle dans une institution.

Identification des étudiants en difficulté

La visualisation du type de *commits* pour chaque élève permet d'identifier rapidement les élèves ayant le plus de difficultés. Par exemple,

dans la figure 4, on peut voir que la quatrième barre est dominée par l'orange et le rouge, ce qui indique que l'élève correspondant fait rarement les exercices tout seul et, dans la plupart des cas, après la publication de la solution par l'enseignant. Lors de séances en face à face avec de petits groupes d'élèves, l'enseignant peut ne pas avoir besoin d'un tel outil pour identifier les élèves en difficulté. Cependant, Git4School donne rapidement des indicateurs pour identifier les élèves en difficulté avec lesquels l'enseignant peut décider d'interagir sans attendre qu'ils demandent de l'aide. Git4School permet ainsi aux enseignants de mieux répartir le temps passé avec les élèves sur la base d'indicateurs objectifs sur leurs besoins et a donc été utilisé comme aide pour réaliser des interventions non planifiées ciblant les élèves en difficulté.

Dans le contexte de l'enseignement hybride ou à distance, induit par les confinements liés à l'épidémie COVID19 par exemple, Git4School a été très utile pour identifier les élèves en difficulté en compensant en partie l'absence de signaux donnés par les élèves traditionnellement lors des sessions en face à face.

Information sur l'activité des étudiants en dehors des cours

La vue d'ensemble (figure 3) affiche tous les *commits* effectués par les étudiants lors des sessions en face à face (fond bleu clair) et en dehors de la classe (fond blanc). Git4School permet ainsi d'identifier les étudiants qui poursuivent leur travail en dehors des sessions en face à face. Pour l'étude de cas 2, il s'agissait en fait d'un moyen de découvrir que moins de 5 % des étudiants avaient l'habitude de travailler en dehors des sessions en face à face, même lorsqu'il leur était explicitement demandé de réaliser un travail à la maison. L'enseignant a questionné ce manque d'engagement des étudiants. Le paragraphe suivant décrit comment Git4School a permis d'identifier la cause la plus probable : un niveau de difficulté du travail demandé inapproprié au regard du niveau d'expertise des étudiants.

Qualification de la difficulté d'un exercice

La visualisation du taux d'achèvement par exercice (figure 5) peut aider à identifier une situation anormale pour un exercice donné. Dans l'étude de cas 1, le seuil de déclenchement de la prochaine intervention prévue n'était pas atteint quand les élèves ne pouvaient pas effectuer l'exercice par eux-mêmes. Cette observation, pouvant être faite en temps réel par l'enseignant en consultant le tableau de bord, offre alors la possibilité de réaliser une remédiation sur les concepts qui sont visés dans l'exercice.

Si la visualisation du taux d'achèvement sur l'ensemble des exercices est largement dominée par la couleur rouge, cela signifie que les élèves n'ont réalisé les exercices qu'après la publication de la solution fournie par l'enseignant. Cette situation peut remettre en cause la conception pédagogique de l'ensemble des exercices proposés. En effet, dans l'étude de cas 2, comme les premiers exercices ont pris plus de temps que prévu, l'enseignant a anticipé la publication de la solution de ces exercices afin que les étudiants puissent consacrer plus de temps aux exercices suivants jugés plus importants par l'enseignant. Dans cette situation, Git4School fournit des indicateurs objectifs pour aider à identifier les ressources pédagogiques inadaptées pour atteindre les objectifs d'apprentissage initiaux.

Comparaison de niveaux entre groupes d'étudiants

La deuxième étude de cas a impliqué deux groupes d'étudiants travaillant en parallèle sur les mêmes exercices. Assez rapidement, les visualisations sont apparues très différentes dans les deux groupes. Après les deux premières sessions, alors que le premier groupe montrait une distribution compacte des *commits* verts sur les quatre premiers exercices, le second groupe montrait une distribution plus large et inégale des *commits* verts jusqu'au septième exercice. Git4School est apparu comme un outil inattendu pour identifier les différences dans les progrès des étudiants d'un même groupe ainsi que l'hétérogénéité entre deux groupes inscrits au même cours. Ainsi, l'enseignant a pu donner du travail supplémentaire aux élèves les plus avancés afin de maintenir un rythme plus cohérent entre les deux groupes. D'autres choix auraient pu être faits (par exemple, mobiliser les étudiants avancés pour aider les autres), mais l'important ici est de constater que Git4School a permis la mise en œuvre d'interventions correctives imprévues déclenchées sur la base d'informations exactes.

Éthique et gestion des données personnelles

Git4School extrait des dépôts GitHub les données correspondant à la progression des étudiants, que ce soit lors de sessions en face à face ou en dehors des cours. Pour cela, les étudiants ont accepté les conditions d'utilisation de GitHub et de GitHub Classroom. Aucune condition d'utilisation supplémentaire n'est validée par les étudiants pour Git4School. Techniquement, aucune donnée n'est stockée sur le serveur qui héberge Git4School. Les données utilisées par le tableau de bord sont

uniquement stockées localement pendant la période où l'enseignant utilise Git4School.

Limites

Actuellement, Git4School est uniquement compatible avec GitHub. Ce choix de conception n'est pas un problème pour les enseignants et les institutions souhaitant travailler avec cette plateforme populaire, mais il pourrait constituer un obstacle à l'adoption plus large de Git4School. En cas de frein avéré à son adoption, il sera possible d'adapter Git4School afin de le rendre compatible avec d'autres plateformes.

6. Conclusion

Nous avons présenté Git4school, un tableau de bord conçu pour soutenir les interventions des enseignants dans le cadre des cours de génie logiciel en fonction de l'activité des étudiants tracée via l'utilisation de dépôts Git. Une évaluation expérimentale nous a d'abord permis de fournir des réponses à (**RQ1**) en montrant l'adoption de Git et GitHub par les étudiants. Nous avons également montré en réponse à (**RQ2**) que Git4School peut aider les enseignants à prendre des décisions pour déclencher des interventions planifiées ou non planifiées. De plus, nous avons montré que Git4school a été utilisé pour évaluer l'impact des interventions planifiées et pour identifier les faiblesses dans la conception d'une situation d'apprentissage en permettant ainsi d'en cibler des améliorations.

Pendant les séances de la deuxième étude de cas, l'enseignant a montré à plusieurs reprises aux élèves la visualisation du taux d'achèvement par exercice afin de commenter la progression globale du groupe. Il n'a pas été possible de mesurer l'impact de cette utilisation particulière de Git4shool comme outil de sensibilisation des élèves, mais l'attention et la curiosité des élèves à l'égard de l'outil nous ont incités à réfléchir à la conception de visualisations dédiées aux élèves dans le cadre de telles activités. Ainsi, nous souhaiterions à l'avenir fournir aux élèves des visualisations spécifiques pour les aider à autoréguler leur apprentissage.

De nombreuses données ont été collectées au cours des expériences et d'autres viendront rapidement des futures utilisations de Git4school. Il sera bientôt possible d'effectuer une exploration statistique de certaines données afin de découvrir dans quelle mesure certains modèles de travail pourraient être corrélés avec les résultats d'apprentissage. Ce travail sur les données collectées est l'une de nos priorités pour améliorer la

compréhension et l'évaluation des modèles d'apprentissage destinés aux cours de génie logiciel.

RÉFÉRENCES

Bakharia, A., Corrin, L., De Barba, P., Kennedy, G., Gašević, D., Mulder, R., Williams, D., Dawson, S. et Lockyer, L. (2016). A conceptual framework linking learning design with learning analytics. Dans *Proceeding of the International Conference on Learning Analytics & Knowledge (LAK)* (p. 329-338). ACM Press.

Bourque, P. et Fairley, R. E. (2014). SWEBOK: *Guide to the Software Engineering Body of Knowledge*. IEEE Computer Society. http://www.swebok.org/

Choquet, C., Delozanne, E. et Luengo, V. (2007). Numéro spécial : Analyse des traces d'utilisation dans les EIAH. *Sticef, 14.*

Emin, V., Pernin, J.-P. et Guéraud, V. (2011). Scénarisation pédagogique dirigée par les intentions. *Sticef, 18*(1), 195-227.

Gašević, D., Dawson, S. et Siemens, G. (2015). Let's not forget: Learning analytics are about learning. *TechTrends, 59,* 64-71.

Glassy, L. (2006). Using version control to observe student software development processes. *Journal of Computing Sciences in Colleges, 21,* 99-106.

ISCED (2012). *International standard classification of education: ISCED 2011.* UNESCO Institute for Statistics. http://uis.unesco.org/sites/default/files/documents/international-standard-classification-of-education-isced-2011-en.pdf

Jones, C. (2010). Using subversion as an aid in evaluating individuals working on a group coding project. *Journal of Computing Sciences in Colleges, 25,* 18-23.

Kay, J., Maisonneuve, N., Yacef, K. et Zaïane, O. (2006). Mining patterns of events in students' teamwork data. Dans J. Kay, N. Maisonneuve, K. Yacef et O. Zaïane (dir.), *In Educational Data Mining Workshop, held in conjunction with Intelligent Tutoring Systems (ITS)* (p. 45-52).

Liu, Y., Stroulia, E., Wong, K. et German, D. (2004). Using CVS Historical Information to Understand How Students Develop Software. Dans *Proceeding of the International Workshop on Mining Repositories (MSR)* (p. 32-36).

Lockyer, L., Heathcote, E. et Dawson, S. (2013). Informing pedagogical action: Aligning learning analytics with learning design. *American Behavioral Scientist, 57,* 1439-1459.

Mierle, K., Laven, K., Roweis, S. et Wilson, G. (2005). Mining student CVS repositories for performance indicators. *ACM SIGSOFT Software Engineering Notes, 30,* 1-5.

Mittal, M. et Sureka, A. (2014). Process Mining Software Repositories from Student Projects in an Undergraduate Software Engineering Course. Dans *Proceedings of the International Conference on Software Engineering (ICSE)* (p. 344-353). ACM Press.

Poncin, W., Serebrenik, A. et Van den Brand, M. (2011). Mining student capstone projects with FRASR and ProM. Dans *Proceedings of the ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA)* (p. 87-96). ACM Press.

Poumay, M. (2014). Six leviers pour améliorer l'apprentissage des étudiants du supérieur. *Revue internationale de pédagogie du supérieur, 30*(1), 1-19.

Robles, G. et Gonzalez-Barahona, J. (2013, 3). Mining student repositories to gain learning analytics. An experience report. Dans *Proceedings of the IEEE Global Engineering Education Conference (EDUCON)*, IEEE (p. 1249-1254).

Schwendimann, B. A., Rodríguez-Triana, M. J., Vozniuk, A., Prieto, L. P., Boroujeni, M. S., Holzer, A. et Dillenbourg, P. (2017). Perceiving Learning at a Glance: A Systematic Literature Review of Learning Dashboard Research. *IEEE Transactions on Learning Technologies, 10,* 30-41.

Silvestre, F., Vidal, P. et Broisin, J. (2015). Reflexive learning, socio-cognitive conflict and peer-assessment to improve the quality of feedbacks in online tests. Dans G. Conole, T. Klobučar, C. Rensing, J. Konert et E. Lavoué (dir.), *Design for Teaching and Learning in a Networked World*, (p. 339-351). Springer.

Tchounikine, P. (2013). Clarifying design for orchestration: orchestration and orchestrable technology, scripting and conducting. *Computers & Education, 69,* 500-503.

Tissenbaum, M. et Slotta, J. (2019). Supporting classroom orchestration with real-time feedback: A role for teacher dashboards and real-time agents. *International Journal of Computer-Supported Collaborative Learning, 14,* 325-351.

Van Harmelen, M. et Workman, D. (2012). Analytics for learning and teaching. *CETIS Analytics Series, 1,* 1-40.

Wise, A. F. (2014). Designing pedagogical interventions to support student use of learning analytics. Dans *Proceedings of the International conference on learning analytics and knowledge (LAK)* (p. 203-211). ACM Press.

Didactique de l'informatique : une formation nécessaire

▶ **Christophe DECLERCQ** (LIM, INSPE de l'académie de la Réunion, Université de la Réunion)

■ **RÉSUMÉ** • À partir de l'expérience de l'auteur en formation d'enseignants d'informatique, cette rubrique propose un parcours subjectif parmi les travaux pouvant être référencés et utilisés en didactique de l'informatique pour répondre aux questions professionnelles des enseignants d'informatique au lycée en France. En conclusion des pistes pour la recherche et la formation continue des enseignants sont esquissées.

■ **MOTS-CLÉS** • didactique de l'informatique, formation des enseignants.

■ ***ABSTRACT*** • *This paper gives some references used in computer science teachers professional development in France. In conclusion, it made proposals for research and the continuing education of teachers.*

■ ***KEYWORDS*** • *computer science, teacher training.*

1. Introduction

En 1988, Jacques Arsac (1988) situait la didactique de l'informatique comme un problème ouvert et en évoquait une des difficultés fondamentales : « *Il semble à peu près compris que programmer n'est pas résoudre un problème, c'est le faire résoudre par une machine* ». Quelques années et réformes plus tard, Georges-Louis Baron et Éric Bruillard (2001) interrogeaient encore l'existence de cette didactique dans un contexte de va-et-vient entre une informatique « *objet d'enseignement* » et une informatique « *outil* ». En 2015, Janine Rogalski (2015) proposait un parcours historique de quelques décennies de travaux en didactique de l'informatique pouvant pour la plupart être encore utilisés aujourd'hui.

Alors que les conditions d'émergence de l'informatique en tant que discipline scolaire avaient été posées dès 1987 par Baron (1987), « *un corps d'enseignants professionnels, dont la compétence est garantie par la possession d'un grade, des horaires fixés réglementairement, des programmes nationaux d'enseignement, qui définissent le savoir à enseigner, une Inspection générale, et des examens finaux* », ce n'est finalement qu'en 2020 que l'ensemble de ces conditions étaient effectivement réunies en France après la création de la spécialité « Numérique et sciences informatiques » au lycée et d'une spécialité homonyme du concours de recrutement du Capes. La formation préparant à ce concours, le master « Métiers de l'enseignement de l'éducation et de la formation » (MEEF 2d degré), comportant une part importante de didactique disciplinaire, la question de l'existence d'une didactique de l'informatique ne se pose plus : il s'agit maintenant d'une nécessité. On n'oserait supposer qu'une discipline enseignée uniquement dans l'enseignement supérieur n'a pas besoin de didactique, mais a contrario une discipline scolaire en a une nécessité inscrite réglementairement dans le programme des formations d'enseignants.

La problématique, qui se pose aux formateurs d'enseignants, est alors de rassembler dans un corpus cohérent de « didactique de l'informatique », des travaux publiés, à différentes époques et dans différents contextes, sur l'enseignement de l'informatique. Il n'est pas question ici d'une approche académique visant l'exhaustivité, mais plus prosaïquement de rendre accessibles des travaux permettant de répondre à des questions professionnelles de futurs enseignants. Ces questions peuvent porter sur le savoir à enseigner, sur les compétences attendues des élèves, sur les obstacles à anticiper, les situations didactiques à inventer, les raisons de choisir un environnement d'apprentissage, l'évaluation, ainsi que sur les

formes d'enseignement. L'absence, à l'heure actuelle, de cadre de référence unique pour une didactique de l'informatique scolaire (Fluckiger, 2019) incite à emprunter des références aux sciences de l'éducation, aux didactiques voisines, à la psychologie et à l'ergonomie. Cette rubrique propose un parcours de ces références en les situant dans le contexte historique de l'introduction de l'informatique au niveau scolaire en France, et en illustrant l'usage qui peut en être fait pour tenter des réponses aux questions professionnelles des enseignants d'informatique.

1.1. Quelques repères historiques, épistémologiques et curriculaires

L'histoire de l'enseignement de l'informatique en France entre 1970 et 1985 a été relatée par un de ses acteurs, Claude Pair (1987). L'émergence d'une communauté francophone de didactique de l'informatique entre 1970 et 2000 a été décrite en particulier par Baron et Bruillard (2001). La série de colloques Didapro, qui faisait suite aux colloques organisés par l'association francophone de didactique de l'informatique, a remis à l'ordre du jour les travaux sur l'enseignement de l'informatique en particulier depuis les éditions 2018 (Parriaux *et al.,* 2018) et 2020 (Caron *et al.,* 2020).

Les programmes du lycée publiés en 2018 dans le cadre de la réforme du baccalauréat reposent explicitement sur la description épistémologique de l'informatique proposée par Gilles Dowek (2011) qui en a décrit les quatre piliers – information, algorithme, langage et machine – ainsi que les relations entre ces concepts fondamentaux. Des développements épistémologiques plus récents (Delmas-Rigoutsos, 2018, 2020) permettent aussi d'approfondir le caractère fondamental de ces concepts et leurs liens avec les autres disciplines scientifiques.

On remarque que certaines notions méritent d'être reconstruites dans le contexte de l'informatique. C'est le cas en particulier de la notion d'algorithme qui avait, dans le contexte des mathématiques, une définition beaucoup plus restrictive (Modeste *et al.,* 2010) liée au contexte d'usage des algorithmes en mathématiques principalement pour faire du calcul. En informatique, on considère aussi des algorithmes non terminant pour décrire des procédés de contrôle ainsi que des algorithmes probabilistes, donc non déterministes. On s'accorde ainsi pour définir l'algorithme comme la description d'un procédé de traitement ou d'une méthode de calcul ou de résolution de problèmes. On ajoute aussitôt pour éviter toute confusion avec la notion de programme que cette description est destinée au lecteur humain et qu'elle sera jugée suffisamment précise si

on peut l'utiliser pour effectuer la preuve de correction ou l'évaluation de la complexité de cet algorithme.

Une description détaillée des enseignements du lycée figure au bulletin officiel (Bulletin officiel, 2019). Pour la formation des enseignants, on s'intéresse en particulier à l'enseignement de « Sciences numériques et technologie » (SNT) obligatoire en classe de seconde (élèves de 15/16 ans) et à l'enseignement de spécialité « Numérique et sciences informatiques » (NSI) optionnelle en classes de première et terminale (élèves de 16 à 18 ans) du lycée.

Si on se réfère aux spécialistes des curricula qui décrivent l'informatique à la fois comme une science et un ensemble de techniques (Bruillard, 2017), l'enseignement SNT aurait dû être nommé « Science et technologies du numérique », la science (au singulier) du numérique étant l'informatique. C'est d'ailleurs précisément un curriculum d'initiation à l'informatique adossé à une exploration de quelques champs d'usage des technologies du numérique (internet, photographie, objets connectés…). L'enseignement SNT peut ainsi être présenté comme un enseignement d'informatique pour comprendre le monde numérique. Cela permet d'en proposer une didactique fondée sur les pratiques sociales de référence de Jean-Louis Martinand comme le proposait déjà Christian Orange en 1990.

L'enseignement de spécialité « Numérique et sciences informatiques », dont le pluriel intrigue jusqu'à l'Académie des sciences, est un enseignement disciplinaire d'informatique dont la présentation se conforme à l'épistémologie de la discipline. La description du programme inclut principalement une présentation en termes savants des notions et capacités attendues précédée d'un préambule concernant les compétences et les formes d'enseignement, dont le travail par projet. L'enseignant a la responsabilité de mettre en correspondance formes d'enseignement et objectifs en termes de compétences avec les exigences en termes de contenus du programme. On verra quels cadres théoriques peuvent accompagner l'enseignant dans cette voie.

1.2. À la recherche d'un modèle des situations d'enseignement/apprentissage de l'informatique

Selon les auteurs (Fluckiger, 2019), la formalisation d'une situation d'enseignement/apprentissage inclut le savoir, un (ou plusieurs) élève(s), un (ou plusieurs) enseignant(s), des situations d'apprentissage ainsi qu'éventuellement des instruments utilisés pour l'apprentissage. Selon les théories, l'accent est porté principalement sur l'élève, sur le savoir, sur les

situations problèmes dont la résolution peut mener à la construction du savoir ou des compétences visées, ou sur l'activité de l'enseignant. Le déroulement des activités d'apprentissage peut s'effectuer de manière synchrone ou asynchrone, les différents acteurs se trouvant en présence ou à distance.

La complexité des situations d'enseignement/apprentissage de l'informatique, outre le fait déjà évoqué qu'il ne s'agit pas de résoudre des problèmes, mais de décrire comment une machine pourrait les résoudre, tient aux multiples rôles que peut avoir l'informatique dans ces situations. L'informatique est bien sûr d'abord objet d'enseignement. L'outil informatique peut aussi être un moyen de communication permettant de désynchroniser, mettre à distance, ou faire collaborer les élèves dans leurs apprentissages. L'enseignement de notions informatiques nécessite aussi des pratiques qui requièrent l'usage d'instruments permettant d'exécuter ou interpréter des programmes, des requêtes ou des commandes et d'en visualiser les effets.

Il y a ainsi de multiples sources de quiproquos si l'élève confond l'un et l'autre, ce qui demande de la part de l'enseignant la plus grande rigueur dans la rédaction des consignes. On distinguera aisément : « calcule… », « écrit un programme qui calcule… », « envoie un programme qui calcule… », « écrit un programme qui envoie le résultat du calcul… », « écrit un programme qui écrit un programme qui… ».

On a pu observer (Declercq et Tort, 2018) dès l'initiation à l'informatique, des confusions entre les postures d'utilisateur de télécommande, de programmeur de télécommande et de programmeur. Ces confusions perdurent jusqu'au lycée, quand une même consigne peut être interprétée par l'élève comme une action à décrire dans un langage de programmation ou une interaction avec un environnement de programmation. Cette confusion peut se produire par exemple pour l'affichage de résultats intermédiaires en cours d'exécution que l'on peut confier à l'environnement ou décrire par des instructions du langage.

Face à cette complexité d'une situation générale d'enseignement/apprentissage de l'informatique, et en l'absence de théorie générale, les formateurs d'enseignants ont besoin de s'appuyer sur des cadres théoriques variés. Les parties suivantes vont s'intéresser successivement aux compétences (partie 2), à l'activité de l'élève (partie 3), aux situations et aux savoirs (partie 4), puis aux instruments indispensables aux apprentissages en informatique (partie 5).

2. Les compétences de la « pensée informatique »

L'expression « *computational thinking* » introduite par Jeannette Wing (2006) fait référence à des compétences et des habiletés humaines, pouvant être développées à l'occasion d'activités de programmation, et transférables à d'autres situations de type « résolution de problème ». Il ne s'agit pas de penser comme une machine, mais de décrire les compétences cognitives en jeu pour résoudre, entre autres, un problème informatique, ou plutôt pour le faire résoudre par une machine. Il s'agit d'une activité cognitive de haut niveau et donc bien d'une activité humaine. L'énumération des compétences en jeu fait débat. La traduction française aussi fait débat. « *Thinking* » aurait pu être traduit par « réflexion ». Le terme « computationnel » n'existant pas, l'adjectif « calculatoire » est le plus proche, mais semble réducteur.

Dans le contexte de l'apprentissage de la programmation, on utilise les compétences proposées par Selby et Woollard (2014) pour décrire et qualifier les activités proposées à des élèves. On a reformulé les compétences sous forme verbale de la manière suivante :

– **évaluer** : capacité à attribuer mentalement une valeur (résultat, type...) à un programme donné ;

– **anticiper** : capacité à se mettre dans la posture du programmeur qui doit décrire dans un algorithme l'enchaînement séquentiel/répétitif/conditionnel des instructions, avant même le début de son exécution. On préfère ce terme au terme original de « algorithmic thinking » qui encourt le risque de confusion avec « computational thinking » ;

– **décomposer** : capacité à transformer un problème complexe en un ensemble de problèmes plus simples équivalents au problème initial ;

– **généraliser** : capacité à inférer un problème général à partir d'une instance de ce problème et à repérer dans un problème particulier la répétition de traitements ou de données suivant un même schéma ;

– **abstraire** : capacité à « faire abstraction » des informations non pertinentes et à créer des solutions où la manière dont un problème est résolu peut être « abstraite » à l'aide d'une interface pertinente.

Évaluer un programme (lui donner une valeur) peut se faire de différentes manières dans différents domaines. Le plus simple est bien sûr d'évaluer le résultat que donne le programme à l'exécution. Mais il est aussi utile de savoir évaluer le type du résultat sans chercher nécessairement à connaître sa valeur : « *It is interpreting code as data and data as code. It is type checking as the*

generalization of dimensional analysis[1] » (Wing, 2006). De manière générale, évaluer un programme consiste donc à regarder ce programme comme une donnée et à en calculer une valeur par une méthode particulière. Ce changement de plan du programmeur consiste à regarder son programme tel qu'il est, et non tel qu'il aurait voulu qu'il soit. La compétence « évaluer » est fondamentale pour mettre au point un programme.

Anticiper est une compétence fondamentale pour la conception d'algorithmes. C'est aussi un des principaux obstacles didactiques rencontrés par les programmeurs débutants : « *Une propriété difficile à intégrer [...] est le caractère différé d'une exécution du programme* » (Rogalski et Samurçay, 1986). C'est la part créative du travail du programmeur d'imaginer par quel chemin de calculs intermédiaires on peut passer, pour aller des informations disponibles aux informations que l'on souhaite calculer. Anticiper les étapes successives du traitement et les différents cas à envisager, c'est tout l'art de programmer.

Décomposer permet d'envisager le traitement de problèmes arbitrairement complexes par réduction à des problèmes plus simples ou déjà connus. La décomposition par cas peut se pratiquer à tous les niveaux de la conception d'un programme : du niveau le plus fin pour séparer quelques instructions au niveau le plus global pour séparer des traitements demandant chacun plusieurs centaines ou milliers de lignes de code. La décomposition séquentielle peut être mise en œuvre avec des variables intermédiaires et des fonctions ou procédures définies pour résoudre les problèmes élémentaires. Le résultat de la démarche de conception peut ainsi être montré explicitement.

Généraliser est une compétence de haut niveau qui permet au programmeur de résoudre des problèmes plus généraux et ensuite de réutiliser pour des instances particulières des parties de programme déjà écrites. La notion de paramètre, utilisée pour instancier des valeurs ou des fonctions particulières est le mécanisme permettant de mettre en œuvre la généralisation. Dans le contexte de la programmation objet, un autre mécanisme permet de prévoir des solutions générales à toute une classe de problèmes, c'est le mécanisme de l'héritage entre classes.

---

[1] Traduction : « C'est interpréter un programme comme une donnée et une donnée comme un programme. On considère la vérification de types comme une généralisation de l'analyse dimensionnelle ».

Abstraire avec les données consiste à encapsuler un certain nombre d'informations en utilisant une structure de données composée, qui peut éventuellement masquer le détail du codage des informations. Abstraire avec les fonctions permet de masquer la méthode de calcul utilisée. La combinaison des deux méthodes peut aboutir à la programmation orientée objet où données et méthodes sont encapsulées à l'intérieur d'une classe.

Ces compétences sont citées dans le préambule commun aux programmes de première et de terminale. Leur maîtrise par l'enseignant est nécessaire pour qualifier les activités proposées aux élèves et aussi pour en mesurer la difficulté. Les activités d'évaluation – « quel est le résultat de... » – sont les plus simples. Les tâches de programmation mobilisent le plus souvent les compétences « anticiper » et « décomposer ». Les compétences « abstraire » et « généraliser » sont plus exigeantes au niveau cognitif car de plus haut niveau.

La maîtrise de ce cadre permet à l'enseignant, à la fois, de s'engager dans une démarche d'évaluation par compétences et de varier les activités proposées aux élèves sur la base de cette typologie.

3. Les apports de la psychologie de la programmation

La psychologie de la programmation (Hoc, 1982) est un domaine de recherche qui a émergé dans le contexte de l'ergonomie et de la didactique professionnelle et s'est consacré à l'étude des situations de travail des programmeurs. Ce n'étaient donc pas, à l'origine, des travaux liés à l'enseignement, mais rapidement la question de l'enseignement de méthodes de programmation s'est posée (Rogalski, 1988), (Rogalski *et al.,* 1988). Les difficultés cognitives rencontrées par les programmeurs débutants ont été signalées (Rogalski et Samurçay, 1986).

En particulier, les difficultés de conceptualisation par les élèves de la notion de variable ont été mises en évidence par Renan Samurçay (1985) :

– « *la construction de la signification et des opérations sur les variables : la déclaration, l'affectation des valeurs, l'entrée et la sortie des données sur l'écran ;*

– *le contrôle des valeurs particulières qui doivent être prises par les variables lors de l'exécution du programme : ce contrôle intervient dans la planification des instructions par les structures que sont la répétition, le choix et la séquentialité* ».

La typologie introduite par Samurçay distingue les « variables données », qui sont des données explicites du problème, et ne varient pas, des variables nécessitées par la résolution informatique, avec parmi elles les

« variables compteurs », les « variables accumulateurs » et les « variables intermédiaires ». Cette distinction par rôle est utile pour l'enseignant pour graduer les difficultés lors de l'introduction des variables des situations les plus simples (variables données puis compteurs) aux situations les plus générales (variables accumulateurs et variables intermédiaires).

Le rôle précurseur de notions déjà vues par les élèves en mathématiques dans l'assimilation ou l'accommodation de nouvelles notions en informatique a été étudié par Janine Rogalski (Rogalski, 1987) concernant les notions de variable et de fonction : « *La variable a comme précurseur possible la variable mathématique.* » Les précurseurs ont un double rôle : producteur et réducteur. « *Le caractère statique de la variable mathématique peut constituer un obstacle à la représentation de la modification possible de la valeur d'une variable lors de l'exécution d'un programme. [...] L'existence de la représentation symbolique = de l'égalité des variables numériques joue un rôle producteur dans les acquisitions initiales des tests en programmation... mais peut rendre difficile le changement de point de vue qui consiste non pas à comparer... mais à tester si une certaine propriété est vraie.* »

Une synthèse des travaux de ce domaine, qui a été particulièrement actif pendant les années de l'option informatique des lycées, a été publiée récemment (Lagrange et Rogalski, 2017) et reste d'actualité pour anticiper nombre de difficultés dans les apprentissages de l'informatique au lycée.

On retiendra en particulier :

– La difficulté pour les élèves de conceptualiser le calcul booléen puisque « *les élèves conçoivent les expressions booléennes comme des "conditions" telles qu'elles s'expriment dans le langage usuel, plutôt que comme un calcul sur des valeurs logiques.* »

– Le risque de confusion entre égalité et affectation. Leurs notations sont proches et varient selon les langages. En particulier le symbole = peut dénoter l'affectation dans un langage et l'égalité dans un autre.

– La question de « *comment articuler l'élaboration de situations didactiques centrées sur l'acquisition de concepts déterminés (variables, boucles, conditionnelles) avec une activité de programmation partant de problèmes consistants, plus ouverts* » reste vive, car les programmes de NSI incitent à une pédagogie par projets tout en étant principalement définis par une liste de notions à étudier.

Dans leur conclusion, Rogalski et Lagrange soulignent : « *Il apparaît que l'enjeu central est, pour les élèves que nous avons observés, de comprendre la construction d'un programme ou d'un algorithme comme l'organisation d'un*

traitement sur un dispositif ; ils doivent percevoir ce dispositif comme un ensemble de variables, et concevoir ces variables comme des objets "calculables" et le traitement comme l'évolution de leurs valeurs. »

Dans le contexte actuel de la programmation en Python au lycée, il convient aussi le moment venu de passer d'un modèle de dispositif basé uniquement sur les variables en mémoire, à un modèle incluant l'environnement et la mémoire. Cette étape est nécessaire pour bien comprendre les données mutables et les mécanismes de passage de paramètres. Cette dernière question ne peut être éludée, car elle est la source de questions professionnelles pour l'enseignant.

4. Des cadres de référence empruntés aux didactiques des mathématiques

Dans l'étude des situations d'apprentissage proposées aux élèves, c'est sans surprise la théorie des situations didactiques de Guy Brousseau (1998), empruntée à la didactique des mathématiques, qui semble la plus utilisée pour analyser les activités. Les notions de milieu, de variable didactique, de situation, d'obstacle sont indispensables pour analyser *a priori* les activités proposées aux élèves. En particulier, la maîtrise des variables didactiques d'une situation permet à l'enseignant de différentier les activités proposées selon les capacités des élèves. Une communication récente (Meyer et Modeste, 2020), qui propose une situation fondamentale pour l'étude de la complexité des algorithmes, s'y réfère explicitement et cible des apprentissages au programme de la spécialité NSI. La recherche a permis de confirmer l'hypothèse des auteurs concernant l'intérêt des jeux à deux joueurs pour aborder dans le cadre scolaire la notion de complexité.

De nombreux autres travaux restent à mener pour proposer des situations couvrant l'ensemble des contenus au programme de la spécialité NSI.

C'est aussi aux didactiques des mathématiques que l'on doit l'emprunt de la théorie anthropologique du didactique de Yves Chevallard (1991). Cette approche a été explorée par Fabienne Viallet et Patrice Venturini dans une étude sur la transposition de la notion de boucle (2010). La distinction entre savoir savant et savoir enseigné est particulièrement féconde pour l'enseignement de l'informatique dans un contexte où la plus grande partie des savoirs à enseigner ne l'ont été auparavant que dans l'enseignement supérieur, ce qui explique que le travail de transposition didactique n'ait pas encore été accompli. Ceci est particulièrement flagrant dans le domaine de l'informatique théorique où des enseignants de terminale s'interrogent sur

la nécessité d'introduire les machines de Turing pour aborder l'indécidabilité du problème de l'arrêt. Une proposition de preuve de l'indécidabilité du problème de l'arrêt transposée au lycée a été proposée par Journault *et al.* (2020). C'est aussi prégnant dans le domaine des réseaux informatiques où les filières d'enseignement supérieur délivrent une formation professionnelle alors que les objectifs des programmes de lycées se limitent à une compréhension des technologies employées et de leurs enjeux. L'accompagnement d'enseignants débutants nécessite leur prise de conscience que cette transposition est indispensable pour construire, dans chaque domaine, un savoir à enseigner au lycée à partir du savoir savant qu'ils ont rencontré à l'Université.

Cette proximité entre les mathématiques et l'informatique marquée par des emprunts de théories didactiques ne doit cependant pas occulter l'émancipation de l'informatique par rapport aux mathématiques en tant que discipline. Ce processus d'émancipation achevé depuis plus de trente ans à l'Université est encore en cours dans l'enseignement scolaire, sachant qu'une partie des enseignements d'informatique au collège et en classe de seconde est confiée aux professeurs de mathématiques ou de technologie. De très nombreux enseignants de mathématiques, formés en formation continue via le diplôme interuniversitaire « Enseigner l'informatique au lycée » (Marquet et Declercq, 2019), enseignent d'ailleurs les deux disciplines et doivent à ce titre jongler avec les injonctions contradictoires concernant la manière d'organiser les apprentissages pour la conception d'algorithmes. Une anecdote significative de cette difficulté mérite d'être signalée : une directive concernant l'écriture de programmes à l'épreuve de mathématiques du baccalauréat proscrivant l'usage du *print* et encourageant le *return*, a été interprétée par des enseignants comme s'appliquant aussi à l'informatique. Ceci n'a évidemment pas de sens, car la programmation de systèmes interactifs ou réactifs ne peut s'envisager sans l'utilisation d'entrées-sorties.

Le partage de cadres théoriques permet alors d'expliquer pourquoi telle situation est plus propice dans tel milieu et pourquoi le savoir enseigné concernant les algorithmes peut différer en mathématiques et en informatique.

5. L'analyse instrumentale et les environnements interactifs pour l'apprentissage humain

Les apports de la théorie instrumentale de Pierre Rabardel (1995) ont été résumés dans (Nijimbere, 2013). Conçu initialement pour comprendre

l'usage d'instruments dans le travail humain, ces recherches sont particulièrement fécondes aussi pour comprendre l'usage d'instruments dans le cadre d'une activité d'apprentissage.

La compréhension de phénomène d'appropriation d'un artefact pour en faire un instrument (genèse instrumentale), processus dirigé à la fois vers le sujet qui modifie ses schèmes d'action (instrumentation) et vers l'artefact qui est alors adapté à l'action du sujet (instrumentalisation), est utile pour comprendre comment un élève, ou un enseignant, s'approprie par exemple un environnement de programmation. Cela permet de relativiser la perception que l'utilisateur a d'un instrument, perception d'autant plus positive que sa genèse instrumentale est avancée et de mieux comprendre pourquoi le meilleur instrument pour le professeur n'est pas forcément le meilleur pour l'élève.

Pour objectiver les raisons du choix d'un instrument à proposer à l'élève, la notion de « retour instrumental » est fondamentale. Quel retour l'instrument procure-t-il à l'élève lors de son usage ? Ce retour peut-il le guider dans l'apprentissage ?

Appliqué au choix d'un environnement de programmation, les questions de retour instrumental à se poser sont les suivantes :

– Quel retour lors de l'édition : colorisation syntaxique, indentation automatique, parenthésage automatique ? Ce retour aide-t-il à l'écriture d'un programme correct syntaxiquement ?

– Quel retour avant, ou lors de l'exécution, concernant la correction du programme : problèmes liés au typage, effets de bord non désirés, débordements dans les tableaux, instruction conditionnelle non exhaustive, choix des inégalités, comparaisons et calculs entre flottants, mauvais nommage des variables ? Ces retours sont-ils utilisables par l'élève ?

– Quel retour à l'exécution : distinction *print/return*, exécution pas à pas au niveau instruction ou au niveau expression, visualisation de la mémoire, de l'environnement, de la pile ?

Une étude attentive permet de choisir un environnement en fonction de ce qu'il donne à voir à l'élève des programmes et de leur exécution. Ce choix peut bien sur différer de la seconde à la terminale. En particulier, le modèle de machine que l'environnement laisse entrevoir peut évoluer de la « machine qui associe des valeurs à des variables » à celle qui « montre l'environnement associant des références aux noms, et la mémoire associant des valeurs aux références ».

Pour les premiers apprentissages en programmation, on a proposé un environnement dédié dont les retours instrumentaux garantissent des programmes corrects par construction (Declercq et Nény, 2020).

L'analyse des retours instrumentaux gagnerait aussi à être menée à propos de l'enseignement des systèmes d'exploitation : en effet selon les choix retenus (un simulateur de console ou la console du système) la nature des retours, différés ou non, visuels dans l'explorateur de fichier ou textuel, est très différente.

L'analyse instrumentale est depuis longtemps utilisée dans la communauté qui s'intéresse aux environnements interactifs pour l'apprentissage humain. Ses résultats ne sont pas spécifiques à l'apprentissage de l'informatique, mais lui sont applicables en prenant soin de distinguer l'informatique moyen et/ou objet d'enseignement. En particulier les travaux sur les jeux sérieux, les exerciseurs, les environnements personnels d'apprentissage ou ceux permettant la collaboration sont applicables même s'ils ne sont pas spécifiques et ne sont pas à ce titre répertoriés en didactique de l'informatique.

Des travaux récents sur la collaboration en apprentissage de l'informatique utilisant soit des systèmes issus de l'industrie du logiciel (Raclet *et al.,* 2020) soit des systèmes construits à cette fin (Broisin *et al.,* 2015) relatent des expérimentations dans l'enseignement supérieur. Leur transposition au lycée serait à étudier, vu l'importance accordée aux projets parmi les formes d'enseignement de l'informatique au lycée, et vu la difficulté de gérer la conduite et la documentation d'un projet sans la médiation d'instruments adaptés.

L'analyse instrumentale, même si cela est parfois oublié, s'applique aussi aux instruments tangibles en bois ou en papier, développés dans le cadre du courant dit de l'informatique débranchée (*computer science unplugged*) ou informatique sans ordinateur. Une revue de littérature (Drot-Delange, 2013) a été publiée qui recense de nombreux travaux et les bénéfices de cette approche pour les apprentissages en informatique. Destinée à l'origine plutôt à l'enseignement primaire, l'informatique débranchée a conquis l'enseignement secondaire et est explicitement citée comme modalité d'enseignement dans les programmes de première et de terminale.

Les situations issues de concours telles celles du concours Castor (Drot-Delange et Tort, 2018) constituent aussi une forme originale de situations

instrumentées où la résolution de défis peut être utilisée pour construire des apprentissages en informatique.

Dans la diversité des situations évoquées, l'analyse instrumentale est ainsi un cadre indispensable pour l'enseignant qui veut fonder ses choix d'environnements à proposer aux élèves, non sur ses préférences personnelles, mais sur une analyse rationnelle des processus cognitifs en jeu.

6. En guise de conclusion, des pistes pour la recherche et la formation

La présente recension de travaux en didactique de l'informatique ne prétend pas à l'exhaustivité : elle a été construite à partir des références réellement utilisées en formation d'enseignants d'informatique en partie dans le cadre du DIU et surtout dans celui du parcours informatique du master MEEF de l'INSPE de l'académie de Nantes. Cette analyse emprunte aussi, sans pouvoir encore les citer, à des travaux en cours d'étudiants préparant leur mémoire de master et peut également fournir des références et des idées de problématiques aux promotions suivantes. C'est en particulier à cause de son usage en enseignement que la littérature francophone a été presque exclusivement utilisée.

Pour constituer un corpus plus complet, il conviendrait maintenant de comparer au niveau national les références utilisées en formation d'enseignants d'informatique, et aussi au niveau de la communauté francophone, les hautes écoles pédagogiques en Suisse ou en Belgique ayant déjà une longue expérience de formation d'enseignants du secondaire dans un contexte un peu différent.

On peut aussi formuler l'hypothèse qu'une demande de formation en didactique de l'informatique est en cours d'émergence parmi les enseignants d'informatique au lycée dont la formation par le DIU a été, faute de temps, très majoritairement disciplinaire. C'est à ce public que cette rubrique est dédiée en souhaitant qu'elle suscitera des envies de formation voire de participation à des groupes de recherche-action comme le proposent de nombreux IREM (Institut de recherche en Enseignement des Mathématiques) dont certains deviennent des IREMI (Institut de recherche en Enseignement des Mathématiques et de l'Informatique) incluant l'informatique dans leurs préoccupations.

Gageons aussi que la communauté de recherche francophone en didactique de l'informatique va se renforcer fortement, maintenant que son objet d'étude est devenu une discipline scolaire à part entière. Des

signes en attestent déjà avec la constitution d'un groupe de travail sur l'enseignement de l'informatique de la maternelle à l'Université (Broisin *et al.,* 2021) au sein de l'ATIEF et la parution de ce numéro spécial.

La prochaine conférence francophone de didactique de l'informatique « Didapro 9 » qui va avoir lieu en mai 2022 à l'INSPE de l'académie de Nantes, présentera de nombreuses contributions sur ces thèmes.

RÉFÉRENCES

Arsac, J. (1988). La didactique de l'informatique : Un problème ouvert ? Dans *Actes du Colloque francophone sur la didactique de l'informatique* (p. 9-18).

Baron, G.-L. (1987). *La constitution de l'informatique comme discipline scolaire, le cas des lycées* [thèse de doctorat, Université Paris Cité].

Baron, G.-L. et Bruillard, E. (2001). Une didactique de l'informatique ? *Revue française de pédagogie, 135*, 163-172.

Broisin, J., Declercq, C., Fluckiger, C., Parmentier, Y., Peter, Y et Secq, Y. (2021). Dans *Actes de l'atelier Apimu 2021 @ EIAH : Apprendre la pensée informatique de la maternelle à l'Université*. HAL.

Broisin, J., Venant, R. et Vidal, P. (2015). Lab4CE: a remote laboratory for computer education. *International Journal of Artificial Intelligence in Education, 25*(4), 154-180.

Brousseau, G. (1998). *Théorie des situations didactiques*. La Pensée sauvage.

Bruillard, E. (2017). Enseignement de l'informatique entre science et usages créatifs : Quelle scolarisation ? Dans H. J. Nguyen et E. Vandeput (.dir), *L'informatique et le numérique dans la classe. Qui, quoi, comment ?* Presses universitaires de Namur.

Bulletin officiel (2019). Spécial nº 1 du 22 janvier.

Caron, P.-A., Fluckiger, C., Marquet, P., Peter, Y. et Secq, Y. (2020). *L'informatique, objets d'enseignements enjeux épistémologiques, didactiques et de formation* [communication]. Colloque Didapro 8 – DidaSTIC L'informatique, objets d'enseignements – enjeux épistémologiques, didactiques et de formation, Lille, France.

Chevallard, Y. (1991). *La transposition didactique, du savoir savant au savoir enseigné*. La Pensée sauvage.

Declercq, C et Nény, F. (2020). *Block2Py, un éditeur de blocs pour l'apprentissage du langage Python* [communication]. Colloque Didapro 8 – DidaSTIC L'informatique, objets d'enseignements – enjeux épistémologiques, didactiques et de formation, Lille, France.

Declercq, C. et Tort, F. (2018). Organiser l'apprentissage de la programmation au cycle 3 avec des activités guidées et/ou créatives. Dans *Actes des Ateliers RJC EIAH 2018*. https://wikis.univ-lille.fr/computational-teaching/_media/wiki/actions/2018/rjceiah/christophe-declercq-francoise-tort.pdf

Delmas-Rigoutsos, Y. (2018). Proposition de structuration historique des concepts de la pensée informatique fondamentale [communication]. Dans G. Parriaux, J. P. Pellet, G. L. Baron, É. Bruillard et V. Komis (dir.), *De 0 à 1 ou l'heure de l'informatique à l'école. Actes du Colloque Didapro 7 - DidaSTIC* (p. 31-60). Peter Lang.

Delmas-Rigoutsos, Y. (2020). *Variables, grandeurs et types* [communication]. Colloque Didapro 8 – DidaSTIC L'informatique, objets d'enseignements – enjeux épistémologiques, didactiques et de formation, Lille, France.

Dowek, G. (2011). Les quatre concepts de l'informatique. Dans G.-L. Baron, E. Bruillard, et V. Komis (dir.), *Sciences et technologies de l'information et de la communication en milieu éducatif : Analyse de pratiques et enjeux didactiques.* (p. 21-29). New Technologies Editions.

Drot-Delange, B. (2013). Enseigner l'informatique débranchée : Analyse didactique d'activités. *AREF*, 1-13.

Drot-Delange, B. et Tort, F. (2018). *Concours Castor, ressource pédagogique pour l'enseignement de l'informatique ? Étude exploratoire auprès d'enseignants* [communication]. Didapro 7 – DidaSTIC. De 0 à 1 ou l'heure de l'informatique à l'école, Lausanne, Suisse.

Fluckiger. C. (2019). *Une approche didactique de l'informatique scolaire.* Presses universitaires de Rennes.

Hoc, J. M. (1982). L'étude psychologique de l'activité de programmation : Une revue de la question. *Technique et Science Informatique, 38*(2-3), 213-221.

Journault, M., Lafourcade, P., More, M. et Poulain, R. (2020). *Une preuve pour le lycée de l'indécidabilité du problème de l'arrêt* [communication]. Colloque Didapro 8 – DidaSTIC L'informatique, objets d'enseignements – enjeux épistémologiques, didactiques et de formation, Lille, France.

Lagrange, J.-B. et Rogalski, J. (2017). Savoirs, concepts et situations dans les premiers apprentissages en programmation et en algorithmique. *Annales de didactiques et de sciences cognitives, 22,* 119-158.

Marquet, P. et Declercq, C. (2019). DIU Enseigner l'informatique au lycée. *Bulletin de la société informatique de France, 1024, 47-56*

Meyer, A. et Modeste, S. (2020, février). *Analyse didactique d'un jeu de recherche : Vers une situation fondamentale pour la complexité d'algorithmes et de problèmes* [communication]. Colloque Didapro 8 – DidaSTIC L'informatique, objets d'enseignements – enjeux épistémologiques, didactiques et de formation, Lille, France.

Modeste, S., Gravier, S. et Ouvrier-Buffet, C. (2010). Algorithmique et apprentissage de la preuve. *Repères IREM, 79,* 51-72.

Nijimbere, C. (2013). Approche instrumentale et didactiques : Apports de Pierre Rabardel. *Adjectif.net.* https://adjectif.net/spip/spip.php?article202

Orange, C. (1990). Didactique de l'informatique et pratiques sociales de référence. *Bulletin de l'EPI, 60.*

Pair, C. (1987). Informatique et enseignement = hier, aujourd'hui et demain. *Bulletin de l'EPI, 47*, 85-97.

Parriaux, G., Pellet, J.-P., Baron, G.-L., Bruillard, É et Komis, V. (2018). De 0 à 1 ou l'heure de l'informatique à l'école. Dans *Actes du colloque Didapro 7 – DidaSTIC.* Peter Lang

Rabardel, P. (1995). *Les hommes et les technologies. Approche cognitive des instruments contemporains.* Armand Colin.

Raclet, J.-B., Silvestre, F. et Pons, M. (2020). Mise en œuvre d'approches pédagogiques fondées sur des pratiques de l'industrie du logiciel pour l'apprentissage de la programmation. Dans *Actes du Colloque Didapro 8 – DidaSTIC : L'informatique, objets d'enseignements* (p. 1-12).

Rogalski, J. (1987). Acquisition de savoirs et de savoir-faire en informatique. *Cahier de didactique des mathématiques, 43.* IREM Paris VII.

Rogalski, J. (1988). Enseignement de méthodes de programmation dans l'initiation à l'informatique. Dans G.-L. Baron, J. Baudé et P. Cornu (dir.), *Colloque francophone sur la didactique de l'informatique* (p. 61-72). Association EPI.

Rogalski, J. (2015). Psychologie de la programmation, didactique de l'informatique. Déjà une histoire... *Informatique en éducation : perspectives curriculaires et didactiques.*

Rogalski, J. et Samurçay, R. (1986). Les problèmes cognitifs rencontrés par des élèves de l'enseignement secondaire dans l'apprentissage de l'informatique. *European Journal of Psychology of Education, 1*(2), 97-110.

Rogalski, J., Samurçay, R. et Hoc, J. M. (1988). L'apprentissage des méthodes de programmation comme méthodes de résolution de problème. *Le Travail Humain, 51*(4), 309-320.

Samurçay, R. (1985). Signification et fonctionnement du concept de variable informatique chez des élèves débutants. *Educational Studies in Mathematics, 16*, 143–161.

Selby, C. et Woollard, J. (2014). Computational thinking: The developing definition. Dans *Actes du SIGCSE.* https://people.cs.vt.edu/˜kafura/CS6604/Papers/CT-Developing-Definition.pdf

Viallet, F et Venturini, P. (2010). *Didactique comparée et enseignement de l'informatique* [communication]. Congrès International Actualité de la Recherche en Éducation et en Formation (AREF), Borgeaux, France.

Wing, J. M. (2006). Computational Thinking. *Communications of the ACM, 49*(3), 33-35.

Comités

Rédactrice en chef

Élise LAVOUÉ • LIRIS, Université Jean Moulin Lyon 3

Comité de rédaction

Monique BARON • LIP6, Sorbonne Université
Laetitia BOULC'H • EDA, Université Paris Cité
Éric BRUILLARD • EDA, Université Paris Cité
Pierre-André CARON • CIREL, Université Lille
Christophe DESPRÈS • LIUM, Le Mans Université
Michel DESMARAIS • Polytechnique Montréal
Béatrice DROT-DELANGE • ACTé, Université Clermont Auvergne
Nour EL MAWAS • CIREL, Université Lille
Sébastien GEORGE • LIUM, Le Mans Université, Laval
Monique GRANDBASTIEN • LORIA, Université de Lorraine
Richard HOTTE • LICEF, Télé-université, Université du Québec, Montréal, Canada
Pierre JACOBONI • LIUM, Le Mans Université
Élise LAVOUÉ • LIRIS, Université Jean Moulin Lyon 3
Vanda LUENGO • LIP6, Sorbonne Université
Agathe MERCERON • Université de Berlin, Allemagne
Gaëlle MOLINARI • TECFA, Unidistance, Genève, Suisse
Chrysta PÉLISSIER • Praxiling, Université Montpellier 3
Jean-Luc RINAUDO • Civiic, Université de Rouen

Comité de parrainage scientifique

Nicolas BALACHEFF • Laboratoire d'Informatique de Grenoble, CNRS
Stefano CERRI • LIRMM, Université de Montpellier 2
Christian DEPOVER • Université de Mons, Belgique
Alain DERYCKE • TRIGONE, Université de Lille
Pierre DILLENBOURG • École polytechnique fédérale de Lausanne, Suisse
Claude FRASSON • Université de Montréal, Canada
Catherine GARBAY • CNRS, laboratoire d'Informatique de Grenoble
Gilles GAUTHIER • UQAM, Canada
Guy GOUARDÈRES • ISIHM, Université de Pau

Comité de lecture

Viviane GUÉRAUD • LIG, Université Grenoble Alpes
Nicolas GUICHON • ICAR, Université Lumière Lyon 2
Nathalie GUIN • LIRIS, Université Lyon 1
France HENRI • LICEF, Télé-université, Université du Québec, Montréal, Canada
Pierre JARRAUD • FOAD, Sorbonne Université
Michelle JOAB • LIRMM, Université Montpellier 2
Céline JOIRON • MIS, Université de Picardie Jules Verne, Amiens
Mehdi KHANEBOUBI • STEF, ENS Paris-Saclay
Vassilis KOMIS • Université de Patras, Grèce
Thérèse LAFERRIÈRE • TACT, Université Laval, Canada
Françoise LE CALVEZ • LIP 6, Sorbonne Université
Marie LEFÈVRE • LIRIS, Université Lyon 1
Dominique LENNE • Heudiasyc, Université de Technologie de Compiègne Pascal LEROUX • CREN, Le Mans Université
Paul LIBBRECHT•Leibniz Institute for Research and Information in Education, Allemagne
Cabral LIMA • Université Fédéral de Rio de Janeiro, Brésil
Domitile LOURDEAUX • Heudiasyc, Université de Technologie deCompiègne
Pascal MARQUET • LISEC, Université de Strasbourg
Jean-Charles MARTY • LIRIS, Université de Savoie
André MAYERS • Université de Sherbrooke, Canada
Christine MICHEL • TECHNÉ, Université de Poitiers
Roger NKAMBOU • GDAC, Université du Québec à Montréal, Canada
Thierry NODENOT • LIUPPA, Université de Pau et des Pays de l'Adour, Bayonne
Daniel PERAYA • TECFA, Université de Genève, Suisse
Yvan PETER • CRIStAL, Université de Lille
Julia PILET • LDAR, INSPÉ, Université Paris Est Créteil Val-de-Marne
Dominique PY • LIUM, Le Mans Université
Christophe REFFAY • ELLIAD, INSPÉ, Université de Franche-Comté
Éric SANCHEZ • TECFA, Université de Genève, Suisse
Karim SEHABA • LIRIS, Université Lyon 2
Nicolas SZILAS • TECFA, Université de Genève, Suisse
Pierre TCHOUNIKINE • LIG, Université Grenoble Alpes
André TRICOT • EPSYLON, Université Paul-Valéry Montpellier 3
Nicolas VAN LABEKE • Learning Sciences Research Institute, University of Nottingham, UK

ISBN 978-2-901384-05-2
DOI: 10.23709/sticef.28.3