L'informatique, objets d'enseignement et d'apprentissage. Quelles nouvelles perspectives pour la recherche ? Actes du colloque DIDAPRO 9

Christophe Declercq, Jérôme Lehuen, Pascal Leroux, Valérie Renault, Arnauld Sejourne

▶ **To cite this version:**

Christophe Declercq, Jérôme Lehuen, Pascal Leroux, Valérie Renault, Arnauld Sejourne. L'informatique, objets d'enseignement et d'apprentissage. Quelles nouvelles perspectives pour la recherche ? Actes du colloque DIDAPRO 9. L'informatique, objets d'enseignement et d'apprentissage. Quelles nouvelles perspectives pour la recherche ?, Le Mans, France. , 137 p., 2022. hal-03697888

HAL Id: hal-03697888
https://hal.science/hal-03697888

Submitted on 17 Jun 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

DIDAPRO 9

L’informatique, objets d’enseignement et d'apprentissage. Quelles nouvelles perspectives pour la recherche ?

18-20 mai 2022 Le Mans, France

https://www.didapro.org/9/

Actes du colloque

édités par Christophe DECLERCQ, Jérôme LEHUEN, Pascal LEROUX, Valérie RENAULT, Arnauld SEJOURNE

Le colloque DIDAPRO 9 est organisé par le CREN, l'INSPE de l'académie de Nantes et Le Mans Université en partenariat avec l'Académie de Nantes et l'INSPE de l'académie de la Réunion

Préface

La neuvième édition du colloque Didapro confirme et amplifie la tendance amorcée avec les deux éditions précédentes du colloque, d'un retour de l'informatique en tant qu'objet d'enseignement - apprentissage. Les travaux présentés témoignent d'un renouveau des questions posées par la ré-introduction de l'informatique en tant que discipline scolaire dans les curricula de l'enseignement primaire et secondaire dans la communauté francophone.

Alors que les décennies précédentes avaient été marquées par un mouvement de balancier entre une informatique outil et une informatique objet d'enseignements, une synthèse semble s'opérer - dans le champ de la didactique de l'informatique - avec des travaux complémentaires dans le domaine du numérique et de l'informatique, qui se nourissent mutuellement.

Cette édition du colloque a reçu ving-deux propositions de communications, parmi lesquelles onze ont été acceptées pour présentation et publication dans les actes. De plus, six travaux en cours ont pu être présentés sous forme de posters.

Les communications ont été réparties en trois sessions. On peut en déduire quelques-unes des questions vives du colloque ainsi qu'à l'envers les thématiques moins abordées.

La session consacrée à l'éducation au numérique et aux ressources éducatives témoigne d'une imbrication croissante des recherches sur le numérique et sur l'informatique. Les auteurs des communications ont en effet proposé d'"ouvrir la boite noire", que ce soit pour l'éducation à la sécurité, l'éducation aux données ou la problématique de l'enseignement de l'accessibilité web. Notons en particulier l'émergence de ce dernier thème dans la cadre de Didapro.

La session consacrée à l'analyse de l'activité et à la didactique de l'informatique a rassemblé des communications traitant d'analyse de l'activité des élèves ou des enseignants dans le cadre d'enseignement/apprentissage de la science informatique à l'école primaire ou au lycée. Les communications reposent chacune sur un ou des cadres théoriques explicités mais distincts d'une communication à l'autre, ce qui nous semble témoigner à la fois de la richesse des cadres théoriques mobilisables en didactique de l'informatique, et d'une absence à l'heure actuelle de cadre théorique unificateur pour une "didactique de l'informatique".

La session consacrée aux jeux et à l'apprentissage de l'informatique confirme l'importance de ce thème - déjà présent lors de la précédente édition du colloque - et peut-être sa généralisation. Les deux communications de 2020 portaient sur l'enseignement primaire, alors que lors de l'édition présente, deux des quatre communications portent sur l'enseignement au lycée.

Le programme scientifique du collogue a été complété par les contributions invitées de Pierre Eric Mounier-Kuhn, sur "La construction sociale d'une discipline académique : l'informatique", de Françoise Tort, sur "Évaluation automatique de savoirs et savoir-faire numériques. Questions et perspectives ouvertes par le dispositif PIX" et de Engin Bumbacher, sur "Nouveaux types d'évaluation pour mieux avancer la Pensée Computationnelle dans les écoles : Une approche prometteuse ou une impasse ?".

Par rapport aux thèmes annoncés du colloque, on note que les travaux se concentrent sur les pratiques d'enseignement/apprentissage ainsi que sur les ressources, dispositifs et environnements associés. On peut observer l'absence de communication sur les politiques publiques et l'analyse curriculaire.

L'étude des logiques des acteurs, bien que n'ayant pas fait l'objet de communication, a cependant été au coeur d'une des tables-rondes du colloque : "Les enseignants.e.s d'informatique au lycée : un nouveau groupe professionnel en quête d'identité et une communauté d'apprentissage en développement".

De même, la formation des enseignants, n'a été abordée que lors de la table-ronde: "Quelles perspectives pour la recherche en didactique de l'informatique, quels usages en formation d'enseignants ?". Nous espérons que ces débats pourront inspirer et préfigurer des travaux de recherche futurs.

Pour conclure, nous souhaitons souligner un indicateur précieux pour l'avenir et le développement de notre - relativement petite - communauté de recherche francophone en didactique de l'informatique. Six communications ont été présentées ou co-signées par des doctorantes ou doctorants en didactique de l'informatique, ce qui annonce un potentiel renouvellement et une ouverture de cette communauté.

Enfin nous remercions le laboratoire et les tutelles responsables de l'organisation : le CREN, Le Mans Université et l'INSPE de l'académie de Nantes ainsi que nos partenaires : l'académie de Nantes et l'INSPE de l'académie de la Réunion.

Le comité d'organisation.

Comité scientifique

- Jean-Pierre ARCHAMBAULT, Enseignement public et informatique
- Georges-Louis BARON, Université Paris Descartes
- Monique BARON, Sorbonne Université
- Jacques BEZIAT, Université de Caen Normandie
- Lætitia BOULC'H, Université Paris Descartes
- Pascale BRANDT-POMARES, Aix-Marseille Université
- Julien BROISIN, Université de Toulouse
- Eric BRUILLARD, ENS Paris-Saclay
- Pierre-André CARON, Université de Lille
- Thierry DANQUIGNY, Université de Lille
- Christophe DECLERCQ, INSPÉ de l'académie de la Réunion
- Emmanuel DESMONTILS, Université de Nantes
- Christian DEPOVER, Université de Mons
- Béatrice DROT-DELANGE, Université Clermont Auvergne
- Cédric FLUCKIGER, Université de Lille
- Monique GRANDBASTIEN, Université Poincaré, Nancy
- Julie HENRY, Université de Namur et Université de Liège
- Magali HERSANT, INSPÉ Nantes
- Vassilis KOMIS, Université de Patras
- Pascal LEROUX, Le Mans Université
- Philippe MARQUET, Université de Lille
- Mathieu MURATET, INSHEA
- Sandra NOGRY, Université de Cergy-Pontoise
- Gabriel PARRIAUX, HEP Vaud
- Jean-Philippe PELLET, HEP Vaud
- Yvan PETER, Université de Lille
- Christophe REFFAY, Université de Franche-Comté
- Valérie RENAULT, Le Mans Université
- Margarida ROMERO, INSPÉ de Nice
- Yann SECQ, Université de Lille
- Arnauld SEJOURNE, INSPÉ de l'académie de Nantes
- Françoise TORT, ENS Paris-Saclay
- Emmanuelle VOULGRE, Université Descartes Sorbonne Paris Cité
- Amel YESSAD, Sorbonne Université

Comité d'organisation

- Christophe DECLERCQ, INSPÉ de l'académie de la Réunion
- Jérôme LEHUEN, Le Mans Université
- Pascal LEROUX, Le Mans Université
- Valérie RENAULT, Le Mans Université
- Arnauld SEJOURNE, INSPÉ de l'académie de Nantes

Table des matières

Education au numérique et ressources éducatives

- Julie Henry, Alyson Hernalesteen, Anne-Sophie Collard : « Stop Hackers », un jeu de rôle pour éduquer les enfants à la cybersécurité de manière critique. 6
- Béatrice Drot-Delange, Françoise Tort : Les datasprints, un dispositif d'éducation aux données ? Une étude exploratoire via le cas de « Traces de Soldats ». 19
- Caroline Ladage, Mehdi Khaneboubi, Cécile Redondo : Les rapports aux ressources éducatives des enseignants en informatique dans les IUT en France. 29
- Béatrice Drot-Delange : Enseignement de l'accessibilité web et ressources pédagogiques. Le cas de la formation « Métiers du multimédia et internet » en IUT. 42

Analyse de l'activité et didactique de l'informatique

- Marie Valorge : Possibilités d'agir d'un environnement d'apprentissage de la programmation informatique à l'école primaire. 49
- Julien Bugmann, Morgane Chevalier, Jean-Philippe Pellet, Gabriel Parriaux : Difficultés d'enseignement et d'apprentissage de la science informatique au primaire. 61
- Olivier Goletti, Florian De Pierpont, Kim Mens : Création d'exemples résolus avec objectifs étiquetés pour l'apprentissage de la programmation avec Python. 77

Jeux et apprentissage de l'informatique

- Matthieu Branthome : Conception et évaluation du jeu sérieux Pyrates : agencement du milieu didactique pour la transition Scratch-Python. 86
- Antoine Meyer, Simon Modeste : Situation didactique autour d'un jeu de recherche : expérimentation en classes de NSI. 100
- Marielle Léonard, Yann Secq, Yvan Peter, Cédric Fluckiger : Pensée informatique : approche didactique de l'identification de motifs. 113
- Hajar Saddoug, Aryan Rahimian, Bertrand Marne, Mathieu Muratet, Karim Sehaba, Sébastien Jolivet : Analyse de l'adaptabilité de jeux pour l'apprentissage de la pensée informatique ou de la programmation. . 126

“Stop Hackers”, un jeu de rôle pour éduquer les enfants à la cybersécurité de manière critique*

Henry, Julie[0000−0003−4354−9848], Hernalesteen, Alyson, and Collard, Anne-Sophie[0000−0003−3457−6908]

Namur Digital Institute (NADI), Université de Namur, Belgique
prénom.nom@unamur.be

Abstract. Le jeu de rôles “Stop Hackers” a pour objectif d'éduquer les enfants de 10-14 ans à la cybersécurité. Basé sur un modèle théorique d'éducation au numérique critique et citoyenne, ce dispositif questionne l'intention humaine derrière une cyberattaque, les conditions de son succès, ainsi que la valeur des données visées. D'une part, il s'agit de réfléchir à la forme d'une éducation à la cybersécurité qui entraînerait un changement de représentation de ce concept informatique et un questionnement critique chez les jeunes. D'autre part, il s'agit de déterminer si les enseignants se sentent capables de mettre en œuvre une telle éducation critique, au-delà des aspects techniques. “Stop Hackers” a été conçu à partir d'une démarche de type recherche orientée par la conception. Des données ont été collectées lors des phases d'expérimentation, par observations et via des interviews, auprès de quatre enseignants et 107 élèves. Les résultats obtenus montrent : 1) que les enseignants sont en mesure de mettre en place le dispositif au sein de leur classe, 2) que le dispositif permet d'atteindre, partiellement, les objectifs d'apprentissage, et 3) que la contextualisation et l'expérience de jeu peuvent servir de base à un questionnement critique.

Keywords: Éducation au numérique · Éducation aux médias · Enseignement de l'informatique · Activité débranchée

Si les jeunes utilisent, de plus en plus tôt, Internet et ses outils de communication (médias sociaux, jeux en ligne, courrier électronique et messagerie instantanée), il est interpellant de constater qu'un grand nombre d'entre eux n'ont jamais entendu parler du phishing et de ses dérivés[1].

Prendre conscience des risques auxquels ils sont exposés sur Internet est une étape importante pour que les jeunes naviguent en toute sécurité et comprennent les différentes cyber-menaces auxquels ils pourraient être confrontés. Cela peut se faire à travers des campagnes de sensibilisation et des initiatives ponctuelles, mais la solution la plus efficace pour toucher un maximum de jeunes reste d'organiser une éducation à la cybersécurité à l'école.

* Supported by organization x.

[1] 30% des jeunes Belges, selon le rapport de Febelfin (2021). There is plenty of ‘phish’ in the sea., consulté en ligne le 23/12/2021

Cette recherche vise à développer un dispositif éducatif pour les 10-14 ans croisant l'enseignement de l'informatique et l'éducation aux médias (EAM) et questionnant l'intention humaine derrière une cyberattaque, les conditions de son succès, ainsi que la valeur des données visées. D'une part, il s'agit de réfléchir à la forme d'une éducation à la cybersécurité qui entraînerait un changement de représentation de ce concept informatique et un questionnement critique chez les jeunes. D'autre part, il s'agit de déterminer si les enseignants se sentent capables de mettre en œuvre une telle éducation critique, au-delà des aspects techniques.

Les principaux apports de cet article sont :

- la validation d'un modèle théorique d'éducation citoyenne critique à la technologie qui intègre les approches de l'enseignement de l'informatique, ici dans le domaine de la cybersécurité, et de l'EAM [X,X] ;
- la conception d'un dispositif éducatif questionnant l'intention humaine à l'origine d'une cyberattaque, les conditions de son succès, ainsi que la valeur des données visées, et pouvant être mis en place par un enseignant ;
- la formulation de pistes pour d'autres concepteurs et chercheurs afin de créer des activités éducatives critiques en cybersécurité pour les enfants, s'appuyant sur l'analyse des données préliminaires recueillies.

1 L'éducation à la cybersécurité : intérêt et défis

Accélérée par la crise du covid, la numérisation de la vie quotidienne se poursuit et augmente les cyber-risques. Une éducation à la cybersécurité apparait, dès lors, essentielle pour faire prendre conscience aux jeunes internautes de leur vulnérabilité lorsqu'ils utilisent des médias sociaux (en opposition à leur sentiment d'invincibilité [19]), mais aussi leur apprendre à prévenir certaines cyberattaques [1]. En outre, il apparait que cette éducation devrait se faire dès le plus jeune âge. En effet, les enfants ne sachant pas encore lire ni écrire, couraient un risque beaucoup plus élevé que leurs aînés face aux menaces et aux dangers du cyberespace parce qu'ils ne possèdent pas les connaissances pour se protéger [22]. L'intérêt pour une telle éducation est partagé par les enseignants, conscients que leurs élèves doivent être préparés à identifier les risques qu'ils encourent lorsqu'ils utilisent les technologies numériques [4].

Si l'aspect essentiel d'une éducation à la cybersécurité n'est donc plus à discuter, sa mise en œuvre rencontre plusieurs défis : le manque de compétences des enseignants et l'insuffisance de ressources pour les soutenir, mais aussi un manque de diversité dans le choix des thèmes à aborder.

Si, de façon générale, les enseignants ont la volonté d'enseigner la cybersécurité, ils déclarent ne pas avoir les connaissances suffisantes pour le faire. Le manque de ressources et de soutien pour les y aider ne favorise pas la mise en place d'une telle éducation [4,16,19] : très peu de ressources s'adressent aux jeunes enfants [22], les approches proposées manquent de rigueur dans l'évaluation de leurs effets [18] et n'ont pas toujours le succès escompté [12,13,10,9].

Concernant les thèmes composant cette éducation, cyberintimidation, protection des données personnelles et fiabilité des informations sur les médias sociaux sont ceux qui sont les plus couramment évoqués par les enseignants [4]. Ces thèmes sont également parmi les plus étudiés au niveau de la recherche [18], avec la vie privée et les données personnelles, les problèmes des cyberprédateurs utilisant les médias sociaux et les jeux, les sextos [11,15,25,8], ainsi que le phishing (reconnaissance d'un e-mail frauduleux) [9]. Les compétences plus techniques, telles que le maintien de la sécurité des comptes (gestion des mots de passe et craquage de ceux-ci), des logiciels et des appareils (mises à jour), d'un réseau, ou encore la connaissance des techniques possibles de cyberattaque, sont peu présentes [7,15] ou réalisées par des experts [14].

2 Un modèle théorique d'éducation au numérique critique et citoyenne

Compte tenu de l'âge des élèves, Corradini et Nardelli trouvent excessif de parler d'éducation à la cybersécurité à l'école primaire et secondaire. Pour eux, il s'agit avant tout de conscience numérique (*digital awareness*) : apprendre aux élèves à comprendre le concept de risque numérique et souligner l'importance du comportement en ligne [4]. Cela passe, selon eux par l'apprentissage d'une utilisation responsable des technologies numériques.

Cette vision s'inscrit dans dans une approche de l'éducation au numérique qui vise les compétences permettant d'utiliser, de comprendre et d'évaluer les technologies [23,24]. Avec le développement de technologies complexes, dont le fonctionnement devient de plus en plus opaque pour le grand public, la compréhension des aspects techniques est un enjeu fondamental. Cette dimension technique est prise en charge, au niveau de l'éducation, par l'enseignement de l'informatique qui soutient l'acquisition de compétences suffisamment avancées pour comprendre les concepts fondamentaux et identifier les logiques de ces technologies [5].

Cependant, enseigner les aspects techniques ne suffit pas pour amener les usagers à questionner la place des technologies dans la société et la manière dont elles s'inscrivent dans des pratiques qu'elles façonnent en même temps. Faire appel aux cadres de l'EAM permet d'envisager une éducation aux technologies qui soit critique par rapport aux problématiques éthiques et sociétales, et réflexive par rapport aux pratiques de chacun. C'est une approche qui envisage les risques mais aussi les opportunités que représente l'évolution du numérique dans la société. Concernant la cybersécurité, il ne s'agit donc pas seulement d'alerter sur les problèmes liés aux attaques mais aussi d'envisager de continuer à utiliser les technologiques numériques tout en étant conscient des risques. L'EAM vise en effet à développer les compétences nécessaires pour être critique, créatif, autonome et socialisé dans l'environnement médiatique contemporain, en prenant en compte la dimension technique des médias numériques mais également leurs dimensions informationnelle (sémiotique) et sociale (pragmatique) [6].

Pour répondre aux exigences de la formation de citoyens critiques et autonomes en cybersécurité, le modèle théorique sur lequel nous nous appuyons croise les perspectives de l'enseignement de l'informatique et de l'EAM [X]. Il s'inscrit dans la proposition d'une éducation critique à la technologie formulée par Saariketo [20,21], qui vise à prendre en compte de manière réflexive le rôle de la technologie dans la société et la vie quotidienne. Il s'agit donc non seulement de soutenir une compréhension des aspects de la cybersécurité au niveau de son fonctionnement informatique réel et actuel, mais aussi d'en développer une compréhension qui s'inscrit dans un contexte d'usage des technologies numériques, mettant en jeu des intérêts et des intentions humaines, prenant place au sein de relations sociales et s'inscrivant dans la compréhension que nous pouvons avoir d'une situation. En d'autres termes, cela consiste à ouvrir la "boîte noire" pour saisir les failles et les stratégies mises au point techniquement, mais aussi à la "déplier" dans un contexte médiatique et social pour mettre au jour la manière dont est envisagée la sécurité numérique en tant que construction sociale. Cette approche intégrée permet de développer une réflexivité (1) sur notre compréhension technique de la cybersécurité et des risques dans lesquels nos usages évoluent, et (2) sur le rôle du contexte médiatique et social dans lequel elle prend place, au sein duquel s'entremêlent les risques et les opportunités liés à nos usages des technologies numériques.

3 Méthodologie de recherche

S'appuyant sur ce modèle théorique de l'éducation au numérique critique et citoyenne, la recherche mise en place vise, d'une part, à concevoir une activité d'éducation à la cybersécurité pour les 10-14 ans et, d'autre part, à répondre aux deux questions suivantes :

- L'activité "Stop Hackers" est-elle réalisable en classe par un enseignant ? Est-ce que le dispositif fonctionne tant que jeu éducatif ? Comment les enseignants se l'approprient-ils ?
- L'activité permet-elle de remplir ses objectifs éducatifs ? Quels sont les apprentissages développés par les élèves, au niveau technique et au niveau critique ?

Après avoir expliqué le contexte et la démarche de recherche, nous présentons le dispositif éducatif "Stop Hackers" puis la méthode de collecte et d'analyse des données.

3.1 Contexte

Le dispositif "Stop Hackers" accompagne la mise en œuvre d'une réforme de l'enseignement en Belgique francophone[2] et vise à outiller les enseignants pour la mise en place d'un nouveau référentiel de compétences incluant une éducation

[2] Le Pacte pour un Enseignement d'Excellence, consulté le 11 janvier 2022.

au numérique, le référentiel “Formation Manuelle Technique, Technologique et Numérique” (FMTTN)[3]. En ce qui concerne l'éducation à la cybersécurité, le référentiel FMTTN annonce deux compétences à développer chez les enfants de 11 à 13 ans (élèves de 6e primaire et de début de secondaire) : “adopter un comportement responsable face à des situations de cyberattaque” et “réagir de manière responsable face aux risques de cyberattaque”. Les savoirs associés à ces compétences consistent principalement en du vocabulaire, utilisé de façon adéquate et en contexte (hameçonnage, virus, routeur, pirates, attaque en ligne, etc.). Les savoir-faire énoncés sont “reconnaitre des situations de cyberattaque” et “proposer et mettre en place des pistes d'actions pour faire face à des situations de cyberattaque”.

3.2 Une recherche orientée par la conception

Le dispositif “Stop Hackers” a été développé selon une approche de type recherche orientée par la conception (RoC) [2,3]. Cette approche consiste à mener un processus itératif qui articule des phases de conception, d'expérimentation dans différents contextes (dans le cas de cette étude, dans des écoles mais également dans d'autres contextes d'éducation non formelle), et d'analyse des données collectées durant ces expérimentations en vue d'améliorer le dispositif. Dans le cadre de cette recherche, la phase de conception repose sur une collaboration entre les chercheurs, les praticiens (enseignants) et des experts en cybersécurité (l'expertise en EAM est déjà représentée au sein de l'équipe de recherche) [3]. Elle est structurée en quatre étapes, les trois dernières étant itératives et suivant un processus de “recherche par les erreurs” [2] :

- les chercheurs **s'approprient** le sujet à partir de ressources validées par les experts ;
- ils **prennent du recul** par rapport à la connaissance experte et sélectionnent les concepts pertinents à aborder suivant le modèle théorique suivi (Section 2) et les compétences numériques visées (Section 3.1) ; il s'agit plus précisément de développer une connaissance des attaques possibles et de pouvoir identifier le rôle du contexte social dans la compréhension des risques liés à ces attaques ;
- les chercheurs **définissent la séquence** de l'activité qu'ils soumettent ensuite aux experts pour une première validation ;
- ils **créent du matériel pédagogique**.

L'activité est **séquencée** en trois temps : la contextualisation, l'expérience de jeu et le débriefing. La **contextualisation** part des représentations des enfants et de leurs pratiques médiatiques. Ensuite, à la manière d'un jeu de rôles, les enfants vivent une **expérience** tangible dans une situation définie. Enfin, le **débriefing** consiste en un retour sur le contexte, sur les représentations initiales et sur l'expérience de jeu. Il s'agit aussi d'élargir le questionnement à des problématiques sociétales contemporaines.

[3] Le référentiel FMTTN, version provisoire consultée le 11 janvier 2022

Les données collectées au fur et à mesure des expérimentations menées avec les enseignants permettent de consolider l'activité ainsi que le matériel éducatif **créé**, y compris la documentation pour les aider à mettre en œuvre l'activité au niveau pédagogique, didactique ou organisationnel. Cette approche a déjà été éprouvée dans une autre recherche visant le développement de dispositifs d'éducation critique au numérique [X] : un dispositif éduquant à l'intelligence artificielle [X,X].

3.3 Le dispositif "Stop Hackers"

"Stop Hackers" est un jeu de rôle débranché, inspiré du jeu "Les loups-garous de Thiercelieux". Ce dispositif vise à faire prendre conscience aux élèves que :

- Il existe différentes menaces sur Internet.
- Ces menaces sont mises en oeuvre par des personnes (et non des machines) qui ont des intentions.
- Les données volées ou endommagées ont une certaine valeur.
- Pour que ces menaces réussissent et deviennent des cyberattaques, il faut des conditions et un contexte social qui les rendent plausibles. Dans le cas contraire, elles échoueront.

Le temps de **contextualisation** de l'activité se réfère aux pratiques médiatiques de partage d'informations, en particulier sur les réseaux sociaux et via les applications de communication. La consigne donnée pour le temps du jeu est de partager le plus d'informations avec ses amis, afin que les cyberattaques s'inscrivent dans des pratiques et un contexte social familiers aux enfants.

Lors du jeu (**expérience**), les élèves sont répartis en groupes de 6 à 10 joueurs. Trois rôles sont disponibles : les amis, les pirates et les routeurs. Chaque groupe doit être composé des trois rôles, dans des proportions définies : de 3 à 5 amis, de 2 à 3 pirates et de 1 à 2 routeurs. Au sein d'un même groupe, amis et pirates s'affrontent.

Les amis ont pour mission de s'envoyer des messages et du contenu (photos et vidéos à visionner, contenu à télécharger). Chaque ami a une personnalité (six disponibles : Alice, Bob, Carole, David, Greg et Fanny) et un lot de cartes "messages" à sa disposition(cfr Figure 1) et un plateau pour les disposer (cfr Figure 2). Chaque ami est représenté par une couleur que l'on retrouve également en bordure de ses cartes : par exemple, Bob est "rouge" et Alice, "violet". Pour chaque message reçu, un ami doit décider s'il souhaite en consulter le contenu (à savoir le déposer sur son plateau, d'après la pré-visualisation sur la carte) ou le jeter. Dans les deux cas, il lui est demandé de justifier par écrit sa décision. Chaque contenu consulté (c'est-à-dire visionné ou téléchargé) permet de gagner un point au groupe d'amis à condition qu'il soit sécurisé, sinon il en fait perdre. Les amis ne découvrent qu'en fin de partie les contenus qui n'étaient pas sécurisés.

Les routeurs sont neutres. Chaque groupe en possède au moins un. Ils sont les maîtres du temps dans le jeu : ils gèrent les tours d'échanges de messages entre amis. À chaque tour, le routeur collecte les messages des amis de son groupe,

Fig. 1. Cartes “messages” de Bob

Fig. 2. Plateau de jeu d’Alice

les trie et les transfère aux bons destinataires. Il peut y ajouter des publicités, représentées par des cartes “messages” disposant d’une bordure grise. Chaque plateau dispose d’un emplacement pour déposer ces messages sans expéditeur précis.

Les pirates doivent lancer des menaces. Les trois profils de pirate représentent trois menaces différentes, avec des missions précises. Eve est une écouteuse externe (*eavesdropper*) et a pour mission de relever, dans les échanges entre amis, un maximum d’informations personnelles. Elle ne dispose pas de cartes “messages”. Oscar peut usurper l’identité d’un ami et substituer les messages de ce dernier par les siens qui sont infectés de virus. Pour se faire, ses cartes messages sont les copies conformes des messages des amis, couleurs comprises. Enfin, Peggy pratique le *phishing* : à travers ses messages qui ressemblent à de la publicité (de couleur grise), elle tente de manipuler les amis pour récupérer leurs données personnelles. Chaque menace réussie, et donc devenue cyberattaque, permet de faire gagner un point aux pirates. Pour éviter que les amis identifient les pirates, ceux-ci sont décrits comme des techniciens aidant les routeurs dans leurs tâches.

Dépendant du temps prévu par l’enseignant, le jeu peut prendre fin après sept tours, à savoir que chaque ami a, au minimum, envoyé sept messages. Le comptage des points va permettre de déterminer les vainqueurs dans chaque groupe, mais surtout de lancer le **débriefing**. Il est intéressant de ne pas dévoiler directement l’existence des pirates mais de prendre le temps, d’abord, de recueillir le ressenti des élèves : ont-ils ou non perçu les menaces ? font-ils des liens avec des situations déjà vécues ? Ensuite, des discussions pourront être menées concernant les intentions des pirates, la valeur des données piratées, la “forme” des menaces, les conditions de succès de celles-ci, etc.

La documentation créée à l’intention des enseignants comprend les règles du jeu, mais aussi un dossier pédagogique dans lequel se retrouvent, entre autres,

une clé de répartition des rôles, des connaissances théoriques en cybersécurité, des commentaires d'enseignants ayant testé le dispositif et des suggestions de thèmes à aborder durant le débriefing.

3.4 Phases d'expérimentation et d'analyse

Dans un premier cycle de l'itération, la phase d'expérimentation a été réalisée avec des experts en informatique et des enseignants, en vue de récolter leur évaluation sur la faisabilité du jeu avec le public-cible et la pertinence des concepts en lien avec les objectifs visés. Six chercheurs en informatique ont d'abord été impliqués ; ensuite, une vingtaine d'enseignants issus de l'enseignement primaire, secondaire et spécialisé. Durant ces tests, des observations ont été consignées par l'équipe de recherche et des discussions informelles ont été menées avec les participants.

Dans les cycles suivants, la phase d'expérimentation a pris place dans des écoles, en contexte réel. De février à juin 2021, le dispositif "Stop Hackers" a été testé auprès de quatre enseignants et 107 élèves : 21 en 5e primaire (P5), 44 en en 6e primaire (P6 - deux expérimentations) et 42 en 1re secondaire (S1 - deux expérimentations). Le matériel a évolué entre les différentes itérations, sur le fond et sur la forme. L'activité a été menée par les enseignants, en présence d'un membre de l'équipe de recherche qui intervenait principalement lors du débriefing. Pour chaque test, la collecte de données s'est déroulée en plusieurs étapes. Avant et après le test, des entretiens ont été menés avec les enseignants concernant leurs besoins avant le jeu, mais aussi leur vécu durant celui-ci et leur ressenti du vécu des élèves. Il leur a également été demandé d'évaluer, d'un point de vue pédagogique, le dispositif et de formuler quelques recommandations. Durant le test, des observations ont été réalisées par le chercheur afin de valider la jouabilité du dispositif dans un contexte réel et de vérifier la compréhension du jeu lui-même et des enjeux liés à la cybersécurité par les participants. Afin d'enrichir ces observations, des entretiens individuels ont été menés avec six élèves de P6.

4 Résultats et discussion

L'analyse et la discussion des résultats, illustrés par des verbatims, permet de répondre aux deux questions de recherche : d'une part, la question de la faisabilité de l'activité en classe et, d'autre part, la question des apprentissages.

4.1 Le dispositif éducatif "Stop Hackers"

Si les quatre enseignants ayant testé le dispositif soulignent à l'unanimité le côté ludique de l'activité et l'intérêt d'aborder la thématique de la cybersécurité en classe, ils ne semblent pas toujours au clair avec les objectifs d'apprentissage. Ainsi, lors de la phase de débriefing, les enseignants éprouvent des difficultés à structurer la discussion avec les élèves et ne savent pas toujours comment réagir

face à leurs propos. Ils souffrent d'un manque de maîtrise des enjeux liés à la cybersécurité. Bien que l'enseignant de S1 paraisse mieux connaitre le sujet et structurer, de ce fait, le débriefing, il éprouve tout de même des difficultés à gérer les échanges.

Concernant les élèves, certains semblent en difficulté avec leur rôle durant l'activité. Deux raisons pourraient l'expliquer. La première est un manque de précisions dans les consignes données par l'enseignant au début de l'activité et dans les réponses aux questions des élèves à propos de ces consignes. Un enseignant de P6 témoigne : *"Au début de l'activité j'étais vraiment énervé car je n'arrivais pas à gérer toutes les questions tout en continuant à donner les consignes."*. En outre, les élèves ne parviennent pas toujours à faire le lien entre un rôle et son matériel spécifique. Une évolution des supports a été rapidement proposée. Ainsi, l'enseignant de S1 a bénéficié, lors de sa deuxième expérimentation, d'une version améliorée du matériel pour les pirates mentionnant clairement ce qu'ils pouvaient ou ne pouvaient pas faire. Cette adaptation a notamment rendu les élèves jouant ce rôle plus autonomes, ce qui a eu comme effet direct de soulager l'enseignant en début d'activité. Toutefois, malgré la confusion des enseignants quant aux consignes, les observations ont montré que les élèves étaient capables, en fin d'activité, d'expliquer les règles du jeu et les diférents rôles proposés. Cela a été confirmé dans les entretiens : *"c'est un jeu avec plusieurs rôles. Chaque personne a un rôle spécial : six amis, le routeur qui est une machine qui ne peut pas parler qui fait passer les messages aux amis et les techniciens qui sont censés aider le routeur mais qui sont en fait des pirates. Le but du jeu c'est de s'envoyer des messages entre amis pour cumuler des points sans se faire pirater et perdre des points"* (élève de P6). La seconde raison est relative à un manque d'adaptation du dispositif à la réalité de la classe : les élèves circulent beaucoup durant l'activité, une grande quantité de matériel est nécessaire et celui-ci est très vite dispersé sur les bancs, des confusions sont possibles entre les cartes des amis et des pirates, le timing de l'activité est serré, entre autres. Concernant ce dernier point, la contextualisation (les références aux pratiques médiatiques des élèves, les consignes et la distribution du matériel) est chronophage, durant en moyenne 20 minutes. Il est conseillé d'accorder au moins 30 minutes au débriefing. L'enseignant doit donc gérer le temps de jeu et adapter, en temps réel, le nombre de tours au rythme des élèves.

Enfin, selon le niveau des élèves, l'activité n'a pas été vécue de la même manière. En secondaire, le rôle "ami" semble trop simpliste et les élèves s'ennuient comme le souligne l'enseignant : *"il faut trouver une occupation pour les amis qui attendent que le routeur et les pirates agissent."*. Une solution est proposée par cet enseignant : demander aux "amis" de prendre notes des critères sur lesquels ils reposent leur choix de consulter ou non un message. Lors du débriefing, ces notes alimenteront les discussions. Cet élément n'a pas été relevé dans les classes de primaire. Au-délà de ces quelques moments creux, que ce soit en primaire ou en secondaire, les élèves sont absorbés par l'activité et les tâches à réaliser. Ils ne cherchent pas à gagner mais plutôt à comprendre les rôles et stratégies. Un élève

de P6 résume parfaitement ces observations : *"il n'y a pas vraiment de gagnant. L'objectif, c'est de comprendre ce que sont les hackers et comment ça se passe"*.

4.2 Les apprentissages

Il apparait que les élèves s'inspirent surtout de ce qu'ils vivent au quotidien pour se poser des questions. Lors du débriefing, ils sont nombreux à signaler qu'eux-mêmes ou un membre de leur entourage ont déjà été piratés ou ont subi une tentative de piratage par phishing : *"on reçoit des messages quand on joue comme - si tu mets ton âge, ton nom, ton adresse, tu peux gagner 2000 ou 3000 €- mais c'est pas du tout ça"* (élève de P5) ; *"une amie à mes parents a reçu un message qui disait qu'elle devait retirer de l'argent d'un compte pour mettre sur un autre, mais en fait ils lui ont volé 4000 €"* (élèves de P5). La thématique fait donc sens dans cette tranche d'âges et les élèves semblent avoir compris qu'il existe différentes attaques en ligne et qu'il est possible de les éviter (mais pas forcément comment les éviter). Ainsi, ils comprennent que les publicités peuvent être piégées et qu'il existe des pirates qui espionnent ce qu'ils s'envoient : *"certains tournaient autour de nous et ils notaient des choses sur une feuille."* (élève de P6). Les élèves incarnant les pirates perçoivent mieux que le contexte d'un message est important à analyser : *"Si je savais que le personnage aimait la nourriture, je mettais des publicités sur la nourriture"*. À l'opposé, un élève (P6) qui a joué le rôle d'un ami explique, durant l'entretien, la stratégie qu'il a adoptée pour éviter les pirates. Il confie s'être rendu compte, à la fin de l'activité, que sa stratégie n'était pas la bonne, mais il ne sait pas expliquer pourquoi : *"j'ai tout de suite eu un tilt pour les hackers, donc je me suis dit que je ne devais pas prendre les cartes grises. Je n'ai pris que ceux qui venaient des amis, mais j'ai eu tort de faire comme ça"*.

Ils n'ont donc pas toujours une vision correcte de ce qu'ont représententé les différents rôles joués durant l'activité et des menaces présentes, même s'ils prennent bien conscience de leur existence lors du débriefing. Par exemple, les élèves (P6) interviewés associent, à l'unanimité, les publicités sans expéditeur connu à du piratage, mais jamais les contenus envoyés par des amis. Pourtant, le pirate Oscar usurpe l'identité des amis pour leur transférer des contenus malveillants. De plus, ces mêmes élèves lient majoritairement les intentions des pirates au vol d'argent et ne donnent de la valeur qu'à des données personnelles non illustrées dans l'activité : *"certaines sont plus importantes, non ? Ce serait des codes de banque ou de téléphone, mais dans le jeu, il n'y en avait pas trop"* ; *"Je ne sais pas trop. Je dirais que les informations intimes comme des photos, c'est ce qu'on pourrait voler"* ; *"Certaines données sont plus importantes, comme savoir ce que tu aimes bien... [il se reprend] non, savoir des codes VISA et tout ça"* ; *"Je n'ai pas beaucoup de données importantes, donc je ne sais pas ce qu'on pourrait me voler"*. Dès lors, ils ne savent pas toujours identifier les données qu'eux-mêmes transmettent et la valeur qu'elles peuvent avoir pour les pirates.

Ces différents résultats montrent que la contextualisation permet d'ancrer la problématique de la cybersécurité dans les pratiques et le vécu des élèves, tout en introduisant l'expérience du jeu. Lors de celui-ci, les élèves se rendent compte

de certains aspects, ils se posent des questions, sans que le jeu leur apporte toutes les réponses, ce qui permet de nourrir le temps de débriefing. Celui-ci est donc un moment essentiel pour approfondir les questions et asseoir les apprentissages. Or les résultats montrent que ce troisième temps du dispositif comporte certaines limites. Durant le débriefing, les enseignants de primaire, par manque de connaissances, se sont principalement focalisés sur les témoignages des élèves ou se sont référés à leur propre vécu, sans gagner en généralisation au niveau du questionnement et sans revenir sur les différents types de menace en ligne. De manière générale, les aspects techniques (notamment, comment éviter une attaque) ont peu été abordés, les enseignants étant moins à l'aise pour en discuter. Seul l'enseignant du secondaire, professeur d'informatique, y a introduit des notions techniques. Renforcer les thèmes plus techniques dans la documentation servant au débriefing semble nécessaire, celle-ci étant plutôt orientée sur la dimension critique. Ces thèmes devront être accompagnés de notions théoriques pour donner une base minimale de connaissances aux enseignants.

5 Conclusion

A partir du modèle d'une éducation au numérique critique et citoyenne, le dispositif "Stop Hackers" a pour objectif d'éduquer les enfants de 10-14 ans à la cybersécurité. Le projet de recherche a montré que l'activité, conçue à partir d'une démarche de type recherche orientée par la conception, peut être menée par des enseignants dans leur classe et permet de rencontrer, partiellement, les objectifs d'apprentissage. La contextualisation et l'expérience de jeu permettent de développer un questionnement critique, même s'il pourrait davantage monter en généralisation au cours du débriefing. Les aspects liés à la perspective de l'éducation à l'informatique doivent encore être renforcés, notamment à partir de la documentation fournie. Une autre possibilité serait de poursuivre l'activité par une leçon spécifique sur les notions plus techniques. Les enseignants seraient également plus confiants s'ils pouvaient bénéficier d'une formation sur la thématique.

References

1. Amankwa, E.: Relevance of Cybersecurity Education at Pedagogy Levels in Schools. Journal of Information Security, 12(4), 233-249. (2021)
2. Anderson, T., & Shattuck, J.: Design-based research: A decade of progress in education research?. Educational researcher, 41(1), 16-25. (2012)
3. Cobb, P., Confrey, J., DiSessa, A., Lehrer, R., & Schauble, L.: Design experiments in educational research. Educational researcher, 32(1), 9-13. (2003)
4. Corradini, I., & Nardelli, E.: Developing digital awareness at school: a fundamental step for cybersecurity education. In International Conference on Applied Human Factors and Ergonomics (pp. 102-110). Springer, Cham. (2020, July)
5. De la Higuera, C.: A l'ecole, doit-on enseigner l'informatique ou le coding ?Retrieved from http://www.slate.fr/story/110897/ecole-enseigner-informatique-coding, accessed December 7, 2021. (2015)

6. Fastrez, P.: Quelles compétences le concept de littératie médiatique englobe-t-il ? Une proposition de définition matricielle. Recherches en communication, 33, 35-52. (2010)
7. James, C., Weinstein, E., & Mendoza, K.: Teaching digital citizens in today's world: Research and insights behind the common sense K–12 digital citizenship curriculum. Common Sense Media. (2019)
8. Kumar, P., Vitak, J., Chetty, M., Clegg, T. L., Yang, J., McNally, B., & Bonsignore, E.: Co-designing online privacy-related games and stories with children. In Proceedings of the 17th ACM Conference IDC (pp. 67-79). (2018)
9. Lastdrager, E., Gallardo, I. C., Hartel, P., & Junger, M.: How effective is anti-phishing training for children?. In Thirteenth Symposium on Usable Privacy and Security (SOUPS 2017) (pp. 229-239). (2017)
10. Maqsood, S., Biddle, R., Maqsood, S., & Chiasson, S.: An exploratory study of children's online password behaviours. In Proceedings of the 17th ACM Conference on Interaction Design and Children (pp. 539-544). (2018)
11. Martin, F., Gezer, T., & Wang, C.: Educators' perceptions of student digital citizenship practices. Computers in the Schools, 36(4), 238-254. (2019)
12. Martin, F., Gezer, T., Wang, W. C., Petty, T., & Wang, C.: Examining K-12 educator experiences from digital citizenship professional development. Journal of Research on Technology in Education, 1-18. (2020)
13. Nicholson, J., Javed, Y., Dixon, M., Coventry, L., Ajayi, O. D., & Anderson, P.: Investigating teenagers' ability to detect phishing messages. In 2020 EuroSPW (pp. 140-149). IEEE. (2020)
14. Nicholson, J., Terry, J., Beckett, H., & Kumar, P.: Understanding Young People's Experiences of Cybersecurity. In European Symposium on Usable Security 2021 (pp. 200-210). (2021)
15. Orlando, J.: Kids need to learn about cybersecurity, but teachers only have so much time in the day. The Conversation. Retrieved January 10, 2022 from http://theconversation.com/kids-need-to-learn-about-cybersecurity-but-teachers-only-have-so-much-time-in-the-day-112136 (2019)
16. Pencheva, D., Hallett, J., & Rashid, A.: Bringing cyber to school: Integrating cybersecurity into secondary school education. IEEE Security & Privacy, 18(2), 68-74. (2020)
17. Putnam, C.: Teaching in a Digital Age: Internet Safety Education. (2019)
18. Quayyum, F., Cruzes, D. S., & Jaccheri, L.: Cybersecurity awareness for children: A systematic literature review. International Journal of Child-Computer Interaction, 100343. (2021)
19. Rahman, A., Sairi, I. H., Zizi, N. A. M., & Khalid, F.: The importance of cybersecurity education in school. International Journal of Information and Education Technology, 10(5), 378-382. (2020)
20. Saariketo, M.: Imagining alternative agency in techno-society: outlining the basis of critical technology education (en). Media practice and everyday agency in Europe, 129-138. (2014)
21. Saariketo, M.: Reflections on the question of technology in media literacy education. In Reflections on media education futures: contributions to the Conference Media Education Futures in Tampere, Finland (pp. 51-61). (2014)
22. Snyman, D. P., Drevin, G. R., Kruger, H. A., Drevin, L., & Allers, J.: A Wolf, Hyena, and Fox Game to Raise Cybersecurity Awareness Among Pre-school Children. In International Symposium on Human Aspects of Information Security and Assurance (pp. 91-101). Springer, Cham. (2021, July)

23. Voogt, J., & Roblin, N. P.; A comparative analysis of international frameworks for 21st century competences: Implications for national curriculum policies. Journal of curriculum studies, 44(3), 299-321. (2012)
24. Vuorikari, R., Punie, Y., Gomez, S. C., & Van Den Brande, G.: DigComp 2.0: The digital competence framework for citizens. Update phase 1: The conceptual reference model (No. JRC101254). Joint Research Centre (Seville site). (2016)
25. Zhao, J., Wang, G., Dally, C., Slovak, P., Edbrooke-Childs, J., Van Kleek, M., & Shadbolt, N.: I make up a silly name' Understanding Children's Perception of Privacy Risks Online. In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (pp. 1-13). (2019)

Les *datasprints*, un dispositif d'éducation aux données ? Une étude exploratoire via le cas de « Traces de Soldats »

Béatrice Drot-Delange[1] et Françoise Tort[2]

[1] ACTé, Université Clermont Auvergne
[2] ENS Paris Saclay
beatrice.drot-delange@uca.fr francoise.tort@ens-paris-saclay.fr

Résumé. L'avènement de la société de la donnée représente de véritables enjeux pour l'éducation, tels que l'identification des compétences nécessaires. Nous analysons, à l'aide de la modélisation des concepts et pratiques de l'éducation aux données de Grillenberger et Romeike, le cas du *datasprint* « Traces de soldats » créé par l'Atelier Canopé 94. L'analyse porte sur un corpus hétérogène de documents rassemblant un document d'intention, un document d'accompagnement et des productions de collégiens. Les résultats obtenus mettent en évidence que la quasi-totalité des compétences proposées par le modèle sont mobilisables dans le *datasprint.* A l'inverse, des compétences susceptibles d'être mises en œuvre dans le *datasprint* ne sont pas prises en charge par le modèle. Des pistes de recherche sont proposées pour contribuer à une didactique de la donnée.

Mots clés : donnée, didactique, enseignement secondaire, situation authentique, compétences, littératie

1 Introduction

Les données, et leurs traitements, s'invitent régulièrement dans le débat public. L'exigence de publication des algorithmes, tels ceux utilisés pour l'affectation des étudiants après le baccalauréat, la mise en œuvre du règlement général sur la protection des données, les amendes infligées aux grands groupes internationaux par la Commission Nationale de l'Informatique et des Libertés en sont quelques exemples récents. Le développement de l'Internet des objets accroît de manière vertigineuse la production, le stockage et l'exploitation de données. Chaque individu ou collectif produit, plus ou moins consciemment, des données qui seront ensuite exploitées par des industries ou des services. Si la valeur économique des données n'est plus à démontrer, leur valeur patrimoniale est probablement moins connue. Dans ce contexte, on peut qualifier notre société contemporaine de société de la donnée.

Ces évolutions sociétales et techniques représentent de véritables enjeux pour l'éducation et la formation. L'une des questions qui se posent est celle des compétences nécessaires dans cette société de la donnée. Un cadre général pour cette réflexion est proposé par ce que des chercheurs nomment la littératie des données et l'éducation aux données.

Crusoe [2] en propose une définition qu'il élabore en synthétisant les différentes facettes repérées dans la littérature et qu'il juge essentielles : « La littératie des données est la connaissance de ce que sont les données, de la manière dont elles sont collectées, analysées, visualisées et partagées, et est la compréhension de la manière dont les données sont utilisées à des fins bénéfiques ou défavorables, dans le contexte culturel de la sécurité et de la vie privée » (p. 38). Cette définition intègre notamment la connaissance de l'étendue des phénomènes ou objets que l'on peut décrire par des données, des différentes méthodes de collecte, des analyses statistiques et des représentations visuelles, au-delà des graphiques canoniques, des différents modèles de partage des données, en prenant en compte les contextes réglementaires s'appliquant à la protection des données, leur sécurité et leur confidentialité.

L'éducation aux données n'existe pas en tant que telle dans les enseignements scolaires en France. Toutefois, l'enseignement d'informatique introduit depuis 2015 et reconfiguré en 2017 constitue une introduction à quelques dimensions d'une éducation aux données, dans une approche disciplinaire. Nous avons montré [3] que le programme de l'enseignement des Sciences Numériques et Technologie (SNT) et ses manuels scolaires abordent certaines phases du cycle de vie des données, telles que l'acquisition, l'implémentation, l'analyse et le partage des données. Ils n'abordent pas ce qui pourrait relever de l'éducation aux médias et à l'information. Le programme ne propose pas de développer un esprit critique vis-à-vis des données, au sens de Womack [7], à savoir les compétences d'évaluation critique de l'information et la capacité à l'utiliser dans un objectif précis. C'est un enseignement disciplinaire qui vise avant tout les concepts et éventuellement certaines pratiques associées aux données.

D'autres approches de l'éducation aux données s'inscrivent davantage dans cette dimension analytique et critique. C'est le cas des *datasprints* pédagogiques. Ils sont définis comme « un dispositif de médiation numérique des savoirs contributif limité dans le temps où les participants unissent leurs compétences pour explorer, augmenter un jeu de données et proposer des visualisations éclairant une question déterminée » [1]. Ces dispositifs sont réputés ne pas nécessiter d'expertise technique ou scientifique. Leurs promoteurs estiment qu'ils favoriseraient la créativité, la collaboration et le questionnement. Les participants développeraient des habiletés numériques et critiques tout en construisant collectivement un savoir. Il s'agit pour leurs promoteurs de faire travailler les élèves dans le champ des humanités numériques [1]. Il ne s'agit pas d'un enseignement disciplinaire, mais d'une approche qui considère les données comme un matériau.

Dans la perspective d'une contribution à une didactique de la donnée, cet article présente une analyse des concepts et des pratiques relevant d'une éducation aux données mobilisés par les activités du *datasprint* « Traces de soldat ». Nous mobilisons la méthode que nous avons élaborée pour analyser les programmes de SNT [3] et qui s'appuie sur le modèle proposé par Grillenberger et Romeike [5] pour concevoir une éducation aux données. Il s'agit d'identifier les compétences nécessaires aux élèves mais aussi aux enseignants pour se lancer dans ce type de projets. Cette analyse vise également à accroître la robustesse du modèle, pour un niveau d'enseignement secondaire, voire primaire, dans le contexte scolaire français.

2 Cadre d'analyse : un modèle d'éducation aux données

L'identification des compétences nécessaires dans le champ de la littératie des données est un travail mené par de nombreux chercheurs. Parmi eux, les travaux de Grillenberger et Romeike portent sur l'identification des idées fondamentales de ce champ. Ils élaborent un modèle d'éducation aux données [5]. Selon eux, ce modèle serait plus adapté à l'enseignement primaire et secondaire que les travaux déjà existants dans le champ, s'adressant plutôt à l'enseignement supérieur.
Ils proposent un modèle qui distingue d'une part les concepts scientifiques sous-jacents à l'éducation aux données – intitulés « *content area* » dans le modèle– et d'autre part les pratiques avec et sur les données – intitulés « *process* » – basées sur le cycle de vie des données.

Les concepts relèvent principalement de l'informatique. Ils sont regroupés en quatre domaines. Le premier (C1) concerne les concepts de base tels que la distinction entre donnée et information ou la représentation numérique de l'information, par exemple. Le deuxième (C2), recouvre les concepts relevant du stockage et de l'accès aux données, et aborde la duplication ou la synchronisation des données. Le troisième (C3) se focalise sur les méthodes, algorithmes et principes nécessaires à l'analyse des données. Enfin le quatrième (C4) inclut les questions d'éthique, de sécurité et de confidentialité des données.

Quant aux pratiques, elles sont définies en référence au cycle de vie des données. Elles couvrent, selon les auteurs, l'ensemble des pratiques généralement mentionnées dans les définitions de l'éducation aux données : acquisition, nettoyage, modélisation, implémentation, optimisation, analyse, visualisation, évaluation, partage, suppression ou archivage. Le cycle de vie sous cette forme n'étant pas facilement utilisable en contexte scolaire, les auteurs l'ont adapté en identifiant, avec des enseignants et des chercheurs, les phases utiles lors de la mise en œuvre d'une leçon concernant l'éducation aux données. Dans leur modèle, les pratiques sont regroupés en quatre phases : « Collecter, modéliser et nettoyer » (P1), « Implémenter et optimiser » (P2), « Analyser, visualiser et interpréter » (P3) et « Partager, archiver et effacer » (P4).

Le modèle proposé combine ces deux dimensions (concepts et pratiques) sous forme d'une matrice permettant d'identifier précisément les compétences à l'intersection de chacune de ces deux dimensions. Ainsi, le domaine P1 recouvre les capacités de répondre aux questions suivantes : quelles caractéristiques du système à modéliser doit-on collecter en tant que données (P1C1) ? Comment peut-on les collecter (P1C2) ? Comment stocker ces données de manière à pouvoir les utiliser plus tard ? Les données collectées sont-elles utiles à l'atteinte des objectifs visés (P1C3) ? Comment peut-on pratiquement assurer la sécurité des données collectées (P1C4) ? Ces questions sont détaillées pour chaque pratique dans Drot-Delange et Tort [3]. Nous utilisons cette matrice pour identifier les concepts et les compétences mobilisées dans les activités proposées par un *datasprint*.

3 Méthodologie et corpus : le *datasprint* « Traces de soldat »

Le *datasprint* « Traces de soldats », a été créé par l'Atelier Canopé 94. Il proposait aux participants, élèves et enseignants de travailler à partir des traces inscrites sur les monuments commémoratifs et des données numériques pour comprendre les parcours de « Poilus » de la Grande Guerre. Il s'est tenu pendant l'année 2018.

Des ressources à destination des enseignants ont été produites, des productions d'élèves publiées. L'activité a également généré de nombreux échanges sur les réseaux sociaux. Dans cette étude exploratoire, nous avons constitué et analysé quatre corpus constitués de documents hétérogènes. Les corpus 1 et 2 comportent des documents d'accompagnement des enseignants qui nous renseignent sur les intentions pédagogiques du projet, les corpus 3 et 4 des productions d'élèves

Le corpus 1 est un article de Franck Bodin, directeur de l'atelier Canopé 94, publié sur medium.com, qui présente la démarche et l'intérêt pédagogique du travail avec les données[1]. Il comporte plusieurs vidéos et des liens vers d'autres articles. Le corpus 2 est un kit pédagogique produit par Philippe Chadefaux, enseignant de Sciences numériques et technologie (SNT), et publié sur la plateforme Canoprof[2]. Il comporte plusieurs pages et des tutoriels vidéo réalisés par l'auteur ou disponibles en ligne.

Ce projet a été réalisé dans de nombreux établissements, écoles, collèges, lycées, de toutes les régions de France. Il a abouti à la publication, par le réseau Canopé d'un jeu de données sur les poilus et de 62 « réutilisations » de ce jeu, correspondant à des productions d'élèves[3]. Pour chaque réutilisation, un court texte descriptif présente le travail sur les données réalisé par les élèves. De plus, les enseignants ont témoigné de l'avancée des travaux de leurs élèves sur tweeter (#tracesDeSoldats), postant parfois des photographies de productions intermédiaires sur papiers ou sur ordinateur.

Pour mener un premier travail exploratoire d'analyse des productions des élèves, nous avons sélectionné les productions de deux établissements scolaires, pour lesquelles nous avions accès à du matériel varié : copies d'écran, vidéos interactives, interview des élèves, affiches, etc. Ainsi, nous avons construit un troisième corpus 3 avec la production publiée du collège Simone de Beauvoir de Créteil (v. figure 1) publiée et 4 photographies d'écrans présentant un travail intermédiaire sur tableur (v. figure 2), tweetées par l'enseignant[4]. Un quatrième corpus 4 comporte les onze productions publiées du collège Achille Mauzan et du lycée Aristide Briand de Gap, ainsi qu'une vidéo

[1] Apprendre avec les données numériques, les *datasprints* pédagogiques : le cas de Traces de Soldats, Bodin F., 29 mai 2018, medium.com, consulté le 10/01/22.

[2] Traces de soldats - le *datasprint* historique, Chadefaux P, consulté le 10/01/22 : https://philippe-chadefaux.canoprof.fr/eleve/TdS/

[3] Jeu de données "Première Guerre mondiale - Les Poilus morts pour la France (à compléter)", Réseau Canopé, data.gouv.fr

[4] Tweet du 12/10/2018 : https://twitter.com/GirardinYG/status/1050676563602071552 et du 16/10/2018 : https://twitter.com/GirardinYG/status/1052223734860394496i

interactive comportant les visualisations produites, et en bande son des témoignages des élèves répondant à des questions sur leurs activités et leurs résultats (v. tableau 1)[5].

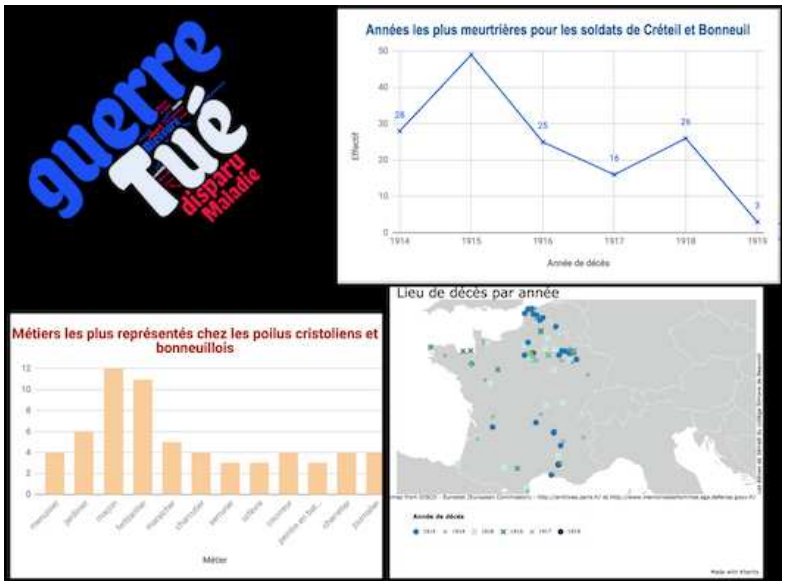


Fig. 1. Extrait de la production du Collège Simone de Beauvoir de Créteil, publiée sur le site data.gouv.fr, comme cas d'utilisation du jeu de données.

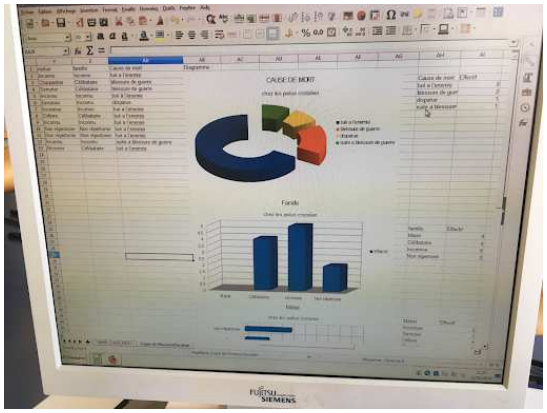


Fig. 2. Une photographie de l'écran d'ordinateur d'une session de travail des élèves du Collège Simone de Beauvoir de Créteil, postée sur Twitter par l'enseignant.

[5] Traces de soldats - Le datasprint historique Hautes Alpes, consulté le 10/01/22 : https://www.hautes-alpes.fr/evenement/1416/1639-actualites.htm

Le travail d'analyse a consisté en une analyse thématique descendante du discours des corpus 1 et 2 (texte et description du contenu des vidéos avec verbatim). Pour chaque portion de texte, nous en avons inféré les concepts et les pratiques en matière d'éducation aux données. Ensuite nous les avons classés selon la matrice de Grillenberger et Romeike (op. cit.). Les corpus 3 et 4 ont été utilisés, lors de la synthèse de l'interprétation pour émettre des hypothèses sur le travail effectivement réalisé par les élèves.

Tableau. 1. Extrait des témoignages des élèves, retranscrits de la bande son de la vidéo interactive produite par le collège Achille Mauzan et le lycée Aristide Briand de Gap

"On est parti d'une base de données, la fiche synthèse des matricules. C'était un tableau Excel pas complété. On a dû faire des recherches pour le compléter. On avait leur matricule, date de mort, de naissance, endroit d'où ils venaient, leur lieu de mort, des fois c'était pas en France."
"Le prof nous a donné des thèmes pour qu'on soit pas perdus. Par exemple : le premier mort et le dernier mort, le nombre de soldat par l'année de naissance, le déplacement dans les tranchées."
"On a commencé par certains sites : mémoire haute alpes, mémoire des hommes. On a trouvé toutes les infos. On pouvait accéder à une photo de leur pièce d'identité et chercher ce qui n'était pas forcément visible"
"On a fait des moyennes, et avec pictochart on a réussi à faire comme une présentation pour des affiches, et c'est plus agréable à lire. On a été voir des anciens combattants qui ont pu faire des *témoignages. Il y avait des poèmes. Des lycéens ont vu notre travail."*

4 Résultats

L'interprétation proposée porte sur la couverture par les activités du *datasprint* du domaine de l'éducation aux données, tel que modélisé par Grillenberger et Romeike. Une synthèse est proposée dans la figure 3.

La pratique « Collecter, modéliser et nettoyer » (P1) est très largement couverte par les activités proposées. Le corpus 1 précise que « le fichier de données fourni initialement doit être enrichi par un travail de *crowdsourcing* effectué en classe », qui met en œuvre la compétence de sélection et d'exploitation de sources de données (P1C1). Dans une section intitulée « récolter des données », le corpus 2 montre comment l'enseignant peut réaliser le jeu de données initial depuis le site du ministère des armées « Mémoires des hommes »[6] et précise « Les élèves auront pour travail de compléter la base de données déjà constituées avec d'autres sources et de l'uniformiser pour pouvoir l'exploiter ». Il s'agit ici de sélectionner et formater les données pour respecter le modèle proposé (P1C2). Plusieurs vidéos montrent comment repérer les données manquantes dans un tableau de données et comment les compléter en allant les chercher sur des sites spécialisés. Il est également question d'aller collecter des données sur le terrain (auprès des mairies, des familles, etc.). Dans le corpus 3, le texte descriptif de la réutilisation précise que les élèves ont complété le jeu de données des causes de la mort, présentes sur la fiche numérisée accessible via « Mémoire des Hommes », et des métiers des

[6] https://www.memoiredeshommes.sga.defense.gouv.fr/

Poilus, trouvés dans les registres matricules. Dans le corpus 4, une élève témoigne d'une pratique effective de collecte de données pour compléter un fichier au moyen d'un tableur (v. tableau 1, ligne 1). Enfin, nous associons à la compétence d'analyse lors de la collecte des données (P1C3) la vérification de la fiabilité des données. Celle-ci est abordée dans les deux corpus. Le corpus 1 précise que « Repérer la donnée fiable, connaître les sources primaires, vérifier, croiser les informations secondaires sont des compétences développées à travers le *datasprint* ». Le corpus 2 reprend la notion de sources primaires pour insister sur leur fiabilité, et les processus de vérification.

Pratiques / Concepts	P1 Collecter, modéliser et nettoyer	P2 Implémenter et optimiser	P3 Analyser, visualiser et interpréter	P4 Partager, archiver et effacer
C1 Données, informations				
C2 Accès et stockage des données				
C3 Analyse des données				
C4 Éthique et sécurité des données				

Compétences couvertes Compétences non couvertes

Fig. 3. Compétences relevant de l'éducation aux données dans le *datasprint* Traces de soldats

La pratique « Implémenter et optimiser » (P2) est également couverte par les activités proposées. Dans une section intitulée « Traitement des données », le corpus 2 propose des tutoriels vidéo détaillant comment réaliser pratiquement les recherches d'informations complémentaires, par *web scraping* ou par recherche d'informations et leur ajout dans le jeu de données ; comment mettre en forme le jeu de données (scinder des données en plusieurs colonnes) ; comment réaliser certains calculs statistiques simples avec un tableur ; comment filtrer et trier des données. Ces activités relèvent de l'implémentation et l'optimisation de la mise en forme des données pour la collecte (P2C1) et pour l'analyse (P2C3). Il est précisé : « D'autres outils [que le tableur] existent mais ils sont plus difficiles à prendre en main pour des élèves et le tableur est un outil au programme dès le collège ». Les copies d'écrans du corpus 3 (v. figure 2) montrent des travaux d'élèves en cours de réalisation avec un tableur, laissant supposer qu'ils ont réalisé ce type de traitement sur les données. On peut supposer que le travail collaboratif a amené les élèves et les enseignants à choisir les formats de fichiers à partager (P2C2).

La pratique « Analyser, visualiser et interpréter » (P3) est centrale dans les intentions énoncées par le corpus 1. Les élèves doivent « être en mesure d'analyser les données disponibles, de leur donner un sens, de formuler des hypothèses, puis de les vérifier en

opérant des choix : quelles données garder, quelles données compléter, quelles données vérifier, croiser. » (P3C1) Cette analyse et ces choix sont guidés par l'objectif de produire une ou plusieurs visualisations de données. Dans une section intitulée « faire parler les données : datavisualisation », le corpus 2 détaille les différentes représentations graphiques et leur utilisation. Pour chacune, sont énoncées des règles permettant d'en améliorer la lisibilité et une vidéo présente un kit pédagogique sur le design graphique qui introduit les notions de typographie, couleur, mise en page. Ceci doit permettre aux élèves de choisir les visualisations adaptées à leur objectif (P3C3).

La pratique « Partager, archiver et effacer » (P4) est absente des corpus 1 et 2. Pourtant le projet a comporté une activité de partage des données et des visualisations produites sur le site data.gouv.fr. Il semble que la préparation du jeu de données, les choix afférents au partage, et le dépôt ont été réalisés par les membres des ateliers Canopé. Nous n'avons pas d'information sur la façon dont les élèves et les enseignants y ont été associés.

Les concepts relevant de l'éthique et de la sécurité des données (C4) ne sont pas abordés dans le corpus. Ceci peut s'expliquer par le sujet lui-même : les données portent sur des individus décédés pour lesquels les questions de protection de données personnelles et d'anonymisation ne se posent pas[7].

5 Discussion et perspectives

L'ensemble des supports analysés dans notre étude exploratoire est constitué d'un document que l'on pourrait qualifier d'intention (corpus 1), d'une trame qui explore de manière quasi-exhaustive les activités pouvant être réalisées avec les élèves et qui fournit des ressources pour les mener à bien (corpus 2), de différentes traces des activités des élèves de deux établissements (corpus 3 et 4). Il s'agit des productions des élèves, publiées sur le site des données ouvertes data.gouv.fr par le réseau Canopé, des photographies d'écran d'ordinateur prises par l'enseignant pendant une séance de travail avec ses élèves, et de témoignages des élèves enregistrées pour la bande son d'une production interactive. Nous n'avons pas réalisé d'observation directe de l'activité.

Le corpus 1 comporte beaucoup d'expressions pour qualifier l'approche pédagogique et ses objectifs, que l'on ne retrouve pas dans l'enseignement disciplinaire de SNT : la rupture avec l'approche disciplinaire des savoirs (même si le programme de SNT incite les enseignants de cette discipline à collaborer avec des collègues d'autres disciplines), la co-réalisation en équipe, la participation à un commun numérique, le développement de l'esprit critique, l'expérience d'éducation par la recherche, etc. Nous avons relevé ce qui nous apparaît comme une contradiction entre les corpus 1 et 2. Le premier précise que « Paradoxalement les compétences les plus largement travaillées relèvent moins des habiletés numériques que des compétences critiques », alors que le corpus 2 comporte des tutoriels présentant des habiletés techniques assez poussées de

[7] Voir sur le site de l'INSEE: https://www.data.gouv.fr/fr/datasets/fichier-des-personnes-decedees/

recherche d'information sur le Web, d'utilisation d'outils dédiées ou de l'outil tableur. Cette contradiction est en fait révélatrice de l'adaptabilité offerte par le *datasprint* en termes d'objectifs pédagogiques et d'appropriation par les enseignants et les élèves. L'analyse des déclinaisons d'un même *datasprint* serait intéressante à mener du point de vue de la discipline enseignée par l'enseignant qui le propose. Les corpus 3 et 4 n'ont pas été analysés systématiquement, mais il ressort des similitudes dans les réalisations. Si la ville concernée change, ce sont les mêmes questions qui sont posées, les mêmes informations complémentaires qui sont recherchées. Dès lors, on peut s'interroger sur l'objectif de créativité énoncé dans le corpus. On pourrait parler d'autonomie guidée, mais aussi contrainte par ce que permet ou non le jeu de données.

En tant que dispositif d'éducation aux données, l'analyse des corpus et des productions réalisées montre que le *datasprint* « Traces de soldats » s'apparente à la conception d'une situation authentique d'apprentissage. Duval et Pagé [4] synthétisent, à partir d'une revue de la littérature, les cinq caractéristiques de ces situations par les traits suivants que nous déclinons dans le cas de « Traces de soldats ». (1) Le contexte d'apprentissage a un caractère réaliste. Travailler avec des données historiques et locales assure le caractère réaliste de la situation dans le cas de « Traces de soldats ». (2) La situation demande l'accomplissement d'une réalisation plutôt qu'une simple reprise d'information. Toutes les actions des élèves sont ici sous-tendues par l'objectif de réalisation d'une production partageable, diffusable et contribuant à la production de connaissances. (3) La situation authentique propose des tâches complexes qui favorisent le jugement et l'innovation. L'analyse du corpus montre que les élèves sont sollicités tout au long d'un processus assez complet d'analyse de données : depuis le choix des sources de collecte de nouvelles données, jusqu'à l'interprétation permise par une représentation graphique. Les différentes pratiques de l'éducation aux données, à l'exception du partage, sont mobilisées dans « Traces de soldats », nécessitant des habiletés informatiques développées. Les compétences, dans le *datasprint*, deviennent effectives, celles-ci résultant « de la mobilisation, de la sélection, de la coordination, de la mise en œuvre et des nombreux ajustements des ressources utiles pour le traitement des tâches dans une situation donnée ou une classe de situations » [6, p. 40]. Néanmoins, l'analyse du corpus ne permet pas toujours de savoir quelle est la part d'autonomie dans les choix réalisés par les élèves. (4) La situation authentique nécessite une consultation entre les élèves et une rétroaction en vue de l'amélioration de la réalisation. Cette caractéristique n'est pas vraiment identifiable dans notre corpus. En effet, nous n'avons pas d'information sur les modalités d'accompagnement des apprentissages par les enseignants ou les évaluations menées. (5) La situation authentique génère une forte motivation, qui dépasse le désir d'obtenir une bonne note. Ce point, comme le précédent, n'est pas directement identifiable dans l'analyse de nos corpus, puisqu'il nécessiterait d'observer l'activité effectivement menée.

Notre analyse exploratoire montre la richesse de ces situations d'apprentissage en matière d'éducation aux données que fournissent les *datasprints*. Cependant, nous nous interrogeons pour savoir si le cas emblématique que nous avons étudié est représentatif

des *datasprints*. Une étude serait à mener pour savoir quels rôles jouent la mise à disposition de ressources et la mise en place d'ateliers dans l'adoption par les enseignants de cette modalité pédagogique. Nous avons également souligné les limites de notre étude qui ne se base que sur une partie des ressources et des productions d'élèves disponibles. Outre l'activité des élèves, l'activité conjointe enseignants-élèves lors du déroulement de ces *datasprints* constitue une piste de recherche.

Enfin si le *datasprint* étudié couvre la quasi-totalité des compétences proposées par le modèle de Grillenberger et Romeike, celui-ci ne prend pas en compte les compétences liées à la réutilisation de données collectées par d'autres et la question de la vérification de la fiabilité des sources. Les pistes évoquées pourront contribuer à l'élaboration d'une didactique de la donnée.

Contribution

Cette recherche participe des travaux menés par le GIS 2IF (Innovation, interdisciplinarité et formation) dans le cadre du Groupe Thématique Numérique 2020-2022 « Humanités numériques : entre recherche et formation », bénéficiant du soutien du Ministère de l'Éducation nationale, de la Jeunesse et des Sports.

Références

1. Bodin, F. (2018). « Apprendre avec les données numériques, les *datasprints* pédagogiques : le cas de Traces de Soldats ». *Medium*. 17 octobre 2018.
2. Crusoe, D. (2016). Data Literacy defined pro populo : To read this article, please provide a little information. *The Journal of Community Informatics*, 12(3).
3. Drot-Delange, B., Tort, F. (à paraitre). Éducation aux données vs. Enseignement des données : une contribution aux humanités numériques à l'école ? *Humanités numériques*, vol. 5, « Enseigner et apprendre les humanités numériques ».
4. Duval, A.-M., Pagé, M. (2013). *La situation authentique : de la conception à l'évaluation. Une formule pédagogique pour toutes les disciplines*. Montréal. AQPC
5. Grillenberger, A. et Romeike, R. (2018). « Developing a theoretically founded data literacy competency model ». Dans *Proceedings of the 13th Workshop in Primary and Secondary Computing Education*, 1-10.
6. Jonnaert, P., & Vander Borght, C. (2003). Créer des conditions d'apprentissage : Un cadre de référence socioconstructiviste pour une formation didactique des enseignants. De Boeck.
7. Womack, Ryan. 2014. « Data Visualization and Information Literacy ». *IASSIST Quarterly* 38 (1) : 12-17.

Les rapports aux ressources éducatives des enseignants en informatique dans les IUT en France

Caroline Ladage[1], Mehdi Khaneboubi[2] et Cécile Redondo[1]

[1] Aix Marseille université, UR 4671 ADEF, France
caroline.ladage@univ-amu.fr, cecile.redondo@univ-amu.fr
[2] Laboratoire EDA, Université Paris Cité, France
mehdi.khaneboubi@cyu.fr

Résumé. La recherche présentée s'intéresse aux ressources pour l'enseignement dans les formations des Instituts universitaires de technologie (IUT) en France et propose d'identifier les perceptions des enseignants issus des filières en informatique, parmi celles caractéristiques des enseignants de l'ensemble des filières ayant participé à une enquête par questionnaire sur la conception et les usages de ressources éducatives en IUT. Dans une approche anthropologique du didactique la recherche présentée questionne les conditions et contraintes de la diffusion des connaissances et des praxéologies au sein des IUT. Il étudie les usages des ressources éducatives des enseignants et ce qui peut déterminer les choix qu'ils font et les collaborations qu'ils mettent en place, ou non, pour leur élaboration. Une enquête par questionnaire a été menée recueillant 1033 participations. Les résultats font apparaître des pratiques plutôt personnelles dans la conception des ressources et peu de recours à des ressources émanant des instances nationales de l'IUT et de ses réseaux, dont la volonté est pourtant d'encourager la collaboration autour de la conception des ressources et leur partage auprès de l'ensemble des enseignants en IUT. Il ressort également un manque de questionnement des spécificités des contenus à enseigner et quelques rares témoignages de l'intérêt des projets pour répondre aux exigences de professionnalisation auxquelles doivent faire face les formations en IUT.

Mots clés: didactique de l'informatique, ressources éducatives, théorie anthropologique du didactique, enseignant en IUT

1 Introduction

Notre proposition de contribution au colloque Didapro/didaSTIC s'inscrit dans la thématique « Ressources, dispositifs, scénarios et environnements pour l'enseignement de l'informatique » et se situe dans le contexte de l'enseignement de l'informatique dans les formations des Instituts universitaires de technologie (IUT) en France.
La recherche est menée dans le cadre du projet ANR Renoir – IUT dont l'un des axes de questionnement est l'étude de l'utilisation des ressources éducatives par une sélection de filières et disciplines enseignées à l'IUT, dont l'informatique. Nous avons mis

en œuvre une enquête par questionnaire diffusée au niveau national à l'ensemble des IUT, toutes filières confondues.

Notre questionnaire interroge les pratiques d'enseignement, la production de ressources et leur utilisation par les enseignants d'IUT. Nous nous intéressons ici plus particulièrement aux profils des personnes ayant déclaré enseigner dans les filières en informatique, tout en positionnant leurs déclarations par rapport aux autres filières étudiées, là où elles apparaissent comme distinctives.

Les résultats de l'enquête contribuent au moins partiellement aux questions qui sont au cœur de l'axe 2 du colloque, à savoir celles ayant trait aux ressources disponibles pour former à/par l'informatique, celles aussi des activités d'enseignement pour consolider les connaissances, ici dans le contexte des enseignements en IUT. Notre recherche propose de contribuer aux approches didactiques de l'étude des ressources éducatives en s'appuyant sur une enquête par questionnaire en ligne comme méthode de recueil de données dont l'objectif est d'encourager une participation la plus large possible.

2 La question des ressources éducatives dans une approche didactique

L'approche didactique de l'étude des ressources éducative peut se décliner en un nombre important d'axes de questionnement s'intéressant aussi bien aux programmes que ces ressources sont appelées à servir qu'à leurs acteurs, comprenant toute la chaîne des personnes impliquées dans leur conception, leur diffusion et leurs utilisations. Notre recherche s'intéresse à la diffusion des connaissances et des pratiques dans la société dans une approche anthropologique du didactique (Chevallard, 2007) dont l'une des théorisations majeures porte sur les phénomènes de transposition institutionnelle et didactique des savoirs dans la société et des rapports personnels et institutionnels aux savoirs (dont la notion doit être prise en un sens large, incluant les connaissances et praxéologies). Nous considérons toute l'importance des rapports institutionnels et personnels aux ressources éducatives sur la diffusion des connaissances dans nos sociétés. Ces rapports agissent comme un ensemble de conditions favorables ou non et de contraintes sur la nature et la qualité de la diffusion des connaissances. Or le travail des enseignants sur les ressources est complexe (Bruillard, 2019) et peut se trouver en tension entre démarches personnelles et injonctions institutionnelles au regard des exigences d'évolutivité, comme l'énonce Bruillard (2013) lorsqu'il souligne que

> « Ce qui importe, c'est que les ressources soient vivantes, c'est-à-dire qu'elles soient utilisées, discutées, modifiées, ce qui dépend des individus et des collectifs qui les portent ainsi que des mécanismes qui les réunissent. On peut ainsi douter de l'utilité de créer un répertoire national de ressources, si une animation suffisante autour de celles-ci n'est pas mise en place. »

Le contexte des formations en IUT est fortement marqué par des enjeux de professionnalisation (Tralongo, 2018) et par une exigence d'harmonisation des formations au niveau national ce qui confère un statut central aux programmes nationaux et pose également la question du choix des contenus d'enseignement et, partant de là, de la conception des ressources éducatives. Par ailleurs l'organisation de la formation en IUT

est fortement marquée par le fonctionnement complexe et imbriqué d'un réseau institutionnel et d'un réseau associatif national (Le Nir, 2015). Ce sont ces conditions qui nous ont incité à tenter de mieux comprendre le rapport des enseignants en IUT à la question des ressources éducatives, dans une approche exploratoire en nous appuyant sur une démarche d'enquête à large diffusion.

3 Une enquête par questionnaire

3.1 Description du dispositif d'enquête

Notre recherche s'appuie sur un questionnaire diffusé en ligne en 2021. Le questionnaire explore plusieurs thématiques pour déterminer les rapports et les perceptions que les participants déclarent au sujet des ressources éducatives en IUT, selon les six axes de questionnement suivants :

1) Quels types de ressources utilisent-ils et où les trouvent-ils ?
2) Avec qui préparent-ils les ressources pour leurs enseignements ?
3) Réalisent-ils des adaptations des ressources existantes ?
4) Qu'est-ce qui détermine le choix des ressources ?
5) Contribuent-ils à l'élaboration de ressources pour des éditeurs ?
6) Quelle est la place de la recherche dans leur rapport aux ressources ?

L'enquête comportait 117 questions avec quatre structures de réponses attendues : fermée unique ; fermée multiple ; fermée échelle ; ouverte textuelle. Les questions étaient présentées selon une progression classique, du plus général au plus spécifique avec un système de renvois, ce qui explique un taux de participation plus faible pour les derniers groupes de questions.

Le questionnaire a été adressé à l'ensemble des 110 IUT répertoriés en France (métropole et outre-mer). La participation à l'enquête était anonyme et est passée par l'accord préalable de la direction de chaque IUT, qui se chargeait ensuite de sa diffusion via un courrier électronique standardisé contenant un lien hypertexte vers le questionnaire en ligne. 96 établissements ont répondu positivement (soit 84 % des structures concernées), tandis que seulement 14 établissements n'ont pas répondu à nos sollicitations. À la fermeture de l'enquête nous comptions 1051 participations, dont nous avons retiré 18 participations non valides (doublons et insuffisamment renseignées). Nous avons arrêté l'échantillon à 1033 participations exploitables. La représentativité de ce recueil de données n'était pas un objectif premier mais plutôt la variété des profils des répondants. Nous voulions une distribution la plus large possible doublée d'une participation basée sur le volontariat et l'intérêt pour la question posée.

3.2 Résultats

Nous présentons ci-après les profils des participants pour proposer ensuite une synthèse des principaux résultats selon nos six axes de questionnement. Pour chaque thématique nous étudions si le profil des enseignants intervenant dans les spécialités en lien avec l'informatique présente des caractéristiques particulières. Il est évident que

l'enseignement de l'informatique peut être réalisé dans l'ensemble des spécialités proposées à l'IUT. Nous retenons pour l'identification des profils des enseignants toutes les déclarations de spécialités, unique ou indiquée en deuxième voire troisième position. Nos données se situent au niveau des spécialités déclarées, le questionnaire ne visait pas une granularité plus fine qui aurait pu interroger les participants faisant intervenir l'informatique quelle que soit la discipline enseignée.

Profils des participants

Notre échantillon est composé de 580 hommes (56,1%) et de 438 femmes (42,4%), et 15 non-réponses. Il compte 438 (42,4%) enseignants du second degré affecté dans le supérieur (PRAG/PRCE) ; 466 enseignants-chercheurs titulaires (45,1%) dont 398 maîtres de conférences (38,5%) et 68 professeurs des universités (6,6%). Le statut de 123 autres participants (11,9 %) se répartit à peu près de manière homogène entre enseignants-chercheurs contractuels, associés et chargés d'enseignements vacataires. L'ancienneté dans l'enseignement à l'IUT est distribuée de manière relativement homogène avec 19,6% entre 6 et 10 ans ; 17,1% entre 1 et 5 ans ; 16,4% entre 11 et 15 ans ; 15,7% entre 21 et 25 ans ; 13,0% entre 16 et 20 ans. Notons que 13,8% déclarent plus de 26 ans d'ancienneté.

Pour la conception du questionnaire, nous avons recensé 24 spécialités d'enseignements, pour lesquelles chacune a trouvé des représentants, allant de seulement 7 participations pour la spécialité « Packaging, emballage, conditionnement » à 146 participants pour « Gestion des entreprises et administrations » (GEA). La question permettait des réponses multiples ce qui a amené certains participants à cocher plusieurs spécialités. Après GEA, les autres spécialités les plus représentées sont par ordre décroissant « Techniques de commercialisation » (n = 135 ; 13,1%) ; « Informatique » (117 ; 11,3%) ; « Génie électrique et informatique industrielle » (100 ; 9,7%) ; « Génie biologique » (80 ; 7,7%). Notons dans le cadre de la population qui nous intéresse ici autour de l'enseignement de l'informatique, la participation de 27 personnes de la spécialité « Statistiques et informatique décisionnelle ».

Le regroupement des trois spécialités explicitement en lien avec l'enseignement de l'informatique (« Informatique », « Génie électrique et informatique industrielle » et « Statistiques et informatique décisionnelle ») concerne 229 participants dont 16 interviennent dans deux spécialités et un seul dans les trois à la fois.

Le traitement des données a été réalisé à l'aide des logiciels Sphinx et Excel. Nous présentons dans les tableaux ci-après la distribution des effectifs obtenues, tout en y intégrant par un code couleur un calcul d'écart à l'effectif théorique attendu par cellule, signalé en rose lorsqu'on est en présence d'éléments sous-représentés, et en bleu pour les éléments sur-représentés en nous nous appuyant sur un test du Khi2 (au seuil de 5 %). Sans pouvoir commenter en détail chaque résultat, nous repérons les résultats saillants en y associant certains résultats qualitatifs recueillis dans les réponses à nos questions ouvertes dont les objectifs étaient de recueillir des justifications et illustrations des réponses aux questions fermées.

Dans ce qui suit nous étudions les réponses de cette sous-population à nos six axes de questionnement au regard des témoignages de l'ensemble de l'échantillon, en

commençant par les résultats au niveau de l'échantillon global et en précisant ensuite le profil de la sous-population d'enseignants en IUT dans les filières en informatique.

1) Quels types de ressources utilisent-ils et où les trouvent-ils ?

Le questionnaire présentait sous forme de deux listes une sélection de types de ressources pour lesquels il était demandé aux participants de déclarer où ils trouvent les ressources nécessaires à la préparation de leurs enseignements, en se positionnant sur une échelle impaire « jamais, rarement, occasionnellement, assez souvent, très souvent ».

Du tableau 1 il ressort que de façon significative les plateformes pédagogiques en ligne ne sont jamais ou rarement utilisées ; il en va de même des sites web institutionnels. À l'inverse, de façon significative, les sites spécialisés, les livres et revues scientifiques et spécialisés figurent parmi les ressources assez et très souvent utilisées.

Tableau 1. Types de ressources 1 sur 2, échantillon global.

	Jamais		Rarement		Occasionnellement		Assez souvent		Très souvent	
	N	% cit.	N	% cit.	N	% cit.	N	% cit.	N	% cit.
Des manuels d'éditeurs universitaires	263	26,8%	203	20,7%	232	23,7%	196	20,0%	86	8,8%
La plateforme pédagogique "IUT en ligne"	458	49,9%	293	31,9%	122	13,3%	36	3,9%	9	1,0%
D'autres plateformes pédagogiques	325	36,4%	209	23,4%	223	24,9%	100	11,2%	37	4,1%
Des sites web institutionnels (Ministère, Académie,...)	374	40,3%	236	25,4%	174	18,7%	113	12,2%	32	3,4%
De la documentation pédagogique : autre	239	26,8%	206	23,1%	232	26,0%	161	18,1%	53	5,9%
Des sites Internet spécialistes	73	7,4%	110	11,1%	250	25,2%	331	33,4%	228	23,0%
Des articles et revues scientifiques	161	16,8%	214	22,4%	204	21,3%	233	24,3%	145	15,2%
Des livres scientifiques et spécialisés	150	15,7%	169	17,7%	217	22,7%	249	26,0%	172	18,0%
Des revues professionnelles	236	25,0%	227	24,0%	223	23,6%	170	18,0%	88	9,3%
Autres ressources de ce type	263	43,7%	110	18,3%	107	17,8%	74	12,3%	48	8,0%
Total	**2542**	**28,0%**	**1977**	**21,8%**	**1984**	**21,9%**	**1663**	**18,3%**	**898**	**9,9%**

Dans le tableau 2, ci-dessous, nous découvrons plus particulièrement les usages de ressources numériques disponibles sur Internet, principalement en accès libre. Les réseaux sociaux ne constituent que très rarement une ressource pour les enseignements, alors qu'une utilisation plus fréquente est déclarée des sites et blogs d'enseignants et des encyclopédies en ligne. Les ressources qui ressortent comme significativement souvent utilisées sont les plateformes de vidéo de type YouTube et les sites d'actualités.

Tableau 2. Types de ressources 2 sur 2, échantillon global.

	Jamais		Rarement		Occasionnellement		Assez souvent		Très souvent	
	N	% cit.	N	% cit.	N	% cit.	N	% cit.	N	% cit.
Des sites ou blogs d'enseignants	310	33,2%	254	27,2%	218	23,3%	113	12,1%	39	4,2%
Des réseaux sociaux : Facebook	732	85,4%	90	10,5%	18	2,1%	9	1,1%	8	0,9%
Des réseaux sociaux : autres	658	76,8%	117	13,7%	55	6,4%	16	1,9%	11	1,3%
Des plateformes de vidéo type Youtube	153	15,7%	306	31,4%	290	29,7%	158	16,2%	68	7,0%
Des abonnements de type newsletters	499	56,6%	203	23,0%	101	11,5%	64	7,3%	15	1,7%
Des manuels grand public	378	42,8%	247	27,9%	174	19,7%	71	8,0%	14	1,6%
Des actualités	204	21,6%	244	25,8%	231	24,4%	173	18,3%	94	9,9%
Des encyclopédies en ligne	269	29,5%	250	27,4%	229	25,1%	122	13,4%	42	4,6%
Autres ressources du type de cette liste	381	64,8%	92	15,6%	71	12,1%	35	6,0%	9	1,5%
Total	**3584**	**45,7%**	**1803**	**23,0%**	**1387**	**17,7%**	**761**	**9,7%**	**300**	**3,8%**

Si nous portons notre attention à la sous-population d'enseignants dans les spécialités en informatique – sans reproduire ici les tableaux faute de place –, nous ne constatons que peu de différences par rapport à l'échantillon global. Le seul contraste significatif que nous repérons dans le premier tableau est l'utilisation plus rare des revues professionnelles et dans le deuxième tableau encore moins de recours aux réseaux sociaux et moins d'utilisation des actualités. Dans les questions ouvertes qui invitaient à signaler d'autres ressources, il est intéressant d'observer que quelques participants citent les logiciels libres, les tutoriels ou encore la « documentation et notes d'applications constructeurs ».

Un groupe de questions était consacré aux plateformes de ressources. Il ressort des résultats que c'est l'IUT-en-ligne qui est le plus souvent utilisé, mais le degré de cette utilisation est très faible (30 participants seulement déclarent l'utiliser très souvent). Cette tendance est confirmée pour la sous-population en informatique.

Il ressort des réponses à la question sur les fonctions perçues de la plateforme IUT en ligne que celle-ci n'est que peu considérée comme une base adéquate pour les enseignements, et plutôt perçue comme un cadre utile pour les étudiants et les professionnels chargés de cours.

2) Avec qui préparent-ils les ressources pour leurs enseignements ?

Nous avons interrogé les participants à plusieurs moments du questionnaire sur leurs collaborations pour la préparation des enseignements et la conception des ressources. À commencer par la préparation des enseignements, le tableau 3 ci-dessous met en lumière qu'au niveau de l'échantillon global les enseignements sont préparés plutôt avec les collègues enseignants de proximité au sein de leur IUT. La collaboration est de manière très significative moins présente entre collègues au sein de réseaux d'IUT et les étudiants n'y occupent qu'une part très faible. Les chargés d'enseignement, les représentants du monde professionnel et les personnels techniques des IUT ne contribuent

majoritairement jamais ou occasionnellement. Les professionnels des lieux de stages sont significativement absents de ces collaborations.

Tableau 3. Collaborations déclarées pour la préparation des enseignements

	Jamais		Rarement		Occasionnellement		Assez souvent		Très souvent	
	N	% cit.	N	% cit.	N	% cit.	N	% cit.	N	% cit.
avec vos collègues enseignants de votre IUT	89	8,9%	225	22,5%	304	30,4%	206	20,6%	177	17,7%
avec vos collègues enseignants au sein de réseaux d'IUT	401	44,4%	305	33,7%	147	16,3%	37	4,1%	14	1,5%
avec les étudiants de votre IUT	339	37,0%	303	33,1%	172	18,8%	66	7,2%	35	3,8%
avec les chargés d'enseignements (issus du monde professionnel)	331	36,3%	257	28,1%	200	21,9%	89	9,7%	36	3,9%
avec des représentants du monde professionnel	319	34,4%	283	30,5%	206	22,2%	83	9,0%	36	3,9%
avec le personnel technique de votre IUT	352	37,8%	256	27,5%	189	20,3%	91	9,8%	43	4,6%
avec les professionnels des lieux de stages	365	39,9%	316	34,6%	161	17,6%	59	6,5%	13	1,4%
avec d'autres profils de personnes que ceux cités dans cette liste	417	63,1%	122	18,5%	80	12,1%	27	4,1%	15	2,3%
Total	**2613**	**36,5%**	**2067**	**28,8%**	**1459**	**20,4%**	**658**	**9,2%**	**369**	**5,1%**

Quant à savoir si les enseignants des filières de l'informatique présentent un profil différent, la réponse est non, notre sous-population ne déclare pas de réponse contrastante sur le sujet.

Des questions ouvertes sondaient les participants plus particulièrement sur la collaboration avec des étudiants, collègues ou autres partenaires pour l'élaboration de ressources. C'est le travail sur le projet (tutoré) qui ressort le plus avec 79 occurrences sur les 477 participants déclarant co-élaborer des ressources avec leurs étudiants. Les filières informatiques représentent 105 de ces enseignants, dont 19 citent le projet. On y trouve aussi « Chercher quelques jeux de données sur Internet sur une certaine thématique » ou encore la réutilisation de travaux d'étudiants en guise d'exemples. Concernant la collaboration avec les collègues, sur les 835 participants concernés, 197 représentent les spécialités informatiques. L'analyse qualitative des réponses fait ressortir une centration sur les collaborations entre équipes pédagogiques de proximité et de la même discipline, sauf dans le cadre des projets, où l'on observe quelques références rares à une collaboration autour de différentes matières : « Pour des ressources mobilisant différentes compétences (ex. informatique et gestion), nous élaborons des cas pratiques d'entreprises et nous déclinons via ces cas nos savoir-faire liés aux matières ». Enfin, la collaboration avec les partenaires est la moins représentée, mais sur les 406 participants concernés, 100 sont en informatique, où l'on trouve une prédominance de références au monde professionnel.

3) Réalisent-ils des adaptations des ressources existantes ?

À la suite de la question sur le type de ressources utilisées dans la préparation des enseignements, et sur les collaborations éventuelles, nous nous intéressons aux

éventuelles adaptations de ces ressources pour les rendre enseignables et utilisables par les étudiants. Cette question concerne d'abord les *contenus* (tableau 4 et 5) et ensuite les *supports* des cours.

Tableau 4. Adaptations des contenus, échantillon global

	Jamais		Rarement		Occasionnellement		Assez souvent		Très souvent	
	N	% cit.	N	% cit.	N	% cit.	N	% cit.	N	% cit.
adapter les contenus des programmes nationaux	**34**	**3,4%**	**123**	**12,2%**	**267**	**26,4%**	**339**	**33,5%**	**249**	**24,6%**
faire évoluer les contenus d'une année sur l'autre	**2**	**0,2%**	**53**	**5,2%**	**236**	**23,0%**	**398**	**38,8%**	**337**	**32,8%**
partager vos contenus de cours avec vos collègues enseignants	**67**	**6,7%**	**168**	**16,7%**	**280**	**27,9%**	**263**	**26,2%**	**227**	**22,6%**
Total	**103**	**3,4%**	**344**	**11,3%**	**783**	**25,7%**	**1000**	**32,9%**	**813**	**26,7%**

Tout d'abord nous notons que les réponses à la question de l'adaptation des contenus des programmes nationaux sont distribuées de manière attendue à tous les niveaux de l'échelle. La majorité des participants font évoluer les contenus de leurs cours d'une année à l'autre. Par contre, il ressort de manière très significative que le partage des contenus de cours avec les collègues enseignants est moins développé qu'attendu. Nous reproduisons dans le tableau 5 les résultats pour les enseignants dans les filières informatique, pour souligner les écarts par rapport à la population globale : moins d'adaptations des contenus des programmes nationaux, moins d'évolutions, mais davantage qu'attendu de partage des contenus.

Tableau 5. Adaptations des contenus, sous-population spécialités informatique

	Jamais		Rarement		Occasionnellement		Assez souvent		Très souvent	
	N	% cit.	N	% cit.	N	% cit.	N	% cit.	N	% cit.
adapter les contenus des programmes nationaux	**6**	**2,7%**	**32**	**14,2%**	**60**	**26,7%**	**63**	**28,0%**	**64**	**28,4%**
faire évoluer les contenus d'une année sur l'autre	**0**	**0,0%**	**14**	**6,1%**	**43**	**18,9%**	**88**	**38,6%**	**83**	**36,4%**
partager vos contenus de cours avec vos collègues enseignants	**9**	**4,1%**	**17**	**7,7%**	**46**	**20,8%**	**76**	**34,4%**	**73**	**33,0%**
Total	**15**	**2,2%**	**63**	**9,3%**	**149**	**22,1%**	**227**	**33,7%**	**220**	**32,6%**

Concernant maintenant les supports des cours, les représentants des filières en informatique présentent sensiblement les mêmes déclarations que celles de la population globale. L'ensemble des participants préfèrent créer eux-mêmes leurs supports de cours (75,4 %, très souvent), ils optent moins massivement pour la personnalisation de cours déjà existants et enfin ils déclarent très majoritairement ne jamais utiliser des supports déjà existants (54,3 %, jamais).

En résumé, nous identifions des pratiques d'adaptation des contenus de cours progressives dans le temps et des démarches plutôt personnelles de préparation des supports de cours.

4) Qu'est-ce qui détermine le choix des ressources ?

La compréhension de l'utilisation de ressources éducatives par les enseignants en IUT portait également sur les causalités déclarées de leurs choix, pour savoir s'ils sont personnels (choix pédagogiques, expérience) ou en référence à des facteurs externes (directives nationales, collègues).

Les résultats présentés dans le tableau 7 font apparaître que les choix sont très significativement (55,1 %, tout à fait d'accord) déterminés par des raisons d'adéquation aux projets pédagogiques personnels des enseignants et de leurs expériences. Les causes externes sont très peu déterminantes, même si les suggestions des collègues sont davantage accueillies que les directives du référentiel national. Notons pour ces deux causes externes que la position « neutre » sur l'échelle reçoit la majorité des réponses, dont nous ne pouvons interpréter le sens autrement qu'une volonté de s'abstenir de répondre.

Tableau 7. Déterminations du choix des ressources, échantillon global

	Pas du tout d'accord		Pas d'accord		Neutre		D'accord		Tout à fait d'accord		**Total**	
	N	% cit.	N	% cit.	N	% cit.	N	% cit.	N	% cit.	N	% cit.
vous estimez qu'elle est la plus adéquate à votre projet pédagogique	10	**1,0%**	18	**1,8%**	99	**9,9%**	324	**32,3%**	553	**55,1%**	**1004**	**100,0%**
vous la maîtrisez au regard de votre expérience	23	**2,3%**	40	**4,0%**	169	**17,0%**	423	**42,6%**	338	**34,0%**	**993**	**100,0%**
vous suivez les directives du référentiel national qui recommande son utilisation	232	**25,0%**	192	**20,7%**	274	**29,5%**	150	**16,1%**	81	**8,7%**	**929**	**100,0%**
vous suivez les suggestions de vos collègues	112	**11,9%**	194	**20,6%**	340	**36,1%**	241	**25,6%**	55	**5,8%**	**942**	**100,0%**
Total	**377**	**9,7%**	**444**	**11,5%**	**882**	**22,8%**	**1138**	**29,4%**	**1027**	**26,6%**	**3868**	

Les enseignants intervenant en informatique suivent le même profil de réponses, mais affirment encore plus (58,4 %, tout à fait d'accord) que leurs choix de ressources est déterminé par l'adéquation à leur projet pédagogique.

5) Contribuent-ils à l'élaboration de ressources pour des éditeurs ?

Parmi les participants à l'enquête 14 % (13,1 %, en informatique) déclare une participation à l'élaboration de ressources. La majorité travaille seul, mais la collaboration avec les collègues de leur propre IUT est occasionnellement et assez souvent présente. Dans une moindre mesure avec des collègues d'autres IUT et des professionnels. Enfin, sans surprise, la collaboration est très rare avec des étudiants. Le profil des enseignants en informatique une fois de plus ne se démarque pas des autres spécialités.

6) Quelle est la place de la recherche dans leur rapport aux ressources ?

La place de la recherche est peu présente dans les déclarations des près de 500 participants concernés statutairement par la recherche (enseignants-chercheurs), avec 34,2 % des participants qui ne fait jamais référence à ses recherches dans les cours écrits en précisant que cette référence est plutôt rare et occasionnelle. Ce n'est pas le cas pour les cours oraux, pour lesquels il ressort de manière significative que les enseignants y font référence aussi bien occasionnellement que très souvent, et de manière très significative assez souvent.

Auprès des enseignants en informatique l'absence de référence à leurs recherches dans les cours écrits monte à 40,9 % et reste une pratique plutôt occasionnelle à l'oral. 30 participants précisent leur réponse dans la question ouverte qui y était associée que la raison se situe majoritairement dans l'écart de niveau, comme en témoignent quelques exemples de verbatim au sujet du lien entre leurs recherches et leurs enseignements :

- « Le niveau Bac+2/+3 laisse peu de place aux recherches scientifiques si ce n'est de dire que telle techno est le fruit de la recherche ou qu'elle travaille sur tels aspects pour demain »
- « Il est très lointain. Mes enseignements relèvent pour moi de la vulgarisation scientifique. »
- « Difficile en raison de l'écart de niveau. Je ne peux rester qu'au niveau des applications, sans rentrer dans le détail. »
- « Ils sont totalement déconnectés, d'où la difficulté d'intégrer ma recherche dans mes enseignements. »
- « J'enseigne en DUT1 et en DUT2. C'est les bases du domaine. Intégrer ou même évoquer des choses pointues du domaine ne serait pas très compréhensible par les étudiants compte tenu de leur bagage. »

Plus rarement on trouve des témoignages portant sur l'actualisation des contenus de cours, comme celui « Les activités de recherche m'incitent à rester attentif aux évolutions du domaine sur lequel j'enseigne et à actualiser mon discours et mes supports de cours au fur et à mesure des années. »

Et en sens inverse : « Il y a des termes communs, mais ce n'est pas du tout le même niveau. L'enseignement aide à s'entraîner à rendre la recherche accessible au plus grand nombre. ». Ou les deux : « Je tente constamment d'enrichir mes enseignements par mon activité de recherche. Avoir un aperçu de cet aspect de mon travail, s'il est en lien avec le contenu du cours, intéresse toujours les étudiants. »

Il ressort de ces réponses que la position de l'enseignant-chercheur en IUT au regard de ses propres recherches est peu développée. Du fait de la prédominance de l'oralité comme support de communication pour faire du lien entre leurs recherches et leurs enseignements, ils ne sont que rarement pérennisés.

En conclusion au questionnaire une question ouverte invitait les participants à s'exprimer librement. Nous ne retenons que quelques exemples de verbatim émanant des enseignants en informatique, et qui recoupent les thématiques au cœur de notre enquête :

- « Il faut continuer à avoir une vision nationale et des adaptations locales. Il faut poursuivre le dialogue et les échanges permanents pour coller au terrain. » ;
- « Côté ressource, il y a beaucoup de possibilités pour construire un cours tant dans le fond que dans la forme. On peut explorer différentes voies lorsque l'équipe pédagogique est sur la même longueur d'onde. La difficulté, si difficulté y a, se pose plus en termes de changement de nos habitudes et de notre adaptabilité. Et ceci est vrai quel que soit l'environnement de formation (IUT, UFR...) » ;
- « Nous n'avons pas assez de temps, avec toutes les tâches qui nous sont imparties, pour faire fonctionner correctement un réseau d'enseignants qui mettraient des ressources à disposition. Pourtant, ce serait très judicieux. »

Enfin pour terminer notre présentation des résultats nous choisissons de reproduire in extenso, malgré sa longueur, le mot de la fin d'un enseignant du second degré affecté dans le supérieur (PRAG/PRCE), avec 10 à 15 ans d'expérience, spécialité informatique :

> « Concevoir des ressources offre une réelle grande conjonction de difficultés.
> - Quels programmes ? Quelles interactions entre les éléments le composant ? Pour Quels objectifs ?
> - Quelle va être l'analyse des programmes ? Va-t-on en avoir une approche conceptuelle, professionnelle, intermédiaire ?
> - Qui vont être les concepteurs et leurs orientations (universitaires, pédagogiques, professionnelle) ? Peuvent-ils travailler en équipe ? Ont-ils de la bonne humeur à revendre ?
> - Quel est le public visé ? Sont-ils déjà des étudiants ou encore ... ? Quel est leur "niveau" d'accueil ? Que veut-on qu'ils gardent du travail qu'ils vont mettre en œuvre (en termes de compétences ?
>   - ... et puis ce que je viens d'évoquer n'est un grand bla-bla
>   - L'enseignement par les ressources pédagogiques est en fait un grand maelstrom extrêmement difficile à gérer qui depuis le BUT ne cesse d'accélérer en mettant ses acteurs en difficultés. »

3.3 Analyse et discussion des résultats

Alors que l'organisation des IUT a suivi des processus encourageant une structuration forte en réseaux à un niveau national (Le Nir, 2015 ; Tralongo, 2018), les résultats de notre enquête ne permettent pas de vérifier un effet de cette structuration sur l'identification et l'utilisation des ressources éducatives. Nous constatons que les participants expriment majoritairement un rapport personnel aux ressources éducatives et que le rapport institutionnel n'est que peu exprimé. Les pratiques qui ressortent majoritairement attestent, d'une part, d'une importante diversité de types de ressources utilisés accessibles au grand public, notamment sur Internet. D'autre part, elles témoignent de démarches de conception de ressources plutôt personnelles et des pratiques collaboratives et de partage peu développées. Notre sous-population d'enseignants en informatique confirme ce profil et témoigne clairement de leur recours aux ressources disponibles sur Internet pour former *à* l'informatique. Cependant, malgré la possibilité offerte aux participants dans les multiples questions ouvertes proposées tout au long du questionnaire (au nombre de 48), de préciser et d'expliquer leurs choix de réponses aux questions fermées, peu de participants ont saisi l'occasion pour entrer dans le détail des questions que posent leurs contenus d'enseignement au regard des enjeux auxquels font face les formations en IUT, dans la société et dans l'environnement universitaire. Il en ressort que les contenus des enseignements et des ressources ne sont que peu discutés – que ce soit au niveau de l'échantillon global ou des enseignants en informatique –, phénomène qui confirme ce que Chevallard (2007) a qualifié de « déni de problématicité » du didactique, pour mettre en lumière un manque de sensibilité et de temps pris pour étudier les conditions et les contraintes qui pèsent sur la diffusion des contenus à enseigner. Cette sensibilité apparaît d'autant plus importante face aux exigences de

professionnalisation des formations, qui sont à l'origine de la création des IUT et qui se trouvent complexifiées dans le contexte récent des réformes en vue notamment d'une généralisation de l'approche par compétences (Tralongo, 2018).
Nous lisons dans nos résultats un début de sensibilité à ces questions qui interrogent les contenus et les praxéologies à enseigner, notamment au sujet de la place du projet tutoré, cité comme activité d'enseignement pour consolider les connaissances avec les étudiants, consolidation pouvant être renforcée par les liens faits par certains enseignants avec leurs recherches. Ces témoignages sont cependant très rares, sans doute faute de temps, et certainement la méthodologie de recherche de l'enquête par questionnaire ne favorise pas des témoignages plus approfondis, ce qui invite à développer le dispositif de recherche en introduisant d'autres méthodes de recueils de données pour approcher au mieux les enjeux du questionnement didactique.

4 Conclusion

L'enquête par questionnaire avait pour objectif de recueillir des témoignages à grande échelle sur la perception et les usages des enseignants en IUT des ressources éducatives – qu'elles soient des productions personnelles, des ressources proposées par l'IUT ou simplement en accès libre sur Internet ou via les canaux usuels de distribution des ressources éducatives. À partir de notre cadre théorique de référence en théorie anthropologique du didactique nous étudions l'importance de la prise en compte des rapports personnels face aux rapports institutionnels aux ressources éducatives dans l'étude de la diffusion des connaissances et des praxéologies dans notre société, et ici plus particulièrement grâce aux formations professionnalisantes des IUT.
Notre étude fait ressortir l'importance d'étudier les conditions et contraintes de la diffusion des connaissances et des praxéologies dans la société et dans les institutions en soulignant que les rapports personnels et institutionnels des enseignants aux ressources éducatives constituent un ensemble de conditions et contraintes qui méritent d'être davantage étudiés pour comprendre ce qui sera réellement enseigné en IUT. Il est important de préciser ici que le contexte des IUT ne constitue qu'un exemple de ce qui peut être observé pour d'autres formations, professionnelles ou universitaires et invite à poursuivre l'enquête pour étudier le rôle central des enseignants face à la question des ressources éducatives.

Références

Bruillard, É. : Understanding teacher activity with educational resources. Selection, creation, modification, use, discussion and sharing. In J. Rodrígez Rodrígez, T. M. Braga Garcia, & É. Bruillard, (Éds.), IARTEM 1991-2016: 25 years developing textbook and educational media research (pp. 343-352). Santiago de Compostela, Espagne: IARTEM, (2019).

Bruillard, E.: Ressources éducatives et travail des enseignants : pour des ressources numériques « vivantes » en éducation. Rapport interne STEF, présentation à l'IARTEM, (2013).

Chevallard, Y.: Passé et présent de la théorie anthropologique du didactique. In L. Ruiz-Higueras, A. Estepa, F. Javier García (ed.), Sociedad, Escuela y Mathemáticas. Aportaciones de la Teoría Antropológica de la Didáctico, Universidad de Jaén, 705-746. (2007).

Le Nir, M. : Évaluation des instituts universitaires de technologie et de leurs départements : retour sur 15 années d'expérience, RIPES, 31-3, (2015).

Tralongo, S. : Ce qui fait et ceux qui font la professionnalisation en IUT, Cahiers de la recherche sur l'éducation et les savoirs, Hors-série n° 6, http://journals.openedition.org/cres/3178 (2018).

Enseignement de l'accessibilité web et ressources pédagogiques. Le cas de la formation « Métiers du multimédia et internet » en IUT.

Béatrice Drot-Delange[1]

[1] ACTé, Université Clermont Auvergne
beatrice.drot-delange@uca.fr

Résumé. L'accessibilité, enjeu sociétal et obligation légale, est encore très peu mise en œuvre sur les sites web. La formation des développeurs web est un des leviers pour améliorer cet état de fait. L'article analyse le curriculum et les ressources pédagogiques prescrites dans la filière Métiers du multimédia et internet dans les Instituts universitaires de technologie en France. A l'instar des recherches, encore rares dans le champ de l'enseignement du développement web, l'analyse des prescriptions montre la primauté donnée aux règles WCAG, sans pour autant que cette approche n'ait montré son efficacité dans la construction par les apprenants d'une meilleure compréhension de l'accessibilité. Cette question reste un enjeu de recherche pour une didactique du web.

Mots clés : enseignement supérieur technologique, didactique, informatique, curriculum prescrit, ressources.

1 Introduction

Alors que le web est très présent dans nos vies au quotidien, les questions relatives à l'enseignement de son développement sont quasi-absentes des recherches en sciences de l'éducation (Smith, 2020) et plus encore en didactique (Drot-Delange, 2016). Le développement web étant un domaine particulièrement vaste (Connolly, 2019), nous avons choisi de focaliser cette communication sur la question de l'accessibilité.

L'accessibilité du web est en enjeu sociétal et une obligation légale. La loi de 2005 en France fixait comme objectif que l'accessibilité ne soit plus un obstacle dès 2011. Cet objectif n'est toujours pas atteint en 2022, avec seulement 13% des sites web accessibles (Vall, 2020).

L'un des facteurs explicatifs de cette situation pourrait être la formation, ou l'absence de formation, des développeurs web. Dans une enquête internationale publiée en 2021[1], les praticiens de l'accessibilité déclarent avoir suivi une formation à l'accessibilité pour 12,5% des répondants. Ce chiffre est en légère augmentation, puisqu'en 2018 ils étaient 5,5%. Des enquêtes menées auprès de webmasters citent d'autres facteurs explicatifs par exemple le manque de temps, de support de la part de l'entreprise, de la

[1] https://webaim.org/projects/practitionersurvey3/. Consulté le 19 janvier 2022

part du client, des logiciels inadéquats, des confusions dans les guides existants (Lazar, Dudley-Sponaugle et Greenidge, 2004).

Parmi les formations diplômant des développeurs web, nous nous intéressons plus particulièrement à la filière « Métiers du multimédia et de l'Internet » (MMI) des Instituts universitaires de technologie (IUT). Cette filière existe depuis 2013 en France. Elle a remplacé la filière « Service réseaux et communication (SRC) », créée en 1993, suite au développement d'Internet.

La réforme des IUT mise en œuvre à la rentrée 2021 instaure une formation sur 3 années et la mise en œuvre d'une approche par compétences. Dans le cadre de cette réforme, la filière MMI propose désormais 3 parcours, « Stratégie de communication numérique et design d'expérience », « Création numérique » et « Développement web et dispositifs interactifs ».

L'accessibilité est très présente dans le curriculum prescrit de ces 3 parcours (MESRI, 2021a), même si seule la première année du programme national du Bachelor universitaire de technologie (BUT) MMI est disponible au moment de l'écriture de cette communication. A titre de comparaison, le programme national du BUT informatique (MESRI, 2021b) n'aborde pas cette question en première année.

Dans cette communication, nous nous posons la question de savoir comment le curriculum prescrit en MMI intègre la question de l'accessibilité web. Dans une première partie, nous mènerons une analyse des contenus, des méthodes, des ressources prescrits le cas échéant dans le programme national. Dans une seconde partie, nous discuterons les résultats obtenus au regard de la littérature.

2 Curriculum prescrit et accessibilité en MMI

Le programme national en MMI (MESRI, 2020a) identifie des compétences qui se déclinent en composantes essentielles. A chaque niveau de développement d'une compétence correspond des apprentissages critiques. Le programme national décline ensuite des « situations d'apprentissage et d'évaluation » (SAE), définies comme « permettant l'évaluation en situation de la compétence » (p.17) et « répond[ant] à une problématique que l'on trouve en milieu professionnel », ainsi que des « ressources » (notées R dans la suite du texte). Les ressources sont définies dans ce programme comme « les savoirs, savoir-faire et savoir-être dont dispose un individu et qui lui permettent de mettre en œuvre la compétence » (p.6).

L'accessibilité est présente à de nombreuses reprises dans le programme national de MMI. Elle figure en effet dès les objectifs de la formation qui précisent que les étudiants sont sensibilisés à « l'accessibilité numérique au plus grand nombre notamment au travers du respect des normes et de standards de qualité » (p. 5). L'une des 5 compétences visées par la formation, « développer pour le web et les médias numériques », inclut la composante essentielle « en se conformant aux standards du web et aux normes d'accessibilité ». Pour autant, aucun des apprentissages critiques correspondant à cette compétence ne renvoie directement à l'accessibilité, sauf à considérer cet intitulé : « Évaluer un site web, un produit multimédia ou un dispositif interactif existant en s'appuyant

sur des guides de bonnes pratiques » (p. 11). Le tableau 1 récapitule les SAE et les ressources du programme pour lesquelles l'accessibilité est mentionnée.

Tableau 1. SAE et Ressources mentionnant l'accessibilité

Référence	Intitulé	Extrait du corpus
SAE 101	Audit communication numérique	« Cette SAE peut être l'occasion d'explorer des outils d'analyse de trafic, de sondages d'opinion pour obtenir des données à analyser ou des référentiels de qualité concernant l'ergonomie et l'accessibilité Web. » « Un des livrables possibles est l'audit d'accessibilité ou d'ergonomie. »
SAE 105	Produire un site web	« Produire un site Web contenant à la fois des pages statiques et des pages générées à partir de jeux de données structurées, respectant les normes du W3C et les recommandations du WCAG. »
R 103	Ergonomie et accessibilité	
R 112	Intégration	« Les bases de l'accessibilité : une partie de la norme WCAG »
SAE 201	Exploration des usages	*La SAE mobilise la ressource 203 (voir ci-après)*
SAE 202	Concevoir un produit ou un service de communication	« Projet intégrateur de fin de première année ». *La ressource R 203 n'est pas mentionnée dans les ressources mobilisées et combinées.*
SAE 203	Site web et bases de données	« Le site proposera une mise en page respectant l'accessibilité et le W3C, mais aussi des interactions. »
R 203	Ergonomie et accessibilité	« Introduction à l'accessibilité numérique (enjeux humains, juridiques et techniques, à quelles étapes d'un projet...) - Introduction à des référentiels de qualité, comprendre la norme internationale WCAG - Méthodes d'audit de l'accessibilité d'un site ou d'un service (sur les plans fonctionnel, graphique et multimédia) »
R 212	Intégration	« Accessibilité : une partie de la norme WCAG »

Le programme renvoie donc principalement aux recommandations du W3C concernant les règles pour l'accessibilité des contenus web (WCAG)[2]. Ces règles visent à faire respecter les 4 grands principes de perceptibilité, d'utilisabilité, de compréhension et de robustesse. Ces principes se déclinent en 12 règles accompagnées de critères de succès. On peut noter que d'autres référentiels sont disponibles en France, tel que le RGAA (Référentiel général d'amélioration de l'accessibilité)[3] créé pour mettre en œuvre l'article 47 de la loi handicap de 2005 et son décret d'application actualisé en 2019.

La lecture de ce programme sur la question de l'accessibilité fait se poser des questions sur la planification entre les deux semestres de l'année. Par exemple, la SAE 105 « Produire un site web », au premier semestre, mobilise les recommandations WCAG, mais celles-ci ne sont qu'en partie abordées par la ressource R112 « Intégration », au premier semestre, puis par la ressource R212 « Intégration » au second semestre. On peut s'interroger sur la manière dont il conviendrait de scinder la norme WCAG en plusieurs parties. De plus, la ressource R103 « Ergonomie et accessibilité » n'est pas mentionnée dans cette SAE, mais le descriptif détaillé n'aborde pas d'item relevant de l'accessibilité.

De même, un audit d'accessibilité est indiqué comme possible dans la SAE 101 « Audit communication numérique » au premier semestre mais les méthodes d'audit ne sont mentionnées qu'au second semestre au sein de la ressource R203 « Ergonomie et accessibilité ».

Les conditions d'écriture, exceptionnellement contraintes temporellement, de ces programmes, dans le cadre de la réforme, expliquent probablement ce qui peut apparaitre aux yeux du lecteur comme des incohérences. Il n'en reste pas moins que le programme national, curriculum prescrit, ne donne pas d'indications sur la manière dont les enseignants vont pouvoir décliner leur enseignement sur cette question.

3 Discussion

Dans le programme de MMI, l'accessibilité fait l'objet d'un cours et est abordée avec l'ergonomie. Elle est également référencée dans les situations d'apprentissage et d'évaluation. Baker, El-Galy et Shinohara (2020) soulignent que la littérature ne permet pas vraiment de connaitre quels sont les aspects de l'accessibilité qui sont couverts par les curricula, les pédagogies communément utilisées et comment sont évalués les étudiants.

A l'instar de l'analyse des publications relatives aux curricula en informatique et la place de l'accessibilité menée par Baker, El-Galy et Shinohara (2020), on peut considérer que le programme de MMI couvre des connaissances techniques (les règles) et une sensibilisation à l'accessibilité.

Dans les curricula, l'accessibilité est le plus souvent abordée par la référence aux règles WCAG, à l'instar du programme de MMI. Comme le soulignent Ferati et Vogel (2020), ces règles ne sont qu'une partie des ressources proposées par le W3C pour l'accessibilité, les autres étant des logiciels et des services. Certains logiciels permettent

[2] https://www.w3.org/Translations/WCAG20-fr/ consulté le 19 janvier 2022

[3] https://www.numerique.gouv.fr/publications/rgaa-accessibilite/ consulté le 19 janvier 2022

par exemple de détecter automatiquement des erreurs en matière d'accessibilité dans le code, contrôles qui seraient particulièrement fastidieux, voire impossible, à réaliser manuellement à partir de la liste des règles. D'autres outils proposent un guide pas-à-pas de l'évaluation.

On peut considérer que ces logiciels et outils sont des ressources potentielles pour l'enseignement de l'accessibilité, mais ne sont pas mentionnés explicitement dans le programme de MMI. Toutefois, l'usage de ces outils est principalement rétrospectif, destiné à l'évaluation, et n'aide pas le concepteur dès le début du projet (Crabb et al., 2019). Ces auteurs relèvent le besoin de ressources pour concevoir et pas seulement pour évaluer les sites web.

Plus généralement, notre étude pose la question des ressources éducatives et des approches pédagogiques qui pourraient favoriser la prise en compte de l'accessibilité dans la formation des concepteurs web. Les recherches dans le domaine sont encore rares. Nous citons quelques résultats d'expérimentations qui pourraient être mises à l'épreuve dans le cas de la formation en IUT.

Baker, El-Galy et Shinohara (2020) prônent la production et l'amélioration des ressources disponibles pour enseigner l'accessibilité. En effet, si les enseignants mobilisent des guides et des outils professionnels dans leurs cours, ceux-ci peuvent être difficiles à comprendre pour des étudiants novices. On pourrait alors avoir un effet contre-productif, voire désincitatif. Les outils existants nécessitent parfois d'être très informé sur l'accessibilité pour interpréter les résultats qu'ils fournissent et juger de la pertinence des propositions ou corrections faites.

Concernant les méthodes pédagogiques, Kawas, Vonessen et Ko (2019) proposent quatre manières d'aborder l'accessibilité dans les curricula : par un cours dédié, des exemples, un contexte et des problèmes motivants. Comme l'indiquent Baker, El-Galy et Shinohara (2020), la littérature se concentre surtout sur la première modalité et produit peu ou pas de résultats sur les autres.

Pour autant, des auteurs proposent des démarches pour aider à mieux prendre en compte les besoins en termes d'accessibilité. Ainsi Motti et Dura (2021) ont proposé des interventions dans des cours auprès de 134 étudiants. Ces interventions sont basées sur l'usage d'un kit de 16 cartes persona, chacun présentant une déficience. L'usage de ces persona motiverait les étudiants à adopter les règles d'accessibilité. Les résultats de l'enquête menée auprès des enseignants après ces interventions montrent l'intérêt de la démarche. Ces interventions ont augmenté l'intérêt et la motivation des étudiants pour appliquer les principes d'accessibilité dans leur travail. Elles permettraient également de répondre au manque de connaissances chez les développeurs des besoins des utilisateurs, mises en évidence par la recherche de Crabb et al. (2019), par une sensibilisation aux difficultés rencontrées par les utilisateurs.

Les ateliers menés par ces chercheurs auprès de 197 développeurs (étudiants et professionnels) apportent des éclairages sur les lacunes dans leurs connaissances des défis posés par l'accessibilité en situation. Par exemple, pour l'accessibilité visuelle, les développeurs évoquent la luminosité, mais très rarement le contraste, la saturation, etc. De même, ils constatent dans leur enquête le manque de connaissances chez les développeurs concernant la manière dont une personne handicapée utilise les technologies,

alors même que cela devrait influer sur la conception des interactions avec les technologies.

4 Conclusion

Notre analyse du programme national de MMI montre que celui-ci accorde une place importante à l'accessibilité. Toutefois, les prescriptions n'indiquent pas explicitement comment enseigner cette question. La principale modalité prescrite est celle de l'usage des guides WCAG et leur mise en œuvre au sein de situations d'apprentissage.

Or ces règles, mais aussi les logiciels, outils et méthodes ne sont pas des ressources conçues pour l'enseignement, mais des ressources mobilisées par les professionnels du champ. Ces ressources, leur usage, leur appropriation, leur compréhension peuvent être difficiles pour des non-initiés. On peut s'interroger pour savoir si la seule entrée par les normes et règles permet d'atteindre les objectifs de professionnalisation attendus, à savoir concevoir une production numérique prenant en compte les besoins d'accessibilité des publics, dans leur diversité et en situation. Cela pose la question de l'efficacité de ces démarches mais aussi celle de l'évaluation des étudiants.

Des recherches dans le champ de l'enseignement de l'informatique sont menées pour proposer de nouvelles modalités pédagogiques pour aborder ces questions, mais elles restent embryonnaires. La question de savoir comment intégrer ces propositions dans le curriculum réel dans le cas de la filière MMI en IUT reste posée. La didactique de l'informatique, et plus spécifiquement la didactique du web, gagnerait à s'en emparer.

Contribution

Cette recherche bénéficie d'un financement de l'Agence Nationale de la Recherche dans le cadre de la convention ANR-18-CE38-0011 pour le projet RENOIR-IUT (Ressources numériques : offre, intermédiations, réseaux en Institut universitaire de technologie).

Références

Baker, C. M., El-Glaly, Y. N., & Shinohara, K. (2020). A Systematic Analysis of Accessibility in Computing Education Research. In Proceedings of the 51st *ACM Technical Symposium on Computer Science Education* (p. 107-113). Association for Computing Machinery. https://doi.org/10.1145/3328778.3366843

Connolly, R. (2019). Facing Backwards While Stumbling Forwards : The Future of Teaching Web Development. Proceedings of the 50th *ACM Technical Symposium on Computer Science Education*, 518-523. https://doi.org/10.1145/3287324.3287433

Crabb, M., Heron, M., Jones, R., Armstrong, M., Reid, H., & Wilson, A. (2019). Developing Accessible Services : Understanding Current Knowledge and Areas for Future Support. In Proceedings of the 2019 *CHI Conference on Human Factors in*

Computing Systems (p. 1-12). Association for Computing Machinery. https://doi.org/10.1145/3290605.3300446

Drot-Delange, B. (2016). *Internet en éducation : Objets de savoirs* [Habilitation à diriger des recherches, Université Blaise Pascal - Clermont II]. https://tel.archives-ouvertes.fr/tel-02870504

Ferati, M., & Vogel, B. (2020). Accessibility in Web Development Courses : A Case Study. *Informatics*, 7(1), 8. https://doi.org/10.3390/informatics7010008

Kawas, S., Vonessen, L., & Ko, A. J. (2019, February). Teaching accessibility: A design exploration of faculty professional development at scale. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (pp. 983-989).

Lazar, J., Dudley-Sponaugle, A., & Greenidge, K.-D. (2004). Improving web accessibility : A study of webmaster perceptions. *Computers in human behavior*, 20(2), 269-288.

MESRI (2021a). Licence professionnelle « Bachelor universitaire de technologie » Métiers du multimédia et de l'internet.

MESRI (2021b). Licence professionnelle « Bachelor universitaire de technologie » Informatique.

Motti, V., & Dura, E. (2021). Evaluating an accessibility intervention based on persona cards with diverse needs to teach accessibility to undergraduate students. *Universal Access in the Information Society*, 1-17.

Smith, J. (2020). Teaching Web Development : A Literature Review. *EdMedia + Innovate Learning 2020*. Netherlands, June 23-26, 2020. 310-314. https://www.learntechlib.org/primary/p/217316/

Vall, R. (2020). *Rapport d'information fait au Sénat au nom de la mission d'information sur la lutte contre l'illectronisme et pour l'inclusion numérique sur la lutte contre l'illectronisme et pour l'inclusion numérique*.

Possibilités d'agir d'un environnement d'apprentissage de la programmation informatique à l'école primaire

Marie Valorge[1]

[1] Éducation, Cultures, Politiques (EA 4571), Université Lumière Lyon 2, France

Résumé. Cet article présente une étude conduite dans une classe de CE1 qui s'initie à la programmation informatique dans le cadre d'une expérimentation où les enseignantes et enseignants ont conçu les scénarios et supports pédagogiques. Pour cette initiation, les élèves utilisent le robot pédagogique Ozobot, suivant une démarche d'investigation, en petit groupe et en autonomie. L'analyse de l'activité d'un groupe de deux élèves s'appuie sur une méthode d'analyse de la tâche issue du champ de l'ergonomie et suivant une approche écologique. Elle permet de relever les conditions qui permettent comme celles qui limitent l'activité des écolières et écoliers, à partir des notions d'*information* et d'*affordance*. Nos résultats préliminaires, axés sur la description exhaustive de l'environnement d'apprentissage et sur les possibilités d'agir offertes aux élèves par cet environnement, questionnent la pertinence d'un enseignement qui repose sur la découverte des élèves par l'utilisation d'outils complexes et dans une classe en difficulté scolaire. La méthode mobilisée est aussi interrogée, par rapport à ses possibilités d'adaptation aux particularités de notre contexte.

Mots-clés : Initiation à la programmation informatique, Robotique pédagogique, École primaire.

1 Introduction

Notre recherche est conduite sur le terrain de l'expérimentation Programmation du robot à l'école primaire (PREP)[1] et en convention avec l'éditeur public de ressources pour la formation des enseignants Réseau Canopé[2]. Elle a pour objet de comprendre comment les élèves du cycle 2 de l'école primaire s'initient à la programmation informatique avec des robots programmables et vise à contribuer à la conception de méthodes qui soient pertinentes pour ces apprentissages comme pour leur enseignement.

Afin d'améliorer les situations de travail, l'ergonomie a développé des outils pour comprendre ces situations. Dans ce domaine, « analyser l'activité à partir de la tâche »

[1] Le projet PREP est une expérimentation conduite entre 2018 et 2022 dans 7 écoles primaires par le laboratoire Éducation, Cultures, Politiques et avec le soutien de la Région Auvergne-Rhône-Alpes.

[2] Notre recherche doctorale est réalisée dans le cadre d'une convention CIFRE établie entre l'Université Lumière Lyon 2 et l'opérateur de l'État Réseau Canopé.

consiste dans un premier temps à définir les conditions externes de l'activité (tâches prescrites et à réaliser) puis dans un second temps à décrire l'activité du sujet (tâches redéfinies et effectives) [5]. Pour analyser des situations d'activité dans le contexte du projet PREP, nous nous appuyons sur la méthode écologique d'analyse de la tâche de la machine de Turing (Turing machine task analysis – TMTA) [9][10][11].

Dans cet article, nous présentons les ancrages théoriques de notre étude en nous arrêtant plus particulièrement sur la méthode mobilisée. Nous exposons ensuite nos questions de recherche, notre méthodologie puis les résultats d'une étude de cas conduite à partir de l'observation filmée d'une séquence d'initiation à la programmation mise en œuvre dans une classe de CE1. Nous concluons en énonçant nos perspectives de recherche, d'un point de vue méthodologique.

2 Ancrages théoriques

2.1 Les origines des concepts d'information et d'affordance et leur rôle dans le développement du sujet

Se donnant pour objectif de comprendre la perception, le psychologue James Gibson [2][3] est à l'origine du concept d'affordance. Selon une perspective écologique, chaque « animal » évolue dans un environnement spécifique et l'information qui émane des composants de cet environnement lui donne à percevoir des possibilités d'agir ou, inversement, le retient d'agir, constituant ainsi un « ensemble d'invites » ou affordances. Selon Eleanor Gibson et Anne Pick [4], une affordance est une propriété objective de l'environnement qui peut être perçue, non perçue, réalisée ou non réalisée. Pour être perçue et réalisée, c'est-à-dire rendre l'environnement *affordant* et donc propice à l'activité, l'information qui en provient doit être à la fois suffisante et signifiante pour le sujet. Le développement de l'enfant repose alors sur ce « prélèvement de l'information » dans l'environnement qui peut être « exploratoire », et ainsi permettre l'acquisition de nouvelles connaissances, ou « exécutoire », et alors confirmer une relation sujet-environnement déjà acquise. Pour saisir la relation des enfants aux objets d'un point de vue écologique, Michael Tomasello [15] propose de distinguer les « objets naturels » des « artefacts matériels » et, suivant Lev Vygotski [19], des « artefacts symboliques » ; les artefacts matériels sont à la fois dotés d'affordances sensori-motrice (comme les objets naturels) et d'autres affordances « intentionnelles » ou « idéelles » (comme les artefacts symboliques).

2.2 L'analyse de l'activité à partir de la tâche dans une perspective écologique

Pour la conception, le concept d'affordance a été développé par Donald Norman [12], afin que l'utilisateur perçoive immédiatement les fonctionnalités de nouveaux objets ; l'analyse de l'activité des utilisateurs est alors un précepte à la conception [13].

Selon Thierry Morineau [8], analyser l'activité à partir de la tâche dans une perspective écologique vise à déterminer « la nature des sollicitations offertes » par l'environnement et de mesurer « leur valeur adaptative » pour le sujet ; il s'agit de considérer l'activité comme « un espace de possibles dans lequel l'opérateur va

naviguer, en mettant en œuvre des stratégies opératoires et des apprentissages ». Dans ce sens, Morineau et ses collègues [9] et Morineau [10][11] ont conçu le modèle TMTA sur la base de deux cadres d'analyses préexistants pour :

- Définir les possibilités d'agir offertes par l'environnement,
- Comprendre l'activité du sujet à partir des stratégies et connaissances qu'il met en œuvre pour réaliser les tâches et selon la qualité de l'information disponible.

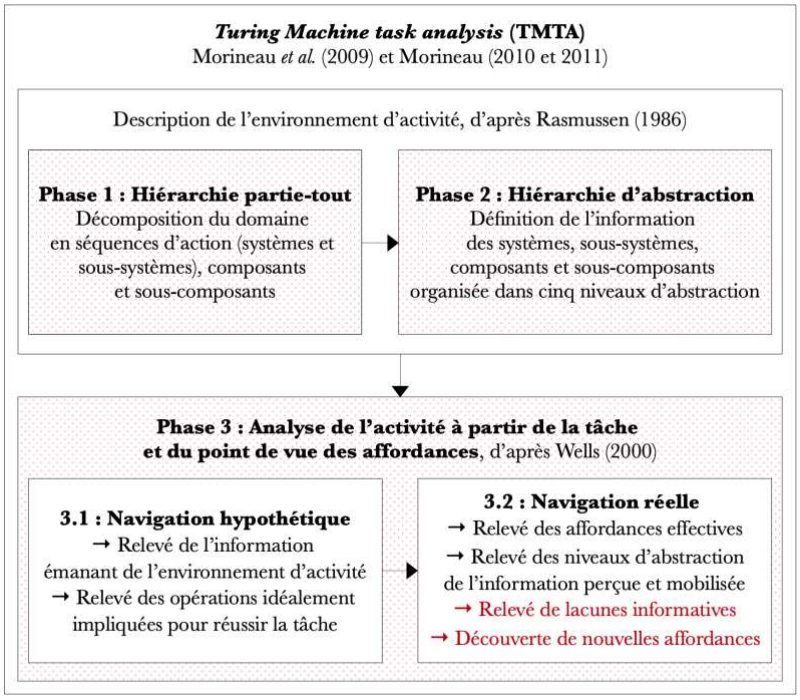


Fig. 1. Phases de notre analyse d'après le modèle d'analyse de la tâche TMTA.

Définir les possibilités d'agir de l'environnement

La TMTA utilise d'une part le modèle de conception d'interface écologique (*ecological interface design* – EID) où, en ingénierie et pour la conception d'interfaces de systèmes homme-machine complexes, Kim J. Vicente et Jens Rasmussen [17][18] ont proposé de décrire l'environnement de l'activité de façon exhaustive, comme un ensemble d'affordances. Morineau et ses collègues [9] ont redéfini les affordances de ce modèle comme de l'« information spécifiant une affordance », puisque l'information qui émane des composants de l'environnement peut ne pas être mobilisée ou même ne pas être pertinente pour l'activité du sujet. Suivant ce premier cadre, l'information est organisée dans une matrice « hiérarchie partie-tout/hiérarchie d'abstraction » :

- L'environnement est d'abord décomposé en séquences d'action (aussi dénommées systèmes et sous-systèmes ou tâches et sous-tâches), en composants et sous-composants (qui sont les éléments disponibles pour l'accomplissement des tâches) (**Fig. 1**, Phase 1).
- Chaque séquence et composant est ensuite décrit en tant qu'information à cinq « niveaux d'abstraction », dans l'objectif de considérer sa perception depuis différentes perspectives (selon ses « formes et apparences », au niveau le plus concret de la hiérarchie, jusqu'à ses « objectifs fonctionnels », au niveau maximal d'abstraction) (**Fig. 1**, Phase 2 ; **Fig. 2**).

Niveaux d'abstraction	Composant « enseignant·e »
1. Objectifs fonctionnels : Compétences des programmes scolaires	**Représenter l'espace :** Situer des objets par rapport à soi, entre eux, par rapport à des objets repères **Concepts et outils des démarches scientifiques :** Connaissance du fonctionnement d'un certain nombre d'objets et de systèmes – Curiosité, envie de se poser des questions, de chercher des réponses **Langages :** S'exprimer en utilisant la langue française à l'oral
2. Fonctions abstraites : Notions ou savoirs à acquérir	Marqueurs spatiaux « droite » et « gauche » Notion technologique de fonctionnement Notion informatique d'instruction Formulation d'un questionnement
3. Fonctions généralisées ou processus : Opérations proposées par le composant à l'élève	L'enseignant·e : - invite les élèves à poser des questions - les aide à recentrer sur le champ : donner des commandes à un robot, réaliser des défis - et à améliorer leur expression orale - rappelle les connaissances acquises qui sont nécessaires pour l'activité : mise en marche du robot, marqueurs spatiaux
4. Fonctions physiques : Sous-composants perceptibles/actionnables par l'élève	L'enseignant·e : - interroge les élèves - leur demande de reformuler leurs réponses
5. Formes et apparences : Informations visuelles, sonores ou kinesthésiques du composant	L'enseignant·e : - parle à voix haute - se déplace dans la classe puis se place devant le tableau - tourne sur lui/elle-même pour indiquer les directions droite et gauche

Fig. 2. Extrait de la matrice hiérarchie partie-tout/hiérarchie d'abstraction représentant l'environnement d'activité : Introduction de la première séance, composant « enseignant ».

Relever les stratégies et connaissances mises en œuvre et évaluer l'information

La méthode TMTA s'appuie d'autre part sur le formalisme des machines de Turing de Wells [20]. Sa mobilisation dans la TMTA [9][10][11] consiste à concevoir un scénario hypothétique décrivant l'activité d'un sujet, tâche par tâche, au sein de la matrice hiérarchie partie tout/hiérarchie d'abstraction définie (**Fig. 1**, Phase 3.1 ; **Fig. 3**), puis à éprouver ce scénario idéal à l'aune de l'activité réelle (**Fig. 1**, Phase 3.2 ; **Fig. 4**).

Pour relever les stratégies et connaissances mobilisées par le sujet, il s'agit de :

— Scénariser l'activité à partir des « transitions » effectuées par le sujet pour passer d'une tâche à l'autre, qui peuvent être un « mouvement » dans ou une « transformation » de l'environnement (observés), un « changement d'état mental » (supposé) ;
— Préciser le niveau où se situe l'information prélevée pour la réalisation de chaque tâche, suivant les cinq « niveaux d'abstraction » de la « matrice hiérarchie partie tout/hiérarchie d'abstraction » précédemment définie.

L'information émanant de l'environnement est quant à elle évaluée selon :

— Qu'elle est mobilisée pour l'effectuation de la tâche (*affordante*) ;
— Qu'elle est lacunaire : non mobilisée malgré sa pertinence ou non pertinente.

Nous précisons notre *évaluation de l'information* par rapport à celle de la TMTA dans nos résultats (*Cf.* 5.1, p. 9).

3 Questions de recherche

L'analyse de la tâche écologique propose donc d'analyser l'activité dans des environnements « sociotechniques complexes » [1], telles que le sont les situations d'apprentissage du terrain PREP, alors qu'elle peut être clairement circonscrite et décomposée en suites d'opérations. La structure des scénarios pédagogiques, préalablement conçus en collaboration par les enseignants engagés dans le projet PREP, nous permet de délimiter la situation en autant d'étapes avec des objectifs donnés ainsi que d'identifier les artefacts mis à disposition des élèves pour atteindre ces objectifs.

Suivant cette perspective, les trois questions de recherche qui guident notre étude empirique sont les suivantes :

— Quels composants de l'environnement d'apprentissage (artefacts matériels et symboliques, propriétés de ces artefacts, autres propriétés de l'environnement) sont proposés aux élèves pour apprendre la programmation informatique ?
— Lesquels de ces composants, organisés selon quelles configurations et en lien à quelles caractéristiques de l'activité des élèves, leur offrent d'acquérir ou de conforter des connaissances ?
— Lesquels autres font, éventuellement, obstacles à ces acquisitions ?

4 Méthodologie

4.1 La séquence d'initiation à la programmation informatique observée

Les observations dont sont issus les résultats présentés dans cet article ont été menées dans une classe à 12 élèves de CE1 d'une école située dans un réseau d'éducation prioritaire (REP+). La scénarisation pédagogique est mise en œuvre sur 4 séances d'approximativement quarante-cinq minutes. Les élèves de CE1 s'y initient progressivement à la programmation informatique en observant d'abord et en

programmant ensuite les déplacements du robot pédagogique Ozobot sur différents tracés imprimés sur des fiches plastifiées. Ces différents supports d'activité proposent aux élèves de suivre une démarche d'investigation, en petit groupe et en autonomie.

Le groupe observé est composé de deux élèves qui sont « bons », selon leur enseignant, dans une classe au niveau général plutôt en-dessous des attentes du programme, notamment à cause de la fermeture des écoles au printemps 2020.

4.2 Recueil et traitement des données

L'ensemble de la séquence (3 heures) a été enregistrée par une caméra fixe placée en hauteur et focalisée sur la table de travail du groupe et par un enregistreur audio qui permet de distinguer les échanges tenus entre les deux élèves. Les enregistrements ont été montés puis transcrits et analysés suivant deux réductions de données [16].

Séquence « Ozobot », CE1
Séance 1, sous-système 1.3 : Bilan intermédiaire en classe entière

L'élève explicite à l'enseignant·e et à la classe :

- son résultat [information pertinente = 1]
- la procédure suivie pour l'obtenir [information pertinente = 1]

Niveau dans la hiérarchie d'abstraction : Processus (3)

Transition : Changement d'état mental – Compréhension → Quelles connaissances ont été acquises lors de cette expérimentation ?

L'élève écoute l'enseignant·e qui l'amène à :

- confirmer la notion informatique d'« instruction » [information pertinente = 1]
- découvrir la notion informatique de condition [information pertinente = 1]

Niveau dans la hiérarchie d'abstraction : Fonctions abstraites (2)

Transition : Changement d'état mental – Planification → Comment formuler notre résultat en termes d'informatique ?

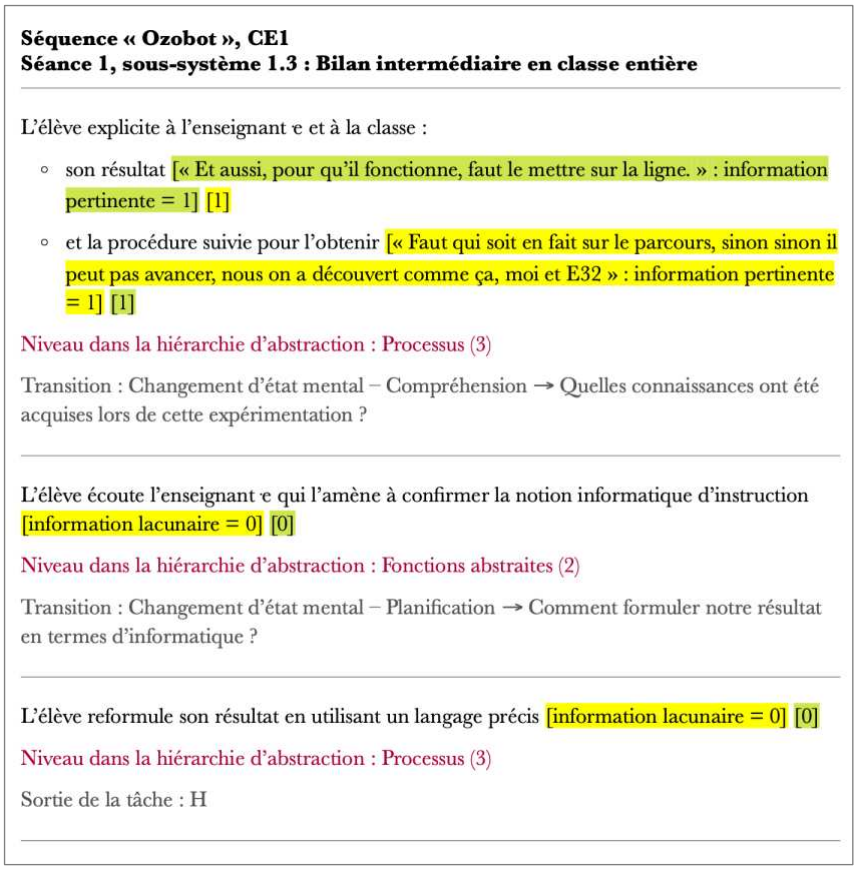
L'élève reformule son résultat en utilisant un langage précis [information pertinente = 1]

Niveau dans la hiérarchie d'abstraction : Processus (3)

Sortie de la tâche : H

Fig. 3. Extrait de la navigation hypothétique : Première séance, bilan intermédiaire en classe entière. Seule l'information pertinente est retenue pour cette description où l'élève accompli chaque tâche prescrite. Le niveau d'abstraction est précisé en référence à la matrice hiérarchie partie-tout/hiérarchie d'abstraction (en rouge) ainsi que l'opération de transition (en gris).

La première réduction des données s'est arrêtée sur les interventions du professeur et sur celles des autres élèves de la classe comme autant d'événements qui constituent l'environnement d'apprentissage. Suivant la TMTA, elle vise à préciser l'environnement particulier de l'activité qui avait été précédemment défini de façon plus générale, à partir du scénario pédagogique de la séance et en référence aux programmes scolaires. Un entretien conduit avec l'enseignant après la séquence a aussi complété cette description exhaustive dans une matrice d'information hiérarchie partie-tout/hiérarchie d'abstraction suivant les premières phases de l'analyse de la TMTA (**Fig. 1**, Phases 1 et 2 ; **Fig. 2**). Cette matrice sert l'écriture du scénario hypothétique de cheminement des élèves dans l'environnement (**Fig. 3**).



Séquence « Ozobot », CE1
Séance 1, sous-système 1.3 : Bilan intermédiaire en classe entière

L'élève explicite à l'enseignant·e et à la classe :

- son résultat [« Et aussi, pour qu'il fonctionne, faut le mettre sur la ligne. » : information pertinente = 1] [1]
- et la procédure suivie pour l'obtenir [« Faut qui soit en fait sur le parcours, sinon sinon il peut pas avancer, nous on a découvert comme ça, moi et E32 » : information pertinente = 1] [1]

Niveau dans la hiérarchie d'abstraction : Processus (3)

Transition : Changement d'état mental – Compréhension → Quelles connaissances ont été acquises lors de cette expérimentation ?

L'élève écoute l'enseignant·e qui l'amène à confirmer la notion informatique d'instruction [information lacunaire = 0] [0]

Niveau dans la hiérarchie d'abstraction : Fonctions abstraites (2)

Transition : Changement d'état mental – Planification → Comment formuler notre résultat en termes d'informatique ?

L'élève reformule son résultat en utilisant un langage précis [information lacunaire = 0] [0]

Niveau dans la hiérarchie d'abstraction : Processus (3)

Sortie de la tâche : H

Fig. 4. Extrait de la navigation réelle de notre groupe de deux élèves : Première séance, bilan intermédiaire en classe entière. Suivant nos premières analyses, nous considérons l'information non mobilisée ou non pertinente comme lacunaire.

La deuxième réduction des données a été focalisée sur l'activité réelle de notre groupe de deux élèves, en référence au scénario hypothétique. Cette navigation réelle donne à lire, comme des écarts entre l'activité prescrite et l'activité réelle, les facilitateurs et les obstacles à la réussite de l'activité d'apprentissage visée par la scénarisation pédagogique (**Fig. 4**). Elle révèle aussi d'autres offres et empêchements qui n'ont pas été prévus par les conceptrices et concepteurs de l'environnement d'apprentissage (**Fig. 5**).

Séquence « Ozobot », CE1
Séance 1, sous-système 1.4 : Expérimentation « Découvrir le fonctionnement d'Ozobot », en autonomie

[…]

En se rapprochant du robot qui roule sur le tracé coloré, l'élève perçoit :

- qu'il est équipé de leds, sous sa coque [information lacunaire = 0] [0]
- que ces leds s'éclairent d'une couleur liée à la couleur du tracé [« Regarde ça change de couleur ! » : information pertinente = 1] [« Le noir ce sera le bleu. Et le bleu ? » : information pertinente = 1]

Niveau dans la hiérarchie d'abstraction : Fonctions physiques (4)

// Interruption de la tâche

Deux élèves de la table 1 qui s'adressent à EN3 : « …il détecte heu la ligne. Parc'qu'on avait vu qui faisait d'la lumière d'en bas. »

E31 et E32 sont interpellés par l'intervention d'autres élèves de la classe [information lacunaire = 0] [0]

Niveau dans la hiérarchie d'abstraction : Fonctions physiques (4)

Transition : Changement d'état mental – Planification → Y a-t-il de la lumière sous le robot ?

Transition : Mouvement → E31 se penche pour regarder sous le robot

---

E31 ne perçoit pas la lumière [« Y a d'la lumière sous le robot ? » : information lacunaire = 0] [0]

Formes physiques (5)

Reprise de la tâche //

Transition : Mouvement → L'élève prend le robot en main, le retourne

[…]

Fig. 5. Extrait de la navigation réelle : Première séance, expérimentation en autonomie. Par rapport au scénario hypothétique, une tâche est interrompue alors qu'une nouvelle information est donnée à percevoir à notre groupe d'élève par d'autres élèves (en vert).

5 Résultats préliminaires

5.1 La TMTA à l'épreuve des particularités de notre contexte d'étude

La mise à l'épreuve de la scénarisation hypothétique à l'activité réelle des élèves qui s'initient à la programmation informatique avec des robots programmables nous permet de considérer :

- L'information émanant de l'environnement d'apprentissage qui n'est pas mobilisée par les élèves et celle qui n'est pas pertinente pour leur activité (distinguées dans le modèle) comme de l'information lacunaire pour les apprentissages visés (insuffisante ou insignifiante suivant Gibson et Pick [4]) ;
- De nouvelles possibilités d'agir [14] qui émergent de l'environnement, compte-tenu qu'il est également structuré par les robots scolaires qui ont leurs propres fonctions d'apprentissage, ainsi que par la coopération conduite entre des élèves qui en ont une perception variée.

5.2 Les composants de l'environnement d'apprentissage

Le scénario pédagogique explicité au regard des prescriptions institutionnelles

- Suivant la TMTA, le scénario pédagogique et les supports des activités ont dû être précisés en référence aux programmes des disciplines [6] et au Socle commun de connaissances de compétences et de culture [7] offrant ainsi une description exhaustive de l'environnement d'enseignement.
- Ce travail de rétro-ingénierie nous a aussi permis de relever des écarts : notamment, alors que la démarche d'investigation est très investie dans l'environnement observé, la dimension technologique de l'initiation à la programmation informatique est partiellement évincée de notre contexte.

L'information émanant des technologies comme source de complexité

- Les robots programmables utilisés par les élèves sur le terrain PREP possèdent des propriétés répondant à des objectifs pédagogiques qui leur sont propres. La navigation réelle des élèves dans l'environnement révèle autant de nouvelles affordances que d'information lacunaire qui résultent de dyssimétries entre ces objectifs et ceux du scénario mis en œuvre par l'enseignant.

5.3 Les offres et les lacunes de l'environnement pour l'activité des élèves

L'information pertinente (les affordances) de l'environnement d'apprentissage

- La démarche d'investigation est adoptée par les élèves qui, dès la première séance, répondent aux questionnements de l'enseignant et s'engagent dans les activités de découverte proposées. Moins soutenue ensuite, elle est réactivée alors que l'enseignant participe à la recherche avec les élèves, en exprimant ses doutes et « *essayant* » avec eux.

– La coopération entre les élèves du groupe observé est effective, elle se manifeste par une découverte symétrique des possibilités offertes par l'environnement, puis par la confrontation des idées et le partage des tâches dans les dernières séances.
– La connaissance de la notion informatique d'instruction est confirmée.

Les nouvelles possibilités d'agir de l'environnement découvertes par les élèves

– D'autres questionnements scientifiques (relatifs à la lumière et la vitesse) sont soulevés par les élèves au cours de la séquence.
– La coopération se réalise avec d'autres élèves de la classe alors que notre groupe les écoute et les interpelle régulièrement au sujet de leurs découvertes respectives.

L'information lacunaire

– La coopération avec les autres groupes d'élèves ne s'avère pas fructueuse alors que ceux-ci donnent des résultats erronés au groupe observé : « — *Tu vois, nous, quand, quand il était au départ il avançait, il s'est arrêté, on a dit "va sur la salade", il a tourné* […] *Tu vois ?* » Ou bien lorsque d'autres élèves partagent leurs découvertes sans les expliciter : « — *Il avance derrière celui à* [autre élève]. *Ouais, mais celui à* [autre élève] *il recule tout seul. — Et oh, Ozobot, tu peux reculer derrière ?* »
– Concernant l'acquisition de connaissances visée, le code que lit le robot sur les tracés n'est pas compris comme un langage, avec une syntaxe donnée qu'il est possible de décoder, puisque les élèves ne comparent pas les différents codes ni ne cherchent à différentier la lecture d'un même code par le robot dans un sens puis dans l'autre.
– Plus généralement, malgré leur implication dans la démarche d'investigation, la progression par essais-erreurs des élèves n'est pas génératrice de connaissances si celles-ci ne sont pas validées par leur enseignant : en autonomie, les élèves ne cherchent pas non plus à comprendre les comportements du robot en dehors des tracés des fiches, ni sur les reflets lumineux ; ils les expliquent par des dysfonctionnements : « — *Il a pas lu* ». Aussi, lorsque le fonctionnement des capteurs optiques est perçu par certains autres élèves (**Fig. 5**), il ne permet l'acquisition d'une nouvelle connaissance par notre groupe que lorsque leur professeur confirme ce fonctionnement, en fin de séance.
– Mais la démarche de recherche n'est pas parfaitement mise en œuvre dans la séquence observée : les hypothèses ne sont pas posées, les notions-outils ne sont pas toutes rappelées dans les introductions ni les bilans des différentes séances, et les élèves ne sont pas toujours invités à généraliser leurs résultats « en utilisant un langage précis » [7].
– D'autres constats issus de cette étude de cas concernent plus spécifiquement la dyssimétrie entre la scénarisation pédagogique et les affordances des robots qui est révélée par l'activité des élèves : certaines découvertes des élèves sur le fonctionnement du robot programmable – (« — *En fait quand on tourne la fiche, il tourne* », « […] *on avait vu qui faisait d'la lumière d'en bas* ») – ne sont pas confirmées ou infirmées par l'enseignant.

5.4 Les points saillants

Ces analyses préliminaires mettent en évidence :

- Un engagement des élèves dans la démarche de recherche, conduite en collaboration au sein du groupe et plus largement de la classe ;
- La nécessité pour les élèves de cette classe de savoir, systématiquement, leurs découvertes validées par l'enseignant ;
- Une activité qui n'est pas strictement organisée par ce dernier (suivant les prescriptions institutionnelles relatifs à la démarche scientifique, plus particulièrement), qui ne répond pas non plus à tous les questionnements de ses élèves.

En effet, à plusieurs reprises, l'enseignant de notre classe semble éviter d'entrer dans des explications trop variées et compliquées. Pourtant, ses élèves, « trop scolaires » selon ses dires, attendent justement et systématiquement qu'il valide leurs découvertes.

Ces résultats préliminaires nous amènent à émettre l'hypothèse que la mise en œuvre d'une démarche d'investigation dans un environnement sociotechnique complexe et au sein d'une classe majoritairement en difficulté scolaire nécessite une scénarisation particulière, dotée d'un équilibre subtil entre la mise en autonomie et accompagnement soutenu.

6 Conclusion

Suivant une méthode d'analyse de l'activité à partir de la tâche du champ de l'ergonomie, nos analyses consistent à lire l'activité à l'intersection des « conditions externes » relatives à la tâche prescrite et des « conditions internes » qui concernent son appropriation par le sujet [5]. En termes d'écologie de la perception, il s'agit de considérer dans un même mouvement les caractéristiques du contexte environnemental et celles des sujets qui sont engagées dans l'activité [8], à partir du « prélèvement de l'information » [4]. L'analyse de cas présentée dans cet article montre de « bons » élèves de CE1 d'une classe en difficulté scolaire qui ont toutefois besoin du soutien de l'enseignant pour rester engagés dans la démarche d'investigation comme pour confirmer leurs apprentissages par découverte. Alors que ce besoin est exprimé et connu de l'enseignant, il ne parait pas en capacité d'y répondre, la nécessité de ne pas complexifier l'activité pour l'ensemble des élèves prévalant. Les modalités de la mise en œuvre en classe d'activités reposant sur la démarche de recherche, instituée dans les programmes scolaires français dès le cycle 2, sont ici questionnées en relation au contexte environnemental de la classe, et plus largement de l'établissement. L'analyse de l'activité d'élèves d'autres classes, d'autres niveaux et d'autres réseaux d'éducation nous amènerons à confirmer ou nuancer ces premiers résultats. Mais pour ce faire, ces analyses seront réalisées selon un « grain » [5] plus épais, pour saisir puis comparer l'activité de façon plus générale, en lien à des objectifs pédagogiques préalablement déterminés. Pour ces comparaisons, en plus de l'information lacunaire ou affordante,

les différents niveaux d'abstraction mobilisés par les élèves pour s'initier à la programmation informatique nous intéresseront plus particulièrement.

Références

1. Albero, B. : Une approche sociotechnique des environnements de formation : Rationalités, modèles et principes d'action. Éducation et didactique, 4(1), 7–24 (2010).
2. Gibson, J. J. : The Theory of Affordances, in R. Shaw, J. Bransford (eds.), The Ecological Approach to Visual Perception, pp. 67-82. Houghton Mifflin (1979).
3. Gibson, J. J. : Approche écologique de la perception visuelle. Éditions Dehors (2014).
4. Gibson E. J., Pick, A. D. : An Ecological Approach to Perceptual Learning and Development. Oxford University Press (2000).
5. Leplat, J. : L'analyse psychologique de l'activité en ergonomie : Aperçu sur son évolution, ses modèles et ses méthodes. Octarès Éditions (2000).
6. Ministère de l'Éducation nationale, de la Jeunesse et des Sports, Le socle commun de connaissances, de compétences et de culture, https://www.education.gouv.fr/cid2770/le-socle-commun-connaissances-competences.html, visité le 15/12/2021
7. Ministère de l'Éducation nationale, de la Jeunesse et des Sports : Programme du cycle 2. En vigueur à la rentrée 2020 (2020).
8. Morineau, T. : Éléments pour une modélisation du concept d'affordance in Actes du colloque ÉPIQUE, 83-95 (2001).
9. Morineau, T., Frénod, E., Blanche, C., Tobin, L : Turing Machine as an ecological model for Task Analysis. Theoretical Issues in Ergonomics Science, 10 (2009).
10. Morineau, T. : La méthode TMTA d'analyse écologique de la tâche et son application à une tâche praxique. Le travail humain, 73(2), 97-122 (2010).
11. Morineau, T. : Turing machine task analysis: A method for modelling affordances in the design process. International Journal of Design Engineering, 4(1), 58-70 (2011).
12. Norman, D. A. : The design of everyday things. Basic Books (2013).
13. Norman, D. A. : The invisible computer: Why good products can fail, the personal computer is so complex, and information appliances are the solution. MIT Press (1999).
14. Simonian, S. : L'affordance socioculturelle : Une approche éco-anthropocentrée des objets techniques. Le cas des Environnements Numériques d'Apprentissage [Habilitation à Diriger les Recherches, Université Rennes 2] (2016).
15. Tomasello, M. : The cultural ecology of young children's interactions with objects and artifacts in E. Winograd, R. Fivush, W. Hirst (eds.), Ecological Approaches to Cognition: Essays in Honor of Ulric Neisser. Lawrence Erlbaum Associates Publishers (1999).
16. Veillard, L. : Les méthodologies de constitution et d'analyse des enregistrements vidéo in L. Veillard, A. Tiberghien (eds.), ViSA : Instrumentation de la recherche en éducation. Éditions de la Maison des sciences de l'homme (2013).
17. Vicente, K. J., Rasmussen, J. : The Ecology of Human-Machine Systems II: Mediating « Direct Perception » in Complex Work Domains. Ecological Psychology, 2(3), 207-249 (1990).
18. Vicente, K. J., Rasmussen, J. : Ecological interface design: Theoretical foundations. IEEE Transactions on Systems, Man, and Cybernetics, 22(4), 589-606 (1992).
19. Vygotski, L. S. : La méthode instrumentale en psychologie in J.-P. Bronckart, B. Schneuwly (eds.), Vygotsky aujourd'hui. Delachaux et Niestlé (1985).
20. Wells, A. J. : Gibson's Affordances and Turing's Theory of Computation. Ecological Psychology, 14(3), 140-180 (2002).

Difficultés d'enseignement et d'apprentissage de la science informatique au primaire

Julien Bugmann[0000−0002−1205−9005], Morgane Chevalier[0000−0002−9115−1992], Jean-Philippe Pellet[0000−0001−7559−397X] et Gabriel Parriaux[0000−0002−8921−5459]

Haute école pédagogique du canton de Vaud, Lausanne, Suisse
{julien.bugmann,morgane.chevalier,jean-philippe.pellet,gabriel.parriaux}@hepl.ch

Résumé Alors que l'enseignement de la science informatique prend une place de plus en plus importante dans les programmes scolaires, et ce dans diverses régions du monde, la Suisse romande, elle aussi, vient d'adopter un nouveau plan d'études intégrant cette nouvelle discipline. Le défi est de taille car il s'agit de former les futurs enseignants du primaire et du secondaire, mais aussi tous les enseignants déjà en poste. Pour ce faire, il est capital de se questionner sur les modalités d'activités proposées en science informatique et leurs difficultés d'enseignement et d'apprentissage relevées par les futurs enseignants. Nous proposons donc dans cet article d'étudier ces différents aspects liés à la discipline de la science informatique au primaire, notamment la programmation et l'algorithmique, à l'issue d'une formation spécifique dispensée dans une Haute école pédagogique, en Suisse. Les résultats montrent que ce sont principalement des activités mixtes qui sont proposées par ces futurs enseignants à leurs élèves et que les principales difficultés d'apprentissage perçues reposent sur la décentration des élèves en situation, la transposition, la maîtrise des outils et la complexité disciplinaire. On constate également que les futurs enseignants éprouvent majoritairement des difficultés d'enseignement en lien avec la maîtrise des savoirs à enseigner, la gestion de classe et ce que nous appelons le défi didactique.

Keywords: science informatique · enseignement · apprentissage

1 Introduction

Le tournant de l'intégration de la science informatique (SI) en tant que discipline à l'école obligatoire a été engagé dans divers pays à travers le monde, depuis plusieurs années déjà [1,2], ou encore très récemment [21,22]. L'intégration de cette nouvelle discipline implique de former en amont les enseignants et les futurs enseignants non seulement à ces nouveaux contenus disciplinaires mais aussi aux approches didactiques pour les enseigner. Il est ainsi attendu des étudiants en formation au métier d'enseignant (étudiants nommés "futurs enseignants" dans cet article) qu'ils orchestrent [10] de telles situations d'enseignement–apprentissage, i.e. qu'ils conçoivent, mettent en œuvre et évaluent les activités permettant d'atteindre les objectifs en SI du plan d'études et qu'ils maîtrisent les concepts informatiques de base [11].

Si la généralisation des contenus de SI à l'école obligatoire est relativement nouvelle, une variété de méthodes permettant de véhiculer de tels contenus est proposée depuis de nombreuses années [3,16,20]. Néanmoins, si le média permet un accès facilité au contenu à enseigner, il ne reste pas moins vrai qu'il n'en garantit pas l'apprentissage [9]. De fait, nous nous intéressons dans cette étude aux liens entre la compréhension de la SI et les modalités de sa mise en œuvre en classe, tout spécialement concernant la programmation et l'algorithmique : qu'en est-il du point de vue des futurs enseignants qui expérimentent ces activités en classe ? Quelles sont les difficultés d'apprentissages des élèves perçues par ces futurs enseignants et quelles sont les difficultés d'enseignement qu'ils mettent en avant ? Ces différents freins sont-ils particulièrement liés à la modalité pédagogique choisie pour les diverses activités ?

Après avoir présenté les différentes modalités d'activités en SI recensées dans la littérature, nous présentons notre problématique et nos questions de recherche. La méthode suivie pour cette étude des représentations des futurs enseignants en formation est ensuite explicitée. Puis, nous présentons et discutons les résultats obtenus avant de conclure en développant les prolongements à cette recherche.

2 Apports théoriques

S'intéresser à la mise en œuvre de la SI à l'école primaire implique plusieurs perspectives. Il convient de se pencher à la fois sur les objets de savoirs visés et leur mise en œuvre au travers des modalités pédagogiques, mais aussi sur les difficultés d'enseignement et d'apprentissage identifiées par les protagonistes que sont respectivement les (futurs) enseignants et les élèves. Dans cette partie, nous rapportons ce qui a été recensé dans la littérature au sujet de ces différentes focales. Le recensement des modalités de mises en œuvre nous permet d'ouvrir à une nouvelle catégorisation.

2.1 La mise en œuvre de la SI à l'école primaire

La SI a connu plusieurs entrées dans l'école obligatoire, sous l'impulsion de projets politiques propres à chaque période de l'histoire de l'éducation [2], mais son intégration dans les curricula dès les premières années scolaires est relativement récente [1,5,14,18]. Sur ces dernières décennies, la mise en œuvre de la SI dans les classes d'élèves de 4 à 12 ans présente ainsi principalement des activités que l'on pourrait classer selon deux modalités pédagogiques, à savoir celle « branchée » et celle « débranchée ». La principale différence entre ces deux modalités réside dans le fait que le recours à une interface écran est nécessaire à la programmation dans le cas de la modalité branchée [3,4].

Les activités de SI dites « débranchées » ou en anglais les « CSU activities » pour « Computer Science Unplugged activities » semblent aujourd'hui dépasser le statut de simple modalité pédagogique à celui d'approche pédagogique en soi [23]. Les apports de la mise en œuvre en classe des activités CSU ne sont pourtant encore que peu documentés dans l'état de l'art en ce qui concerne

l'apprentissage effectif des élèves en SI [12]. Néanmoins, le potentiel pédagogique de cette modalité d'activité est décrit et loué pour sa caractéristique de mise en mouvement de l'élève et d'apprentissage incarné [27].

Les activités dites « branchées », quant à elles, ont initialement été incarnées au moyen des ordinateurs ou encore des robots [20]. Ces derniers s'étant largement diversifiés [6,19], on constate néanmoins que certains de ces robots ne nécessitent pas pour autant une interface écran pour pouvoir les programmer, comme par exemple pour les robots Blue-Bot (TTS Group, Hucknall, Royaume-Uni) et Cubetto (Primo Toys, Londres, Royaume Uni). De fait, la typologie des pratiques pédagogiques semble ne pas être aussi dichotomique et des hybridations d'activités branchées–débranchées apparaissent dans la littérature [13] sous l'appellation de *Robotics Unplugged (RU) activities* pour « activités robotiques débranchées », c'est-à-dire qui recourent à des robots sans pour autant utiliser une interface écran pour les programmer. Cette modalité d'activité vient donc enrichir les moyens de mise en œuvre de la SI dans une approche pédagogique dite « débranchée » [3,4,12].

Cependant, la liste de ces modalités nous parait encore incomplète. En effet, certaines activités d'initiation à la SI intègrent par exemple des robots, mais sans solliciter leurs capteurs ou actionneurs, par exemple dans le cadre d'une simulation de déplacement dans un espace. Il nous paraît donc important de ne pas étiqueter cette pratique comme étant « débranchée » au sens du CSU [3,4]. Nous proposons donc d'ajouter une nouvelle catégorie appelée SMEA, pour « Sans Moyen Électronique Actif » (en anglais, WAE pour *Without Active Electronics*). Dans ce contexte, on considère qu'il est tout à fait possible d'utiliser un robot comme simple simulateur d'un déplacement, mais sans l'activer électroniquement. Cette catégorie contient par essence également toutes les activités papier, crayon ou objets divers, tant qu'elles ne mobilisent aucun moyen électronique actif. Par exemple, des robots tels que Blue-Bot ou Thymio, utilisés sans qu'ils ne soient allumés, et donc pas « électroniquement actifs », en feraient partie. Cela nous permet de distinguer ce type de pratique d'une pratique dite « Robotique » dans laquelle nous considérons que l'outil doit être activé pour être considéré comme un outil dans une activité dite robotique.

Les possibilités de mise en œuvre en classe de la SI semblent donc bien plus denses que la simple distinction en modalités branchées/débranchées (Tableau 1).

Ces différentes modalités peuvent naturellement être couplées et il est donc possible de retrouver des activités SMEA, des activités avec écrans, des activités robotiques, etc. dans une même séquence en classe et pour viser un même objectif pédagogique.

Nous avons donc différents moyens de mise en œuvre en classe, des programmes qui s'adaptent et des formations de plus en plus nombreuses pour les enseignants et les futurs enseignants. Mais alors, que se passe-t-il lorsque l'on met cela en application en contexte scolaire ? Est-ce que tout fonctionne ? Quel regard les (futurs) enseignants portent-ils sur les difficultés d'apprentissage des élèves et celles liées à leurs activités d'enseignement ?

Tableau 1. Affinement des catégorisations branchées/débranchées pour les activités de science informatique.

Modalité de l'activité	**Branchée**		**Débranchée**	
Recours à l'électronique	**oui**		**oui**	**non**
Type d'artefacts mobilisé	Écran	Robot avec interface écran pour programmation	Robot sans usage de l'interface écran pour programmer (interface tangible ou programmes embarqués)	Papier, crayon, objets, etc.
Dénomination	**Écrans**	**Robotique + écrans**	**Robotique Unplugged (RU)**	**Sans Moyen Électronique Actif (SMEA)**
Exemples	• Scratch • Cargo Bot • Code.org • Lightbot • Mimo	• Thymio + VPL • Lego Mindstorms EV3 • Nao Choregraphe	• Thymio préprogramme • Blue-Bot avec interface tangible de programmation ou barre	• Computer Science Unplugged (CSU) • Square • Le jeu du robot • Pixel paravent • Un robot qui ne serait pas allumé

2.2 Difficultés d'enseignement et d'apprentissage

Les pratiques rapportées récemment dans les premières années de la scolarité (école maternelle) et des années suivantes (école primaire) permettent de souligner différents types de difficultés émergentes identifiées par les enseignants et les futurs enseignants pour enseigner la SI auprès de leurs élèves. Parmi celles-ci, on relève notamment le problème d'infrastructure et l'accès au matériel [7,18]. Alors que l'éducation numérique en général et la SI en particulier sont de plus en plus enseignées dans les établissements scolaires, ceux-ci peinent bien souvent à proposer à chaque enseignant les ressources adéquates pour mener à bien leurs activités. Que cela soit en termes de robots, de tablettes, ou de matériel SMEA à disposition, la nouveauté de la discipline, mais aussi le coût du matériel nécessaire

pour la quantité d'élèves et enseignants visés, représentent de véritables freins à une mise en œuvre sereine en classe. En effet, les enseignants sont très souvent contraints de chercher le matériel dans d'autres salles, parfois d'autres bâtiments, et éprouvent bien des difficultés à organiser le tout simplement.

Cette rareté des ressources amène à un autre problème qui est celui du manque de temps de préparation [7,18] pour les activités. D'autant plus que les grilles horaires des enseignants sont surchargées et n'octroient très souvent aucune plage supplémentaire pour enseigner la SI. Le temps est aussi convoqué au travers de l'emboîtement instrumental [17] impliqué dans l'appropriation des outils pour ensuite travailler les concepts visés.

Enfin, un autre frein existant repose sur la résistance des collègues [18] au développement de telles activités en classe.

Concernant les difficultés d'apprentissages des élèves lors de l'initiation à la SI, celles-ci ne sont que rarement listées dans l'état de l'art. Comme le remarque Monique Grandbastien [15], dans une phase de création d'enseignement, il n'est pas étonnant de retrouver dans la littérature davantage de « réflexions quant aux objectifs et des propositions de mise en œuvre que sur des retours d'expérience avec description des succès et des difficultés observés chez les élèves » (p. 14). Néanmoins, quelques difficultés récurrentes sont à pointer, tant au niveau de la compréhension des concepts qui ne sont pas suffisamment introduits par les enseignants [24], que du transfert du langage naturel à celui de la machine [8], soit la transposition.

3 Problématique et questions de recherche

Cette recherche vise à mettre en évidence de potentiels liens entre la compréhension de la SI et les modalités de mise en œuvre en classe. Dans la mesure où les modalités d'activités proposées sont de natures variées (SMEA, robotiques, écrans, voire des activités mixtes), et où les niveaux de connaissances des futurs enseignants sont eux aussi variés, nous nous intéressons aux différents points de vue des futurs enseignants (étudiants en formation d'enseignant) qui expérimentent ces modalités d'activités en classe.

Quelles modalités d'activités sont mises en œuvre ? Quelles difficultés rencontrent ces acteurs du terrain dans leurs enseignements et quelles difficultés d'apprentissages anticipent-ils chez leurs élèves ?

Est-ce que les difficultés d'apprentissages des élèves pressenties par les futurs enseignants sont particulièrement liées à la modalité pédagogique choisie ?

Et enfin, nous nous demandons comment adapter les dispositifs de formation pour permettre au mieux à ces futurs enseignants d'amener leurs élèves vers des apprentissages en SI.

4 Méthodologie

Dans la mesure où le canton de Vaud projetait depuis 2018 l'introduction d'une nouvelle discipline intitulée « Science informatique et projets numériques » à la

grille horaire, le Plan d'études romand (PER) a également connu une profonde révision pour la partie touchant à l'éducation numérique avec, notamment, l'introduction de cette nouvelle discipline qu'est la SI.

En ce qui concerne la formation des futurs enseignants du primaire, les contenus de cette nouvelle discipline étaient déjà pour certains étudiés dans différents modules de la formation, de la 1re à la 3e et dernière année, mais les contenus liés à la SI dans les curricula étaient quant à eux totalement nouveaux pour les futurs enseignants. Pour compléter son dispositif de formation sur ces nouveaux aspects, la HEP Vaud a créé un module temporaire, intitulé « BP63SIP — Science Informatique et Projets numériques » en complément des modules existants pour les étudiants de 3e année du Bachelor primaire.

Dans le cadre de la validation de ce module (en juin 2020), les participants ont présenté oralement, individuellement ou en groupe, leur projet de mise en œuvre (réelle ou fictive) d'une séquence en SI aux cycles 1 ou 2. Dans le cadre de ce module, les futurs enseignants découvraient, entre autres, des activités telles que Pixel Paravent, pour travailler sur la transmission d'information et le binaire, le jeu du robot et les activités de robotique avec Thymio et Bluebot pour découvrir le fonctionnement des machines, capteurs et actionneurs, ou encore l'application Scratch Jr pour travailler les séquences et les boucles. Il est à noter qu'en raison de la pandémie de la COVID-19, 80% des projets ont été simulés et n'ont pas pu avoir lieu en classe. 11% des projets ont été intégralement menés en classe et 9% l'ont été sous une modalité mixte (une partie effectuée en classe, et l'autre non).

À cette occasion et en raison de la COVID-19, l'examen s'est déroulé à distance via une plateforme de visioconférence et chaque présentation orale a été filmée. Lors de cet examen, un chercheur, externe au processus d'évaluation, posait systématiquement deux questions aux étudiants (i.e. futurs enseignants) :

— Quelles étaient ou auraient été (si l'activité n'avait pas pu être menée en classe en raison de la COVID-19) les principales difficultés d'*apprentissage* pour vos élèves dans le cadre de l'activité proposée ?

— Quelles étaient ou auraient été (si l'activité n'avait pas pu être menée en classe en raison de la COVID-19) les principales difficultés d'*enseignement* pour vous dans le cadre de l'activité proposée ?

Après la session d'examen, les séquences vidéo ont été analysées aux fins de la présente recherche. Il était naturellement expliqué aux futurs enseignants que cette participation à la recherche n'impactait en rien l'évaluation du module.

À l'issue de cette recherche, nous avons pu obtenir les données issues de 34 groupes de futurs enseignants, ce qui représente au total 81 participants.

5 Résultats

Nous nous demandions ce que cherchent, reprennent, ou visent les futurs enseignants dans leur mise en œuvre en classe et s'il existait une modalité d'activité favorite pour les élèves.

5.1 Catégories identifiées

Sur les 34 groupes, nous avons obtenu 35 projets, un groupe ayant effectué deux activités avec deux types de publics différents. Sur ces 35 projets, la majorité (71,5%) étaient des projets mixtes, c'est-à-dire intégrant des activités de modalités différentes (Tableau 2). Seules 29,5% des activités ne concernaient qu'une seule modalité d'activité (SMEA, Écrans ou Robotique de manière exclusive). On constate également que les différents groupes concernés ont dans leur grande majorité effectué au moins une activité intégrant la modalité SMEA (80% des groupes).

Tableau 2. Fréquence des différentes modalités d'activités selon les projets concernés.

Type d'activités mises en œuvre	**Nombre de projets concernés**	**Pourcentage**
SMEA	4	11,4%
SMEA et écrans	10	28,6%
SMEA et robotique unplugged	12	34,3%
Écrans	1	2,9%
SMEA, robotique et écrans	2	5,7%
Robotique unplugged	5	14,3%
Robotique et écrans	1	2,9%
Total	**35**	**100%**

5.2 Difficultés d'apprentissage des élèves identifiées par les futurs enseignants

À l'issue des examens du module, nous avons donc posé deux questions aux futurs enseignants, comme évoqué dans la méthodologie puis nous avons effectué une catégorisation de ces données pour procéder à leur traitement et répondre à nos questions de recherche.

En fonction des thématiques abordées par les futurs enseignants interrogés, nous avons pu extraire un certain nombre de catégories (Tableau 3) liées à l'apprentissage, et plus spécifiquement à l'apprentissage de la SI, à savoir :

— la collaboration : lorsque les futurs enseignants parlent de difficultés de collaborations entre élèves

— la complexité disciplinaire : lorsque les futurs enseignants présentent la discipline comme étant, selon eux, particulièrement complexe pour que leurs élèves apprennent (termes, programmes complexes, etc.)

— la culture initiale : quand il est question de compétences et connaissances des élèves qui seraient trop faibles

— la décentration : lorsque les futurs enseignants interrogés évoquent les difficultés qu'ont les élèves à se projeter à la place du robot et à comprendre la démarche algorithmique d'un programme et son issue

— l'intérêt à long terme : lorsque c'est la motivation des élèves sur une plus longue durée pour ces activités qui est remise en question

— la maîtrise de l'outil : quand les futurs enseignants évoquent les difficultés relatives à la maîtrise de l'outil utilisé par les élèves

— le manque de motricité : lorsqu'il est question de difficultés à manipuler les outils par les élèves du fait d'un manque de motricité fine

— le sens de l'activité : lorsque les futurs enseignants mettent en avant un déficit de sens ou de compréhension de l'objectif des activités pour les élèves

— la surcharge cognitive : quand il est question d'un nombre trop important d'informations à traiter pour effectuer une activité

— la transposition : lorsque les futurs enseignants font état de difficultés à transposer les apprentissages, notamment dans le fait de passer d'un langage à un autre

Tableau 3. Répartition des différentes catégories de difficultés d'apprentissages pour les élèves.

Difficulté d'apprentissages	**Nombre d'occurrences**	**Pourcentage**
Collaboration	1	1,5%
Complexité disciplinaire	12	17,9%
Culture initiale	2	3,0%
Décentration	18	26,9%
Intérêt à long terme	1	1,5%
Maîtrise de l'outil	13	19,4%
Manque de motricité	1	1,5%
Sens de l'activité	3	4,5%
Surcharge cognitive	5	7,5%
Transposition	11	16,4%
Total	**67**	**100%**

On constate que les difficultés d'apprentissage les plus souvent relevées par les futurs enseignants (Tableau 3) concernent principalement la décentration (26,9%), la maîtrise de l'outil (19,4%), la complexité disciplinaire (17,9%), mais aussi la transposition (16,4%) pour les élèves, soit le fait de passer des apprentissages réalisés à une représentation concrète en informatique. A contrario, les difficultés les moins souvent évoquées concernent certaines caractéristiques individuelles propres aux élèves (manque de motricité, difficulté à collaborer, manque de culture en informatique).

On relève également que la décentration apparaît majoritairement, et il semble donc être capital de travailler sur ces aspects d'orientation, d'anticipation des programmes, etc. en formation et de donner davantage de temps à ce type d'activités en classe. On note aussi que les futurs enseignants parlent de complexité disciplinaire. On peut penser que cette dernière est surtout liée au manque de maîtrise des savoirs à enseigner par les futurs enseignants (Tableau 4), d'où la nécessité de travailler sur ces savoirs en formation. Enfin, la transposition, elle aussi semble être difficile pour les élèves, ce qui pourrait montrer l'importance d'aborder les liens entre informatique et société pour offrir des outils et des idées permettant de transposer un apprentissage en SI dans une situation concrète du quotidien.

5.3 Difficultés d'enseignement identifiées par les futurs enseignants

Nous avons fait de même avec les difficultés d'enseignement éprouvées par les futurs enseignants en catégorisant les différentes réponses de ces derniers. Nous proposons ainsi différentes catégories, qui sont :

- le défi didactique : lorsqu'un futur enseignant évoque la complexité de transmettre le savoir à ses élèves et de faire du lien avec l'informatique du quotidien
- la différenciation : lorsqu'un futur enseignant a des difficultés à gérer les différences de niveaux des élèves dans la discipline
- l'évaluation : lorsqu'un futur enseignant aborde la ou les difficulté(s) liée(s) aux méthodes d'évaluation des apprentissages
- la gestion de classe : lorsqu'il est question des interventions du futur enseignant en lien avec l'activité des élèves en situation pour atteindre l'objectif pédagogique visé
- la maîtrise de l'outil : lorsque l'utilisation des outils destinés au futur enseignant complique son activité d'enseignement
- la maîtrise des savoirs à enseigner : lorsque le futur enseignant ne s'estime pas suffisamment compétent pour enseigner cette discipline
- le manque de ressources : lorsque le futur enseignant estime ne pas bénéficier de suffisamment de ressources (activités, exercices, etc.) pour mener à bien son activité
- le manque de temps : lorsque le temps à disposition (dans la grille horaire notamment, en préparation, lors de l'activité) n'est pas suffisant
- la scénarisation de l'activité : lorsque le futur enseignant éprouve des difficultés à préparer son activité et à la scénariser

À l'issue de nos analyses, nous constatons (Tableau 4) que les difficultés d'enseignement les plus fréquemment évoquées concernent, pour notre public de futurs enseignants, le défi didactique (20,9%), la gestion de classe (20,9%), la maîtrise des savoirs à enseigner (16,4%) et enfin la maîtrise des outils utilisés dans le cadre de l'activité (13,4%). Les difficultés les moins fréquemment évoquées sont la scénarisation de l'activité (4,5%) ou encore la différenciation (4,5%).

Tableau 4. Répartition des différentes catégories de difficultés d'enseignement pour les futurs enseignants.

Difficulté d'enseignement	**Nombre d'occurrences**	**Pourcentage**
Défi didactique	14	20,9%
Différenciation	3	4,5%
Évaluation	4	6,0%
Gestion de classe	13	19,4%
Maîtrise de l'outil	9	13,4%
Maîtrise des savoirs à enseigner	11	16,4%
Manque de ressources	4	6,0%
Manque de temps	6	9,0%
Scénarisation de l'activité	3	4,5%
Total	**67**	**100%**

On relève ainsi la place forte donnée au défi didactique et à la gestion de classe dans les difficultés évoquées par les futurs enseignants. Le premier pourrait s'expliquer par un grand manque de confiance et potentiellement de maîtrise des savoirs en SI. Pour ce qui en est de la gestion de classe et de la maîtrise de l'activité, cela pourrait correspondre à un déficit d'expérience en formation et surtout en classe mais aussi à des aspects matériels qui viendraient compliquer la gestion de classe.

5.4 Relations entre les différentes modalités d'activités et le degré des élèves

Ces catégories désormais mises en évidence, il nous paraissait important d'en étudier la répartition selon les différents degrés des élèves concernés. En effet, est-ce que les futurs enseignants privilégient une modalité d'activité plutôt qu'une autre selon le cycle dans lequel ils enseignent ?

Après croisement entre la modalité d'activité et le cycle concerné (cycle 1, soit de la 1re à 4e année de primaire, i.e. de 4 à 7 ans, ou cycle 2, de la 5e à la 8e année de primaire, i.e. de 8 à 11 ans), on constate que les activités avec écrans sont principalement destinées aux élèves du cycle 2 et que la robotique seule est plus prisée au cycle 1 (Figure 1). Aucun groupe de futurs enseignants ayant des élèves du cycle 1 n'a proposé d'activité avec des écrans. Les élèves du cycle 1 ont pour la plupart eu des activités avec les robots ou des activités SMEA, voire, dans la majorité des cas, un croisement de ces deux modalités d'activités.

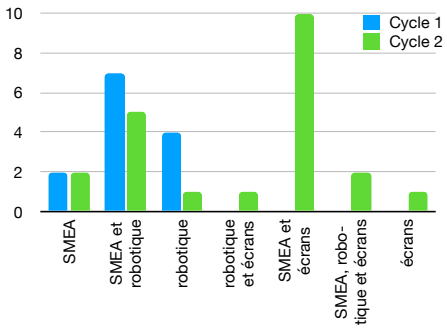


Figure 1. Distribution des modalités par cycle.

En regardant plus précisément les différents degrés concernés (degrés 1–2 équivalent aux deux premières années du cycle 1, degrés 7–8 aux deux dernières du cycle 2), on constate qu'il y a une tendance à faire des activités plus « multi-modales » au fur et à mesure que les degrés augmentent (Figure 2). Ainsi, plus les élèves du primaire seraient âgés, plus les futurs enseignants souhaiteraient articuler plusieurs modalités d'activités.

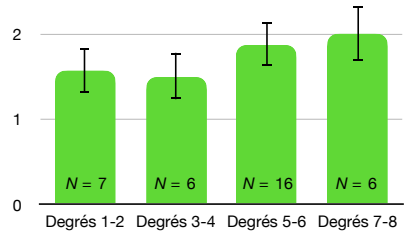


Figure 2. Index de mixité moyen par demi-cycle.

Maintenant que ces éléments sont mis en évidence, nous souhaitons questionner les liens entre les différentes difficultés d'apprentissage évoquées par les futurs enseignants et les activités mises en œuvre.

5.5 Relations entre les difficultés d'apprentissage et d'enseignement et les modalités d'activités mises en œuvre

En croisant les difficultés d'apprentissage des élèves évoquées par les futurs enseignants, nous souhaitons vérifier s'il apparaît une relation entre la modalité d'activité et les difficultés rencontrées ou envisagées. Sur cette base, nous pourrions alors anticiper les propositions faites aux futurs enseignants en formation et

adapter le dispositif pour résoudre au mieux les freins aux apprentissages des élèves en SI.

Il ressort de notre analyse (Figure 3) une différence statistiquement significative entre les trois sous-groupes que sont SMEA, robotique et écrans ($\chi^2(18) = 32.63$, $p = .018$). On relève donc deux données importantes. La première concerne la maîtrise de l'outil qui serait bien plus complexe dans les modalités avec écrans que dans les modalités sans écrans. La deuxième concerne la plus forte difficulté pour les élèves de se décentrer lors des activités réalisées sous les modalités incluant la robotique.

Ces éléments sont particulièrement intéressants car on pourrait penser le contraire, du fait du caractère tangible et manipulable des robots et du fait que l'on soit davantage allocentré sur un écran et davantage autocentré avec un robot. Aussi, il est possible que cela soit potentiellement lié aux choix d'activités menées sur les écrans, et non pas tant à l'écran lui-même.

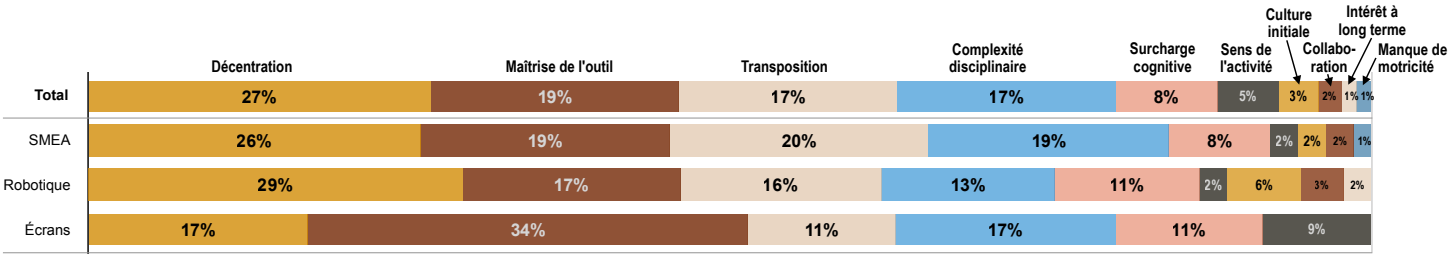


Figure 3. Difficultés d'apprentissage selon les modalités d'activités mises en œuvre.

Certains choix d'activités auraient ou pourraient donc avoir une relation avec certaines difficultés d'apprentissage chez les élèves. Nous avons voulu savoir comment la modalité d'activité influençait, si c'était le cas, les difficultés d'enseignement rencontrées ou anticipées par les futurs enseignants.

Pour cela, nous avons croisé les différentes modalités d'activités et les catégories de difficultés d'enseignement formulées par les participants à la recherche (Figure 4).

On relève que des différences existent selon la modalité d'activité. En robotique, la gestion de classe semble plus complexe que dans une autre activité, tout comme le manque de temps, alors que le défi didactique est moindre. Dans le cadre d'une activité SMEA, c'est la maîtrise de l'outil qui est moins relevée, et l'on constate enfin que l'évaluation est simplifiée dans les activités avec écrans. Cependant, ces différences apparaissent davantage comme des tendances, dans la mesure où elles ne sont pas statistiquement significatives ($\chi^2(16) = 13.53$, $p = .633$).

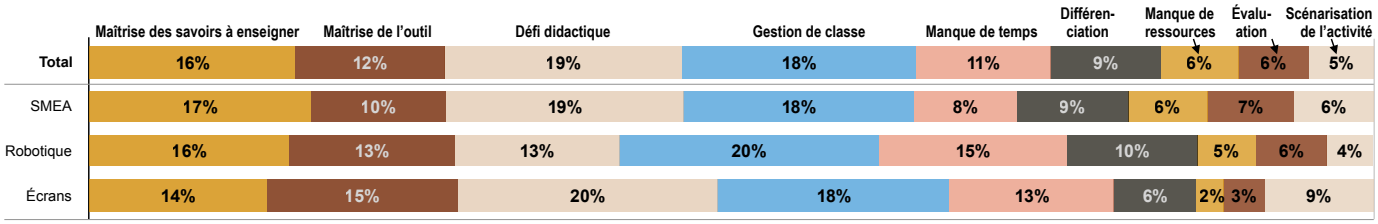


Figure 4. Difficultés d'enseignement selon les modalités d'activités mises en œuvre.

6 Discussion

Tout d'abord, il paraît important de signaler que ce module était optionnel et que le nombre d'inscrits a été particulièrement important, ce qui montre d'un certain côté que la SI attire les futurs enseignants. Est-ce parce que cette discipline est appelée à devenir obligatoire ou parce que la tendance renforcée de la société numérique se renforce ? Cela serait à questionner.

À la lecture des résultats obtenus dans le cadre de cette recherche, on observe que les futurs enseignants ont ce besoin de donner du sens, pour eux et pour leurs élèves. Il apparaît également capital pour ces futurs enseignants de parvenir à transmettre ces apprentissages et le fait de rencontrer une discipline nouvelle, tant pour eux que pour leurs élèves, et de faire face à des difficultés pour atteindre certains objectifs d'apprentissage, leur pose problème.

Comme constaté dans ce travail, certains choix d'activités auraient ou pourraient mettre en exergue certaines difficultés d'apprentissage pour les élèves dans le cadre d'activités en SI, en particulier celles qui touchent à la programmation et à l'algorithmique. En nous appuyant sur ces résultats, il nous faudrait alors adapter les formations proposées en veillant à chercher à réduire au mieux les difficultés rencontrées. Par exemple, avec des activités intégrant des écrans, il faudrait davantage insister sur la maîtrise l'outil et moins sur la décentration.

Pour faire du sens, les futurs enseignants ont donc besoin d'être équipés, tant en ressources qu'en savoirs, ce qui renvoie à la double instrumentation décrite par Trouche [26] dans un autre contexte.

Le manque de maîtrise des savoirs à enseigner demeurant particulièrement problématique pour les futurs enseignants, il nous apparaît capital de démarrer l'initiation aux bases de l'informatique dès le début de la formation des futurs enseignants — à l'instar de ce qui a déjà été expérimenté [22]. Une fois ces bases posées, il est alors possible de développer les aspects didactiques et d'aller vers l'intégration.

Les futurs enseignants semblent ainsi avoir besoin d'être rassurés, confortés dans leurs connaissances, pour réussir le défi didactique et aider leurs élèves à accéder aux apprentissages [18].

Aussi, les difficultés principales d'apprentissage chez les élèves, constituées de difficultés en termes de décentration et de maîtrise des outils, seraient à travailler davantage tout au long de la formation des futurs enseignants. Ceci permettrait

d'assurer une meilleure continuité entre la maîtrise des concepts de base de la SI par les futurs enseignants, indispensables pour permettre de transmettre aux élèves des apprentissages riches, puis d'assurer une mise en situation réussie. C'est la culture de l'enseignant en SI qui serait donc un indicateur valable du transfert d'apprentissage.

Enfin, la mise en perspective semble capitale pour permettre des apprentissages et réussir tout enseignement. Elle se caractérise ici par les difficultés d'apprentissage « Décentration », « Intérêt à long terme », « Sens de l'activité » et surtout « Transposition ». Cela rejoint les travaux de Tardif et Meirieu [25] sur la nécessaire triade « contextualisation, décontextualisation et recontextualisation » nécessaire en vue d'un transfert des connaissances. Ces aspects sont justement traités dans la formation des futurs enseignants au travers de la thématique « informatique et société », dans la mesure où l'informatique est également présente dans l'univers de l'enfant. Le fait de contextualiser le savoir par rapport à sa propre culture facilite la transposition didactique et semble ainsi particulièrement pertinent, voire indispensable.

7 Conclusion

Nous souhaitions, dans le cadre de cette recherche, interroger les difficultés d'apprentissage pour les élèves, et celles d'enseignement pour les futurs enseignants, dans des activités d'initiation à la SI au primaire. Portée sur un échantillon de 34 groupes de futurs enseignants, cette recherche a permis de mettre en évidence des besoins en termes de formation à l'enseignement de la SI pour les futurs enseignants du primaire. En effet, les futurs enseignants et enseignants sont constamment en quête de sens pour ces activités, souffrent d'un déficit de confiance en eux et certainement de maîtrise des concepts à enseigner à leurs élèves. Dans le même temps, nous observons que les futurs enseignants voient comme difficultés pour les élèves le fait d'avoir à se décentrer par rapport à l'activité, à trouver du sens dans ces activités et à pouvoir transposer leurs apprentissages dans un autre contexte. Ces derniers points passent par une mise en relation renforcée sur les liens entre informatique et société afin de les amener à se représenter certaines activités dans un autre contexte. Par ailleurs, cet recherche montre que les futurs enseignants privilégient les activités sans écrans pour les élèves de premier cycle, et ont donc assimilé que l'informatique peut s'enseigner en ayant recours uniquement à des activités SMEA ou robotique unplugged.

Au niveau des perspectives à venir, ce module de formation, et les éléments issus de cette recherche, ont permis d'actualiser notre offre de formation avec le lancement de nouveaux modules qui permettent désormais d'initier les futurs enseignants à la SI dès la première année de formation au sein de notre Haute école pédagogique. Ainsi, cela s'est fait en cohérence avec les résultats issus de cette recherche qui mettent notamment en avant l'importance de poser des bases théoriques solides en informatique dès le début de la formation. Un atelier disciplinaire « Savoirs disciplinaires en science informatique » a donc vu le jour et propose un certain nombre d'apprentissages clés en SI avec le développement de

concepts tels que les machines, les algorithmes, la programmation, les réseaux, etc. En réponse au besoin affiché de lien entre ce qui est vécu en formation et le quotidien des élèves, une focale est posée sur la thématique « Informatique et société » et chaque concept est remis en perspective par rapport à des situations quotidiennes et vécues par les futurs enseignants et leurs élèves. Cet aspect permet justement de répondre au manque de sens affiché dans certaines situations pédagogiques rencontrées dans le cadre du module ayant fait l'objet de cet article.

À l'issue de cet atelier disciplinaire, permettant de lisser les compétences et connaissances des futurs enseignants, nous avons axé les deux années suivantes sur la didactique de l'informatique et sur la mise en situation concrète des apprentissages vécus en formation. Ceci permet aux futurs enseignants d'être tout d'abord rassurés dans la mise en œuvre et la préparation de leurs activités d'enseignement en SI (via une culture informatique commune) et de bénéficier de ressources, notamment, et d'expérience au cours des deux années suivantes pour enseigner en pleine confiance et en maîtrisant les concepts à transmettre.

Références

1. Balanskat, A., Engelhardt, K. : Computing our future : Computer programming and coding-Priorities, school curricula and initiatives across Europe. European Schoolnet (2014)
2. Baron, G.L., Drot-Delange, B. : L'informatique comme objet d'enseignement à l'école primaire française ? Mise en perspective historique. Revue française de pédagogie. Recherches en éducation (195), 51–62 (2016)
3. Bell, T., Alexander, J., Freeman, I., Grimley, M. : Computer science unplugged : School students doing real computing without computers. The New Zealand Journal of Applied Computing and Information Technology **13**(1), 20–29 (2009)
4. Bell, T., Vahrenhold, J. : CS Unplugged—How is it used, and does it work ? In : Adventures between lower bounds and higher altitudes, pp. 497–521. Springer (2018)
5. Bugmann, J. : Quand les robots s'invitent à l'école. L'Éducateur (3), 15–16 (2019)
6. Bugmann, J., Karsenti, T. : Quand les robots entrent en classe. Formation et profession : revue scientifique internationale en éducation **26**(1), 142–145 (2018)
7. Chevalier, M., Gautschi, H. : L'usage des robots en classe, une expérience suisse et belge (2019)
8. Chevalier, M., Giang, C., El-Hamamsy, L., Bonnet, E., Papaspyros, V., Pellet, J.P., Audrin, C., Romero, M., Baumberger, B., Mondada, F. : The role of feedback and guidance as intervention methods to foster computational thinking in educational robotics learning activities for primary school. Computers & Education (2022)
9. Clark, R.E. : Évaluer l'enseignement à distance. Distances et savoirs **7**(1), 93–112 (2009)
10. Dillenbourg, P. : Design for classroom orchestration. Computers & Education (69), 485–492 (2013)
11. Dowek, G. : Les origines de l'informatique. Cahiers philosophiques (2), 7–15 (2015)

12. Drot-Delange, B. : Enseigner l'informatique débranchée : analyse didactique d'activités. In : AREF. pp. 1–13 (2013)
13. El-Hamamsy, L., Chessel-Lazzarotto, F., Bruno, B., Roy, D., Cahlikova, T., Chevalier, M., Parriaux, G., Pellet, J.P., Lanarès, J., Zufferey, J.D., et al. : A computer science and robotics integration model for primary school : evaluation of a large-scale in-service K-4 teacher-training program. Education and Information Technologies **26**(3), 2445–2475 (2021)
14. Goode, J., Margolis, J., Chapman, G. : Curriculum is not enough : The educational theory and research foundation of the exploring computer science professional development model. In : Proceedings of the 45th ACM technical symposium on Computer science education. pp. 493–498 (2014)
15. Grandbastien, M. : Quelle actualité pour les questions abordées lors du premier colloque de didactique de l'informatique de 1988 ? In : Actes du colloque Didapro 8 – DidaSTIC (2020)
16. Greff, É. : Le « jeu de l'enfant-robot » : une démarche et une réflexion en vue du développement de la pensée algorithmique chez les très jeunes enfants. Thèse de doctorat, Paris VII (1996)
17. Marquet, P. : Intérêt du concept de conflit instrumental pour la compréhension des usages des eiah. In : EIAH 2005 (Environnements Informatiques pour l'Apprentissage Humain). pp. 383–388. INRP (2005)
18. Ozturk, Z., Dooley, C.M., Welch, M. : Finding the hook : Computer science education in elementary contexts. Journal of Research on Technology in Education **50**(2), 149–163 (2018)
19. Papadakis, S. : Robots and robotics kits for early childhood and first school age. International Journal of Interactive Mobile Technologies (2020)
20. Papert, S. : Mindstorms : Computers, children, and powerful ideas. Basic Books (1980)
21. Parriaux, G., Pellet, J.P., Chessel-Lazzarotto, F., Chevalier, M., Page, E., Roy, D. : Que pourrait comprendre l'enseignement de la science informatique dans la scolarité obligatoire ? Avec quelle progression ? Bulletin CIIP (5), 13–16 (2020)
22. Repenning, A., Lamprou, A., Petralito, S., Basawapatna, A. : Making computer science education mandatory : Exploring a demographic shift in Switzerland. In : Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education. pp. 422–428 (2019)
23. Romero, M., Duflot-Kremer, M., Viéville, T. : 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 (2018)
24. Spach, M. : Apprentissage d'un concept informatique à l'école primaire : l'automate. AREF 2016 pp. 1256–1258 (2015)
25. Tardif, J., Meirieu, P. : Stratégie en vue de favoriser le transfert des connaissances. Vie pédagogique (98), 4–7 (1996)
26. Trouche, L. : Construction et conduite des instruments dans les apprentissages mathématiques : nécessité des orchestrations. Thèse de doctorat (2003)
27. Tsarava, K., Moeller, K., Pinkwart, N., Butz, M., Trautwein, U., Ninaus, M. : Training computational thinking : Game-based unplugged and plugged-in activities in primary school. In : European conference on games based learning. pp. 687–695. Academic Conferences International Limited (2017)

Création d'exemples résolus avec objectifs étiquetés pour l'apprentissage de la programmation avec Python

Olivier Goletti[1,3][0000−0002−1610−4985], Florian De Pierpont[2] et Kim Mens[1][0000−0003−0303−1630]

[1] ICTEAM/INGI, UCLouvain, Louvain-la-Neuve, Belgique
firstname.lastname@uclouvain.be
[2] EPL, UCLouvain, Louvain-la-Neuve, Belgique
florian.depierpont@student.uclouvain.be
[3] LIACS, Leiden University, Leiden, Pays-Bas

Résumé L'enseignement de la programmation peut profiter de l'utilisation de stratégies d'instruction explicites pour diminuer la charge cognitive et favoriser le transfert des apprentissages. Une de ces stratégies est l'utilisation d'exemples résolus avec objectifs étiquetés. Nous avons utilisé une méthodologie d'analyse de tâche pour extraire à des experts en programmation les connaissances nécessaires à la création de tels exemples résolus. Cet article présente la méthodologie utilisée et propose un document de formation réutilisable par d'autres membres de la communauté enseignante d'informatique.

Mots clés : enseignement de l'informatique · exemples résolus · objectifs étiquetés · stratégie d'instruction explicite

1 Introduction

Comme l'enseignement de l'informatique prend de l'ampleur, la recherche en didactique également. On enseigne maintenant l'informatique de plus en plus tôt et dans de plus en plus de pays [20]. Une méthodologie souvent utilisée est l'apprentissage par problèmes (en anglais : problem based learning - PBL) [12].

Notre analyse d'un cours d'introduction à la programmation utilisant PBL a mis en évidence l'intérêt d'utiliser des stratégies d'instruction explicite pour diminuer la charge cognitive des étudiants et promouvoir le transfert des apprentissages au sein du cours de programmation. Un besoin de techniques d'instructions explicites a été identifié [6,8] . En effet, les étudiants profitent de plus d'accompagnement [18] en particulier dans le cadre de méthodologies d'enseignement moins guidées [10].

Plusieurs stratégies d'instructions explicites ont déjà été testées récemment pour enseigner la programmation [5,11,15,22]. En particulier, l'utilisation d'exemples résolus avec objectifs étiquetés (en anglais : Subgoal Labeled Worked Examples - SLWE) a fait l'objet de plusieurs études par Margulieux et al. [13,15,16]. Les SLWEs utilisés en informatique tels qu'introduits par Margulieux et al. ont été

notamment testés en Java ou pour des langages de programmation visuels (par blocs) seulement [13,15].

Dans le cadre de sa thèse, le premier auteur a identifié des stratégies d'instruction explicite et a formé des tuteurs d'un cours d'informatique de première année de bachelier en ingénieur civil à les utiliser en séance de travaux pratiques, notamment les SLWEs [8,7] . Afin de poursuivre avec la stratégie des SLWEs et dans le but de rédiger un document (appelé document de formation) pour favoriser son utilisation par les tuteurs, nous présentons dans cet article le travail de création de SLWEs qui a été réalisé.

Pour créer les SLWEs, nous avons analysé des résolutions d'exercices faites par des experts. La méthodologie pour cette analyse est l'analyse de tâche (Task Analysis) [3], une méthode efficace pour extraire des procédures implicites que des experts d'un domaine utilisent de façon implicite pour résoudre des tâches complexes. En particulier, nous avons adapté la méthodologie d'analyse de tâche par résolution de problème (Task analysis by problem solving - TAPS) de Catrambone [2] utilisée et recommandée par Margulieux et al. [15].

Cet article présente donc le contexte théorique qui justifie l'utilisation de stratégies d'instruction explicite pour l'enseignement de l'informatique. Nous présentons notre adaptation d'une méthodologie de création de SLWEs ainsi que les SLWEs que nous avons créés.

2 Contexte théorique

Nous présentons d'abord dans cette section deux théories de l'éducation qui sous-tendent le choix de stratégies d'instructions explicites et en particulier d'exemples résolus avec objectifs étiquetés (SLWE) comme stratégie d'instruction. Nous donnons ensuite plus de détails sur cette stratégie.

2.1 Cadres conceptuels

Théorie de la charge cognitive La théorie de la charge cognitive (Cognitive Load Theory - CLT) [21] est une théorie de l'instruction basée sur l'architecture cognitive humaine. La CLT considère que l'humain n'a qu'une mémoire de travail limitée. L'impact sur l'instruction de la CLT est qu'il faut tenter de réduire la charge sur la mémoire de travail quand on enseigne. Trop de contexte ou d'informations superflues augmentent la charge cognitive de l'apprenant. Il faut minimiser cette charge cognitive et soutenir l'application de compétences génériques [17], par exemple en utilisant des exemples résolus [9].

Transfert des apprentissages Le transfert des apprentissage est la remobilisation de connaissances apprises dans un nouveau contexte [4,19]. Ce transfert est un processus actif qui peut être favorisé, notamment par des incitations ou des mises en évidence [4]. L'impact sur l'instruction du transfert des apprentissages est l'utilisation de stratégies qui invitent l'apprenant à se rappeler ses connaissances acquises et à les réappliquer dans d'autres contextes, par exemple

en mettant en évidence les étapes génériques d'une solution avec des objectifs identifiés et étiquetés.

2.2 Exemples résolus avec objectifs étiquetés (SLWEs)

Exemples résolus L'utilisation d'exemples résolus permet d'accélérer l'apprentissage d'une procédure de résolution en illustrant les différentes étapes abstraites d'une procédure de résolution d'un problème dans un exemple concret [15]. Cette méthode a par ailleurs déjà été utilisée dans le cadre de l'apprentissage de l'informatique. Cependant, pour donner un exemple, on prend toujours un contexte spécifique. Un novice amalgame parfois les détails du contexte avec le concept enseigné dans un tel exemple. Une façon d'éviter cet effet est de montrer plusieurs exemples différents et d'identifier ce qui est générique dans la procédure de résolution entre ces exemples.

Apprentissages par objectifs étiquetés L'apprentissage par objectifs identifiés et étiquetés aide l'apprenant à distinguer la procédure générique de résolution d'un problème [1]. Les apprenants sont ainsi guidés dans le transfert de leurs apprentissages lorsqu'ils sont amenés à résoudre des exercices similaires. Les objectifs étiquetés doivent cependant être identifiés avec soin.

Exemples résolus avec objectifs étiquetés (SLWEs) La combinaison de ces deux techniques pour enseigner l'informatique a été proposée par Margulieux et al. [15] en étiquetant les objectifs identifiés dans les exemples résolus directement. Ces objectifs étiquetés permettent d'identifier les étapes génériques de la résolution en les distinguant du contexte spécifique des exemples résolus. Ces étapes peuvent être réutilisées lors de la lecture ou l'écriture de code similaire. Cette stratégie favorise le transfert des apprentissages et diminue la charge cognitive pour les apprenants. Dans leur article, ils proposent une méthodologie pour créer des SLWEs et donnent également des exemples de SLWEs pour la lecture et l'écriture de concepts de base de programmation en Java.

L'utilisation de ces exemples résolus avec objectifs étiquetés (Subgoal Labeled Worked Examples - SLWE) a été montrée comme efficace pour l'enseignement de la programmation [13,15,16] ainsi que dans des supports en ligne de formation d'enseignants [14]. L'objectif du présent article n'est pas d'en discuter l'efficacité mais bien d'adpapter la stratégie et de créer de nouveaux SLWEs pour l'apprentissage du Python.

3 Méthodologie

Nous discutons ici la méthodologie utilisée pour extraire les connaissances des experts à partir de résolutions de problèmes. Ceci permettra de documenter les procédures de résolutions de problèmes, d'identifier les objectifs étiquetés et de rédiger un document de formation avec des exemples résolus. Ce document servira à former des tuteurs à utiliser les SLWEs dans un cours d'informatique.

3.1 Analyse de tâche par résolution de problème

L'objectif de l'analyse de tâche est de faire extraire à un expert d'un domaine ses connaissances implicites et sa façon de résoudre un problème [3]. Plusieurs techniques existent pour ce faire, basées notamment sur l'observation de résolution de problèmes par un expert, la prise de note, la rédaction d'une procédure, etc. En effet, les experts résolvent les problèmes différemment que les novices et leurs explications sont typiquement incomplètes et insuffisantes pour un novice. Même des enseignants qui s'y essaient n'y arrivent pas spécialement [1].

La méthodologie utilisée et recommandée par Margulieux et al. est la méthodologie d'analyse de tâche par résolution de problème (TAPS) de Catambone [2]. Dans TAPS, l'expert du sujet est appelé SME (Subject-Matter Expert), on l'appellera "l'expert" dans la suite de cet article. Quant à l'expert en extraction de connaissance (Knowedlge Extractor Expert - KEE), on l'appellera "l'analyste". L'expert doit identifier des tâches qui se rapportent aux concepts à traiter et les résoudre devant l'analyste. L'analyste est de préférence novice dans le sujet à analyser et doit prendre des notes sur la technique de résolution de l'expert ou des experts.

Les étapes recommandées par TAPS sont les suivantes :

— L'expert identifie les exercices qu'un apprenant devrait pourvoir résoudre s'il maîtrise la matière.
— L'expert résout un de ces exercices.
— L'analyste prend des notes détaillées sur la raison de chaque étape de résolution et demande à l'expert de justifier chacune de ces étapes, ce que l'expert a parfois du mal à verbaliser.
— L'analyste reprend ses notes, les réorganise, extrait les procédures et justifications.
— L'expert résoud un nouveau problème.
— L'analyste complète ses notes pour combler les lacunes et résoudre les incohérences.
— L'analyste tente de résoudre un problème similaire sur base de ses notes.
— L'analyste réinterroge l'expert jusqu'à pouvoir résoudre les exercices par lui-même.

Finalement, des échanges entre les experts et l'analyste aboutissent à la rédaction d'un document. Ce document décrit pour chaque concept abordé la procédure de résolution, les objectifs étiquetés permettant de structurer les éléments génériques et des exemples résolus annotés illustrant la procédure de résolution.

3.2 TAPS adapté

La méthodologie utilisée dans notre travail est très similaire à TAPS. Les experts étaient au nombre de quatre : le troisième auteur, professeur d'informatique depuis plus de vingt ans et co-titulaire du cours d'introduction à la programmation depuis quatre ans ; le premier auteur, assistant pour le cours d'introduction à la programmation depuis six ans et doctorant en didactique de l'informatique ; ainsi que deux doctorants en informatique, aussi assistants du

cours. L'analyste quant à lui est le second auteur et est étudiant de master en informatique, il n'était donc pas novice en programmation comme recommandé par TAPS, cependant il n'avait jamais suivi de cours en Python.

Les experts ont à chaque fois identifié les exercices à maîtriser pour les différents concepts abordés dans le cours (voir Table 1). L'analyste invitait alors un ou plusieurs experts si nécessaire en visioconférence pour enregistrer sa résolution des exercices sélectionnés. Les experts travaillaient dans un IDE simple avec leur écran partagé. L'analyste posait des questions de justification des différentes étapes aux experts. L'analyste prenait des notes et pouvait éventuellement consulter l'enregistrement après coup. Si nécessaire cependant il pouvait faire appel à un autre expert pour le même concept afin de combler les lacunes ou résoudre les incohérences de ses notes.

Comme l'analyste est un étudiant de master en informatique, nous avons décidé de ne pas faire les deux dernières étapes de TAPS qui attendent du KEE qu'il tente de résoudre les problèmes par lui-même sur base de ses notes. Nous sommes conscient que cette expertise en programmation peut introduire un biais dans les SLWEs. Cependant, nous estimons que la richesse des différents profils de l'équipe de recherche (chercheur en didactique de l'informatique et assistant, enseignant, étudiant) a enrichi le regard et l'attention portés aux choix des SLWEs et constitue un avantage pour ce travail.

Concept	**Lecture de code**	**Écriture de code**	**Adaptation Python de [15]**
Affectation	x	x	x
Condition	x	x	x
Boucle	x	x	x
Fonction	x	x	x
Parcours de chaîne ou de liste		x	
Lecture de fichier		x	
Écriture de fichier		x	
Création et mise à jour de dictionnaire		x	
Création de classe		x	
Ajout de noeud dans une liste chaînée		x	
Retrait de noeud dans une liste chaînée		x	

Table 1. Concepts pour lesquels des SLWEs ont été créés en Python

4 Résultats

Nous avons créé des SLWEs pour les concepts donnés dans la Table 1. L'ensemble du document de formation est disponible en ligne [4]. Afin d'illustrer la

4. https://dial.uclouvain.be/pr/boreal/object/boreal:260190

production réalisée, voici les objectifs étiquetés pour écrire un programme qui lit un fichier en Python et l'exemple résolu associé (voir Fig. 1) :

1. **Ouvrir** le fichier
   (a) Identifier le nom et le chemin du fichier (typiquement dans `filename`)
   (b) Choisir le mode `"r"`
   (c) Ouvrir le fichier avec :
   — Soit : `with open(filename , mode) as f :`
   — Soit : `f = open(filename , mode)`
2. **Traitement** du fichier en fonction du format
   (a) Parcours des lignes
   — ligne par ligne avec `f.readline()`
   — en itérant sur les lignes avec `for line in f:`
   (b) Traitement des lignes
   — Retrait des blancs en début et fin de ligne avec `line.strip()`
   — Séparer les éléments en fonction du format avec `line.split()`
   — Convertir les éléments en fonction du type attendu
   (c) Traiter les erreurs de formatage du fichier (ignorer ligne, `raise ValueError`)
3. **Fermeture** du fichier
   — Avec un `with`, il n'y a rien à faire
   — Sinon, avec un `f.close()`
4. Gérer les **exceptions** susceptibles de se produire durant le traitement du fichier (typiquement `IOError`)
   (a) Mettre le code dans un `try : ... except :`
   (b) Traiter les exceptions par le suite avec des `except error_type: ...`

Le document de formation[4] qui contient tous les SLWEs ainsi qu'une description de leur utilisation a déjà servi à la formation de tuteurs au premier semestre de l'année académique 2021-2022. Septs tuteurs ont ainsi utilisé ces SLWEs. La manière dont ils ont utilisé cette stratégie pendant leur séance d'encadrement fera l'objet d'une publication future.

5 Conclusion

Cet article a pour objectif de contribuer à la didactique de l'informatique en proposant des exemples résolus avec objectifs étiquetés (SLWEs) pour l'enseignement de la programmation en Python. L'utilisation de ces SLWEs est justifié dans le cadre des implications en terme d'instruction des théories de la charge cognitive et du transfert des apprentissages. La création de SLWEs était nécessaire car il n'en existait pas encore pour les concepts étudiés en Python. Générer les SLWEs pour les concepts sélectionnés a été possible en utilisant une méthodologie d'analyse de tâche documentée par ailleurs dans la littérature. L'analyse de tâche TAPS a été adaptée pour extraire les connaissances d'un expert en

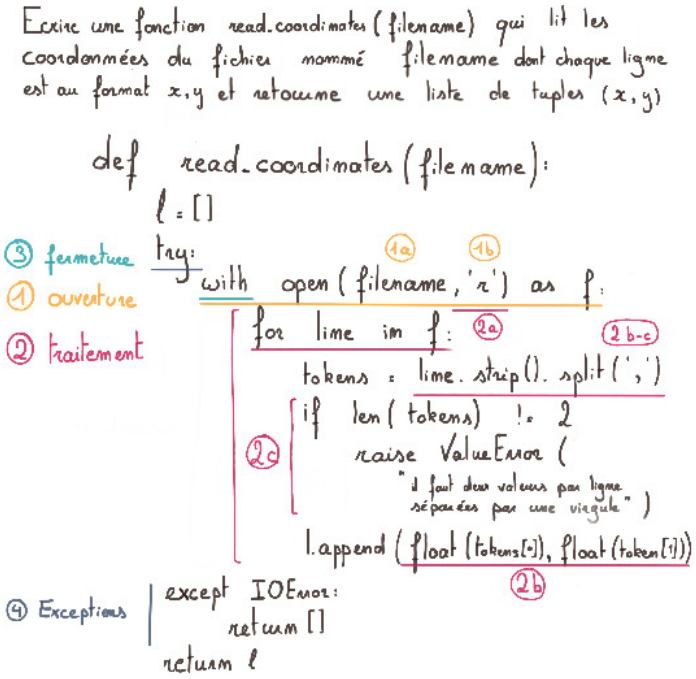


Figure 1. Exemple résolu de lecture de fichier en Python annoté d'objectifs étiquetés

programmation. Une liste de concepts vu dans le cadre d'un premier cours d'informatique de niveau universitaire a été sélectionnée, les enregistrements des résolutions d'exercices pour chaque concept par plusieurs experts ont été analysés par notre analyste, le KEE. Il en a tiré pour chaque concept une procédure de résolution de problèmes et a identifié des objectifs étiquetés pour chaque procédure. Ces procédures et objectifs étiquetés ont été discutés et validés par les auteurs. Un document de formation a ensuite été rédigé pour former les tuteurs du cours qui participaient à l'expérience. Dans ce document, les procédures pour chaque concept sont détaillées, les objectifs étiquetés sont fournis et un ou plusieurs exemples résolus annotés sont fournis. Ce document a été utilisé pour former sept tuteurs à l'utilisation des SLWEs dans leur encadrement d'un cours d'introduction à la programmation en Python. Ce document est publiquement disponible[4] et réutilisable par d'autres membres de la communauté enseignante d'informatique.

Références

1. Catrambone, R. : The Subgoal Learning Model : Creating Better Examples So That Students Can Solve Novel Problems p. 22
2. Catrambone, R. : Task analysis by problem solving (TAPS) : Uncovering expert knowledge to develop high-quality instructional materials and training. In : Learning and Technology Symposium, Columbus, GA (2011)
3. Clark, R., Feldon, D., Van Merrienboer, J.J.G., Yates, K., Early, S. : Cognitive task analysis. In : Handbook of Research on Educational Communications and Technology, chap. 43, pp. 577–593 (Jan 2008)
4. Council, N.R. : How People Learn : Brain, Mind, Experience, and School : Expanded Edition. National Academies Press (2000)
5. Ericson, B.J., Margulieux, L.E., Rick, J. : Solving Parsons Problems Versus Fixing and Writing Code. In : Proceedings of the 17th Koli Calling International Conference on Computing Education Research. pp. 20–29. Koli Calling '17, ACM, New York, NY, USA (2017). https://doi.org/10.1145/3141880.3141895
6. Goletti, O. : En quoi le dispositif mis en œuvre dans le cours d'introduction à l'informatique en BAC1 ingénieur civil basé sur l'apprentissage par problèmes soutient les processus du transfert des apprentissages : l'encodage et l'accessibilité aux connaissances ? Tech. rep., UCLouvain (2019), `http://hdl.handle.net/2078.1/245579`
7. Goletti, O. : Promoting Learning Transfer in Computer Science Education by Training Teachers to use Explicit Programming Strategies. In : ICER '17. pp. 411–412. ACM, Virtual Event USA (Aug 2021). https://doi.org/10.1145/3446871.3469776
8. Goletti, O., Mens, K., Hermans, F. : Tutors' Experiences in Using Explicit Strategies in a Problem-Based Learning Introductory Programming Course. In : ITiCSE '21. p. 7. ACM Press, Virtual Event, Germany (Jun 2021). https://doi.org/10.1145/3430665.3456348
9. Kirschner, P.A. : Cognitive load theory : Implications of cognitive load theory on the design of learning. Learning and Instruction **12**(1), 1–10 (Feb 2002). https://doi.org/10.1016/S0959-4752(01)00014-7
10. Kirschner, P.A., Sweller, J., Clark, R.E. : Why Minimal Guidance During Instruction Does Not Work : An Analysis of the Failure of Constructivist, Discovery, Problem-Based, Experiential, and Inquiry-Based Teaching. Educational Psychologist **41**(2), 75–86 (Jun 2006). https://doi.org/10.1207/s15326985ep4102$_1$
11. Loksa, D., Ko, A.J., Jernigan, W., Oleson, A., Mendez, C.J., Burnett, M.M. : Programming, problem solving, and self-awareness : Effects of explicit guidance. In : Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems. pp. 1449–1461. ACM (2016)
12. Luxton-Reilly, A., Simon, Albluwi, I., Becker, B.A., Giannakos, M., Kumar, A.N., Ott, L., Paterson, J., Scott, M.J., Sheard, J., Szabo, C. : Introductory programming : A systematic literature review. In : ITiCSE '23. pp. 55–106. ITiCSE 2018 Companion, ACM, New York, NY, USA (Jul 2018). https://doi.org/10.1145/3293881.3295779
13. Margulieux, L., Catrambone, R., Guzdial, M. : Subgoal Labeled Worked Examples Improve K-12 Teacher Performance in Computer Programming Training. In : Proceedings of the Annual Meeting of the Cognitive Science Society. vol. 35 (2013)

14. Margulieux, L.E., Catrambone, R., Guzdial, M. : Employing subgoals in computer programming education. Computer Science Education **26**(1), 44–67 (Jan 2016). https://doi.org/10.1080/08993408.2016.1144429, `https://www.tandfonline.com/doi/citedby/10.1080/08993408.2016.1144429`, publisher : Routledge
15. Margulieux, L.E., Morrison, B.B., Decker, A. : Design and Pilot Testing of Subgoal Labeled Worked Examples for Five Core Concepts in CS1. In : ITiCSE '19. pp. 548–554. ACM Press, Aberdeen, Scotland Uk (2019). https://doi.org/10.1145/3304221.3319756
16. Morrison, B.B., Margulieux, L.E., Guzdial, M. : Subgoals, Context, and Worked Examples in Learning Computing Problem Solving. In : ICER '11. pp. 21–29. ACM Press (2015). https://doi.org/10.1145/2787622.2787733
17. Sweller, J., Ayres, P., Kalyuga, S. : Cognitive Load Theory, Volume 1 of Explorations in the Learning Sciences, Instructional Systems and Performance Technologies. Springer, New York (2011)
18. Sweller, J., van Merriënboer, J.J., Paas, F. : Cognitive architecture and instructional design : 20 years later. Educational Psychology Review pp. 1–32 (2019)
19. Tardif, J. : Le Transfert Des Apprentissages. Editions logiques (1999)
20. Vahrenhold, J., Caspersen, M., Berry, G., Gal-Ezer, J., Kölling, M., McGettrick, A., Nardelli, E., Pereira, C., Westermeier, M. : Informatics Education in Europe : Are We All In The Same Boat ? (2017)
21. van Merriënboer, J.J.G., Sweller, J. : Cognitive Load Theory and Complex Learning : Recent Developments and Future Directions. Educational Psychology Review **17**(2), 147–177 (Jun 2005). https://doi.org/10.1007/s10648-005-3951-0
22. Xie, B., Nelson, G.L., Ko, A.J. : An explicit strategy to scaffold novice program tracing. In : SIGCSE TS '49. pp. 344–349. ACM (2018)

Conception et évaluation du jeu sérieux *Pyrates :* agencement du milieu didactique pour la transition Scratch-Python

Matthieu BRANTHÔME[1]

[1] Université de Bretagne Occidentale, CREAD - EA 3875, Brest, France
matthieu.branthome@univ-brest.fr

Résumé. Cette communication présente la conception et l'évaluation de l'application en ligne *Pyrates*, un jeu sérieux qui vise l'introduction de la programmation Python en classe de seconde. Nous y exposons plus particulièrement l'aménagement du milieu didactique avec pour objectif de s'approcher des facilités offertes par les environnements de programmation basés sur les blocs. Afin d'évaluer nos choix, nous avons testé l'application sur le terrain auprès de 240 élèves de seconde et ainsi récolté environ 70.000 traces d'activités générées automatiquement. Ces traces sont complétées par un questionnaire en ligne. Nous montrons que certains choix de conception ont les effets attendus. Ainsi, la création d'un « mémo programmation » permet la découverte des notions algorithmiques tout en offrant un support de référence pour la syntaxe Python. La facilitation des copiés-collés depuis ce mémo limite la saisie au clavier et soutient la tâche de structuration des programmes. L'intégration d'un analyseur syntaxique conçu pour les débutants confère une grande autonomie aux élèves dans le traitement des erreurs. Cependant, d'autres choix ont des impacts plutôt néfastes. La création d'un panneau de commande pour l'exécution des programmes s'avère être entièrement au service d'une démarche de programmation par essais-erreurs ou de stratégies de « contournement didactique ».

Mots-clés : Apprentissage programmation, transition Scratch-Python, milieu didactique, jeu sérieux, EIAH, learning analytics.

1 Introduction

Au fil des ans, en France comme à l'international, la programmation par blocs semble être devenue une des modalités préférentielles d'introduction de la programmation informatique auprès des plus jeunes. La recherche a démontré les bénéfices de cette approche comparativement à l'introduction classique à l'aide de langages textuels (Armoni et al., 2015 ; Price et Barnes, 2015 ; Weintrop et Wilensky, 2017). Dans le même temps, la programmation basée sur du texte reste très majoritairement utilisée au lycée par les élèves plus âgés et par les étudiants à l'université. La plupart des apprenants ayant débuté la programmation sous la forme de blocs devront donc changer de modalité en s'initiant à la programmation textuelle. Comment les aider dans cette transition ?

C'est une des questions vives qui occupent le champ de recherche qui porte sur l'introduction à la programmation (Weintrop, 2019).

Afin d'accompagner les élèves dans ce changement, nous avons développé un jeu sérieux (Alvarez, 2007) visant l'introduction du langage de programmation Python à des élèves de seconde. Ainsi, l'application en ligne *Pyrates* [1,2] prend la forme d'un jeu de plateforme permettant le contrôle d'un avatar en Python. Nous présentons dans cette contribution la conception du jeu et en particulier l'agencement du milieu didactique (Brousseau, 1998). Nous évaluons ensuite nos choix de conception en analysant l'appropriation du jeu par les élèves dans les classes.

Nous exposons d'abord notre cadre théorique en synthétisant quelques concepts de la Théorie des situations didactiques de Brousseau (1998). Ensuite, nous présentons notre revue de travaux en lien avec la transition bloc-texte, avant de détailler notre méthodologie et les résultats en découlant. Finalement, nous concluons et formulons les perspectives et les prolongements de ce travail.

2 Cadre théorique et problématique

Nous nous plaçons dans le cadre de la Théorie des situations didactiques élaborée par Brousseau (1998). Nous exposons ci-dessous les concepts que nous allons mobiliser.

Brousseau définit une **situation** comme : « *une situation problème qui nécessite une adaptation, une réponse de l'élève.* » (1981, p. 112). Il établit le **milieu didactique** comme étant « c*onstitué 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). Cet auteur considère que les activités proposées aux élèves doivent tendre vers « *une sorte d'idéal vers lequel il s'agit de converger* » (1986, p. 50) : la **situation adidactique**. Ce type de problème doit permettre à l'élève d'agir de son propre mouvement, guidé uniquement par la logique interne de la situation. Autrement dit, sans s'appuyer sur les intentions didactiques de l'enseignant qui se refuse à intervenir comme proposeur de la connaissance qu'il cible. Le milieu didactique doit être en mesure de fournir des **rétroactions** à l'élève en réponse à ses actes. Le milieu doit ainsi pouvoir lui dispenser des sanctions positives ou négatives lui permettant d'ajuster son action. (Bessot, 2003).

Chacun des huit niveaux du jeu *Pyrates* est conçu sur le modèle de la situation adidactique : une situation ludique qui doit amener les élèves à écrire un programme mettant en œuvre certaines notions algorithmiques. Nous ne développons pas cet aspect de la conception dans cette communication bien qu'il soit nécessaire de le garder en tête pour comprendre la suite. Nous nous concentrons ici sur l'agencement du milieu didactique des situations. Ainsi, les questions de recherche adressées par ce travail sont :

- QR1 : quels sont les avantages des environnements de programmation basés sur les blocs en comparaison de ceux basés sur du texte ?
- QR2 : comment concevoir un environnement d'apprentissage de Python qui comporte certaines des caractéristiques avantageuses des éditeurs basés sur les blocs ?
- QR3 : lors des expérimentations sur le terrain, les élèves se saisissent-ils de ces caractéristiques ? Si oui, quel usage en font-ils ?

3 État de l'art

Notre revue de travaux est constituée de deux parties. Nous présentons d'abord des applications existantes conçues dans le but d'accompagner la transition des blocs vers le texte. Nous exposons ensuite les résultats de travaux analysant les différences intrinsèques entre ces deux types d'environnements.

3.1 Applications existantes

Afin de soutenir la transition des blocs vers le texte, plusieurs pistes s'appuyant sur des environnements numériques ont été explorées. Parmi ces dispositifs, nous distinguons trois types d'environnements : composés unidirectionnels, composés bidirectionnels et hybrides.

Les environnements composés unidirectionnels comprennent deux vues. L'une permet l'édition des programmes à l'aide de blocs, ces programmes étant convertis automatiquement dans un langage textuel cible dans l'autre vue. Ce langage cible n'est pas directement modifiable, il peut uniquement être consulté et éventuellement exécuté par les utilisateurs. C'est par exemple le cas de l'application *Block2Py* (Declercq et Nény, 2020) dont les blocs miment la syntaxe de Python. L'environnement *Patch* (Robinson, 2016) présente un fonctionnement similaire se basant sur Scratch.

Les environnements composés bidirectionnels sont structurés de la même manière que ceux qualifiés d'unidirectionnels. Ce à quoi s'ajoute la possibilité de créer ou de modifier les programmes directement dans la vue textuelle. Cela entraîne automatiquement la traduction ou la mise à jour du programme dans la vue blocs. Parmi les implémentations existantes, nous pouvons citer *PencilCode* (Bau et al., 2015) qui vise l'apprentissage de Javascript et plus récemment de Python (Andrews et al., 2021), et *BlocEditor* (Matsuzawa et al., 2015) qui propose une approche similaire pour Java. *Blockpy* (Bart et al., 2017) permet également de programmer en Python.

Enfin, les environnements hybrides sont basés sur des cadres (« *framed-based* ») qui combinent blocs et textes en une seule vue. Les structures de haut niveau (boucles, conditionnelles, etc.) peuvent être insérées par glisser-déposer ou au clavier en utilisant des raccourcis. Le code de niveau expression est introduit par édition de texte traditionnelle soutenue par l'auto-complétion. *GP* est une mise en œuvre issue d'une étude exploratoire (Monig et al., 2015). *Stride* met à disposition des enseignants une implémentation opérationnelle pour le langage Java (Kölling et al., 2015, 2017). Depuis peu, *Strype* (Kyfonidis et al., 2021) offre un environnement hybride dédié à l'édition Python.

Certains de ces environnements ont été évalués par des études empiriques. Même s'ils semblent prometteurs (Alrubaye et al., 2019 ; Blanchard et al., 2020), les effets positifs sur les apprentissages ne sont pas toujours démontrés (Brown et al., 2021). Ces logiciels intègrent une traduction automatique vers du texte depuis un environnement d'édition par blocs, ou proposent un environnement mixte blocs-texte. Nous proposons d'explorer une troisième voie en concevant un environnement d'édition purement textuel intégrant certaines caractéristiques avantageuses des éditeurs de blocs. Cette modalité, complémentaire des deux autres, pourra trouver sa place dans une étape plus avancée de la transition blocs-texte.

3.2 Avantages des environnements basés sur les blocs

Plusieurs auteurs (Bau et al., 2017 ; Kolling et al. 2015 ; Weintrop, 2019) ont analysé les différences intrinsèques et les avantages des environnements de programmation par blocs comparés à ceux basés sur du texte. Nous résumons ci-dessous les résultats qui en ressortent. Cela permet de répondre à notre question de recherche QR1.

- **R1.1 : présence d'un catalogue de commande.** Les environnements de programmation par blocs présentent à l'utilisateur un panneau recensant tous les blocs existants organisés de manière thématique et conceptuelle. Les utilisateurs novices peuvent ainsi découvrir de nouveaux concepts ou se remettre en mémoire ceux précédemment acquis. Dans les environnements textuels, l'existence et la syntaxe des structures de code doivent être connues par avance des programmeurs.
- **R1.2 : nombre réduit d'éléments significatifs**. Les langages de programmation textuels sont constitués de nombreux éléments significatifs (mots-clés, signes typographiques, etc.). Cette notation dense constitue un obstacle pour les novices car elle peut surcharger leur mémoire de travail. Les programmeurs expérimentés ont appris au fil du temps à interpréter le code en plus gros morceaux (« *chunks* »). Les blocs permettent de réduire la charge cognitive des programmeurs débutants en leur permettant d'appréhender les commandes en morceaux plus importants.
- **R1.3 : aide à la structuration des programmes**. Les programmes basés sur du texte sont structurés à l'aide de parenthèses, d'accolades ou de l'indentation des lignes. La mécanique d'agencement et de maintien de cette structure est un défi pour les débutants et entraine très souvent des erreurs syntaxiques ou sémantiques. Ces contraintes sont moins présentes dans les langages de blocs dans la mesure ou la structuration des programmes est guidée et maintenue par la forme des blocs.
- **R1.4 : peu de saisies au clavier**. La composition de programmes par glisser-déposer des blocs limite la difficulté de la saisie et de la recherche des signes typographiques sur le clavier. L'acte purement mécanique de taper le texte du programme peut constituer un obstacle cognitif et moteur pour les jeunes apprenants. La nécessité de le faire ajoute une charge supplémentaire et des distractions cognitives lorsqu'il faut corriger les inévitables erreurs de frappe.
- **R1.5 : absence d'erreurs syntaxiques.** Les systèmes basés sur des blocs permettent d'éviter la plupart des erreurs de syntaxe à la faveur d'une manipulation globale et contrainte des structures. Dans les systèmes textuels, ces erreurs sont nombreuses et les messages d'erreur sont généralement vagues dans leur formulation. L'interprétation de ces messages est une compétence non triviale que les novices mettent beaucoup de temps à maîtriser.
- **R1.6 : contrôle et visibilité de l'exécution**. Les environnements basés sur les blocs facilitent le contrôle et améliorent la visibilité de l'exécution des programmes. Ils permettent de mettre en évidence le bloc en cours d'exécution afin de rendre visible la correspondance entre le code et l'action, d'offrir un mode pas-à-pas (régler la vitesse, arrêter et reprendre l'exécution) ou de rendre visible l'état courant des variables. Ces fonctionnalités, que l'on ne rencontre pas nécessairement dans les environnements basés sur du texte, permettent aux débutants de mieux comprendre l'exécution des programmes.

Ces comparaisons s'appuient sur des éditeurs de code basiques. Notons que certains environnements de développement pédagogiques utilisés en seconde, tel que *EduPython* [3], proposent des fonctionnalités facilitantes comme la coloration syntaxique, la complétion automatique ou la vérification de la syntaxe pendant la saisie qui peuvent aider à la structuration des programmes (R1.3) et à limiter la saisie au clavier (R1.4).

4 Méthodologie

Dans cette section, nous décrivons la méthodologie nous permettant de répondre aux questions de recherche relatives à l'agencement du milieu didactique (QR2) et à l'évaluation de nos choix de conception (QR3).

4.1 Un milieu didactique inspiré des environnements basés sur les blocs

Nous avons aménagé le milieu didactique des situations du jeu (QR2) en nous appuyant sur les résultats de la question de recherche QR1. Ainsi, nous avons incorporé dans le milieu les caractéristiques des environnements de programmation par blocs en espérant pouvoir profiter de leurs avantages.

4.2 Évaluation de la conception : traces d'activités et questionnaire

La méthodologie liée à l'évaluation de notre conception (QR3) se base d'abord sur l'analyse des traces de l'activité des utilisateurs. Ces traces sont générées automatiquement au format standardisé *xAPI* (Kevan & Ryan, 2016). Elles témoignent des interactions des élèves avec le milieu didactique : consultations des contenus, copiés-collés, erreurs dans les programmes, aides apportées par l'enseignant, manipulation du panneau de contrôle, etc. Ces traces sont complétées par une enquête en ligne renseignée par les élèves en fin d'expérimentation. Ce questionnaire a pour but de recueillir le point de vue qualitatif des élèves sur l'application.

Nous avons pu expérimenter notre application dans huit classes de seconde auprès de 240 élèves débutants en Python sur deux ou trois séances de 55 minutes chacune. Après, une rapide présentation, il était attendu des élèves qu'ils utilisent le jeu de manière autonome. L'enseignant avait pour consigne de n'intervenir qu'à leur demande. Afin de garder trace de ces interactions, l'enseignant devait renseigner dans l'interface le contenu de l'aide apportée en cliquant sur des boutons qui lui sont réservés (voir Fig. 1-f). Notons que nous avons, lors de toutes les séances, prêté main forte à l'enseignant dans cette tâche.

Notre corpus de données est ainsi constitué de 69 701 traces d'activités et de 224 réponses au questionnaire en ligne (certains étudiants n'ont pas pu répondre pour des raisons techniques). Il a été analysé de façon automatisée au moyen de programmes Python. La manipulation et le traitement des données exploitent la bibliothèque *Pandas*, les graphiques sont générés par les librairies *Matplotlib* et *Seaborn*.

5 Résultats

5.1 Aménagement du milieu didactique

Nous relatons maintenant la manière dont nous avons aménagé le milieu didactique. Notre exposé s'appuie sur la figure Fig. 1 qui reproduit l'interface graphique et les différentes zones de l'application *Pyrates*.

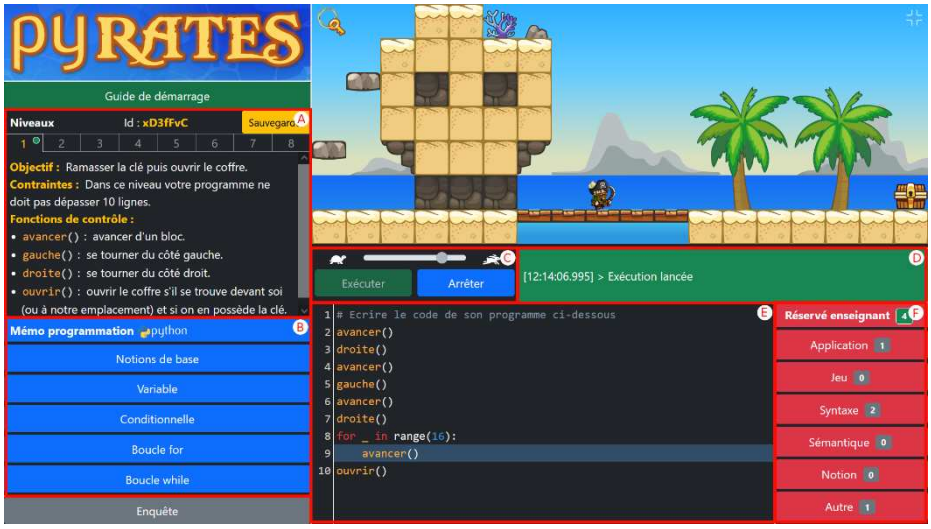


Fig. 1. Différentes zones de l'interface graphique de l'application *Pyrates*.

Nous avons d'abord créé sur la partie gauche de l'interface un bandeau fixe comportant un « Mémo programmation » (Fig. 1-b). Cette zone s'inspire du catalogue de commande présent dans les environnements basés sur les blocs (R1.1). Les contenus y sont classés par notions (notions de base, variable, conditionnelle, boucle for, boucle while) et sont accessibles en cliquant sur les différents boutons bleus. Les concepts présentés ont été choisis en cohérence avec les programmes scolaires du cycle 4 et de la seconde. Notre dispositif visant la transition collège-lycée, nous ne présentons pas la notion de fonction informatique qui est un objectif pour l'année de seconde. De plus, les notions liées aux expressions et aux calculs ne sont pas réinvesties car le contexte du jeu ne s'y prête peu. Afin de guider au mieux les élèves dans l'exploration du milieu didactique, le fait de survoler un bouton à la souris change son intitulé en donnant un aperçu de l'utilité de la notion. Par exemple « variable » devient « garder des informations en mémoire ». Le clic sur un bouton entraine l'apparition d'un panneau latéral qui détaille la notion en sous-notions (voir Fig. 2). Ce découpage a été guidé par la constitution des blocs Scratch. Ainsi, la distinction « Répétition simple » et « Répétition avec compteur » (Fig. 2-a) n'a, par exemple, pas de sens du point de vue de la syntaxe Python. Cependant, les compteurs de boucle n'existant pas en Scratch, il nous semble utile de faire cette différence dans un contexte de transition.

(a) Extrait du panneau relatif à la notion de "Boucle for"

(b) Extrait du panneau relatif à la notion de "Conditionnelle"

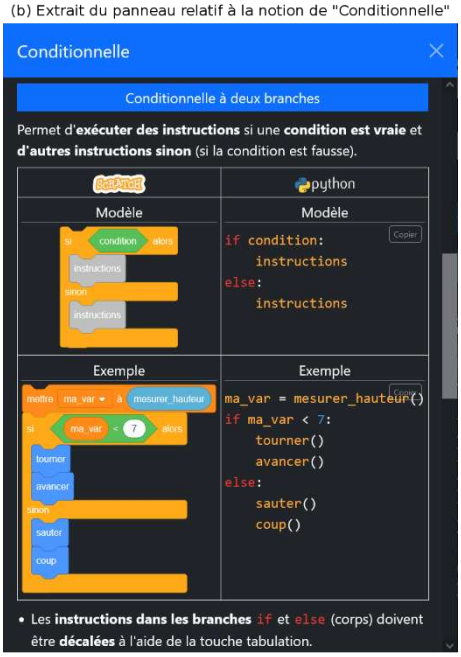


Fig. 2. Exemples d'extraits des panneaux latéraux

Chaque sous-notion est présentée et accompagnée de deux programmes Python : un modèle générique et un exemple dans le contexte du jeu. Nous fournissons de surcroit la traduction de ces éléments dans le langage Scratch. La présence du modèle générique et de l'équivalent Scratch des programmes a pour but d'aider les apprenants à réduire le nombre d'éléments significatifs. Il s'agit d'appréhender le programme Python par morceaux (« *chunk* ») et non élément par élément (R1.2). Par exemple, dans le cas de la répétition simple (Fig. 2-a), les élèves doivent se focaliser sur le nombre entre parenthèses et considérer le reste du code comme un même agrégat. Afin de limiter la saisie au clavier (R1.4), chaque pièce de code Python est accompagnée d'un bouton « Copier » . L'objectif est ici d'encourager la pratique du copier-coller vers l'éditeur de texte. Cet usage doit, d'une certaine manière, prendre le relais du glisser-déposer caractéristique des environnements basés sur les blocs. Cela peut aussi, dans une moindre mesure, soutenir la structuration des programmes (R1.3). L'utilisateur devra cependant veiller au maintien de cette structure au cours de la rédaction de son programme.

En dépit des efforts de conception qui viennent d'être décrits, il semble présomptueux d'envisager la disparition des erreurs syntaxiques. L'interprétation des messages d'erreur constituant un frein pour les programmeurs novices (R1.5), nous avons décidé d'enrichir le milieu didactique d'un analyseur syntaxique conçu pour les débutants (Kohn, 2017). Ce module analyse le code Python saisi par les utilisateurs avant qu'il ne soit exécuté par l'interpréteur. Il a la particularité de formuler des messages d'erreur en français, mais surtout dans un registre pratique et compréhensible des novices. Nous avons également effectué un travail de reformulation des messages afin d'adapter leur

terminologie à celle du mémo programmation. Ainsi, lors de la survenue d'une erreur syntaxique, le message s'affiche dans la zone console de l'interface (Fig. 1-d) et la ligne en cause est surlignée en rouge dans la zone d'édition du code (Fig. 1-e). L'absence d'erreurs détectées par l'analyseur syntaxique ne signifie pas que le code est interprétable. Des erreurs sémantiques (liées par exemple au typage) peuvent toujours apparaitre lors de l'interprétation.

Enfin, nous avons créé un panneau de contrôle (Fig. 1-c) dans le but d'améliorer la maîtrise de l'exécution des programmes (R1.6). Outre le fait de pouvoir lancer l'exécution de leur programme, les utilisateurs ont la capacité d'arrêter l'exécution en cours et de régler sa vitesse à l'aide d'un curseur. Ce curseur modifie la vitesse d'action des personnages en agissant sur un coefficient multiplicateur positionné à 1 (tortue) au lancement du jeu et pouvant aller jusqu'à 3 (lièvre). La visualisation et le suivi de l'exécution (R1.6) sont assurés par la mise en valeur (surlignage) de la ligne en cours d'exécution dans la zone d'édition du code (Fig. 1-e). Ainsi, la correspondance entre le code et l'action en cours est apparente.

5.2 Évaluation des choix de conception

Afin d'évaluer les choix de conception que nous venons d'exposer, nous allons analyser les traces d'utilisations générées automatiquement par l'application. Dans le cadre de cette étude, nous prenons en compte les traces concernant : la consultation du mémo, les copiés-collés du mémo vers l'éditeur de code, les erreurs détectées par l'analyseur syntaxique et par l'interpréteur, les aides syntaxiques et sémantiques renseignées par les enseignants lors de leurs interventions, la manipulation du curseur de vitesse et la vitesse choisie lors de l'exécution des programmes.

Commençons par examiner l'utilisation du mémo programmation. Comme nous pouvons le voir dans la figure Fig. 3-a, ce mémo est fréquemment consulté par les élèves. Nous remarquons également que, à l'instar du catalogue des environnements basés sur les blocs, il est le support de la découverte des notions. Ainsi, chaque fois qu'une nouvelle notion est mise en jeu dans un niveau (niv.1, niv.3, niv4. et niv.8), nous retrouvons une certaine variété dans les notions consultées. C'est en effet la manifestation d'une démarche de recherche. Lorsque les notions ont déjà été utilisées (niv.2, niv.5 et niv.6), la consultation semble plus ciblée sur les notions en jeu. Nous pouvons faire l'hypothèse qu'il s'agit, dans ce cas, pour les élèves de se rappeler de la syntaxe d'implémentation des notions. Nous retrouvons ici la fonction de remémoration du catalogue des environnements basés sur les blocs. En analysant la figure Fig. 3-b, nous pouvons affirmer que les élèves s'emparent quasi-systématiquement du copier-coller lors de l'implémentation d'une notion. En effet, chaque fois qu'une notion est mise en jeu dans un niveau, nous retrouvons, en moyenne, au moins un usage du copier-coller qui lui est associé. Exception faite de la notion de variable qui dispose d'une syntaxe d'implémentation beaucoup plus simple que les autres notions. Cette pratique se rapprochant du glisser-déposer des blocs, elle est en mesure de limiter la saisie au clavier et d'aider l'établissement des structures de code.

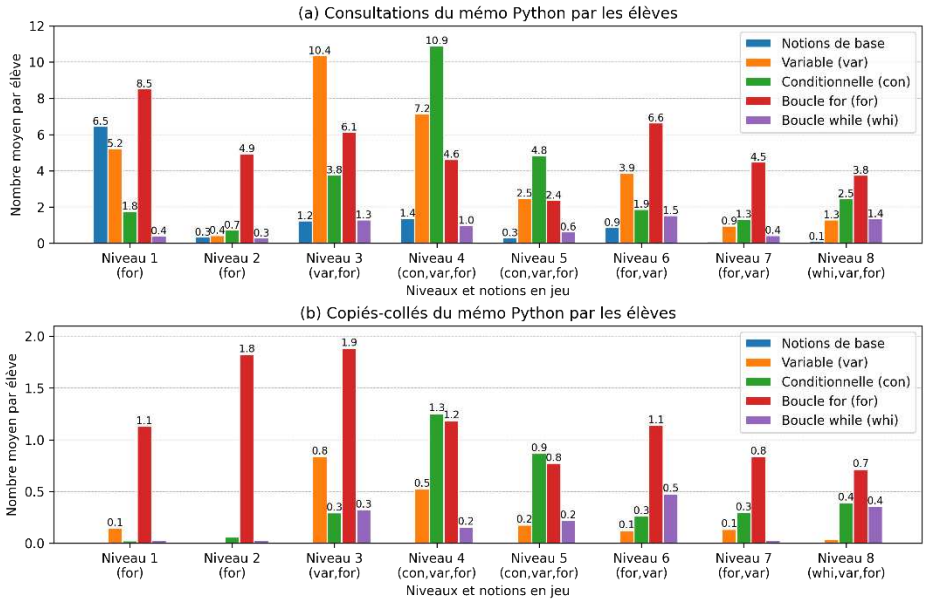


Fig. 3. Consultations et copiés-collés du mémo Python par les élèves par niveau.

Exposons désormais nos analyses se rapportant aux erreurs syntaxiques (issues de l'analyseur syntaxique). L'examen de la figure Fig. 4-a permet d'affirmer qu'elles sont présentes en nombre et dans une proportion bien supérieure à celles portant sur la sémantique (issues de l'interpréteur). En s'intéressant aux aides apportées par les enseignants (Fig. 4-b), il est remarquable de constater que les interventions en rapport avec la syntaxe des programmes sont très peu fréquentes. Cette rareté s'apprécie en comparaison du nombre d'erreurs. Ainsi, on compte une intervention pour trente à quarante erreurs syntaxiques dans les quatre premiers niveaux. Les élèves donc sont en mesure d'ajuster leurs procédures grâce aux rétroactions du milieu, sans solliciter l'enseignant.

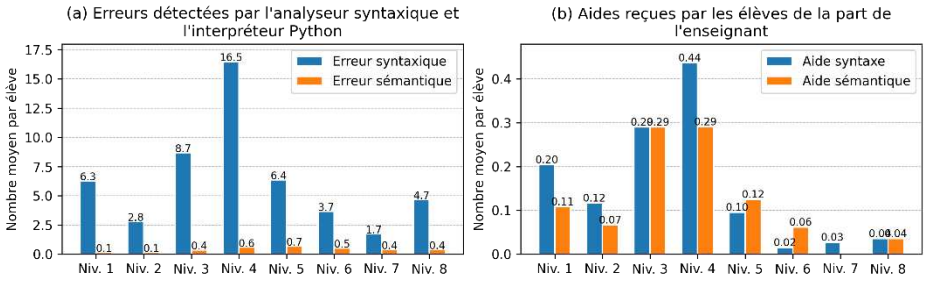


Fig. 4. Erreurs détectées par l'application et aides de l'enseignant reçues par les élèves.

Les traces générées par l'application nous donnent des indications quantitatives relatives à l'utilisation du mémo et à l'interprétation des messages d'erreur. Ces analyses peuvent néanmoins être complétées qualitativement par le questionnaire que nous

avons fait passer aux élèves en fin d'expérimentation. Cette enquête comportait des questions en lien avec le mémo Python et les messages d'erreur. Il s'agissait pour les élèves d'évaluer plusieurs aspects de l'application en plaçant des curseurs entre deux extrêmes (« Pas clair » – « Très clair », « Pas utile » – « Très utile »), ce qui a pour effet de générer un score entre 0 et 100. Nous présentons dans la figure Fig. 5 la répartition (densité) et la médiane des scores concernant les questions qui nous intéressent.

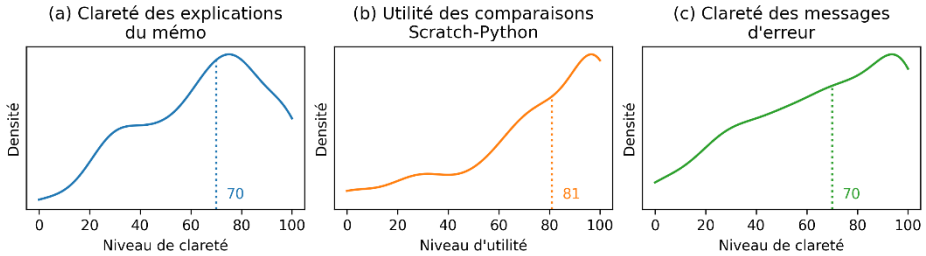


Fig. 5. Extrait des résultats de l'enquête élèves (répartition des réponses et médiane).

En plus d'être beaucoup consultées par les élèves, les explications contenues dans le mémo sont jugées comme étant claires par la majorité des élèves (Fig. 5-a). Nous distinguons néanmoins un groupe d'élèves autour du score de 30 pour qui ces contenus sont plus obscurs. Les comparaisons avec Scratch (Fig. 5-b) sont estimées comme étant utiles, voire très utiles par la grande majorité des élèves. Enfin, les messages d'erreur, dont nous avons montré l'utilité dans l'autonomie des élèves, sont également estimés comme étant clairs par la majorité des répondants.

Évaluons désormais l'utilisation des fonctionnalités de contrôle des programmes. En nous penchant sur la figure Fig. 6-a, nous constatons un très grand nombre de programmes lancés en moyenne par élève. Parmi ces programmes, beaucoup sont erronés (erreur syntaxique ou sémantique, trop de lignes), ce qui semble indiquer l'adoption par les élèves d'une démarche de programmation par essais-erreurs relativement à la correction des programmes. De nombreux programmes corrects sont également lancés (erreur liée au jeu, perte d'un niveau, progression dans un niveau, gain d'un niveau), cela montre que les élèves progressent dans les niveaux de façon incrémentale, par étapes intermédiaires. Les arrêts de programmes sont peu fréquents. Il est cependant possible de distinguer deux types de comportements en fonction du mode de génération du parcours des niveaux. Pour un premier ensemble de niveaux ayant des parcours fixes non aléatoires (niv.1, niv.2, niv.6 et niv.7), les élèves utilisent en moyenne entre quinze et vingt lancements et pratiquement aucun arrêt. Dans les niveaux ayant un parcours aléatoire changeant à chaque exécution (niv.3, niv.4, niv.5 et niv.8), les élèves ont tendance à avoir recours à davantage d'exécutions et à en stopper un certain nombre. Il est probable qu'une partie d'entre eux adoptent pour ces niveaux, de façon transitoire, un mode opératoire consistant à enchaîner les « lancer-arrêter » jusqu'à obtenir une configuration du parcours favorable à leur programme. Cette stratégie que nous qualifions de « contournement didactique » permet de réussir ces niveaux sans mettre œuvre les structures basées sur des tests (conditionnelle, while). Cette procédure a très peu de chances d'aboutir en raison du nombre important de configurations aléatoires

différentes. Ces élèves qui restent « à tout prix » dans le domaine ludique ne souhaitent ou ne peuvent pas rentrer dans l'apprentissage notionnel en explorant le milieu à la recherche d'une notion qui pourrait leur permettre de terminer le niveau.

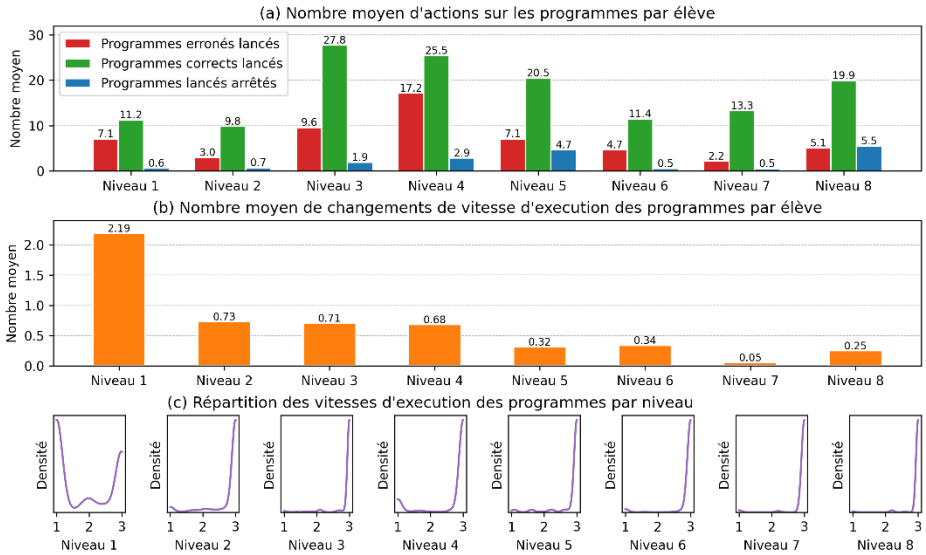


Fig. 6. Données concernant le contrôle de l'exécution des programmes par niveau.

Pour finir, portons notre attention sur le curseur de changement de vitesse. Il est en moyenne peu utilisé (Fig. 6-b) et de manière décroissante dans le temps. La figure Fig. 6-c présente la répartition (densité) des vitesses d'exécution des programmes lancés pour chaque niveau. Elle permet de constater que, dès le niveau 2, les programmes sont presque tous lancés à la vitesse maximale (coefficient multiplicateur 3). La démarche de programmation par essais-erreurs et par étapes incrémentales que nous avons décrit précédemment va de pair avec cette vitesse d'exécution élevée. Trois élèves font d'ailleurs remarquer dans le champ libre du questionnaire que « *le bonhomme n'avance pas assez vite* ». Nous constatons néanmoins une pratique utilisée à la marge dans des niveaux plus avancés (niv.4 et niv.5) qui consiste à revenir à des vitesses d'exécution plus lentes. Nos observations durant les expérimentations indiquent qu'il s'agissait pour certains élèves de pouvoir, ponctuellement, suivre plus facilement les lignes exécutées dans un mode d'action se rapprochant du pas-à-pas.

6 Conclusion et perspectives

En conclusion de cette contribution, nous allons d'abord en rappeler les principaux résultats en revenant à nos questions de recherche. Notre revue de travaux nous a permis d'établir les différences et les avantages des environnements de programmation basés sur les blocs en comparaison de ceux basés sur du texte (QR1). Nous avons ensuite aménagé le milieu didactique des situations de l'application *Pyrates* en y intégrant

certains éléments, supposés profitables pour les élèves, des environnements basés sur les blocs (QR2). Enfin, nous avons évalué notre conception en analysant les traces d'utilisations de l'application et les réponses au questionnaire (QR3). Certains choix d'agencement ont des conséquences positives :

- le mémo programmation est très fréquemment consulté par les élèves, il est le support de la découverte et de la remémoration des notions ;
- les comparaisons avec Scratch qu'il comporte sont considérées comme utiles par une très large majorité d'élèves, elle doivent aider l'appréhension par morceaux des structures Python ;
- le copier-coller depuis le mémo programmation est largement pratiqué, cela a pour effet de limiter la saisie au clavier et, dans une moindre mesure, de soutenir la tâche de structuration des programmes ;
- les rétroactions fournies par l'analyseur syntaxique sous la forme de messages d'erreur jugés « clairs » par les élèves rend possible la correction des programmes en sollicitant très peu l'enseignant.

Le panneau de contrôle devait permettre aux élèves de mieux comprendre l'exécution des programmes. Nous constatons, de façon très marginale, un réduction de la vitesse du personnage afin de suivre les exécutions sur le mode du pas-à-pas. Cependant, de manière générale, il ne produit pas les résultats escomptés :

- le bouton de lancement des programmes est fréquemment utilisé et le curseur de réglage de la vitesse d'exécution est très rapidement positionné au maximum dans le but d'adopter une démarche de programmation par essais-erreurs peu propice à la réflexion ;
- le bouton permettant de stopper les exécutions est peu utilisé, quand il l'est, c'est surtout pour tenter de réussir des niveaux basés sur des parcours aléatoires par « contournement didactique ».

Ces résultats doivent être considérés au regard des limites de notre méthodologie. Ainsi, les élèves étant en contexte écologique, il fut difficile de maintenir des conditions expérimentales totalement similaires entre nos différents groupes, notamment concernant l'activité de l'enseignant et la distance temporelle entre les séances. D'autre part, le raisonnement sur des moyennes permet de dégager des tendances, mais masque les disparités de pratique entre les élèves dont nous avons été témoin dans les classes.

Évoquons pour finir quelques perspectives permettant de prolonger ce travail. Edwards (2004), affirme que les débutants en informatique réussissent mieux à apprendre s'ils passent d'une approche par essais-erreurs à une pratique de « *réflexion en action* ». Il serait ainsi avantageux de modifier les possibilités de contrôle de l'exécution dans notre application de manière à contraindre les élèves à moins d'action et à plus de réflexion. Un moyen pourrait être de limiter le nombre d'exécutions à l'aide de pénalités. Par ailleurs, il serait intéressant d'exploiter nos traces d'activités à l'aide d'algorithme de Data Mining afin de mettre en évidence les différentes stratégies de résolution mises en œuvre. Des algorithmes de clustering pourraient également permettre de faire émerger différents profils d'élèves.

Remerciements

Ce travail a bénéficié du soutien financier de la Région Bretagne et du projet ANR IE-CARE.

Références

1. Alrubaye, H., Ludi, S., & Mkaouer, M. W. (2019). Comparison of block-based and hybrid-based environments in transferring programming skills to text-based environments. In T. Pakfetrat, G.-V. Jourdan, K. Kontogiannis R. Enenkel (dir.), *Proceedings of the 29th Annual International Conference on Computer Science and Software Engineering* (p.100-109). IBM Corp.
2. Alvarez, J. (2007). *Du jeu vidéo au serious game: approches culturelle, pragmatique et formelle* [Thèse de doctorat]. Université de Toulouse 2.
3. Andrews, E., Bau, D., & Blanchard, J. (2021). From Droplet to Lilypad: Present and Future of Dual-Modality Environments. In *2021 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)* (p. 1-2). IEEE.
4. Armoni, M., Meerbaum-Salant, O., & Ben-Ari, M. (2015). From scratch to "real" programming. *ACM Transactions on Computing Education (TOCE)*, 14(4), 1-15.
5. Bart, A. C., Tibau, J., Tilevich, E., Shaffer, C. A., & Kafura, D. (2017). Blockpy: An open access data-science environment for introductory programmers. *Computer*, 50(5), 18-26.
6. Bau, D., Bau, D. A., Dawson, M., & Pickens, C. S. (2015). Pencil code: block code for a text world. In *Proceedings of the 14th International Conference on Interaction Design and Children* (p. 445-448).
7. Bau, D., Gray, J., Kelleher, C., Sheldon, J., & Turbak, F. (2017). Learnable programming: blocks and beyond. *Communications of the ACM*, 60(6), 72-80.
8. Bessot, A. (2003). Une introduction à la théorie des situation didactiques. *Les cahiers du laboratoire Leibniz*, 91, 1-28.
9. Blanchard, J., Gardner-McCune, C., & Anthony, L. (2020, February). Dual-Modality Instruction and Learning: A Case Study in CS1. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (p. 818-824).
10. Brousseau, G. (1981). Problèmes de didactiques des décimaux. *Recherches en Didactique des Mathématiques*, 2(1), 37-125.
11. Brousseau, G. (1986). Fondements et méthodes de la didactique des mathématiques. *Recherches En Didactique Des Mathématiques*, 7(2), 33–115.
12. Brousseau, G. (1998). *Théorie des situations didactiques : Didactique des mathématiques 1970-1990.* La Pensée Sauvage.
13. Brousseau, G. (2010). *Glossaire de quelques concepts de la théorie des situations didactiques en mathématiques.* Site personnel. http://guy-brousseau.com/wp-content/uploads/2010/09/Glossaire_V5.pdf
14. Brown, N., Kyfonidis, C., Weill-Tessier, P., Becker, B., Dillane, J., & Kölling, M. (2021). A Frame of Mind: Frame-based vs. Text-based Editing. In *Proceedings of the 2021 Conference on United Kingdom and Ireland Computing Education Research* (p. 1-7). Association for Computing Machinery.
15. Declercq, C., & Nény, F. (2020). *Block2Py, un éditeur de blocs pour l'apprentissage du langage Python.* Atelier présenté à Didapro 8 – DidaSTIC , Lille, France

16. Edwards, S. H. (2004). Using software testing to move students from trial-and-error to reflection-in-action. In *Proceedings of the 35th SIGCSE technical symposium on Computer science education* (p. 26-30).
17. Kyfonidis, C., Weill-Tessier, P., & Brown, N. (2021). Strype: Frame-Based Editing tool for programming the micro:bit through Python. In *The 16th Workshop in Primary and Secondary Computing Education*. (p. 1–2).
18. Kevan, J. M., & Ryan, P. R. (2016). Experience API: Flexible, decentralized and activity-centric data collection. *Technology, knowledge and learning*, 21(1), 143-149.
19. Kohn, T. (2017). *Teaching Python programming to novices : Addressing misconceptions and creating a development environment* [Thèse de doctorat]. ETH Zurich.
20. Kölling, M., Brown, N. C., & Altadmri, A. (2015). Frame-based editing: Easing the transition from blocks to text-based programming. In *Proceedings of the Workshop in Primary and Secondary Computing Education* (p. 29-38).
21. Kölling, M., Brown, N. C., & Altadmri, A. (2017). Frame-Based Editing. *Journal of Visual Languages and Sentient Systems*, 3, 40-67.
22. Price, T. W., & Barnes, T. (2015). Comparing textual and block interfaces in a novice programming environment. In *Proceedings of the eleventh annual international conference on International Computing Education Research* (p. 91–99). Association for Computing Machinery.
23. Matsuzawa, Y., Ohata, T., Sugiura, M., & Sakai, S. (2015). Language migration in non-cs introductory programming through mutual language translation environment. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (p. 185-190)
24. Monig, J., Ohshima, Y., & Maloney, J. (2015). Blocks at your fingertips: Blurring the line between blocks and text in GP. In *2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond)* (p. 51-53). IEEE.
25. Robinson, W. (2016). From scratch to patch: Easing the blocks-text transition. In *Proceedings of the 11th Workshop in Primary and Secondary Computing Education* (p. 96-99).
26. Weintrop, D., & Wilensky, U. (2017). Comparing block-based and text-based programming in high school computer science classrooms. *ACM Transactions on Computing Education (TOCE)*, 18(1), 1-25.
27. Weintrop, D. (2019). Block-based programming in computer science education. *Communications of the ACM*, 62(8), 22-25.

Pages Internet

1. Page d'accueil de l'application Pyrates, https://py-rates.fr, dernière consultation 15/12/21.
2. Guide pédagogique de l'application Pyrates, https://py-rates.fr/guide/FR/, dernière consultation 15/12/21.
3. Page d'accueil du site EduPython, https://edupython.tuxfamily.org/, dernière consultation 15/12/21.

Situation didactique autour d'un jeu de recherche : expérimentation en classes de NSI

Antoine Meyer[1][0000−0003−4513−4347] and Simon Modeste[2][0000−0002−8578−9287]

[1] LIGM, Univ Gustave Eiffel, CNRS, ESIEE Paris, F-77454 Marne-la-Vallée, France
antoine.meyer@univ-eiffel.fr
[2] IMAG, Université de Montpellier, CNRS, Montpellier – France
simon.modeste@umontpellier.fr

Résumé Ce travail cherche à mettre en évidence les potentialités didactiques, pour l'enseignement de concepts algorithmiques, d'une activité débranchée autour de jeux de devinettes. Grâce à un dispositif de « jeu algorithmique » inspiré de la théorie de la complexité, cette activité vise à faire émerger les concepts de complexité au pire d'algorithme et de complexité de problème. Elle mobilise aussi d'autres concepts présents dans les programmes de NSI, comme l'approche « diviser pour régner », ainsi que des savoir-faires plus généraux comme celui de formuler et prouver un énoncé. Nous décrivons une expérimentation de cette situation didactique dans plusieurs classes de lycée françaises de la spécialité Numérique et Sciences Informatiques (NSI), et fournissons des éléments d'analyse des données recueillies. Nous nous appuyons sur divers éléments théoriques et méthodologiques issus de la didactique des disciplines, ainsi que sur une analyse épistémologique et didactique préalable.

Keywords : algorithmique · dichotomie · complexité · didactique · lycée · informatique débranchée

1 Introduction

Au cours de ces dernières années, l'informatique s'est structurée en France en tant que discipline scolaire à part entière. Depuis 2009 déjà, des éléments d'informatique (essentiellement d'algorithmique et de programmation) sont apparus dans les programmes de disciplines existantes du secondaire, en particulier en mathématiques et en technologie. Lors de la dernière réforme du lycée, une nouvelle matière baptisée Numérique et Sciences Informatiques (NSI), dotée d'un volume horaire important, est apparue en classes de 1ère et de Terminale. Une nouvelle filière des classes préparatoires aux grandes écoles[3] a également vu le jour. Parallèlement ont été créés un CAPES puis une agrégation d'informatique afin de recruter des enseignant · es à même de prendre en charge ces contenus.

Le présent travail a pour objectif de contribuer, à son échelle, à une réflexion épistémologique et didactique sur les concepts présents (ou susceptibles de l'être)

[3] Mathématiques, Physique, Ingénierie et Informatique (MP2I) en première année, puis Mathématiques, Physique et Informatique (MPI) en 2e année.

dans les programmes d'informatique scolaire. Cet effort de recherche global pourrait viser à préciser en particulier les enjeux de *transposition didactique* de ces concepts depuis la science informatique vers l'enseignement, et pourrait permettre par exemple de contribuer à la formation des enseignants, d'apporter un regard critique sur les programmes, ou de contribuer à la réflexion sur la transition lycée – supérieur.

L'objet visé dans ce travail est l'algorithme de recherche binaire (ou par dichotomie), qui permet de résoudre le problème de recherche d'un élément dans une collection triée. Son étude fait explicitement partie du programme de première NSI et met en jeu un certain nombre d'autres concepts liés des programmes de première et de terminale, comme la complexité, le paradigme « diviser pour régner » ou indirectement la structure d'arbre binaire de recherche. On pourra se référer par exemple à [6] pour une discussion plus détaillée.

Nous faisons l'hypothèse qu'une situation adidactique (au sens de la *théorie des situations didactiques*), organisée sous la forme d'un jeu à deux joueurs, est de nature à faire émerger la stratégie de dichotomie à la fois en tant qu'outil (pour gagner le jeu) et en tant qu'objet (pour justifier l'efficacité de cette stratégie), ce qui peut permettre des apprentissages relatifs aux notions d'algorithme, de problème, de complexité et de preuve. Notre expérimentation a pour objectif de mettre à l'épreuve, et le cas échéant de raffiner, cette hypothèse de recherche.

Nous présentons à la section 2 quelques outils théoriques issus de la didactique et de l'informatique qui servent de support à cette étude. La section 3 détaille la situation didactique envisagée et quelques éléments de sa résolution, et la section 4 présente notre méthodologie de recueil de données. Nous donnons en section 5 quelques exemples de productions d'élèves recueillies au cours de l'expérimentation et quelques pistes d'analyse, avant de conclure.

2 Outillage théorique

On s'appuie dans ce travail sur des concepts issus de la didactique des disciplines, en particulier la *théorie des situations didactiques* développée par Brousseau [4] et ses successeurs, et sur les modèles épistémologiques de l'algorithme proposés par Modeste [8]. Nous sommes aussi influencés par des concepts tirés de la théorie de la complexité, qui nous amènent à considérer une famille de situations didactiques que nous qualifions de *jeux algorithmiques*.

2.1 Théorie des situations didactiques

Sans nous lancer dans une description détaillée de cette théorie bien connue des didacticiens (en particulier des mathématiques), nous relevons quelques concepts clés qui ont guidé l'élaboration de ce travail et l'analyse à suivre.

Milieu et rétroactions. Dans la théorie des situations didactiques ou TSD [4] (voir [3] pour une introduction succincte), la connaissance est envisagée comme une propriété du système { sujet, milieu }. Le *milieu* est un environnement dans

lequel agit le sujet, et qui lui apporte des *rétroactions* qui peuvent l'amener à développer de nouvelles stratégies d'action.

Situations adidactiques. Brousseau décrit une situation adidactique comme une situation porteuse d'un problème à résoudre, mais dans laquelle l'enseignant n'intervient pas directement. Il élabore en amont un milieu qui, par ses rétroactions, est susceptible d'amener l'élève à former de nouvelles stratégies, ce qui dans cette théorie constituerait un apprentissage. L'aspect *adidactique* de la situation tient au fait que l'élève est confrontée à un problème dont la responsabilité de la résolution lui revient (on parle de *dévolution*).

Variables didactiques. Modèle pour le chercheur autant que pour l'enseignant, le concept de situation fait également intervenir celui de *variable didactique*, qui représente un paramètre de la situation et dont le réglage peut avoir un impact sur les apprentissages potentiels des sujets, sur les stratégies valides ou non valides pour résoudre le problème posé, etc. Le développement de situations didactiques à des fins de recherche, par exemple pour explorer les enjeux d'enseignement d'une notion particulière comme c'est le cas dans ce travail, requiert donc une réflexion particulière sur l'identification des variables didactiques et sur leurs effets. En particulier, il est crucial que le choix des variables didactiques fasse de l'acquisition du savoir visé une *condition nécessaire* à la résolution du problème. Dans notre cas, le choix des variables didactiques devra permettre de faire émerger premièrement la dichotomie comme stratégie souhaitable pour gagner le jeu, et dans un second temps la réflexion sur la complexité de la méthode comme un moyen de répondre au problème posé. Un autre exemple est le choix de la valeur de N dans la situation décrite à la section 3, qui peut nécessiter la mise en œuvre de stratégies différentes chez l'élève.

Phases d'action, formulation, validation. Enfin, nous citons une distinction opérée par la TSD entre trois fonctions, ou statuts, du savoir : action, formulation et validation. Dans une phase d'action, le sujet interagit avec le milieu et reçoit des rétroactions, ce qui l'amène à construire des connaissances implicites. Une phase de formulation donne lieu à la transmission, par exemple d'un sujet à un autre, d'une stratégie explicite de résolution du problème, que nous sommes tentés d'apparenter à un algorithme. Enfin, la validation correspond au fait de présenter des arguments permettant de contrôler ou de démontrer la validité ou la performance d'une stratégie. Ces distinctions ont guidé l'élaboration de la situation présentée à la section 3. Elles met en jeu des composantes d'action (jeu à deux joueurs), de formulation (rédaction d'une stratégie applicable par autrui), et de validation (argumentation sur la performance de la stratégie décrite).

2.2 Modèles épistémologiques pour l'algorithme

Le travail de Modeste [7,8] plaide pour la prise en compte d'un modèle épistémologique de l'algorithme pour étayer le travail didactique en informatique. En s'appuyant sur la dialectique outil-objet mise en avant par Douady [5] et sur le

modèle *cK¢* de Balacheff [2] il propose plusieurs modèles visant à caractériser les éléments constitutifs de ce concept et à questionner sa transposition didactique.

Aspects de l'algorithme et pensée algorithmique. Suite à une revue de définitions de l'algorithmique dans des textes de référence, Modeste établit une liste d'*aspects* de l'algorithme, qui ont pour vocation d'aider à identifier les préoccupations fondamentales de l'algorithmique. Il distingue cinq aspects : les aspects *problème* et *effectivité*, qui ont trait à l'algorithme en tant qu'outil, et les aspects *complexité*, *preuve* et *modèles théoriques* qui ont trait à l'algorithme en tant qu'objet. Modeste complète cette analyse par un questionnement de la nature de l'*activité* algorithmique elle-même, souvent assimilée à une « pensée algorithmique » spécifique. Citant plusieurs textes, il observe que quand bien même la pensée algorithmique pourrait être considérée comme une partie de la pensée mathématique, « voir la pensée algorithmique comme pensée majeure de l'informatique permet un réel enrichissement de son analyse » en particulier dans une perspective d'enseignement, soulignant par là le fait que l'informatique exerce elle-même une influence sur la pratique mathématique.

Conceptions de l'algorithme. Enfin, Modeste développe une structuration des conceptions et usages de l'algorithme en six catégories, organisées selon deux axes : d'une part selon la dialectique outil–objet déjà mentionnée, d'autre part selon trois paradigmes ou « modes de vie » de l'algorithme : en tant que preuve algorithmique (ou preuve mathématique constructive), algorithme mathématique (par exemple exprimé dans un pseudo-code), ou algorithme informatique (implémentable sur machine). Chacune de ces six catégories est porteuse de ses propres exemples et problèmes typiques, opérateurs, systèmes de représentation et modalités de contrôle, conformément au modèle *cK¢* de Balacheff.

2.3 Une famille de situations didactiques pour l'informatique

À la lumière de ces différentes influences théoriques, nous proposons une famille de situations didactiques que nous surnommerons *jeux algorithmiques*, qui repose sur un parallèle entre un jeu à deux joueurs (un élève jouant contre un autre élève, l'enseignant ou un artefact technique) et l'exécution d'un algorithme au sein d'un environnement. Ce parallèle fait l'objet d'un type de raisonnement en théorie de la complexité, appelé *argument d'adversaire*, utilisé pour établir des bornes inférieures sur la complexité de problèmes algorithmiques.

Dans ce contexte, un joueur surnommé « joueur-algorithme » (ou A), cherche à résoudre une instance inconnue d'un certain problème algorithmique p dans un modèle de calcul donné, tandis que son adversaire, que nous appellerons « joueur-environnement » (ou E), joue à la fois le rôle du modèle de calcul et du reste de l'environnement. On suppose que la taille de l'instance à résoudre est connue des deux joueurs dès le début de la partie (contrairement à l'instance elle-même), et l'on notera cette taille n. À chaque tour de jeu, le joueur-environnement fournit les résultats des opérations élémentaires demandées par le joueur-algorithme, et contrôle les éventuels autres facteurs qui ne sont pas du ressort de son adversaire

(aléa, ordonnancement, événements, accès à la mémoire, etc.). Les objectifs des deux joueurs sont opposés : la taille n des instances étant fixée à l'avance, le joueur-algorithme cherche à résoudre le problème en faisant appel au plus petit nombre d'opérations élémentaires possible, tandis que le joueur-environnement cherche à le contraindre à en poser le plus grand nombre.

L'application d'origine de cet argument, issue de la théorie de la complexité, est donc ici instrumentalisée à des fins didactiques dans le but d'explorer les problématiques d'enseignement de l'algorithmique (à la fois en tant qu'outil et en tant qu'objet), de la complexité et de la preuve en informatique.

3 Présentation de la situation didactique

Cette section décrit la situation didactique qui fait l'objet de notre expérimentation. Une discussion plus générale du problème sous-jacent et du choix de paramétrage des variables didactiques sont présentées dans nos travaux antérieurs [6]. La situation envisagée se décompose en deux séances d'une heure. Les élèves sont répartis par groupes de deux ou trois. Au cours de chaque phase, les élèves sont invités, pendant 15 mn environ, à jouer quelques parties d'un jeu à deux joueurs dont les règles leur sont présentées, et à consigner par écrit le déroulement de chaque partie sur une feuille de partie (annexe A.1). Une frise numérique (annexe A.2) est à leur disposition, mais n'est accompagnée d'aucune consigne d'usage. Dans un second temps, les élèves sont confrontés à une tâche de résolution de problème pour laquelle ils disposent d'un temps de recherche d'environ 30mn. Les règles du jeu sont expliquées oralement, et l'énoncé du problème à résoudre écrit au tableau. Chaque séance se termine par un moment de bilan et d'institutionnalisation d'environ 15mn.

3.1 Consignes pour la situation 1

Règles initiales. Il s'agit de deviner un nombre entier compris entre 1 et un certain N, en posant le moins de questions possible. Le jeu se joue par équipes de trois. Une personne est appelée Devinante (**D**), une autre est appelée Répondante (**R**). La troisième personne est l'Arbitre (**A**). Dans la suite, on prendra systématiquement $N = 15$ (on devine donc des nombres compris entre 1 et 15 inclus) mais il est possible de jouer avec des valeurs différentes de N. Chaque partie se déroule de la manière suivante :

1. **R** choisit un nombre entier n compris entre 1 et 15. Il le communique secrètement à **A**.
2. **D** propose à **R** un nombre entier a entre 1 et 15. **A** note ce nombre sur la feuille de partie.
3. **R** peut répondre à chaque proposition par "égal" si le nombre a proposé est égal à n, "plus petit" si n est strictement plus petit que a, ou "plus grand" si n est strictement plus grand que a. **A** vérifie la réponse de **R**.
4. Si **D** a proposé un nombre a égal à n, la partie est terminée. **A** entoure le nombre trouvé et note le score. Sinon, on recommence au point 2.

Le but de **D** est de deviner n en faisant le moins possible de propositions (en comptant la proposition finale, pour laquelle $a = n$). Le nombre de propositions est appelé *score* de la partie. **D** souhaite donc *minimiser* le score.

Phase de jeu (15mn environ). On invite les élèves à jouer quelques parties, en les consignant soigneusement sur la feuille de partie 1 et en changeant régulièrement de rôle (D, R ou A).

Résolution de problème (30 mn environ). On invite les groupes d'élèves à résoudre le problème suivant :

> **Problème 1 :** Décrivez une stratégie pour **D** qui lui assure de gagner *toutes* les parties avec un score inférieur ou égal à 4 quand $N = 15$ (on dit que 4 est le *score maximal* selon cette stratégie).
> Quel est le score maximal pour $N = 31$? Et pour $N = 50$? Et pour N quelconque ?

3.2 Consignes pour la situation 2

Changement de règles. On souhaite renforcer le rôle du joueur **R**. Dans cette version du jeu, **R** cherche à obliger **D** à faire le plus de propositions possibles. **R** souhaite donc *maximiser* le score de la partie. Le rôle de **D** ne change pas : elle cherche toujours à deviner en faisant le moins de propositions possible, c'est-à-dire à *minimiser* le score. On modifie les règles de la manière suivante :

1. **R** *fait semblant de choisir* un nombre entier compris entre 1 et $N = 15$.
2. **D** propose à **R** un nombre entier a entre 1 et 15. **A** note ce nombre.
3. **R** peut répondre à chaque proposition *comme il le souhaite* par "égal", "plus petit" ou "plus grand", à condition de ne pas contredire ses réponses précédentes (cela veut dire que **R** n'a pas le droit par exemple d'affirmer que le nombre recherché est plus petit que 8, puis plus tard qu'il est plus grand que 7). **A** vérifie la cohérence des réponses de **R**.
4. Si **R** a répondu "égal" à la dernière proposition, la partie est terminée. **A** entoure le nombre final et note le score. Sinon, on recommence au point 2.

Phase de jeu (15 mn environ). On invite les groupes d'élèves à jouer quelques parties selon ces nouvelles règles (en les consignant soigneusement sur la feuille de partie 2).

Résolution de problème (30 mn environ). On invite les groupes d'élèves à résoudre le problème suivant :

> **Problème 2 :** Décrivez une stratégie pour **R** qui lui permet d'obliger son adversaire à proposer au moins 4 nombres quand $N = 15$ (on dit que 4 est le *score minimal* selon cette stratégie).
> Quel est le score minimal pour $N = 31$? Et pour $N = 50$? Et pour N quelconque ?
> Attention, on doit prendre en compte toute stratégie possible de **D** !

3.3 Éléments de résolution

Situation 1. À tout moment, l'ensemble des possibilités restantes forme un intervalle. Pendant la phase de jeu, les stratégies attendues pour **D** sont la recherche par balayage (éventuellement par sauts), la dichotomie, ou un mélange des deux[4].

La stratégie par dichotomie pour **D** consiste à toujours proposer un nombre médian parmi les possibilités restantes. Si g et d sont les bornes de l'intervalle de recherche, calculer $(g + d)/2$ (arrondi par défaut ou par excès) fournit un tel nombre. On peut démontrer de plusieurs manières que cette stratégie garantit à **D** de gagner en au plus 4 questions, par exemple en majorant le cardinal de l'intervalle de recherche après chaque réponse : 7 au plus après la première, 3 après la deuxième, et 1 après la troisième (la quatrième question permettant donc de gagner). Une autre justification consiste à dessiner l'arbre des parties possibles lorsque **D** applique cette stratégie. Pour $N = 31$ un raisonnement semblable permet de démontrer que 5 questions suffisent.

Il est possible que certains élèves reconnaissent 31, 15, 7, 3 et 1 comme nombres particuliers (de la forme $2^k - 1$) et en déduisent la conjecture que k questions suffisent dans ce cas. Le cas $N = 50$ est choisi pour déstabiliser ce type de raisonnement. Il impose de surmonter la difficulté liée à l'arrondi dans le calcul du nombre médian. On peut montrer par exemple qu'il reste *au moins* 25 possibilités après une question, puis 12, 6, 3, et enfin 1. Il est également possible de raisonner de proche en proche depuis le cas $N = 1$, pour lequel une seule question suffit, et d'observer qu'une question supplémentaire permet de gagner à coup sûr quand $N = 3$, puis une de plus (donc trois) quand $N = 7$, quatre quand $N = 15$, cinq quand $N = 31$, six quand $N = 63$, et ainsi de suite. En déduire que 6 questions suffisent pour $N = 50$ n'est pas évident et fait appel à un argument de monotonie susceptible de rester implicite.

Quel que soit le raisonnement employé, la conjecture puis la preuve d'un résultat pour N quelconque fait potentiellement appel à des connaissances préalables plus avancées, par exemple sur la numération en binaire, l'algèbre, les suites, les fonctions $\log_2(n)$ ou 2^n, le raisonnement par récurrence, etc.

Situation 2. Pendant la phase de jeu, la stratégie visée pour **R** consiste à toujours répondre de manière à maximiser le cardinal de l'intervalle de recherche restant. Cette stratégie de **R** garantit qu'aucune partie ne peut être gagnée en moins de 4 questions, comme on peut le démontrer en *minorant* le cardinal de l'intervalle de recherche après chaque réponse : 7 puis 3 puis 1. La représentation des parties sous forme d'arbre n'est cependant plus efficiente, car on ne maîtrise plus les propositions de **D**. On peut également conclure de cette observation que la dichotomie est une stratégie *optimale* pour le cas $N = 15$: aucun algorithme ne peut permettre de gagner à coup sûr en moins de coups.

Par un raisonnement de proche en proche, on peut établir que 5 questions sont indispensables pour gagner quand $N = 31$, et 6 questions quand $N = 50$,

[4] En raison de la répétition des parties, il est également possible que certains élèves appliquent des stratégies « psychologiques » cherchant à anticiper les choix adverses.

cependant les calculs sont plus difficiles (comme précédemment). Un raisonnement dans le cas général, par exemple à l'aide de suites, permet de conjecturer ou démontrer qu'au moins $\log_2(n+1)$ questions (arrondi à l'entier supérieur) sont toujours nécessaires dans ce contexte, *quelle que soit* la stratégie de **D**. Ceci montre enfin que la stratégie de dichotomie est optimale pour tout N.

4 Recueil de données

Classes participantes. L'expérimentation a été menée en fin d'année scolaire (mai-juin 2021) auprès d'élèves de trois classes de première NSI et deux classes de terminale NSI, encadrée par quatre professeurs différents. Elle a également eu lieu auprès d'une classe de mathématiques de seconde de l'un des professeurs participants (à sa demande). Les données de cette classe n'ont pas encore fait l'objet d'une analyse spécifique, mais sont susceptibles de fournir un éclairage complémentaire. En raison du contexte sanitaire lié à l'épidémie de Covid au moment de l'expérimentation, les lycées était contraints de n'accueillir que la moitié des élèves en classe, l'autre moitié travaillant chez eux. Il nous a donc été possible de mener l'activité auprès de demi-classes, à raison d'environ quatre ou cinq groupes de trois élèves par session. Au total, une quarantaine d'élèves de première et autant d'élèves de terminale ont participé.

Dispositif. Comme précédemment indiqué, les élèves ont été répartis par groupes de deux ou trois. L'expérimentateur a participé à la majorité des séances, en co-animation avec l'enseignant. Les deux situations ont été présentées à quelques jours d'écart ou immédiatement l'une après l'autre selon les classes. Faute de pouvoir capter en vidéo le travail de chaque groupe d'élèves, nous avons procédé, à chaque fois que cela était possible, à un enregistrement audio des échanges entre les élèves et avec l'enseignant et l'expérimentateur. Le travail a été relativement autonome pendant les 45 premières minutes de chaque situation, à l'exception de l'exposé initial de la situation et d'occasionnelles relances. Le dernier quart d'heure a été comme prévu consacré à des échanges en classe entière (bilan, poursuite du questionnement, institutionnalisation), mais n'a pas fait l'objet d'une planification ni d'une analyse détaillées, ce qui peut constituer une limite de notre approche. Les élèves ont été encouragés à annoter librement les feuilles de partie et les frises de recherche. En phase de résolution de problème, il leur a été demandé de mettre leurs idées par écrit (sans contrainte de forme). L'ensemble des traces écrites ont ensuite été collectées. Nous avons ainsi constitué un corpus (encore en cours d'analyse) dans lequel il nous est possible de recouper les traces de recherche écrites et les extraits sonores correspondants.

5 Exemples de productions d'élèves et éléments d'analyse

Le traitement complet du corpus est en cours, en particulier la transcription des enregistrements audio et l'élaboration d'une grille d'analyse détaillée. Même s'il s'agit d'un travail inachevé, et malgré le manque de place, nous présentons

quelques exemples tirés des productions d'élèves, en tentant de dégager quelques axes d'analyse principaux. La distinction entre phases d'action, de formulation et de validation mise en avant par la TSD nous est ici très utile.

5.1 Stratégies employées lors des phases d'action

Ce premier axe d'analyse s'intéresse à l'émergence de stratégies performantes pour les joueurs D puis R pendant les phases de jeu, grâce aux feuilles de parties. Comme les noms des joueurs y figurent, il nous est possible de questionner la stabilité des stratégies de chaque élève. Il serait trop long de détailler des exemples réels, mais nous sommes à même de relever un certain nombre de tendances.

Émergence de la stratégie de dichotomie en situation 1. L'émergence de la stratégie par dichotomie pour D est soit très rapide (dès la première partie chez certains groupes), soit plus tardive. Dans un premier temps un nombre significatif de parties sont jouées selon une stratégie par balayage ou par dichotomie, mais "bruitée" (irrégularités dans les choix de propositions) ou "mixte" (changement de stratégie apparente en cours de partie). D'autres sont difficilement interprétables, soit parce que le joueur D a un "coup de chance" soit parce que les choix effectués semblent arbitraires. Il ressort néanmoins que la dichotomie émerge dans la très grande majorité des groupes au cours de la situation (parfois pendant la phase de résolution de problème, ou grâce à des indications).

Choix de l'élément médian. Dans un grand nombre de cas, la détermination de l'élément médian a posé problème. Un grand nombre de joueurs a par exemple choisi spontanément le nombre 7 comme première proposition, en divisant la borne supérieure de l'intervalle par deux (et en arrondissant par défaut), au lieu du choix canonique 8 qui divise l'intervalle en deux parties égales.

Une fois cet obstacle surmonté, la décision à prendre reste problématique quand la borne inférieure de l'ensemble est supérieure à 1, ce qui a poussé certains groupes lors de la résolution de problème à uniquement raisonner sur des parties évoluant dans un intervalle de borne inférieure 1 parce que « c'est plus simple ». Une autre difficulté survient quand le nombre de possibilité restantes est pair (faut-il arrondir par défaut ou par excès ?). Une aide a été apportée à certains groupes, par exemple en les invitant à compter le nombre de "cases" restantes de part et d'autre du nombre envisagé. Il ressort de cette analyse rapide que ce point est une difficulté réelle et relativement inattendue de cette situation.

Stratégies pour R (situation 2). Une analyse des données issues de la situation 2 reste à accomplir, mais une première difficulté didactique relevée est que les élèves ne perçoivent pas toujours la différence avec la situation 1 : « on fait encore la dichotomie ». Le changement de point de vue sur l'action du joueur R semble difficile à appréhender pour certains élèves. Ceci suggère une piste de réflexion pour de futurs approfondissements.

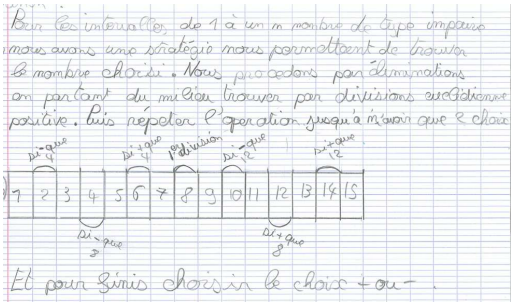


Fig. 1. Stratégie pour $N = 15$ à l'aide d'une frise numérique.

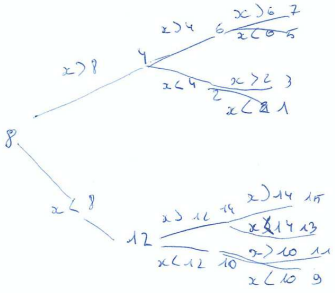


Fig. 2. Stratégie et score pour $N = 15$ à l'aide d'un arbre de décision.

5.2 Formulation des stratégies

Dans ce deuxième axe d'analyse, on se propose d'étudier le type de signes (registres sémiotiques) et les éléments langagiers utilisés pour décrire les stratégies employées par les groupes d'élèves. Le mot « algorithme » n'ayant pas été explicitement prononcé, on peut se demander si des rédactions proches du langage naturel, du pseudo-code ou d'un langage de programmation sont employées. On s'intéresse également, bien sûr, à la correction sémantique des stratégies décrites.

Les élèves auteurs de l'extrait de la figure 1 ont reproduit la frise numérique et s'en sont servi comme support explicatif pour exprimer leur stratégie. On remarque également un doute lié à la parité de N, ce qui rejoint notre discussion précédente. La figure 2 présente un arbre de décision élaboré par un groupe d'élèves (sans que cette terminologie soit invoquée ni qu'aucune consigne en ce sens ait été donnée). Il n'est pas évident de déterminer quelles conclusions les élèves en tirent, mais on peut y voir l'énoncé d'une stratégie pour D puisque chaque nœud de l'arbre indique quel nombre proposer. La figure 3 décrit une stratégie inhabituelle (et néanmoins optimale) consistant à d'abord calculer la première puissance de 2 supérieure à N, avant d'appliquer une stratégie de dichotomie. Par exemple pour $N = 50$, on pourrait supposer que le premier nombre proposé est $(1 + 63)/2 = 32$, et non 24 ou 25. Le score garanti par cette stratégie a l'avantage d'être facile à calculer et à justifier (voir aussi la figure 5), puisqu'aucun problème d'arrondi n'intervient.

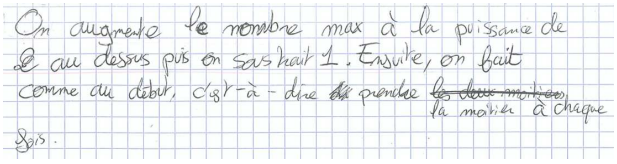


Fig. 3. Stratégie par dichotomie après passage à la puissance suivante.

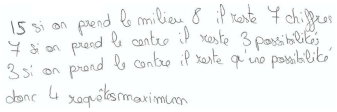


Fig. 4. Justification du score maximal pour $N = 15$ à l'aide d'une suite décroissante.

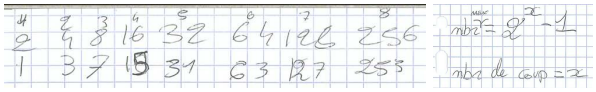


Fig. 5. Justification avec N de la forme $2^x - 1$ à l'aide d'une suite, et généralisation.

5.3 Modes de validation

Le troisième axe d'analyse concerne les phases de validation, notamment relatives à la justification des résultats trouvés : pourquoi telle stratégie garantit-elle un certain score maximal ou minimal, et comment les résultats numériques ou formules annoncés sont-ils valides ? Comme précédemment nous envisageons de prendre ici en compte les aspects sémiotiques et langagiers qu'il est possible de relever dans notre corpus, en particulier les aspects spécifiquement mathématiques des arguments, indices et justifications employés par les élèves.

La Figure 4 présente une justification du score maximal de 4 lorsque $N = 15$. Les outils mathématiques mobilisés sont ici très modestes : absence d'algébrisation, utilisation du langage naturel. Cet argument simple est néanmoins convaincant. La production illustrée figure 5 s'appuie sur la suite 1, 3, 7, 15... pour déterminer le nombre de coups nécessaires quand N prend une valeur de ce type. L'aspect algébrique apparaît ici clairement. L'annotation peu visible « max » laisse penser que les étudiants ont eu l'idée de rechercher la première puissance de 2 supérieure à N, comme le confirme la stratégie formulée par ce groupe (figure 3 ci-dessus). Enfin, on peut interpréter l'arbre de la figure 2 comme une justification du score maximal pour $N = 15$.

6 Conclusion

Nous avons présenté l'expérimentation d'une situation didactique conçue de manière à mobiliser les concepts d'algorithme, de problème, de complexité et de preuve en classe de NSI, et un début d'analyse des données recueillies. Notre hypothèse est que ce type de situation est susceptible de provoquer l'émergence des stratégies et des concepts visés. De par sa portée, cette étude se heurte à certaines limites. De premiers éléments d'analyse montrent que des obstacles mathématiques imprévus, comme la détermination d'un élément médian d'un intervalle d'entiers, peuvent entraver l'accès aux concepts. Les conditions d'expérimentation en classe réelle ne nous permettent pas non plus d'isoler certaines variables, qui restent ici incontrôlées (progression au cours de l'année, spécialités choisies au lycée, pratiques des enseignants, etc.). Nous n'avons pas non plus

cherché à évaluer finement les apprentissages éventuels des élèves ni planifié ou analysé en détail les phases d'institutionnalisation clôturant les séances. Malgré tout, nous pensons que cette situation, et plus généralement la famille des jeux algorithmiques décrite en section 2.3, offrent un potentiel didactique riche, et sont susceptibles d'être adaptées à d'autres problèmes.

Nous souhaitons à court terme affiner le choix des variables didactiques de la situation et les rétroactions du milieu, par exemple en opposant aux élèves un adversaire « expert » (enseignant ou machine). Il nous semblerait également important de déterminer des conditions permettant de favoriser le changement de point de vue entraîné par le passage à la situation 2. Un problème où la stratégie optimale pour D est moins évident, par exemple celui des pesées [9], pourrait le permettre. À plus long terme, nous souhaitons investiguer davantage les enjeux d'institutionnalisation portés par ces situations, en particulier celui de « disparition du jeu » au profit de connaissances plus decontextualisées. Nous pensons également que ce travail met en évidence une relation étroite entre l'enseignement de l'informatique et les connaissances mathématiques des élèves, qui revêt un sens particulier dans le contexte de la réforme actuelle du lycée. Enfin, il nous semblerait important de nous doter d'outils d'analyse plus fine des enjeux relatifs à la preuve, par exemple grâce à la typologie des preuves développée par Balacheff [1].

Références

1. Balacheff, N. : Processus de preuve et situations de validation. Educational Studies in Mathematics **18**(2), 147–176 (1987)
2. Balacheff, N., Margolinas, C. : Modèle de connaissances pour le calcul de situations didactiques. In : Balises en didactique des mathématiques : Cours de la 12e école d'été de didactique des mathémathiques, pp. 1–32. Recherches en Didactique des Mathématiques, La Pensée Sauvage (2005)
3. Bessot, A. : Une introduction à la théorie des situations didactiques (master "mathématiques, informatique" de grenoble 2003-2004). Les cahiers du laboratoire Leibniz **91** (2003)
4. Brousseau, G. : Théorie des situations didactiques. La pensée Sauvage, Grenoble (1998)
5. Douady, R. : Jeux de cadres et dialectique outil-objet. Recherches en Didactique des Mathématiques **7**(2), 5–31 (1986)
6. Meyer, A., Modeste, S. : Analyse didactique d'un jeu de recherche : vers une situation fondamentale pour la complexité d'algorithmes et de problèmes. In : Didapro 8 – DidaSTIC (2020)
7. Modeste, S. : Enseigner l'algorithme pour quoi ? Quelles nouvelles questions pour les mathématiques ? Quels apports pour l'apprentissage de la preuve ? Ph.D. thesis, Université de Grenoble (2012)
8. Modeste, S. : Prendre en compte l'épistémologie de l'algorithme. quels apports d'un modele de conceptions ? quelle transposition didactique ? Recherches en didactique des mathématiques **38**(1), 1–15 (2021)
9. Modeste, S., Ouvrier-Buffet, C., Gravier, S. : Algorithmique et apprentissage de la preuve. Repères IREM (79), pp 51–72 (2010)

A Matériel annexe

A.1 Feuille de partie

-

Date et heure :
Membres du groupe :

Compte-rendu des parties :

Numéro de partie	**Initiales par rôle**			**Borne (N)**	**Liste des propositions de D**												**Score**
	Devine	**Répond**	**Arbitre**		*1e*	*2e*	*3e*	*4e*	*5e*	*6e*	*7e*	*8e*	*9e*	*10e*	*11e*	*...*	
1																	
2																	
3																	
4																	
5																	
6																	
7																	
8																	
9																	
10																	

A.2 Frise de recherche

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Pensée informatique : approche didactique de l'identification de motifs

Marielle LÉONARD[12], Yann SECQ[1], Yvan PETER[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
prenom.nom@univ-lille.fr

Abstract. Cet article concerne l'identification de motifs, capacité relevant de de la pensée informatique, considérée comme essentielle lors de la résolution de problèmes de programmation de type itératif. L'approche didactique adoptée consiste à affiner la définition du concept de motif dans ce contexte en distinguant motif visuel et motif algorithmique. Une analyse quantitative d'une centaine de problèmes posés lors du concours de programmation Algorea (200 000 participants) permet de caractériser et de catégoriser un certain nombre de difficultés relatives à cette activité d'identification de motifs.

Mots clés: pensée informatique, motif, identification de motifs, boucle, didactique de l'informatique, analyse quantitative, large échelle

1 Introduction

La réintroduction de l'informatique dans les programmes scolaires de nombreux pays a provoqué un regain d'intérêt pour les travaux en didactique dans cette discipline. En France, des contenus d'informatique sont présents dans les programmes scolaires de l'école obligatoire depuis 2016. Ainsi, la programmation informatique apparaît dans le programme de mathématiques de cycle 3[1] (classes de CM1, CM2 et 6ème, de 9 à 12 ans) et prescrit l'activité "*Programmer les déplacements d'un robot ou ceux d'un personnage sur un écran en utilisant un logiciel de programmation.*". Pour le cycle 4 (classes de 5ème, 4ème et 3ème, de 12 à 15 ans), l'attendu de fin de cycle mentionné dans le programme[2] est "*Écrire, mettre au point et exécuter un programme simple.*". Mais qu'entend-on par "*un programme simple*" ?

[1] Programme du cycle 3 en vigueur à la rentrée 2020, mathématiques, section espace et géométrie

[2] Programme du cycle 4 en vigueur à la rentrée 2020, mathématiques, thème E – Algorithmique et programmation

Pour ce cycle, la boucle figure parmi les connaissances visées avec la séquence d'instructions, les instructions conditionnelles et les variables informatiques. En première approche, on pourrait donc considérer qu'un programme comportant une seule boucle, sans imbrication, instructions conditionnelles ni gestion de variables, est un programme simple.

Lors d'études précédentes [11] [8], nous avons mis en place des scénarios pédagogiques pour explorer comment des élèves de l'école primaire appréhendent des problèmes de programmation qui présentent une structure itérative, c'est-à-dire dont la solution implique l'utilisation d'une boucle. Les résultats obtenus lors de ces études de cas nous ont amenés à considérer l'identification de motifs, c'est-à-dire de redondances, comme essentielle dans le traitement de ce type de problèmes. En particulier, nous avons repéré un palier de difficulté de manière récurrente : le passage d'une à plusieurs instructions dans le corps de la boucle.

Dans cet article, nous approfondissons l'étude de l'identification de motifs lors de la résolution de problèmes de programmation de type itératif. Nous nous questionnons sur les paramètres qui rendent la résolution de ce type de problèmes plus ou moins difficile. Nous nous demandons aussi si les difficultés identifiées évoluent avec l'âge des élèves. À cette fin, nous mobilisons des éléments de la théorie des champs conceptuels de Vergnaud [17] pour conduire une analyse à priori, puis nous menons une analyse statistique à large échelle à partir des taux de réussite à 101 problèmes de type itératif issus des éditions 2018 à 2021 du concours national de programmation Algorea organisé par l'association France-ioi. Nous concluons en proposant de nouvelles pistes d'études pour affiner notre compréhension de l'apprentissage des bases de la programmation informatique.

2 Identification de motifs

Le terme "*motif*" est un terme polysémique employé dans de nombreux contextes, dont plusieurs domaines artistiques : "*petit élément caractéristique d'une composition musicale, qui en assure l'unité*", "*dessin, ornement, le plus souvent répété, sur un support quelconque*".

En informatique, le terme "*motif*" est associé au mot anglais "*pattern*". D'une part, il est présent dans des travaux menés autour des "*design patterns*" dans le champ du génie logiciel [5]. D'autre part, "*looking for patterns*" [21], "*pattern recognition*" [7], "*identifying and making use of patterns*" [4] l'expression variant suivant les auteurs, désigne une habileté identifiée comme faisant partie de la pensée informatique. Certains travaux mentionnent plus spécifiquement la notion de boucle sur laquelle nous nous concentrons dans cet article. Dans le cadre de description des compétences liées à la pensée informatique défini par Gouws et al. [6] à partir d'une revue de littérature, une catégorie s'intitule "*Patterns and Algorithms*", catégorie dans laquelle la notion de boucle est prise en exemple. Rich et al. [12] définissent des trajectoires d'apprentissage, comprenant des objectifs et des exemples d'activités associées, dont l'une porte sur les structures itératives. Les auteurs mentionnent l'importance de la perception de la redondance car intimement liée à l'initiation à la notion de boucle, mais

aucune analyse de l'activité d'identification de motifs n'est proposée dans ces travaux. Pourtant, cette activité a déjà été repérée comme source de difficultés pour les élèves [1].

En revanche, en didactique des mathématiques, certains travaux abordent cette question d'identification de motifs. Ainsi, pour Collins & Laski [3], un motif est une séquence avec une régularité réplicable, qui peut varier selon une ou plusieurs dimensions. Liljedahl [9] propose de distinguer deux catégories de motifs : les *repeating patterns* et les *number/growing patterns* (Fig. 1). La première correspond à une structure cyclique générée par la répétition d'une unité discernable. Cette définition est reprise dans plusieurs travaux [10][18][20]. La seconde correspond à un motif paramétré par une ou plusieurs informations.

Fig. 1.Catégorisation de motifs en didactique des mathématiques

En nous inspirant des définitions précédentes, nous définissons un « motif » dans notre contexte comme **une entité repérable au sein d'un ensemble car répétée à l'identique ou avec des variations prédictibles**.

Liljedahl [9] répertorie différentes tâches en lien avec le concept de motif : copier une suite de motifs, continuer une suite de motifs, retrouver des éléments manquants dans une suite de motifs, transférer une suite de motifs d'une représentation vers une autre, identifier un motif. En s'appuyant sur des expérimentations menées auprès de jeunes enfants de 3 à 6 ans, Warren & al. [18] construisent une séquence pédagogique puis établissent une progression dans la difficulté de ces tâches [19][20]. Dans cette progression, l'identification de motifs est la tâche la plus difficile et elle est celle qui révèle la compréhension de la structure de la suite de motifs [19]. En effet, la stratégie de correspondance terme à terme, qui consiste à traiter les éléments du motif un par un sans considérer celui-ci dans sa globalité, est systématiquement mise en échec lors de cette activité [3].

Dans notre contexte, nous nous intéressons à l'activité d'identification de motifs dans le champ de la didactique de l'informatique, en rapport avec la confrontation à des problèmes de programmation de type itératif. Nous considérons que la distinction proposée par Liljedahl [9] est un début de caractérisation des formes de complexité d'abstraction des motifs, notamment la transition de motifs directement observables (visuels) vers des motifs non observables (changements d'état de l'environnement, voire similarité de processus dans le cadre de *design patterns*) et nous proposons d'étudier plus en détails les caractéristiques des motifs à identifier.

3 Concept de classe de situation

Pour caractériser et catégoriser les motifs à identifier lors de problèmes de programmation de type itératif, nous nous appuyons sur le concept de classe de situation développé par Vergnaud [16] au sein de la théorie des champs conceptuels. Celle-ci s'inscrit dans une approche constructiviste et cognitiviste de l'apprentissage. Elle vise à comprendre la conceptualisation, en particulier dans le cas des activités cognitives complexes, dont la programmation informatique fait partie. L'unité d'analyse est le couple sujet / situation au sens de tâche. L'hypothèse de Vergnaud est que toute action finalisée repose sur une "*conceptualisation-en-acte*", c'est-à-dire que les actions du sujet rendent compte d'une activité cognitive restant le plus souvent implicite, y compris pour le sujet lui-même. En didactique de l'informatique, la théorie des champs conceptuels a été mobilisée par Rogalski [13][14] pour étudier "l'alphabétisation informatique" au lycée et plus récemment par Spach [15] pour analyser des situations de robotique pédagogique. Dans notre contexte, nous nous plaçons dans ce cadre d'analyse pour étudier des situations où le but du sujet est de concevoir un programme informatique pour résoudre un problème de type itératif.

Vergnaud invite à analyser les situations auxquelles le sujet est confronté en les regroupant en *classes*. Cette catégorisation peut être envisagée du point de vue de l'expert, par une analyse des caractéristiques des situations, et du point de vue du sujet, en étudiant la manière dont celui-ci traite les situations auxquelles il est confronté. L'expert s'appuie sur l'identification de *variables de situation* visant à différencier des situations proches. Le changement de valeur d'une variable de situation peut affecter ou non la structure du traitement de la situation par le sujet. Si c'est le cas, cela permet de définir deux classes de situation distinctes.

Vergnaud insiste aussi sur la progressivité de la conceptualisation, qu'il convient de considérer sur un temps long. Dans une étude sur les structures additives, Vergnaud & Durand [17] ont demandé à 28 élèves de chaque niveau du CP au CM2 de résoudre des problèmes additifs dont la réponse est strictement la même numériquement, mais pour lesquels la formulation du problème induit un raisonnement différent. Ils ont ainsi identifié des classes de situation qui correspondent à des paliers de difficulté dans la résolution de ces problèmes additifs. Leurs résultats montrent également un effet de l'âge sur la capacité des élèves à résoudre ces problèmes.

Dans cet article, nous proposons d'affiner la définition du concept de motif et de caractériser certaines difficultés relatives à l'activité d'identification de motifs lors du traitement de situations de type itératif dans un contexte de programmation. Nous nous appuyons sur les travaux réalisés autour de ce concept de motif et nous mobilisons le concept de *classe de situation* pour catégoriser ces problèmes de type itératif. Nous nous inspirons aussi de l'étude de Vergnaud & Durand (1976) que nous avons transposée dans notre contexte. La section suivante détaille la méthodologie et le cadre expérimental qui a été mis en œuvre pour réaliser cette étude.

4 Méthodologie et cadre expérimental

Nous travaillons à partir de 101 situations de type itératif qui sont issues des éditions 2018 à 2021 du concours national de programmation Algorea. Toutes ces situations consistent à programmer des actions et des déplacements d'un robot virtuel sur une grille en langage *Scratch*. Leur point commun est que la séquence d'actions à faire exécuter au robot virtuel comporte de la redondance, qu'il est nécessaire d'identifier pour résoudre le problème. La solution de référence implique donc une boucle ou plusieurs boucles en séquence (mais elle ne nécessite pas de boucles imbriquées). Pour l'étude de ces situations, nous envisageons les deux points de vue indiqués par Vergnaud [16]. D'une part, nous réalisons une analyse du point de vue de l'expert, appelée aussi analyse à priori, de ces 101 situations. D'autre part, nous analysons l'activité des sujets confrontés à ces situations lors de leur participation au concours Algorea, à travers les taux de réussite relevés pour ces problèmes.

4.1 Analyse à priori : motif visuel et motif algorithmique

Lorsqu'une situation de programmation d'un robot virtuel sur une grille est de type itératif, deux motifs distincts sont à considérer lors du traitement de la situation. Parmi les concepts piliers de l'informatique [2], le premier motif est lié au concept de données. Dans notre contexte, il s'agit d'un **motif visuel**, qui est observable sur la grille. Il est constitué de cases adjacentes, contenant ou non un élément saillant visuellement (case marquée, ou contenant un objet). Le second motif est lié aux concepts d'algorithme et de machine. Il est constitué d'actions à faire exécuter les unes après les autres par la machine, actions qui sont induites à la fois à partir du motif repéré dans les données et à partir des spécificités de cette machine. C'est la répétition d'une suite d'actions dans le même ordre chronologique qui détermine ce second motif, que nous appelons **motif algorithmique**. Celui-ci n'est observable que lors de l'exécution effective des actions. Dans notre contexte, le motif algorithmique dépend du motif visuel et du système d'orientation du robot virtuel. Dans un programme conçu en langage *Scratch*, il correspond à la séquence de blocs dans le corps de la boucle.

Résoudre un problème de programmation de type itératif dans notre contexte requiert donc d'identifier le motif visuel sur la grille, de mettre en correspondance ce motif visuel avec les actions à faire réaliser au robot virtuel sur cette même grille, puis d'exprimer ce motif algorithmique avec le langage de programmation *Scratch*.

Pour chacun de ces motifs, visuel puis algorithmique, nous identifions plusieurs paramètres ou caractéristiques, qui correspondent à des *variables de situation* au sens de Vergnaud [16]. Pour le motif visuel, nous considérons le nombre de cases qu'il occupe sur la grille, la présence d'éléments saillants visuellement au sein des motifs visuels et la présence d'éléments de décor sur la grille. Pour le motif algorithmique, nous retenons le nombre d'actions constituant le motif et la présence d'actions ne faisant pas partie de la suite de motifs (correspondant aux instructions hors de la boucle). Comme variable de situation, nous étudions aussi le degré de correspondance entre le motif visuel et le motif algorithmique.

4.2 Cadre expérimental

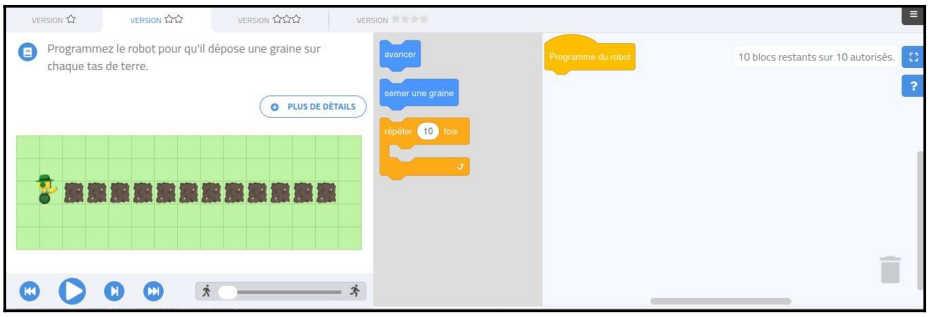


Fig. 2. Environnement de programmation du concours Algorea (situation 1, où la case délimite le motif visuel)

Les situations de déplacement de robot virtuel que nous étudions sont issues du concours de programmation en ligne Algorea, dont l'environnement de programmation est représenté sur la figure 2.

Cet environnement est adapté pour notre étude sur l'identification de motifs. D'une part, lors des problèmes de type itératif, le bloc *répéter* est le seul bloc de structure de contrôle disponible. Le sujet induit rapidement qu'il se trouve dans la situation où il a besoin d'utiliser ce bloc *répéter*. D'autre part, le nombre de blocs autorisé pour concevoir un programme est limité, ce qui contraint l'utilisation de ce bloc *répéter*. En revanche, le nombre d'essais n'est pas limité, ce qui permet le tâtonnement.

Suivant les tours (trois par an), le concours Algorea implique jusqu'à plus de 200 000 participants du CM1 à la terminale (de 9 à 18 ans). Dans le cadre de cette étude, nous nous intéressons seulement aux résultats individuels en langage *Scratch* pour les élèves du CM1 à la 3ème (de 9 à 15 ans), ce qui représente entre 6 000 et 75 000 participants suivant les tours, répartis sur les 6 niveaux étudiés, avec une surreprésentation des élèves de collège. Ainsi, nous ne maîtrisons pas la taille de l'échantillon étudié qui varie suivant les tours, mais reste conséquente. De plus, la situation est totalement écologique, la passation du concours ayant lieu en milieu scolaire ou à la maison. Toutefois, nous considérons que cette taille conséquente d'échantillon compense les variations de contexte de passation.

5 Résultats et analyse

Pour chaque situation, nous disposons du taux de réussite par niveau de classe. De manière préliminaire à l'étude sur l'identification de motifs, nous procédons à quelques analyses d'ordre plus général. Nous vérifions d'une part la robustesse de nos données concernant les taux de réussite. Lorsque l'on considère tous les niveaux de classe ensemble, un test d'indépendance de *khi2* nous permet de vérifier que tous les écarts de taux de réussite entre deux situations sont statistiquement significatifs avec

une *p-value* inférieure à 0,01. Pour un niveau de classe particulier, un écart de taux de réussite de 5 unités de pourcentage entre deux situations est significatif pour le collège (*p-value*<0,02). Seules quelques situations pour le niveau élémentaire dont l'effectif est plus réduit amènent à des écarts de taux de réussite de 5 unités de pourcentage moins robustes statistiquement.

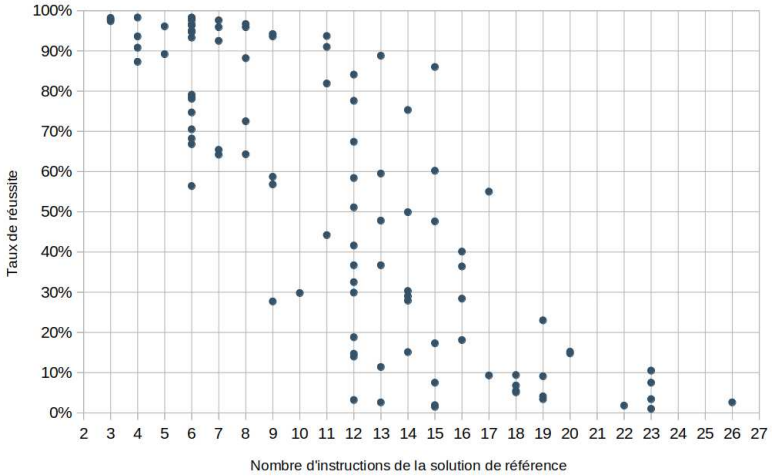


Fig. 3. Diagramme de dispersion du taux de réussite en fonction du nombre d'instructions de la solution de référence (taux de corrélation linéaire -0,81; p-value<0,05 au test de Bravais-Pearson)

D'autre part, la figure 3 confirme, que de manière attendue, le taux de réussite baisse lorsque le nombre d'instructions dans la solution de référence augmente. Cependant, nous remarquons une dispersion des valeurs importante sur l'axe vertical, parfois de plus de 50 unités de pourcentage, ce qui nous indique que d'autres variables de situation ont un effet sur le taux de réussite. L'identification et l'étude de ces variables sont l'objet des sections suivantes. À cette fin, pour chaque caractéristique relevée, nous calculons la médiane des taux de réussite et l'écart interquartile comme indicateurs de la distribution des données.

5.1 Motif visuel

Dans cette section, c'est l'aspect visuel du motif qui importe, indépendamment des actions que doit exécuter le robot virtuel.

Concernant le nombre de cases sur lesquels s'étend le motif visuel, nous pouvons distinguer deux classes de situation de manière très marquée (Fig. 4). Pour une première classe de situation, le motif visuel est constitué d'une seule case de la grille (exemple Fig. 2). Le taux de réussite de ces problèmes est élevé dès l'école élémentaire. L'écart interquartile est faible, ce qui signifie que cette caractéristique est prégnante dans l'explication du taux de réussite. En revanche, l'écart interquartile est beaucoup plus élevé si le motif visuel s'étend sur plusieurs cases (exemples Fig. 6).

Dans ce cas, d'autres variables contribuent significativement à la valeur du taux de réussite.

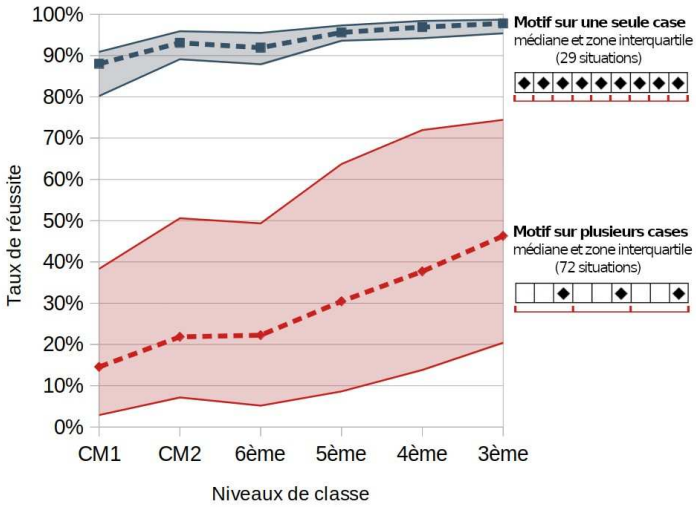


Fig. 4. Deux classes de situation : situations où la case délimite le motif, et situations pour lesquelles le motif s'étend sur plusieurs cases

Pour 70 situations pour lesquelles le motif visuel s'étend sur plusieurs cases, nous étudions les cases adjacentes qui ne font pas partie du même motif.

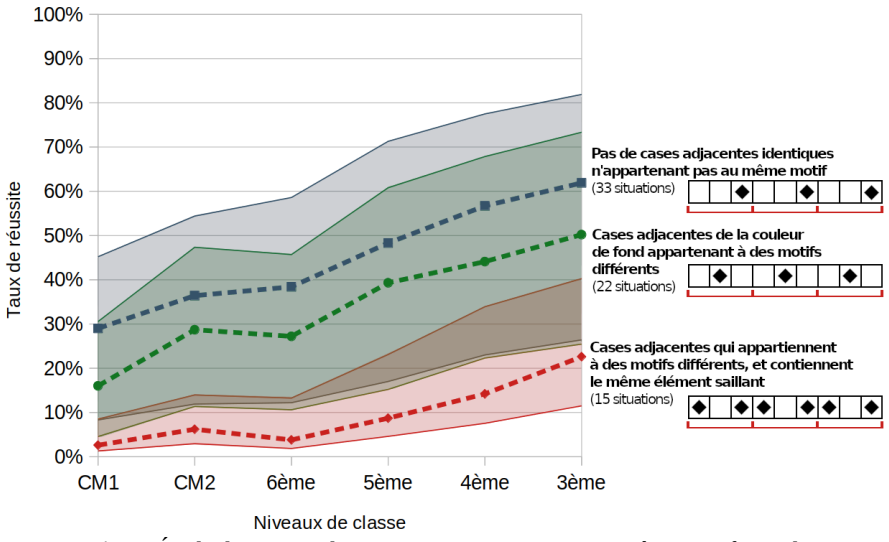


Fig. 5. Étude des cases adjacentes n'appartenant pas au même motif visuel

Lorsque des cases identiques adjacentes comportant un élément saillant visuellement n'appartiennent pas au même motif (exemples Fig. 6 : situations 3 et 4), le taux de réussite est bas (Fig. 5 : courbe rouge), et ce de manière plus marquée chez les plus

jeunes élèves. En revanche, lorsque ce sont deux cases adjacentes de la couleur du fond de la grille qui appartiennent à des motifs différents, le taux de réussite est proche de celui des situations sans cases adjacentes identiques appartenant à des motifs différents. Ce résultat nous amène à penser que les éléments saillants sont pris comme points de repère privilégiés lors de l'identification du motif visuel. Des éléments saillants identiques sur des cases adjacentes sont perçus comme faisant partie d'une même entité visuelle. Lorsque ceux-ci n'appartiennent pas au même motif, cela rend le motif moins visible et donc son identification plus difficile.

Nous montrons de la même manière l'effet de la présence d'éléments de décor sur la grille. Faute de place, nous ne donnons pour chaque modalité que la valeur de la médiane (Q_2) et de l'écart interquartile (EI) pour tous les niveaux de classe pris ensemble, l'unité étant le point de pourcentage du taux de réussite. Suivant la manière dont ils sont disposés, les éléments de décor sont plutôt une aide ou une source de difficulté. Lorsqu'ils contraignent complètement le parcours du robot (Q_2: 60.0, EI: 31.3), ils constituent une aide par rapport à des situations sans éléments de décor (Q_2: 51.1, EI: 0.58). Si ce n'est pas le cas, ils semblent agir comme distracteurs et constituent une source de difficultés (Q_2: 27.7, EI: 49.0). Cette difficulté devient massive lorsque ces éléments de décors rendent certains motifs visuellement différents (Q_2: 3.2, EI: 3.3).

Ainsi l'étude des caractéristiques des motifs visuels montre que la nature des éléments présents sur la grille a un effet sur la complexité de la situation. Plus le motif est facilement isolé visuellement, et plus la situation est bien résolue. À l'inverse, les facteurs qui perturbent la visibilité du motif impactent négativement le taux de réussite de la situation.

5.2 Correspondance entre motif visuel et motif algorithmique

Après avoir identifié le motif visuel sur la grille, il est nécessaire d'en déduire le motif algorithmique correspondant. Concernant la correspondance entre motif visuel et motif algorithmique, nous distinguons 5 classes de situation. Pour les trois premières classes, tous les motifs visuels sont identiques, ce qui n'est plus vrai pour les deux dernières classes.

Une première classe de situation, très distincte, et que nous avons déjà identifiée dans la section précédente concerne les situations où la case délimite le motif (exemple sur la figure 2). Les classes suivantes sont représentées figure 6. Nous regroupons dans une deuxième classe les situations pour lesquelles nous avons une correspondance stricte entre motif visuel et motif algorithmique. Chaque action de déplacement est repérable par la limite entre deux cases et les autres actions sont identifiables par un élément saillant visuellement. Ce sont les situations où le déplacement du robot n'est possible que dans une seule direction et les situations de déplacements en orientation absolue (nord, sud, est, ouest). Une troisième classe correspond aux situations où plusieurs états du robot virtuel sur une même case sont visuellement identiques, rendant partielle la correspondance entre motif visuel et mo-

tif algorithmique. Ce sont les situations en orientation relative pour lesquelles les actions de pivotement du robot ne sont pas observables avant l'exécution du programme. Il est nécessaire de simuler mentalement les actions de pivotement du robot, en se les représentant sur les cases adéquates et en maintenant l'orientation du robot en mémoire. La quatrième classe concerne les situations en orientation relative pour lesquelles la disposition des motifs est cyclique. Les motifs visuels ne sont donc plus identiques qu'à une rotation d'un quart de tour près. Enfin, pour les situations regroupées dans une cinquième classe, la correspondance entre motif visuel et motif algorithmique est entravée. Il est nécessaire de faire abstraction de certains éléments visuels. Soit des éléments saillants ou des éléments de décor sont équivalents mais visuellement différents, soit plusieurs motifs visuels sont partiellement superposés, perturbant la visibilité de chacun d'eux.

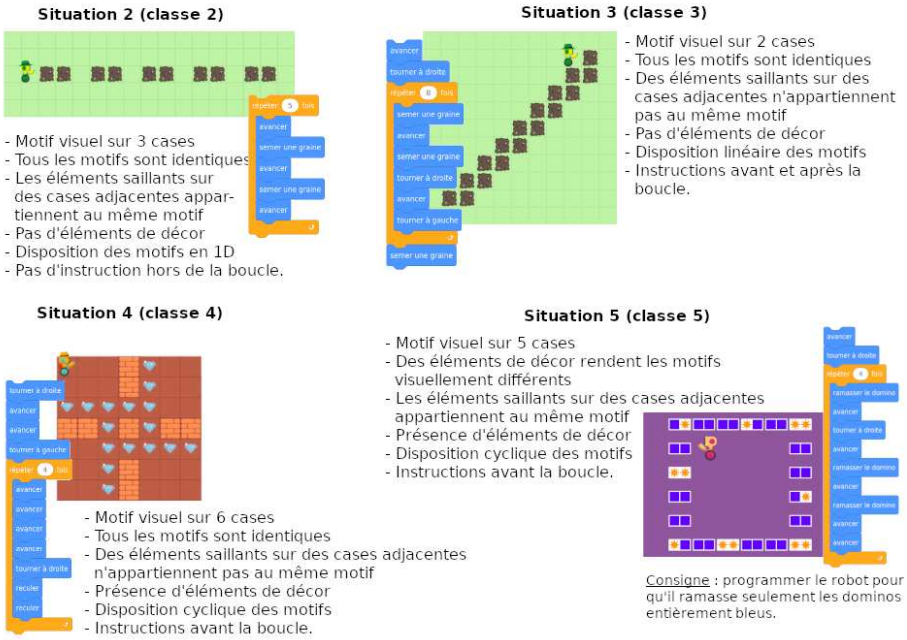


Fig. 6. Exemple type de situation pour chaque classe définie pour la correspondance entre motif visuel et motif algorithmique

Les 5 classes de situation définies précédemment correspondent à une gradation dans la difficulté à mettre en correspondance motif visuel et motif algorithmique (Fig. 7). Les situations de la classe 1, pour lesquelles la correspondance entre les deux motifs est attachée à la case, sont bien réussies par la majorité des élèves dès l'école élémentaire. En revanche, les situations de classe 5, qui nécessitent beaucoup plus de capacités d'abstraction, sont encore difficiles pour la plupart des élèves de fin de collège. Les zones interquartiles des classes 1 et 5 ne chevauchent pas celle des autres classes de situation. Nous en déduisons que le degré de correspondance entre motif visuel et motif algorithmique détermine fortement la difficulté de ces situations. En revanche, les classes 2, 3 et 4 ont des zones interquartiles partiellement superposées,

ce qui signifie que d'autres variables impactent aussi la difficulté de ces situations de manière significative. Ce sont aussi les classes de situation où l'on observe la plus forte progression au cours des 6 niveaux de classe étudiés.

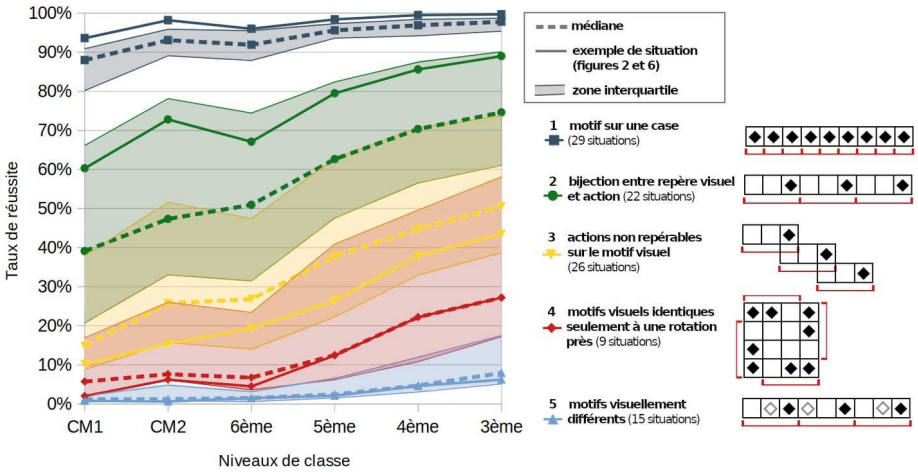


Fig. 7. Étude de la correspondance entre motif visuel et motif algorithmique

Concernant le motif algorithmique exprimé en langage *Scratch*, nous montrons en outre que le taux de réussite est corrélé au nombre d'instructions dans la boucle (taux de corrélation linéaire de -0,79) et que la situation est significativement moins bien résolue lorsqu'il est nécessaire de placer des instructions hors de la boucle, en particulier avant. Nous pensons que cette dernière difficulté est liée au repérage de la position du robot à considérer pour le début de la série de motifs visuels, position qu'il faut anticiper mentalement.

6 Discussion et perspectives

Nous avons montré dans cette étude qu'un problème de programmation de type itératif, même si la solution comporte seulement une boucle, n'est pas forcément un problème simple. Lors de la résolution de ce type de problème, l'identification d'un motif visuel et celle du motif algorithmique correspondant sont essentiels. Nous avons caractérisé des facteurs qui rendent difficile l'identification du motif visuel et nous avons établi une gradation dans les difficultés rencontrées, en particulier pour la mise en correspondance des motifs visuel et algorithmique. Parmi les difficultés identifiées, nous retrouvons celle, déjà repérée dans une étude précédente [8], liée à l'association de la situation de programmation avec de l'orientation dans l'espace.

D'autres travaux sont en cours pour approfondir cette étude. D'une part, peut-on considérer qu'un élève maîtrise la notion de boucle lorsqu'il a résolu les problèmes de programmation en tâtonnant, ce qui est possible dans ce contexte ? D'autre part, nous savons que l'identification de motif n'est pas seule en jeu dans le traitement des situations de type itératif. Une fois le motif identifié, il faut dénombrer les motifs, ce qui

peut induire d'autres difficultés qui restent à analyser. Pour affiner notre compréhension, nous avons besoin de données plus précises. C'est pourquoi, nous avons mis en place une collecte de traces d'activité à plusieurs échelles. Hormis les taux de réussite collectés à l'échelle nationale analysés dans cet article, nous disposons de traces d'activités à l'échelle de classes et d'enregistrements vidéo de participation au concours à l'échelle individuelle. Les traces d'activité à l'échelle des classes devraient nous permettre de distinguer les procédures de résolution expertes et les réussites par tâtonnement. Quant à l'analyse des enregistrements vidéos, nous cherchons à identifier des indicateurs qui traduisent le raisonnement, la *conceptualisation-en-acte* [16] du participant (procédure experte, erreurs). L'objectif sera ensuite d'apparier ces indicateurs avec les traces d'activité afin de passer à l'échelle, c'est-à-dire de faire le lien entre les trois échelles de collecte.

Remerciements

Ce travail est soutenu par le projet Interreg Teach Transition (https://teachtransition.eu). Nous tenons aussi à remercier l'association France-IOI pour la mise à disposition des taux de réussite au concours Algorea.

References

1. Baron, G. L., Drot-Delange, B.: L'informatique comme objet d'enseignement à l'école primaire française? Mise en perspective historique. Revue française de pédagogie. Recherches en éducation, (195), 51-62. (2016).
2. Berry, G. L' Hyperpuissance de l'informatique : Algorithmes, données, machines, réseaux. Odile Jacob. (2017).
3. Collins, M. A., & Laski, E. V.: Preschoolers' strategies for solving visual pattern tasks. Early Childhood Research Quarterly, 32, 204-214. (2015).
4. Csizmadia, A., Curzon, P., Dorling, M., Humphreys, S., Ng, T., Selby, C., Woollard, J.: Computational thinking-A guide for teachers. (2015).
5. Gamma, E., Johnson R., Vlissides J., Helm R.: Design Patterns: Elements of Reusable Object-Oriented Software. (1994).
6. Gouws, L. A., Bradshaw, K., Wentworth, P.: Computational thinking in educational activities : An evaluation of the educational game light-bot. In: Proceedings of the 18th ACM conference on Innovation and technology in computer science education, 10-15. (2013).
7. Hsu, T.-C., Chang, S.-C., Hung, Y.-T.: How to learn and how to teach computational thinking : Suggestions based on a review of the literature. In: Computers & Education, 126, 296-310. (2018).
8. Léonard, M., Peter, Y., Secq, Y.: Reconnaissance et synthèse de motifs redondants avec des élèves de 6-7 ans MOTIFS.MOTIFS.MOTIFS. ⟹ 3 x MOTIFS. 3 x MOTIFS. In : Colloque DIDAPRO 8 -DIDASTIC - L'informatique, objets d'enseignements enjeux épistémologiques, didactiques et de formation. (2020).
9. Liljedahl, P.: Repeating pattern or number pattern : The distinction is blurred. Focus on learning problems in mathematics, 26(3), 24-42. (2004).
10. Papic, M.: Promoting Repeating Patterns with Young Children—More Than Just Alternating Colours! Australian Primary Mathematics Classroom, 12(3), 8. (2007).

11. Peter, Y., Secq, Y., Léonard, M. : Reconnaissance de motifs redondants et répétitions : Introduction à la Pensée Informatique. STICEF (Sciences et Technologies de l'Information et de la Communication pour l'Éducation et la Formation), 27(2) (2020).
12. Rich, K. M., Strickland, C., Binkowski, T. A., Moran, C., Franklin, D.: K-8 Learning Trajectories Derived from Research Literature : Sequence, Repetition, Conditionals. Proceedings of the 2017 ACM Conference on International Computing Education Research, 182-190. (2017).
13. Rogalski, J.: Acquisition de savoirs et de savoir-faire en informatique. In: Cahiers de Didactique des Mathématiques, 43. (1987).
14. Rogalski, J., Vergnaud, G.: Didactique de l'informatique et acquisitions cognitives en programmation. Psychologie française, 32.4. (1987).
15. Spach, M.: Activités robotiques à l'école primaire et apprentissage de concepts informatiques: quelle place du scénario pédagogique? Les limites du co-apprentissage (Doctoral dissertation, Université Sorbonne Paris Cité). (2017).
16. Vergnaud, G.: La théorie des champs conceptuels. In Recherches en didactique des mathématiques: Vol. 10/2.3. La Pensée Sauvage. (1991).
17. Vergnaud, G., Durand, C.: Structures additives et complexité psychogénétique. Revue française de pédagogie, 28-43. (1976).
18. Warren, E., Cooper, T.: Using Repeating Patterns to Explore Functional Thinking. Australian Primary Mathematics Classroom, 11(1), 9. (2006).
19. Warren, E., Miller, J.: Exploring Four Year Old Indigenous Students' Ability to Pattern. International Research in Early Childhood Education, 1(2), 42-56. (2010).
20. Warren, E., Miller, J., Cooper, T.: Repeating patterns : Strategies to assist young students to generalise the mathematical structure. Australasian Journal of Early Childhood, 37(3), 111-120. (2012).
21. Wing, J. M.: Computational thinking. In: Communications of the ACM, 49(3), 33-35. (2006).

Analyse de l'adaptabilité de jeux pour l'apprentissage de la pensée informatique ou de la programmation

Hajar Saddoug[1], Aryan Rahimian[1], Marne Bertrand[2][0000-0002-4953-9360], Mathieu Muratet[3][0000-0001-6101-5132], Karim Sehaba[4][0000-0002-6541-1877] and Sébastien Jolivet[5][0000-0003-3915-8465]

[1] Sorbonne Université, Place Jussieu, Paris, France
hajar.saddoug@etu.sorbonne-universite.fr,
rahimian_aryan@yahoo.fr
[2] ICAR UMR 5191, Université Lumière Lyon 2, Parvis René Descartes, Lyon, France
bertrand.marne@ens-lyon.fr
[3] Sorbonne Université, CNRS, INS HEA, LIP6, F-75005 Paris, France
mathieu.muratet@lip6.fr
[4] LIRIS - Université Lumière Lyon 2, avenue Pierre Mendès-France, Lyon, France
karim.sehaba@liris.cnrs.fr
[5] IUFE, Université de Genève, rue du Général-Dufour, Genève, Switzerland
sebastien.jolivet@unige.ch

Résumé. Ce travail s'inscrit dans la problématique générale de l'appropriation par les enseignants et les formateurs des jeux sérieux dédiés à l'apprentissage de la pensée informatique ou de la programmation. Pour favoriser cette appropriation, nous souhaitons mettre en œuvre une approche méta-design favorisant la genèse instrumentale. Dans cet article, nous cherchons à tester à quel point une sélection de jeux sérieux se prête à ce type d'approche. Nous y identifions des critères d'analyse destinés à caractériser la capacité à l'instrumentalisation de jeux sérieux et avec ceux-ci nous expertisons en profondeur 10 jeux. Nos résultats montrent que les capacités d'adaptation de la plupart de ces 10 jeux restent faibles et hors de portée de beaucoup d'enseignants et formateurs.

Mots-clés : Jeux sérieux, Méta-design, Genèse instrumentale, Pensée informatique, Programmation.

1 Introduction

La question de l'enseignement de l'informatique, qui remonte aux années 70, se caractérise par un mouvement de balancier entre deux conceptions de ce qu'il faut enseigner. D'un côté, une vision d'une informatique-outil qui promeut la formation à l'usage des outils. De l'autre, une vision de l'informatique en tant qu'objet d'enseignement qui considère qu'il faut avant tout enseigner les concepts et méthodes de l'informatique [1]. Depuis maintenant quelques années, l'informatique disciplinaire réapparaît dans les programmes scolaires, à l'école on parle plus particulièrement de *pensée informatique*.

Selon Jeannette Wing [2], la pensée informatique met en jeu un répertoire de cinq capacités cognitives : (1) la pensée algorithmique, (2) l'abstraction, (3) l'évaluation, (4) la décomposition, et (5) la généralisation. Gilles Dowek [3] quant à lui propose une structuration de l'informatique en quatre concepts afin que les contenus enseignés donnent une image fidèle de la discipline elle-même : (1) l'information numérique, (2) les algorithmes, (3) les langages, et (4) les machines informatiques. Ainsi, l'informatique ne se réduit pas au codage qui ne constitue que la phase finale de traduction dans un langage de programmation. Elle est avant tout une activité conceptuelle. Ainsi conçue, l'informatique quitte le statut d'outil pour ce qu'elle est en réalité : une science ayant ses spécificités et nécessitant un apprentissage propre. Cela étant, enseigner la pensée informatique au primaire et l'informatique au secondaire demande un investissement particulièrement important de la part des enseignants [4] pour aborder cette nouvelle discipline, du fait d'un manque de formation (initiale et continue).

En parallèle de ces évolutions, les jeux sérieux d'apprentissage ont émergé et proposent de nombreux avantages par rapport aux outils d'enseignement classique. En effet, de nombreux auteurs considèrent les jeux sérieux d'apprentissage comme prometteurs, notamment pour augmenter l'engagement et la motivation des apprenants [5], d'autres envisagent ces outils pour favoriser un apprentissage plus constructiviste [6]. Concernant le thème de l'informatique et de la programmation, de nombreux jeux sérieux existent [7, 8], mais leur appropriation par les enseignants reste un enjeu majeur. Nous faisons l'hypothèse qu'une réponse à ce défi est d'impliquer les enseignants à l'aide d'une méthode de conception participative telle que le méta-design.

Le méta-design est une méthode de conception participative avancée, dans laquelle les utilisateurs finaux (« *owners of problems* ») sont impliqués de façon centrale dans les phases initiales de la conception. Mais, et c'est la raison de sa pertinence pour résoudre les problèmes d'appropriation, les utilisateurs doivent aussi avoir les moyens de continuer à être concepteurs durant les phases d'utilisation des artefacts, grâce à *l'underdesign* que Fischer et al. [9] définissent comme : « [...] *underdesign aims to provide social and technical instruments for the owners of problems to create the solutions [of their problems] themselves at use time.* ». Il s'agit donc, dans notre cas de figure, d'identifier si des enseignants et formateurs, considérés comme une des catégories d'utilisateurs finaux de jeux sérieux dédiés à la programmation, parviennent à s'inscrire dans une démarche de méta-design (les apprenants-joueurs sont une autre catégorie d'utilisateurs finaux non traitée ici). Nous faisons l'hypothèse que cette démarche peut favoriser la genèse instrumentale [10] et plus particulièrement l'instrumentalisation, et donc une forme d'appropriation [11].

Néanmoins pour pouvoir mettre en œuvre cette démarche de méta-design, il est nécessaire de disposer d'artefacts adaptables (dans notre cas de jeux sérieux) fournissant les fonctionnalités permettant à ses utilisateurs finaux (enseignants ou des formateurs) d'observer leurs usages à différents niveaux d'abstraction et de les adapter en fonction, en prenant en compte leur contexte et leurs besoins. Dans cet article, nous réalisons une analyse de jeux sérieux sur le thème de la pensée informatique avec comme focale les outils mis à disposition pour favoriser la genèse instrumentale et l'appropriation de ces jeux par les enseignants [11].

Dans la première partie, nous présentons notre méthode de travail, c'est-à-dire comment nous avons élaboré nos critères d'analyse, comment nous avons construit notre sélection de jeux à expertiser et comment nous avons mené l'analyse. Dans la seconde partie, nous présentons et discutons les résultats de cette analyse, avant de conclure dans la dernière partie.

2 Méthode

La littérature propose déjà plusieurs travaux de recensement et d'analyse de jeux sérieux destinés à la programmation. Parmi les travaux que nous avons pu consulter, nous remarquons que l'attention est principalement portée sur les thèmes et les compétences enseignées par les jeux sérieux analysés [7, 8, 12, 13]. Parmi ces travaux, certains comme Lindberg et al. [12], identifient un mauvais alignement entre les compétences enseignées dans les jeux sérieux (29 ont été analysés) et celles présentes dans les programmes (notamment français). On retrouve cette tendance chez Malliarakis et al. et Miljanovic et Bradbury [7, 13] avec respectivement 12 et 49 jeux expertisés (et de nombreux recoupements entre ces travaux). Cependant, ces travaux ne cherchent jamais à caractériser la genèse instrumentale, notamment par l'instrumentalisation. Seuls Vahldick et al. [8] évoquent l'importance que peuvent jouer des éditeurs dans certains cas, sans, pour autant, en faire un critère d'analyse des jeux expertisés (40 jeux analysés).

Nous avons donc entrepris de conduire notre propre analyse en nous concentrant sur la capacité des jeux expertisés à disposer, d'une part, des moyens d'observation permettant à l'enseignant de comprendre les usages du jeu, et d'autre part, des mécanismes d'adaptation lui permettant de réajuster le jeu en fonction de ces usages observés.

2.1 Sélection des jeux

Pour construire la liste des jeux traitant de la pensée informatique à analyser, nous avons associé les jeux que nous avions déjà identifiés dans d'autres travaux [14, 15], à ceux qui proviennent des revues systématiques citées plus tôt, et à d'autres, recherchés sur le web avec les mots-clés « jeux introduction programmation », mais également « jeux codage », « coding game ». Nous n'avons conservé que les jeux sérieux actuellement fonctionnels (par exemple, les jeux utilisant la technologie *Flash* sont désormais très difficilement utilisables), peu chers ou gratuits. Ainsi, nous avons pu recenser 48 jeux permettant l'apprentissage du code ou de la pensée informatique.

Comme nous nous concentrons sur les jeux sérieux destinés à l'apprentissage de la pensée informatique et de la programmation dans un contexte scolaire, nous avons aussi choisi d'exclure les jeux ne présentant aucun accès direct à la programmation par le joueur, ainsi que ceux seulement disponibles sur téléphone mobile ou tablette (dont l'utilisation est plus contrainte en situation scolaire). Nous avons également regroupé les jeux qui proposent un fonctionnement similaire à Scratch [16] ou Blockly [17] qui sont des micromondes [18] proposant une programmation par blocs d'instructions.

Dans un second temps, nous avons passé en revue la liste de jeux, pour les trier et les catégoriser en prenant en compte les éléments suivants :

— Ressemblances et dissemblances entre les jeux ;
— Avantages et inconvénients des outils-auteurs et de suivi présents dans les jeux ;
— Catégories (ex. : âge, discipline, coût, etc.) ;
— Type de formation (lorsque celle-ci est proposée par le jeu) : autoformation, enseignement classique, etc.

Cette catégorisation nous a permis d'affiner la liste pour nous concentrer sur une sélection de 10 jeux (parmi les 48) présentant peu de points communs et correspondant bien à notre objectif : identifier des jeux sérieux centrés sur l'apprentissage de la pensée informatique et de la programmation, qui pourraient être adaptés ou facilement appropriés par des enseignants et des formateurs, c'est-à-dire permettant l'instauration d'un contexte de méta-design. Désormais, chaque jeu retenu pour l'analyse est associé à une catégorie distincte.

Table 1. Liste initiale des 48 jeux, les 10 jeux sélectionnés sont mis en évidence.

Algoblocs	Code hunt	CodeWars	Elevator Saga	Hocus Focus	**Pyrates**	SQL Murder Mystery
AlgoPython	Code Moji	**Codin'Game**	Empire Of Code	Human Resource Machine	**RoboCode**	StarLogo
Bee Bot	Code Monkey	Collabots	Flexbox Defense	Imagi	Ruby Warrior	Tynker
Blockly	Code Monster	**Compute It**	Flexbox Froggy	**KoduGame**	Run Marco	Unruly Splats
Ceebot	CodeCombat	CSS Diner	Gladiabots	Le chevalier de la programmation	Scratch	Untrusted
CheckIO	Codefi	Cyber Dojo	Grid Garden	Pixel	Screeps	Vim adventures
Code	CodeGym	Duskers	Heartbreak	ProgAndPlay	**SPY**	

2.2 Pré-création de la grille d'analyse

Afin d'établir notre grille d'analyse, nous avons identifié plusieurs critères. Dans un premier temps, nous avons identifié trois temporalités liées aux modifications potentielles qui pourraient être opérées par des enseignants ou des formateurs dans les jeux de la liste. À chacune de ces trois temporalités (avant, pendant et après l'instrumentalisation), nous avons associé des types d'assistance à la modification qui nous serviront de critère :

— **Pré-modification** : présence de tutoriels et d'explications de différents formats, concernant la manipulation du jeu et/ou son utilisation à d'autres fins formatives.

— **Durant la modification** : capacité à la détection des erreurs pendant la manipulation du jeu.
— **Post-modification** : possibilité d'avoir une vue d'ensemble des modifications appliquées au jeu et/ou la présence d'une aide complémentaire (maintenance humaine ou numérique interactive type forum, FAQ, etc.)

Dans un second temps, et pour donner suite à cette étape préparatoire, nous avons formulé d'autres critères plus fins et facilement mesurables. Ces critères sont détaillés dans la section suivante.

2.3 Critères d'analyse

Notre but est d'analyser les jeux, non pas pour en sélectionner certains et en disqualifier d'autres, mais de les catégoriser avec des critères fins, afin d'offrir une grille claire rendant compte des convergences et divergences entre jeux adaptables et dotés de moyens de collecte de traces, dédiés à l'enseignement de la programmation ou de la pensée informatique.

Nous avons hésité à utiliser un système de score en raison de sa subjectivité, d'autant plus que l'essentiel dans notre grille était d'indiquer la présence effective ou non du critère défini. Ainsi, nous avons opté pour les appellations « Présence » ou « Absence » du critère, et dans le cas de la présence, nous indiquons toujours en commentaire sous quelle forme le critère est présent. Néanmoins, pour l'un des critères, le « Degré » de scénarisation, nous devions pouvoir être plus précis, sans pour autant indiquer trop de détails inutiles à une analyse rapide. Par conséquent, nous avons opté pour 3 états possibles : un niveau faible, un niveau fort ou une absence de scénario.

Au fur et à mesure que des critères apparaissaient lors de nos explorations, nous avons décidé de les ranger par classe. Nous avons ainsi identifié 7 classes comprenant chacune différents critères dont voici la description.

Table 2. Récapitulatif des 7 classes de critères utilisés dans notre analyse

Classes	Critères associés
Adaptabilité	Code source libre, Profil enseignant, Modification IHM, Modalité d'interaction
Édition	Modification de tâches, Ajout de tâches, Organisation de tâches, Création de scénario, Éditeur fourni
Capacité formative	Guide explicatif (pour jouer), Guide pédagogique (pour éditer), Assistance didactique, Assistance pédagogique
Traces	Progression, Performance, Informations générales, Format de trace
Diversité de choix	Langage de programmation
Communauté	Forum amateurs, Contact avec l'auteur / éditeur
Scénarisation	Degré, Tâches indépendantes

L'adaptabilité est une classe de critères centrale pour nos travaux, elle décrit la capacité du jeu à être modifié. Celle-ci se décompose en 4 critères :

— Code source libre : nous avons cherché à identifier la disponibilité du code source, sa licence, et la présence de ressources numériques ayant été utilisées.
— Profil enseignant : nous avons identifié si un compte et un profil spécifique existaient pour des formateurs ou des enseignants, leur donnant accès à des fonctionnalités spécifiques (créer des leçons, les organiser, visualiser des traces, scores, etc.).
— Modification IHM : nous avons identifié si l'interface du jeu était modifiable pour être adaptée.
— Modalité d'interaction : nous avons identifié s'il était possible de modifier les interactions existantes dans le jeu (par exemple : l'utilisation d'autres outils que le clavier et la souris).

La classe « Édition » se distingue de la classe « Adaptabilité », car elle est moins générale et vise spécifiquement la modification du contenu du jeu (tâche et scénario). Ses critères sont :

— Modification de tâches : la possibilité de modifier le contenu d'une tâche existante, d'un niveau, son niveau de difficulté, etc.
— Ajout de tâches : la possibilité d'ajouter des tâches à celles existantes dans le jeu.
— Organisation de tâches : la possibilité de réorganiser les tâches existantes ou celles que l'on a soi-même créées (quand c'est possible), pour éventuellement créer un scénario.
— Création de scénario : la possibilité de créer un enchaînement de tâches.
— Éditeur fourni : la mise à disposition ou non d'un éditeur facilitant la création, la modification et l'organisation des tâches.

Dans le contexte de nos travaux, nous avons mis en avant une autre classe de critères qui nous paraît importante : « Capacité formative ». Elle a pour but d'identifier, par exemple, si le système forme l'enseignant, ou s'il l'aide à enseigner la programmation. Est-ce que le système lui indique clairement ce qu'il peut faire avec lui ? Si oui, comment le fait-il ? Lui fournit-il des feedbacks durant sa conception pédagogique ? Lui propose-t-il des exercices adaptés aux notions visées ? Le corrige-t-il dans la difficulté de ces derniers ? etc. Donc, il s'agit d'identifier toute information ou tout moyen facilitant la prise en main du système et l'enseignement par celui-ci. Que ce soit avant même d'élaborer et de composer sa leçon dans le système ou pendant son élaboration. Dans cette classe, nous avons 4 critères :

— Guide explicatif (pour jouer) : décrit si sont présentes des informations potentiellement utiles à la compréhension et au bon usage du jeu dans le système.
— Guide pédagogique (pour éditer) : décrit si sont présentes des informations potentiellement utiles à la compréhension et au bon usage de l'enseignement dans le système.
— Assistance didactique : décrit si sont présents des moyens d'aide à l'enseignant pour mieux enseigner les notions visées (en proposant notamment des exercices adaptés)
— Assistance pédagogique : décrit si est présente une assistance qui corrige et guide l'enseignant pendant qu'il compose sa leçon dans le jeu, afin de la rendre plus efficiente.

La classe « Traces » décrit au travers de plusieurs critères les moyens mis à disposition pour collecter, consulter et analyser des traces des actions des apprenants dans le jeu. En effet, l'adaptation d'un jeu par l'enseignant ou le formateur est souvent motivée par des écarts entre des comportements supposés (des apprenants-joueurs) durant la conception du jeu et les pratiques réelles qui peuvent surgir durant son déploiement. De tels écarts ne sont perceptibles, dans certains cas, qu'à travers la mise en place d'un système de collecte de traces, représentant les interactions entre l'utilisateur et le jeu. Les traces ainsi collectées peuvent faire l'objet de transformation afin de mettre en évidence des indicateurs de haut niveau d'abstraction assistant l'enseignant dans la mise en œuvre des adaptations qui s'imposent pour le bon déroulement de l'apprentissage.

Cette classe n'est pas indépendante de la « Capacité formative » ni de « Adaptabilité » (et plus particulièrement son critère « Profil enseignant »). En effet, la présence d'une interface enseignant semble nécessaire à l'accès aux traces. Les critères de cette classe sont :

- Progression : identifie la présence de traces (indicateurs) de la progression des apprenants dans le jeu (les niveaux passés, débloqués, les niveaux rejoués, etc.).
- Performance : identifie la présence de traces (indicateurs) de la performance de chaque apprenant (les scores, les badges obtenus, etc.).
- Informations générales : identifie la présence d'informations sur les apprenants (noms, nombres, classe, groupe, etc.)
- Format de trace : décrit le format des traces fournies (brut ou raffiné). Ici, nous distinguons le format raffiné, à savoir des informations rassemblées et présentées dans le but d'informer l'utilisateur sur les traces ; et le format brut qui s'illustre par des informations partielles et dispersées dans le système.

La classe « Communauté » ne regroupe que deux critères, l'existence d'un lieu d'échanges entre les utilisateurs d'une part (réseau social, forums…) et, d'autre part, celle de coordonnées de contact des auteurs ou de l'éditeur du jeu. Nous avons ajouté cette classe pour identifier si de l'aide extérieure était éventuellement accessible à des enseignants ou des formateurs qui voudraient se lancer dans des modifications.

La classe « Scénarisation » porte sur la scénarisation du jeu et indique le degré de dépendance entre les tâches. Les 2 critères sont :

- Degré : qui indique à quel point la scénarisation est forte, c'est-à-dire l'importance d'une histoire ou d'un contexte qui relie les différentes étapes unitaires du jeu (niveaux, tableaux, exercices, etc.) entre-elles. Ce critère peut avoir 3 valeurs (forte, faible, absence).
- Tâches indépendantes : qui indique le degré de dépendance des tâches (les unités qui composent la scénarisation). Dans le cas d'une scénarisation forte, on s'interrogera sur l'indépendance des tâches du scénario. Peuvent-elles être modifiées séparément et sans affecter la scénarisation générale ?

Un dernier critère ne pouvait être classé ailleurs : « Langage de programmation ». Nous lui proposons donc une classe spécifique « Diversité de choix ». Il s'agit d'indiquer si le jeu permet l'utilisation d'un ou plusieurs langages de programmation. Nous

ne l'avons pas placé dans la classe « Édition », car il s'agit de la diversité de langage pour jouer et non pour éditer. La classe « Diversité de choix » est la seule classe qui est spécifique aux jeux dédiés à l'apprentissage de la pensée informatique et de la programmation. En effet, toutes les autres classes ne contiennent que des critères qui pourraient être utilisés pour analyser l'adaptabilité de jeux sérieux dédiés à d'autres types d'apprentissages.

2.4 Procédure d'analyse

Pour l'analyse de chaque jeu avec la grille, nous avons procédé de la manière suivante : dans un premier temps, le même jeu a été analysé par deux examinateurs séparément, avec une prise de notes des éléments pertinents à retenir. Dans un second temps, la grille a été complétée conjointement. Le temps accordé pour l'examen de chaque jeu fut différent. En effet, nous avons remarqué que selon le type du jeu, le délai d'un examen complet était en moyenne d'une heure. Pour d'autres jeux, étant donné leur caractère simple et leur interface minimaliste, la durée de leur analyse était sensiblement plus courte.

La phase d'analyse n'a pas seulement été de jouer au jeu, mais a aussi nécessité un temps d'appropriation et de maîtrise par les examinateurs : lecture de guides, visionnage de tutoriels avant de jouer et de tester l'éditeur de niveau quand il était présent. Cette analyse a eu comme particularité une approche méta-ludique, dans laquelle les examinateurs devaient être conscients du jeu et d'eux-mêmes en tant que joueurs pendant l'acte de jeu. Cette attitude est nécessaire pour identifier les avantages et les points d'amélioration de chaque jeu et de constater, au fur et à mesure, les points convergents et divergents entre ces derniers.

En effet, l'analyse s'est réalisée par classe de critères, dès que toutes les cases d'une même classe étaient remplies, l'examinateur passait à l'analyse d'une autre classe de la grille. Certaines classes, comme « Adaptabilité » et « Communauté » demandaient plus de temps à l'examinateur que les autres, car des recherches annexes, notamment sur le web, devaient être réalisées pour juger de l'existence de certains critères en dehors du jeu lui-même. Par exemple, la disponibilité du code source libre, la présence de forums dédiés au jeu ou toute information annexe présente sur des blogs d'amateurs.

Une colonne commentaire a été ajoutée à chaque classe, pour indiquer des informations complémentaires sur la présence ou l'absence de certains critères, mais aussi pour ajouter des particularités observées dans le jeu.

3 Résultats et discussion

Par manque de place, nous ne présentons pas ici le détail de l'analyse des 10 jeux retenus, mais la grille d'analyse complète est disponible à l'adresse web suivante : https://recherche.univ-lyon2.fr/meta-dect/Analyse_Adaptabilite_Jeux_Serieux.ods.

Concernant la classe « Adaptabilité », nous constatons que de nombreux jeux ont un code source ouvert (6/10) ce qui est un élément positif pour favoriser leur évolution. En revanche, en l'état, peu de jeux (3/10) fournissent des interfaces spécifiques aux

enseignants (Profil enseignant). Les fonctionnalités proposées sont très différentes entre les jeux : Algoblocs permet à l'enseignant de publier des défis ou d'activer/désactiver des options de commentaires et de forum ; AlgoPython quant à lui permet aux enseignants de créer des classes et d'y associer des élèves ; enfin, le jeu Code permet à l'enseignant d'organiser ses projets, de structurer des cours décomposés en unités et en chapitres. Les deux derniers critères de cette catégorie sont très peu identifiés. Aucun des jeux étudiés n'offre la possibilité de modifier l'interface du jeu et seul KoduGame permet d'utiliser des périphériques d'interaction autre que le couple clavier/souris.

Concernant la classe « Édition », la plupart des jeux ne donnent aucun contrôle sur la scénarisation, ils permettent seulement au joueur de débloquer les niveaux au fur et à mesure de leur progression. Deux jeux sortent du lot : SPY et KoduGame. Le premier offre la possibilité d'ajouter/supprimer/modifier des niveaux du scénario à travers l'édition de fichiers XML. Le second permet aux enseignants de créer des mondes dans lesquels ils pourront définir des tâches, les organiser et préciser des consignes. Nous avons néanmoins noté la possibilité d'ajouter des tâches dans Code, les tâches (appelées unités dans le site) peuvent être assignées à un cours, mais leur contenu ne peut être ni modifié ni réorganisé.

Pour la classe « Capacité formative », 6 jeux sur 10 proposent des guides explicatifs et seulement 3 (Code, PyRates et Ceebot) sont accompagnés de guides pédagogiques qui donnent des informations sur les notions abordées à travers le jeu. Les deux derniers critères de cette classe sont dépendants de la possibilité de créer/modifier une tâche. Aucun des deux jeux identifiés dans la classe « Édition » ne propose d'outils permettant d'assister l'enseignant tant sur les dimensions pédagogiques que didactiques.

Dans la classe « Traces », trois jeux sortent du lot (AlgoPython, CodinGame et Algoblocs). Les actions des joueurs sont tracées afin de construire des tableaux de bord à destination des enseignants affichant des indicateurs de progression et de performance. En revanche, aucun des jeux ne rend accessibles les données tracées rendant impossibles des traitements/analyses personnels.

Pour la classe « Diversité de choix », un seul jeu propose au joueur de pouvoir manipuler différents langages de programmation : Codin'Game. Seuls des langages textuels sont proposés (pas de langages à base de blocs), mais plusieurs paradigmes de programmation sont possibles comme des approches orientées objet, fonctionnelles et impératives. L'enseignant est donc libre de choisir son langage parmi une large palette de 27 langages de programmation différents.

Enfin, la classe « Scénarisation » nous montre que la plupart des jeux (7/10) proposent une forme de scénarisation impliquant des liens entre les différents niveaux de jeu. Ce lien peut être indépendant du contenu des tâches comme dans PyRates où les niveaux sont structurés par une histoire incluant une thématique et des personnages, mais où chaque niveau peut aussi être joué indépendamment des autres niveaux.

4 Conclusion

Nos travaux s'inscrivent dans la problématique générale de l'appropriation par les enseignants et les formateurs des jeux sérieux d'apprentissage de la pensée informatique

et de la programmation. Plus précisément, nous cherchons à caractériser le degré d'adaptabilité de chaque jeu afin de favoriser la dimension instrumentalisation de la genèse instrumentale à travers une approche méta-design.

Dans ce contexte, cet article présente un travail préliminaire d'identification de critères d'adaptabilité de jeux sérieux et leur utilisation pour l'analyse d'une sélection de jeux destinés à ces apprentissages.

Nous présentons une grille d'analyse comportant 7 classes de 1 à 5 critères. Chaque classe décrit une composante d'adaptabilité pour d'éventuels utilisateurs aux profils divers, allant d'un enseignant de primaire qui est novice en informatique à un informaticien expérimenté. À l'exception de l'une des classes et de son critère associé spécifique de la pensée informatique et de la programmation (diversité du choix dans les langages de programmation), nous pensons que l'un des apports principaux de ce travail est de fournir un ensemble structuré de critères d'analyse de l'adaptabilité réutilisables pour n'importe quels types de jeux sérieux d'apprentissage.

Nous avons identifié 48 jeux destinés à l'apprentissage de la pensée informatique et de la programmation. Nous avons réalisé une sélection de 10 jeux sur lesquels nous avons utilisé notre grille d'analyse.

Grâce à cette analyse des 10 jeux sélectionnés avec nos critères, nous avons observé que même s'ils sont nombreux à proposer un code source ouvert (critère « Code source libre »), peu d'entre eux sont facilement adaptables (critères « Modification IHM », « Modalité d'interaction », « Modification de tâches », « Ajout de tâches », « Organisation de tâches », « Création de scénario »). Quand elles sont possibles (ex. SPY, PyRates, Kodu Game), les modifications nécessitent le plus souvent une bonne connaissance de la programmation (éditeurs inexistants, sauf pour Kodu Game) et il n'y a pas de ressources formatives à disposition pour être assisté dans ces modifications.

Au-delà des apports directs de ces travaux (un ensemble de critères d'analyse de l'adaptabilité réutilisables dans d'autres contextes et sa mise en œuvre sur 10 jeux), ces résultats nous permettent d'identifier que les exigences pour avoir un jeu d'apprentissage de la pensée informatique ou de la programmation permettant le méta-design sont élevées et qu'aucun de ceux expertisés ici ne les satisfait complètement. Nos futures recherches seront donc orientées vers l'identification de moyens pour rendre les jeux sérieux destinés à ces apprentissages plus propices à la mise place d'une approche méta-design dans l'optique de favoriser leur appropriation par les enseignants et les formateurs.

Remerciements. Les auteurs remercient l'Université Lumière Lyon 2 pour son soutient financier dans le cadre du programme APPI 2020.

Références

1. Baron G-L, Drot-Delange B (2016) L'informatique comme objet d'enseignement à l'école primaire française ? Mise en perspective historique. Revue française de pédagogie Recherches en éducation 51–62. https://doi.org/10.4000/rfp.5032
2. Wing JM (2006) Computational thinking. Commun ACM 49:33–35. https://doi.org/10.1145/1118178.1118215

3. Dowek G (2011) Les quatre concepts de l'informatique. Athènes : New Technologies Editions, p 21
4. Kradolfer S, Dubois S, Riedo F, Mondada F, Fassa F (2014) A Sociological Contribution to Understanding the Use of Robots in Schools: The Thymio Robot. In: Beetz M, Johnston B, Williams M-A (eds) Social Robotics. Springer International Publishing, Cham, pp 217–228
5. Bouvier P, Sehaba K, Elise L (2014) A trace-based approach to identifying users' engagement and qualifying their engaged-behaviours in interactive systems. Application to a social game. User Modeling and User-Adapted Interaction (UMUAI'14) 45:413–451
6. Bouvier P, Sehaba K, Lavoué É (2014) A trace-based approach to identifying users' engagement and qualifying their engaged-behaviours in interactive systems: application to a social game. User Model User-Adap Inter 24:413–451. https://doi.org/10.1007/s11257-014-9150-2
7. Miljanovic M, Bradbury J (2018) A Review of Serious Games for Programming
8. Vahldick A, Mendes AJ, Marcelino MJ (2014) A review of games designed to improve introductory computer programming competencies. In: 2014 IEEE Frontiers in Education Conference (FIE) Proceedings. IEEE, Madrid, Spain, pp 1–7
9. Fischer G, Giaccardi E, Ye Y, Sutcliffe AG, Mehandjiev N (2004) Meta-design: a manifesto for end-user development. Communications of the ACM 47:33–37
10. Rabardel P (1995) Les hommes et les technologies : une approche cognitive des instruments contemporains. Armand Colin, Paris, France
11. Folcher V, Rabardel P (2004) Hommes, artefacts, activités : perspective instrumentale. Presses Universitaires de France
12. Lindberg RS, Laine TH, Haaranen L (2019) Gamifying programming education in K-12: A review of programming curricula in seven countries and programming games. British Journal of Educational Technology 50:1979–1995
13. Malliarakis C, Satratzemi M, Xinogalos S (2014) Educational games for teaching computer programming. In: Research on e-Learning and ICT in Education. Springer, pp 87–98
14. Marne B, Sehaba K, Muratet M (2021) Vers une méthode de méta-design de jeux sérieux : Application pour l'apprentissage de la programmation à travers Blockly Maze. p 342
15. Nédélec X, Marne B, Muratet M, Sehaba K, Lapostolle J (2021) Une approche méta-design du jeu sérieux pour l'enseignement de l'informatique à l'école élémentaire. In: Broisin J, Declercq C, Fluckiger C, Parmentier Y, Peter Y, Secq Y (eds) Atelier " Apprendre la Pensée Informatique de la Maternelle à l'Université ", dans le cadre de la conférence Environnements Informatiques pour l'Apprentissage Humain (EIAH). Fribourg, Switzerland, pp 96–106
16. Resnick M, Maloney J, Monroy-Hernández A, Rusk N, Eastmond E, Brennan K, Millner A, Rosenbaum E, Silver J, Silverman B, Kafai Y (2009) Scratch: Programming for All. Communications of the ACM 52:60–67. https://doi.org/10.1145/1592761.1592779
17. Fraser N (2015) Ten things we've learned from Blockly. In: 2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond). IEEE, pp 49–50
18. Papert S (1987) Microworlds: transforming education. In: Artificial intelligence and education. Ablex Publishing Corp.355 Chestnut St. Norwood, NJUnited States, pp 79–94