DOI : https://doi.org/10.26034/cm.iptic.2025.11756

L'AUTOCONFRONTATION CROISÉE ENTRE ÉTUDIANTS POUR FORMER AU DÉBOGAGE INFORMATIQUE

Olivier Neuhaus, Haute Ecole Fédérale en Formation Professionnelle, Suisse, olivier.neuhaus@hefp.swiss

Mario Ramalho, Ecole de Métiers de Fribourg, Suisse, mario.ramalho@edufr.ch

Résumé

Le débogage du code d'un autre développeur est une activité centrale sur le terrain des professionnels de l'informatique. Cependant il n'est que rarement abordé en formation ou alors souvent de manière artificielle (par exemple déboguer son propre code). Afin de former des apprentis informaticiens en école plein temps au débogage informatique, un dispositif basé sur des exercices de dépannage réalisés par les élèves et filmés a été mis en place dans un centre de formation en Suisse. En s'appuyant sur la trace vidéo aussi bien du flux à l'écran que de la webcam réalisée durant l'exercice, une autoconfrontation croisée entre pairs permet de faire ressortir non pas le résultat mais bien la démarche ayant mené les étudiants à la détection puis la correction rapide d'un bug informatique dans un programme amenant ainsi à un réel partage de pratique entre les étudiants.

Mots-clés : Formation professionnelle ; Autoconfrontation croisée ; Analyse vidéo ; Programmation informatique ; Débogage

1. Introduction

En Suisse, le cursus de formation professionnelle permet, dès la fin de la formation scolaire obligatoire, d'apprendre un métier au sein d'une entreprise, en alternance avec des cours dispensés en centres de formation. Cette voie offre à des jeunes, dès la fin de leur parcours scolaire la possibilité de débuter une formation dans une entreprise en dual (alternance entre un emploi en entreprise et une formation en école professionnelle) ou en plein temps à l'école (offrant aussi bien la dimension pratique que théorique en école) dans l'optique de l'obtention d'un CFC (Certificat Fédéral de Capacité) ou AFP (Attestation Fédérale de Formation Professionnelle) leur ouvrant directement les portes du monde de l'emploi.

Pour les étudiants informaticiens[1], l'apprentissage de la programmation représente un enjeu important de la formation. Si cet apprentissage connaît depuis plusieurs années de grandes évolutions grâce à divers outils permettant par exemple de rendre visible son aspect abstrait, des compétences comme celles liées au dépannage ou à la correction de bug sont néanmoins toujours complexes à acquérir (Siegmund et al., 2014). Ce sont pourtant bien des « compétences essentielles pour la maintenance et les modifications de programmes » (Rogalski, 2015) pour les informaticiens en formation et futurs professionnels qui consacreront entre 20 et 40% de leur temps d'activité sur cette tâche (Michaeli et Romeike, 2019). Or, cette tâche est difficile à réaliser. Keringhan et Plauger (1995) présentent l'activité de débogage comme deux fois plus difficile à réaliser que celle de développement. En formation, si les bases de l'algorithmique et de la programmation sont généralement apprises en institut, aussi bien pour la formation duale qu'en école à plein temps, la faculté à prendre en main le code d'un autre professionnel, à l'analyser pour en faire une maintenance ou à le dépanner en cas de problème, sont généralement des tâches apprises au contact du terrain en entreprise ou apprises « à la dure » (Michaeli et Romeike, 2019) sans avoir abordé en classe les stratégies adéquates.

Pour les étudiants en école plein-temps, sans contact direct avec le terrain cette compétence est d'autant plus difficile à acquérir. Une approche prometteuse consiste à mettre ces étudiants en situation de dépannage durant les cours et à déployer ensuite une didactique basée sur l'alternance et situations (Gremion, 2022), les poussant à analyser leur propre style et à le comparer à un genre du métier (Clot et Faïta, 2000). Cependant, le constat qui ressort fréquemment des enseignants qui empruntent cette approche en classe est que, si certains étudiants parviennent rapidement à détecter et corriger le code qui pose problème, d'autres n'y parviennent toujours pas même après une durée plus longue. Selon notre propre expérience, le fait de créer des espaces d'échanges entre les élèves qui y parviennent rapidement et les autres n'est, dans ce contexte, que peu porteur, la tentation étant forte pour les plus rapides de non pas partager l'analyse qu'ils ont réalisée mais d'uniquement donner la solution en indiquant la ou les lignes qu'il s'agit de corriger. Ces échanges laissent ainsi cachée toute la question de l'analyse du code ayant mené au débogage, élément pourtant central de cette problématique.

C'est la raison pour laquelle le dispositif présenté dans la suite de cet article a été imaginé. Expérimenté à l'Ecole des Métiers de Fribourg, nous en présenterons les principes et les premiers résultats pour ouvrir sur les perspectives que de telles approches peuvent offrir dans le champ de la formation en algorithmique.

2. Contexte

2.2 La formation des informaticiens en Suisse

La formation professionnelle suisse est fréquemment citée en exemple à l'échelle internationale (Kriesi et al., 2022). En 2022, la voie de la formation professionnelle a été choisie par près de 60% des jeunes en fin de leur scolarité obligatoire (SEFRI, 2022), la modalité duale (avec une formation en alternance entre un emploi en entreprise et des cours en école) étant majoritairement celle empruntée au détriment d'une formation en école à plein temps.

Pour chaque métier, les ordonnances de formation sont gérées par le Secrétariat d'Etat à la Formation, à la Recherche et à l'Innovation (SEFRI) qui s'appuie sur les Organisations nationales du monde du Travail (ORTRA) constituées de représentants des milieux économiques et professionnels. Pour la formation des informaticiens, « ICT-Formation professionnelle Suisse » est l'ORTRA en charge de proposer une ordonnance de formation en lien, tant avec le niveau des étudiants, qu'avec les besoins du terrain et de l'économie. Cette ordonnance est révisée à une fréquence de 4 ans, donnant lieu à de nouveaux contenus à aborder en formation.

[1] Afin d'alléger le style d'écriture, nous avons décidé d'utiliser une écriture exclusivement au masculin. Cependant lorsque le masculin est utilisé, il désigne autant les femmes que les hommes. Ainsi, par exemple, les termes informaticiens ou enseignants dans cet article désignent autant les femmes que les hommes même s'ils ne sont pas rédigés sous la forme enseignant·e·s, informaticien·e·s.

Suite aux deux dernières révisions de l'ordonnance fédérale (RO 2020 5411)[2], l'apprentissage de la programmation et de l'algorithmique se présente en plusieurs modules de formation de niveau progressif. Les contenus sont principalement axés sur l'apprentissage a) des concepts de base de la programmation (boucles, tests, …), b) du concept de programmation orientée objet et c) de la programmation web.

2.2 La compétence de débogage informatique pour les étudiants

Comme précisé précédemment, afin d'assurer la qualité du code développé en vue d'une intégration, le fait de tester son algorithme est une pratique importante pour un informaticien et fait intégralement partie du plan de formation. En formation, celle-ci est abordée majoritairement lors de l'étape de développement (l'étudiant développe un code qu'il doit lui-même tester pour s'assurer de son bon fonctionnement). Or, sur le terrain, le travail de l'informaticien consiste très fréquemment à reprendre le programme d'un autre développeur pour corriger d'éventuels problèmes, améliorer les performances ou y ajouter de nouvelles fonctionnalités (Michaeli et Romeike, 2019).

Pour les étudiants en formation de type dual, les compétences mobilisées pour réaliser un débogage sont souvent acquises dans des projets concrets au sein de l'entreprise formatrice, en étant amenés à réaliser ces actions sur les programmes gérés par l'entreprise, en binôme avec un formateur de terrain dans un premier temps puis de manière autonome si l'étudiant a acquis l'expérience nécessaire.

Elles sont cependant plus difficiles à traiter dans le cadre d'une formation en école à plein temps puisqu'il faut concevoir les codes sources d'erreur à corriger qui soient utiles pour l'apprentissage. Pour ces étudiants (qui constituent notre corpus de recherche), l'implication vers des projets concrets est plus difficile. Au contact des entreprises formatrices qui les accueillent en fin de formation durant quelques mois de stages pour une première immersion sur le terrain, nous avons constaté que ce manque d'expérience sur des projets concrets et dans la reprise et correction du code d'un autre développeur dans le cursus en école était problématique et rendait plus difficiles leur premiers pas dans le monde professionnel.

Les rencontres réalisées avec certains stagiaires ont de plus relevé que, conscients de cette lacune, une gêne était ressentie face à ce manque de compétence dans cette activité de débogage pour laquelle ils ne se sentent pas à la hauteur des attentes du marché de l'emploi.

3. Cadre théorique

3. 1 La didactique de l'informatique

L'apprentissage de l'informatique et plus spécifiquement de la programmation n'est pas facile à aborder pour les nouveaux professionnels de la branche. Fluckiger (2019) présente comme première difficulté la complexité d'aborder en formation le même objet qu'est la machine informatique, à la fois comme objet de savoir que comme outil d'apprentissage.

En abordant l'apprentissage de l'informatique comme objet de savoir, une des premières difficultés réside dans le fait de traiter des concepts, majoritairement abstraits (Hartmann et al., 2011). Rogalski (2015) présente ainsi la difficulté pour le programmeur de franchir le pas « du faire à faire faire », programmant sa pensée en code que l'ordinateur devra ensuite exécuter.

Dès lors, afin de faciliter la prise en main de ces concepts abstraits, de nombreux outils tels que des robots (Nijimbere et al., 2015 ; Romero et DeBlois, 2022), des représentations graphiques (Gautier et Wrobel-Dautcourt, 2015), des jeux ou des activités débranchées (Vincent et Cohen, 2017) ont vu le jour. L'apport de ces approches réside avant tout dans le fait de donner à voir ce qui est fait par l'ordinateur. A chaque fois, l'importance du debriefing post-activité avec les étudiants permettant de revenir sur les concepts abordés, apparait néanmoins comme centrale (Plumettaz, 2024).

3.2 L'enseignement du débogage informatique

Personne n'écrit de logiciels parfaits, le travail de débogage informatique occupe ainsi une partie importante du temps d'un informaticien (Thomas et Hunt, 2024, p. 121).

L'entreprise informatique IBM (2024) présente le débogage comme un processus qui a pour but de « découvrir la cause des erreurs de codage, de prévenir les problèmes de fonctionnement des logiciels et d'en améliorer les performances générales ». Ce processus se décompose en six étapes différentes : a)

[2] *La dernière ordonnance de formation https://www.fedlex.admin.ch/eli/oc/2020/941/fr*

Reproduire les conditions dans lesquelles le bug est apparu, b) trouver le bug, c) déterminer l'origine du problème, d) corriger le bug, e) tester pour valider le correctif, f) documenter le processus.

En abordant la compétence de débogage informatique, la partie abstraite présentée dans le chapitre précédent réside non seulement dans le fonctionnement non-visible de l'exécution du code mais également dans le raisonnement du professionnel qui débogue et dont on ne peut observer que le résultat de son processus (par exemple une correction effectuée dans une ligne ou partie du code permettant de corriger l'erreur). Or, l'action de débogage consiste principalement à mener une analyse, faisant intervenir non seulement les connaissances des professionnels mais aussi les connaissances des caractéristiques globales du système informatique (Romero et DeBlois, 2022). McCauley et al. (2008, p. 76) présentent sept connaissances utilisées durant la phase de localisation d'un bug et du débogage : 1) connaissance du programme prévu, 2) connaissance du programme réel, 3) compréhension du langage d'implémentation, 4) expertise générale en programmation, 5) connaissance du domaine d'application, 6) connaissance des bugs possibles, 7) connaissance des méthodes de débogage.

Fitzgerald et al. (2008) relèvent également des compétences différentes à mobiliser pour effectuer des tâches de développement et de débogage, mettant en évidence une complexité plus grande pour la deuxième.

Dès lors, si des étudiants ou professionnels sont à l'aise dans l'activité de débogage, ils le seront généralement également dans celle du développement algorithmique. L'inverse n'est cependant pas nécessairement le cas (Fitzgerald et al., 2008). Aborder l'activité de débogage permet ainsi de développer indirectement les compétences de développement et de développer l'esprit critique des professionnels face à des logiciels existants (Vitali-Rosati, 2024, p. 119).

En formation, le débogage informatique est reconnu comme étant une compétence importante mais qui reste à la fois difficile à apprendre pour les programmeurs novices et à enseigner pour les professeurs d'informatique (McCauley et al., 2008). Ceci peut expliquer la raison pour laquelle elle est souvent négligée dans les cursus de formation (Yang et al., 2024). McCauley et al. mettent néanmoins en lumière le fait que les bonnes pratiques menant à un débogage de qualité peuvent être enseignées et relèvent les pistes suivantes pour y parvenir (2008, p. 86) : 1) utiliser un enseignement explicite, 2) partir d'exemples de programmes concrets, 3) donner la possibilité aux étudiants de comparer leurs résultats, 4) amener à penser en mode de test d'hypothèses en évitant d'aller directement vers l'envie de corriger le bug.

Ce sont autant d'enjeux dont notre dispositif expérimental devra tenir compte.

3.3 L'autoconfrontation croisée ou collective

Nous nous appuyons sur les travaux de Clot et al. (2000) pour mettre en place un dispositif sous forme d'autoconfrontation croisée entre les étudiants. Le principe consiste à confronter des professionnels à l'activité de leur pair.

Clot présente les apports de l'autoconfrontation croisée sur base vidéo de la manière suivante :

> Des controverses professionnelles peuvent alors s'engager, portant sur les styles des actions de chacun d'entre eux. Les écarts stylistiques étant évalués par rapport aux formes génériques propres au groupe professionnel, ce sont alors le genre professionnel et ses variantes que les sujets font entrer dans une zone de développement. (Clot et al., 2000, p. 5)

En comparaison avec une autoconfrontation simple qui consiste à mettre un professionnel en confrontation avec son activité, Clot relève qu'une approche croisée entre pairs apporte une modification du niveau d'analyse, à destination non plus d'un chercheur mais d'un collègue, l'amenant à penser, à sentir et à agir selon sa perspective à lui. Clot parle dès lors d'un dispositif ouvrant à une co-analyse du travail quand « on reprend l'analyse en commun d'un même enregistrement vidéo avec un autre expert du domaine, un collègue de travail de même niveau d'expertise » (2018, p. 142). Dans notre expérimentation, nous le présenterons en section 4.1, des groupes de 2 ou 3 étudiants ont été formés, raison pour laquelle nous pouvons aussi bien parler d'autoconfrontation croisée que collective.

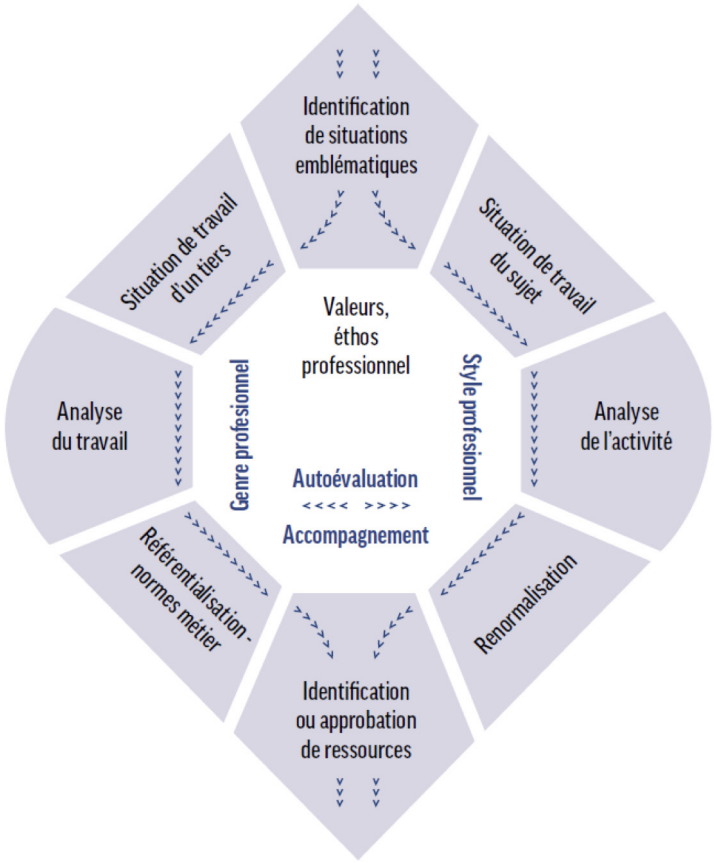
Le dispositif mis en place a ainsi principalement comme ambition de donner accès à la partie non visible de l'analyse menant au débogage informatique en confrontant les activités des pairs, ouvrant à des explications, des arrêts sur image et des compléments d'informations.

Le fait d'utiliser la trace vidéo comme support à cette activité permet de s'appuyer tant sur les fonctions de visibilisation que d'élucidation (Olry et al., 2021).

3.4 Didactique professionnelle par alternance et situation

Enfin, nous nous appuyons sur le dispositif DPAS (Modèle de didactique professionnelle par alternance et situations) de Gremion (2022) qui propose une approche didactique partant de situations emblématiques du métier et qui se divise en deux axes distincts : a) l'axe du genre professionnel de la didactique, b) l'axe du style professionnel de l'activité (Figure 1). L'apprentissage d'un métier passe par l'analyse de l'écart entre l'axe du genre et celui du style réalisé par le professionnel en formation : les parties autoévaluation et accompagnement se trouvant au centre de la Figure 1 dans le schéma adapté par Gremion et Maubant (à paraître, p. 64).

Figure 1 : Modèle DPAS (Gremion et Maubant, à paraître)



Ainsi, le fait de mettre les étudiants dans la situation emblématique du métier d'informaticien qu'est le débogage d'une application existante ne suffit pas à créer un apprentissage. Il s'agit pour les professionnels en devenir de mesurer l'écart entre leur style (présent dans la vidéo qu'ils ont eux-mêmes tournée lors de l'activité de débogage) et le genre professionnel (présent dans la vidéo d'un professionnel expérimenté, dans les normes professionnelles, dans les manuels, ...).

En confrontant les différents styles des apprentis durant l'activité d'analyse croisée des vidéos, la comparaison va au-delà d'une simple superposition de différentes pratiques mais permet de mesurer l'écart des pratiques de l'étudiant avec le genre du métier. La confrontation des différents styles permet ainsi de faire émerger un genre commun entre les étudiants (qui sera institutionnalisé lors de l'étape 6 en fin d'activité).

4. La recherche

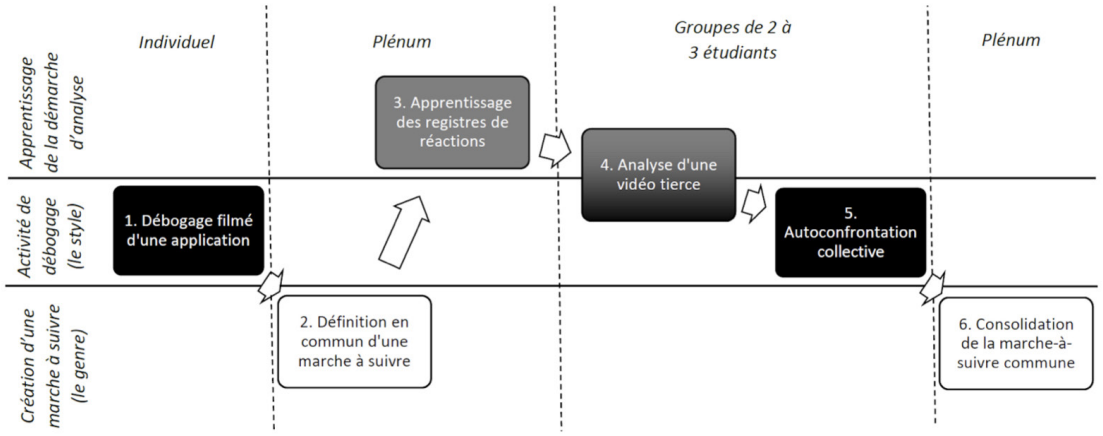
La recherche porte sur l'analyse du dispositif « Atelier du débogueur » mis en place. Ce chapitre permettra de présenter le dispositif, l'approche méthodologique ainsi que les résultats obtenus.

4.1 Dispositif pédagogique mis en place

Pour pallier tant les problèmes liés aux difficultés d'apprentissage du concept de débogage que les problèmes liés à la sensation d'incompétence ressentie par les étudiants dans ce domaine (soulignées dans la partie 2.2), un dispositif utilisant la trace vidéo et l'autoconfrontation croisée (Clot et al., 2000) a été mis en place depuis 2023 à l'Ecole des Métiers de Fribourg (Ramalho, 2023). Le choix de ce dispositif a été effectué en partant de l'hypothèse que la trace vidéo permette de relever, visionner voire revisionner plusieurs fois les étapes clés (et souvent cachées ou réalisées inconsciemment) se trouvant derrière l'activité et qu'une description orale de l'étudiant n'aurait pas forcément relevées. La vidéo permet ainsi au professionnel de voir, se voir et donner à voir son activité (Thievenaz, 2021). Notre dispositif appelé « l'atelier du débogueur » consiste à faire réaliser à toute une classe d'apprentis informaticiens le dépannage d'une même application tout en se filmant via webcam et en enregistrant le flux vidéo des actions menées sur l'écran. Des autoconfrontations croisées entre pairs sont ensuite organisées, afin de faire ressortir les étapes ayant mené à la détection puis à la correction du problème. Ce dispositif qui s'appuie sur d'autres expériences réalisées dans le cadre de la formation des enseignants professionnels (Neuhaus, 2024a) a pour visée de faire émerger la face cachée de l'activité de débogage.

L'atelier du débogueur se déroule en plusieurs étapes, individuelles, en plénum et en sous-groupes qui peuvent être séparées en 3 familles différentes : a) les étapes de film et d'analyse de l'activité de débogage, b) les étapes de création d'une marche à suivre commune, c) les étapes d'apprentissage de la démarche d'analyse. Ces étapes sont présentées dans la figure ci-dessous :

Figure 2 : Etapes de l'activité mise en place



Nous allons ci-dessous présenter en détail le contenu de ces différentes étapes.

1. pour débuter, les apprentis sont confrontés au débogage d'une application identique, contenant une erreur courante. Cette application est développée par l'enseignant dans le langage de programmation « Java », déjà connu des élèves. Volontairement l'erreur ajoutée par l'enseignant est cachée et ne laisse pas de « trace » (aucune information ou message d'erreur présent dans la console de développement) et nécessite ainsi une investigation plus grande pour la détecter et la corriger. Durant cette étape, il est demandé aux étudiants de se filmer individuellement via webcam et d'enregistrer simultanément le flux affiché à l'écran. Un temps de quinze minutes est mis à disposition des étudiants pour réaliser l'exercice ;
2. durant une séance collective, les étudiants définissent la liste des tâches essentielles à réaliser lors d'un processus de débogage. Le but de cette étape est de partir des conceptions et représentations des étudiants au lieu de s'appuyer immédiatement sur une liste de bonnes pratiques tirées d'un référentiel ;

3. l'apprentissage des registres de réactions à utiliser lors de l'analyse d'une vidéo se fait également de manière collective. Le risque en utilisant la trace vidéo pour réaliser l'analyse de l'activité d'un pair réside dans l'aspect évaluatif qui pourrait provoquer un sentiment de malaise, ce qui nécessite d'initialement se former et s'entraîner à l'exercice (Paul, 2022, p. 63). Muller et Lussi Borer (2018) relèvent trois registres de réactions différents lorsque des professionnels analysent une vidéo : descriptif, interprétatif et évaluatif. Le registre évaluatif est celui qui est le plus souvent spontanément mobilisé dans un premier temps ce qui risque de mener à un jugement rapide, peu constructif. Ces différentes étapes sont ainsi présentées aux étudiants par l'enseignant, en insistant sur l'importance de débuter l'analyse par une description de l'activité ;
4. avant d'entrer dans le traitement des vidéos des étudiants, une analyse est effectuée en commun sur une vidéo tournée par l'enseignant lui-même le montrant en situation de débogage d'une autre application. Les buts de cette étape sont multiples. Il s'agit tout d'abord pour les étudiants de s'approprier collectivement les registres de réactions présentées lors du point précédent mais également d'entrer pour la première fois dans l'analyse d'une activité de débogage, raison pour laquelle elle se trouve au milieu des deux zones dédiées dans la Figure 2 ;
5. enfin, des groupes de 2 ou 3 étudiants sont formés. Dans la mesure du possible ils sont volontairement constitués d'étudiants de niveaux différents (des étudiants ayant réussi à corriger l'application avec ceux qui n'y sont pas parvenus). Dans chaque groupe, la consigne est donnée de visualiser tour à tour la vidéo de chaque étudiant avant que ses collègues ne réagissent sous forme d'autoconfrontation collective (Clot, 2018) ; en fin d'activité, un moment collectif permet à la classe de coconstruire une marche à suivre optimisée pour le débogage, intégrant les apprentissages et les observations de chacun.

4.2 La question de recherche

La question de recherche présentée dans cet article est la suivante : Est-ce qu'un dispositif axé sur l'utilisation de la vidéo et de l'autoconfrontation collective tel qu'il est présenté dans l'atelier du débogueur aide les apprentis à faire ressortir les étapes principales menant au débogage d'une application et à se sentir au final davantage aptes à prendre en main et déboguer le programme d'un autre développeur ?

Notre hypothèse est que, confrontés à l'activité de leurs pairs, dans la même situation de débogage, les étudiants parviennent à identifier, non seulement la solution du débogage mais également les étapes clés qui ont mené à sa découverte.

De plus, ayant constaté sur le terrain, principalement lors du départ en stage de formation, des difficultés pour plusieurs étudiants face à la reprise de code d'un autre développeur, nous avançons l'hypothèse que, suite à cet exercice, les étudiants se sentiront davantage en confiance pour prendre en main le programme d'un autre professionnel dans le futur.

4.3 Le corpus de recherche

Le corpus de recherche est composé de 2 groupes-classe différents. Un de 2e année avec 17 étudiants et un autre de 3e année avec 19 étudiants. Les deux groupes sont constituées uniquement de garçons entre 16 et 18 ans et ont le même enseignant (corédacteur de cet article) pour les cours de programmation. Ce choix a été arrêté en fonction des classes auxquelles l'enseignant-chercheur enseignait durant la période de récolte de données.

Les étudiants étant en formation professionnelle d'informaticien, ils ont, avant cette expérimentation, déjà vécu entre 2 et 4 modules de formation sur le thème de la programmation. Chaque module dure 5 semaines, à raison de 1,5 jour bloc par semaine. Ils maitrisent ainsi les bases de la programmation (les tests, les boucles, les méthodes et fonctions, les bases de la programmation orientée objet) et ont déjà indirectement été amenés à mener une activité de débogage mais sur leurs propres programmes lors de la réalisation des exercices demandés dans les modules précédents.

4.4 Méthodologie

La méthodologie de recherche se dessine en plusieurs étapes qualitatives et quantitatives :

a. les différentes vidéos tournées par les étudiants avec leur webcam et la résolution de l'exercice de débogage ont été réceptionnées par l'enseignant et ont permis une analyse globale.
b. le moment d'autoconfrontation collective en sous-groupe a été filmé pour en garder une trace. Pour des raisons logistiques principalement (les autoconfrontations collectives ayant eu lieu en

même temps dans la classe), cette prise vidéo n'a cependant pas été réalisée sur tous les groupes mais uniquement sur certains.

c. deux sondages en ligne ont été réalisés pour sonder *a priori* et *a posteriori* le sentiment de capabilité dont font preuve les étudiants face aux activités de débogage et leur aisance avec l'outil de débogage.

Nous allons préciser ci-dessous la manière dont, pour ces trois étapes, les données ont été acquises, analysées et la manière dont les résultats ont été traités.

4.4.1 Le film des activités de débogage

Les vidéos ont été réalisées en captant aussi bien le flux affiché sur l'écran (montrant les différentes actions réalisées) que le flux capturé par la webcam. Si le but initial avec ces vidéos était d'obtenir un matériel sur lequel s'appuyer pour les autoconfrontations, après l'expérimentation, une visualisation de toutes les vidéos a été réalisée par l'équipe de recherche. Sans grille d'analyse, le but était de relever les étapes principales présentes lors du débogage.

Cette analyse a été réalisée tant en visualisant les activités faites par l'étudiant qu'en analysant les attitudes non-verbales et les mimiques présentes dans les vidéos par webcam. Nous nous appuyons sur les travaux de Thievenaz pour qui « les gestes, postures, regards et intonations peuvent, sous certaines conditions, être considérés comme porteurs de signification et d'intentionnalité ». (2019, p. 88). Ainsi l'analyse de la position du corps, des mains, du regard et des mimiques de la bouche sont autant d'indicateurs permettant d'aider la compréhension du cheminement cognitif de l'étudiant durant l'activité de debogage.

Par ailleurs, pour des raisons de droits à l'image, il est à relever que les vidéos ont été tournées par les étudiants eux-mêmes et restent en leur unique possession. Ainsi, après un visionnage global par l'équipe de recherche, elles ont été effacées et aucune sauvegarde n'en a été faite.

4.4.2 Les autoconfrontations collectives

Les moments d'autoconfrontation collectives ont été menés de manière autonome dans les groupes. La méthodologie d'autoconfrontation a été choisie pour favoriser la richesse des échanges et des apprentissages. En effet, dans notre corpus, 50% des apprentis n'ont pas réussi à découvrir le bug dans le temps imparti. Les groupes ont ainsi été formés en cherchant à mélanger des étudiants ayant trouvé la solution avec les autres. Pour en garder une trace permettant principalement d'illustrer le processus, l'enseignant-chercheur a filmé quelques interactions au hasard de son passage dans les rangs et a noté quelques remarques qui apparaissent comme représentatives du déroulé de l'expérience.

4.4.3 Sondages a priori et a posteriori

Des sondages ont été réalisés au début et à la fin de l'activité pour évaluer la perception des apprentis concernant leur aptitude au débogage. L'objectif était de mesurer leur sentiment de capabilité avant et après l'activité. Les questions posées étaient les suivantes :

a. Comment jugez-vous votre efficacité sur l'utilisation du débogueur (touche raccourcie, parcours de code, …) ?
b. Est-ce que vous vous sentez capables de déboguer un programme informatique d'un autre développeur ?

Les réponses à donner à ces deux questions étaient basées sur une échelle de Likert allant de 1 à 5 (1 = pas bon et 5 = très bon).

La première question est volontairement axée sur l'utilisation de l'outil de débogage. Celle-ci permettra de déterminer si la prise en main de l'outil représente un frein à la réalisation du débogage. La deuxième question s'articule sur la sensation générale de capabilité à déboguer le programme d'un autre développeur.

Les sondages tant *a priori* qu'*a posteriori* sont réalisés depuis une plateforme numérique sur laquelle les étudiants sont connectés. Les données permettent ainsi d'identifier les évolutions pour chaque étudiant entre le début et la fin de l'activité. Elles sont néanmoins anonymisées pour la présentation des résultats dans cet article.

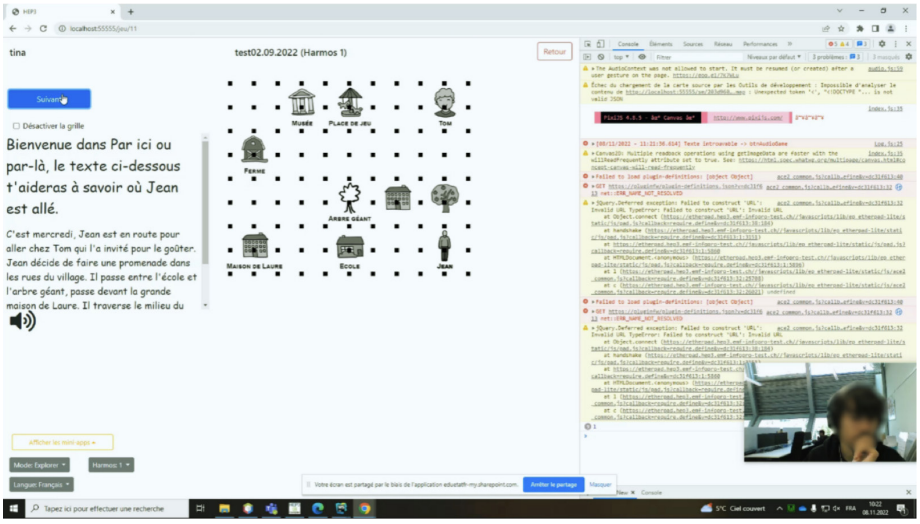
4.5 Résultats obtenus

Nous présenterons notre rendu en trois étapes : a) nous reviendrons sur les films réalisés par les apprentis sur l'activité de débogage ; b) nous présenterons les remarques relevées lors de de l'étape d'autoconfrontation collective ; c) enfin, nous présenterons les analyses des sondages *a priori* et *a posteriori* réalisés.

4.5.1 Débogage filmé par l'apprenti

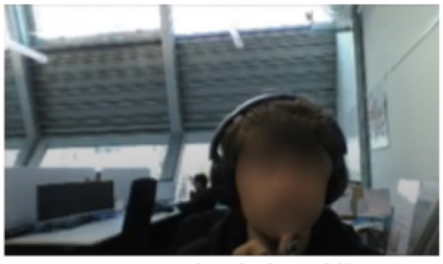
Cette vidéo a été utilisée prioritairement afin de permettre l'autoconfrontation entre les pratiques des étudiants. Cependant, cette expérimentation permet de relever certains éléments qui nous semblent importants par la superposition entre l'écran de l'étudiant et sa webcam (Figure 3).

Figure 3 : Exemple de film réalisé par les étudiants

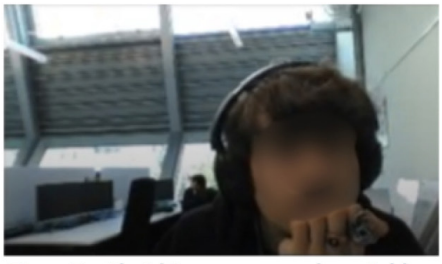


Le fait de demander aux étudiants de filmer simultanément l'écran et le visage permet une analyse complémentaire entre les actions entreprises par l'étudiant et son attitude non verbale (Figure 4).

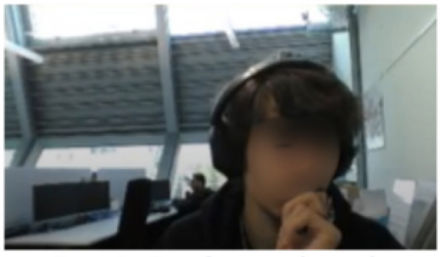
Figure 4 : Images représentant le déroulement du débogage pour un étudiant



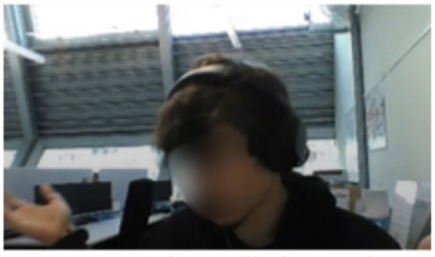
Etape 1 « Je recherche le problème »



Etape 2 « C'est bizarre ce nom de variable »



Etape 3 « Peut-être que c'est ça ! »



Etape 4 « Et bien voilà, j'ai trouvé »

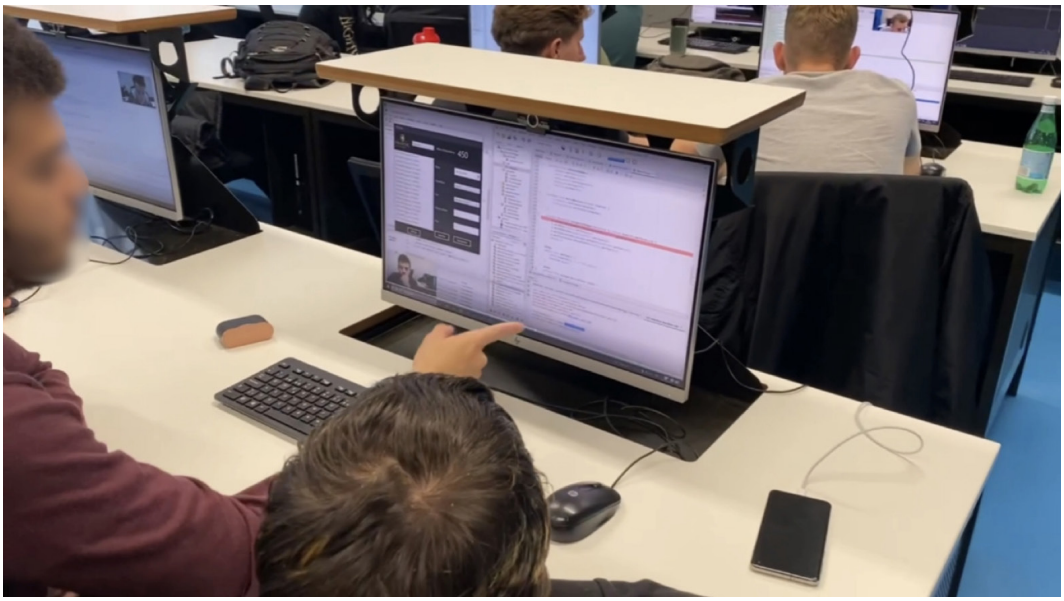
Nous pouvons relever des indicateurs intéressants visibles dans les quatre images ci-dessus, tant par la position des mains que la position corporelle. Ainsi la première image montre l'étudiant, l'index sur la bouche que nous interprétons comme un moment de réflexion. Le passage vers la 4e image montre les mains se plaçant sous le menton, devant la bouche pour terminer par un mouvement illustrant la satisfaction de la réussite.

Nous émettons l'hypothèse que ces indices non-verbaux permettent non seulement au développeur qui revisionne sa vidéo de se replonger au plus proche de son état et ses doutes durant l'activité mais également à son collègue visualisant la vidéo d'en prendre conscience plus facilement qu'avec les seules actions à l'écran.

4.5.2 Autoconfrontation collective

Durant cette étape, chaque étudiant montre sa vidéo à ses collègues de groupe avant que ceux-ci ne réagissent (Figure 5).

Figure 5 Autoconfrontation entre les étudiants



Cette étape n'a pas été systématiquement filmée et analysée pour tous les binômes. Cependant, lors de nos passages dans les rangs, nous avons constaté que les apprentis ont rencontré des difficultés à suivre rigoureusement le processus structuré de description, d'interprétation et d'évaluation. La partie évaluation était très rapidement abordée dans certains groupes. Néanmoins, sans en avoir fait un référencement systématique, nous avons relevé que les remarques échangées étaient bienveillantes et les conseils prodigués ont enrichi le groupe. Nous avons également relevé que de nombreux échanges portaient sur le processus abordé lors du débogage et non plus la simple correction de l'erreur. Certaines phrases représentatives ont été notées lors de nos passages telles que « Tu peux tester le programme en l'exécutant une première fois, avant de lancer le débogueur pour voir le problème », « Moi, j'ai plutôt lancé le débogueur, plutôt que de rechercher dans le code ce qui n'allait pas », « J'ai utilisé les touches de raccourci F7/F8 et relancé plusieurs fois l'application pour voir quel était le problème ».

Ces témoignages illustrent un partage d'expériences précieux qui contribue à l'amélioration des compétences de débogage de chacun. Nous pouvons relever que ces remarques abordent aussi bien le processus permettant de mener à bien cette activité que l'utilisation de l'outil de débogage. En permettant aux apprentis de confronter leurs méthodes et résultats, cette activité semble ainsi favoriser non seulement l'apprentissage collaboratif, mais aussi le développement de compétences critiques et analytiques essentielles au débogage.

4.5.3 Sondage sur la sensation de capabilité

Les résultats des sondages avant et après l'activité ont relevé une évolution importante des réponses des apprentis quant à leur capacité d'analyse et de prise en main d'un code tout comme dans la confiance affichée par ces derniers sur leur capacité à intervenir sur le terrain pour corriger le code d'un autre développeur.

Ainsi à la question a) « Comment jugez-vous votre efficacité sur l'utilisation du débogueur (touche raccourcie, parcours de code, ...) ? », les statistiques des réponses montrent l'évolution suivante :

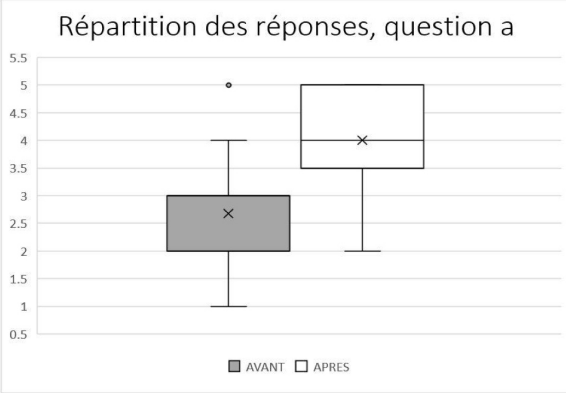
Tableau 1 Statistique des réponses à la question a (1 : pas du tout efficace ; 5 : très efficace)

		N	Moyenne	Médiane	[Min,Max]	Ecart type	Erreur standard
Score	Avant	28	2.6875	3	[1, 5]	0.9833	0.1858
	Après	29	4.0000	4	[2, 5]	0.8018	0.1489

La sensation d'efficacité avec l'outil de débogage est beaucoup plus élevée après l'activité et ceci en présentant moins de différences (écart type) entre les étudiants. Plus aucun étudiant ne se retrouve par ailleurs dans une sensation d'inefficacité totale.

Le graphique en Figure 6 illustre une sensation de capabilité qui se trouve globalement plus haute après l'activité mais avec un spectre général plus grand allant de 3,5 à 5.

Figure 6 : Schéma représentant les réponses à la question a



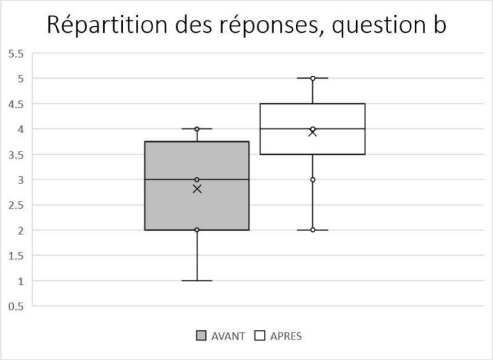
De la même manière, pour la question b) « Est-ce que vous vous sentez capables de déboguer un programme informatique d'un autre développeur ? » les résultats sont les suivants :

Tableau 2 : Statistique des réponses à la question b
(1 : pas du tout capable ; 5 : complément capable)

		N	Moyenne	Médiane	[Min,Max]	Ecart type	Erreur standard
Score	Avant	28	2.8214	3	[1, 4]	0.8929	0.1630
	Après	29	3.9310	4	[2, 5]	0.8422	0.1564

A nouveau, une évolution nette se dessine à ce sujet entre avant et après l'activité. Après l'activité, plus aucun étudiant ne se sent totalement incapable dans cette activité et la moyenne obtenue se rapprochant du score de 4 avec un écart type plus faible. Le graphique en Figure 7 montre de plus une normalisation des retours après l'activité avec des conceptions plus proches entre les étudiants.

Figure 7 Schéma représentant les réponses à la question b



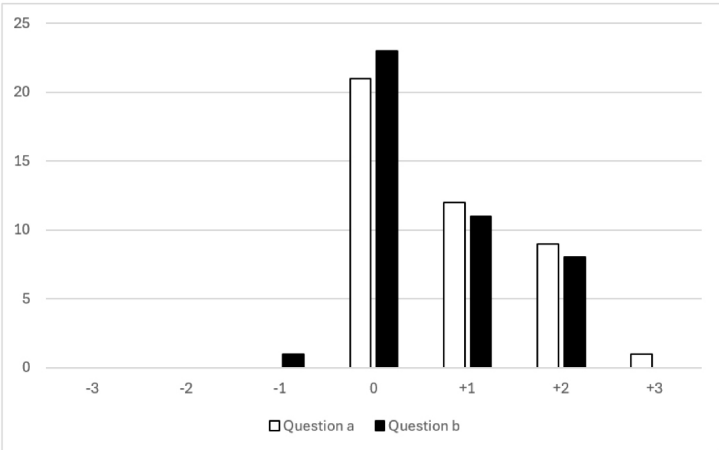
Enfin, afin de détecter si l'activité a été profitable individuellement pour tous les participants, nous avons également vérifié l'évolution individuelle de chaque étudiant. Le tableau ci-dessous présente le nombre d'étudiants ayant amélioré (de +1, +2, +3 ou +4) ou non sa réponse entre les deux questionnaires.

Tableau 3 : Evolution des réponses individuelles avant ou après l'activité

	-3	-2	-1	0	+1	+2	+3	+4
Question a	0	0	0	21	12	9	1	0
Question b	0	0	1	23	11	8	0	0

Le graphique suivant reprenant les mêmes données illustre une stagnation ou évolution positive pour presque tous les étudiants entre avant et après l'activité.

Figure 8 : Graphique de l'évolution des réponses individuelles avant ou après l'activité



Nous pouvons relever qu'à la question a) « Comment jugez-vous votre efficacité sur l'utilisation du débogueur (touche raccourcie, parcours de code, ...) ? », 48% des personnes n'ont pas vu de changement, 28% ont augmenté de 1 point et 21% ont augmenté de 2 points.

Pour la question b) « Est-ce que vous vous sentez capables de déboguer un programme informatique d'un autre développeur ? », 2% ont diminué de 1 point, 53% n'ont pas vu de changement, 16% ont augmenté de 1 point et 19% ont augmenté de 2 points.

4.6 Discussions

Les résultats présentés dans cet article sont encourageants quant à l'utilisation de tels dispositifs avec les apprentis informaticiens. Le dispositif mis en place permet non seulement aux étudiants d'aborder en cours des concepts difficiles, mais, comme le montrent les statistiques post activité, contribue également à donner aux étudiants une confiance leur permettant de se sentir à même de reprendre le code d'un autre développeur.

Cette première expérimentation ouvre néanmoins à des questions nouvelles sur le fonctionnement et les apports du dispositif, étape après étape. Ces questions ne peuvent néanmoins pas être répondues par la méthodologie adoptée dans cette recherche qui présente comme limite principale le fait que le dispositif d'analyse de l'étape d'autoconfrontation croisée est non systématique. Ainsi nous avons relevé certaines phrases d'étudiants au gré de nos passages dans les rangs mais pour mieux saisir l'apport de cette étape, il serait important que celle-ci soit également filmée et analysée de manière systématique. Cependant ces quelques phrases ont permis de mettre en évidence les étapes difficiles à enseigner en école et relevées dans notre cadre théorique en nous basant sur les prescriptions d'IBM (2024) et sur les connaissances à mobiliser relevées par McCauley et al. (2008). Comme évoqué par Fitzgerald et al. (2008), nous pouvons ainsi émettre l'hypothèse que le développement de cette compétence en analyse et débogage de code contribuera également à améliorer les compétences en développement. Cette tendance devra cependant être vérifiée dans de prochaines recherches.

Les résultats qui découlent de cette première expérimentation nous encouragent à poursuivre nos recherches de manière plus poussée. Nous pouvons évoquer en priorité les directions suivantes qui seront données à nos prochaines recherches : a) de telles expérimentations gagneraient à être non seulement reproduites à plusieurs reprises mais également en immersion réelle sur le terrain. Ainsi, pour les étudiants étant en stage en entreprise durant leur 4e année de formation, le fait de réaborder ces concepts à cette occasion pourrait être particulièrement fertile en termes formatifs, b) ensuite, si l'expérimentation a été réalisée avec des étudiants en formation en école à plein temps, une expérimentation en classe duale serait également à réaliser. En effet, celle-ci permettrait de potentiellement relever non pas des pratiques émergentes personnelles des candidats mais également de confronter les différentes approches ou habitudes des entreprises formatrices portées par les apprentis. Cette confrontation pourrait contribuer à apporter aux étudiants des approches différentes facilitant leur entrée dans le monde professionnel une fois la formation terminée, c) dans l'optique de la mise en place d'un portfolio à visée formative (Dévé et al., 2009) dans l'établissement, il pourrait être intéressant de demander aux étudiants de garder ces différentes vidéos réalisées au fil de leur apprentissage pour y revenir au fil de leur évolution, relevant ainsi de manière réflexive leur progrès (Gauthier, 2009). Ceci contribuerait à donner aux étudiants davantage de confiance envers leur capabilité à agir sur le terrain de manière adéquate.

5. Conclusions et perspectives

L'expérimentation réalisée sur dans le dispositif « l'atelier du débogueur » est riche de belles perspectives.

Comme relevé dans la discussion ci-dessus, nos résultats s'appuient sur un corpus relativement limité et une acquisition de données qui gagnerait à être davantage approfondie, ce qui sera fait dans de futures recherches. Cette première expérimentation laisse cependant entrevoir un impact positif de l'autoconfrontation croisée en utilisant la trace vidéo, tant sur la sensation d'efficacité des étudiants avec les outils de débogage que sur leur confiance face à cette tâche essentielle pour leur futur professionnel.

Par ailleurs, ces premiers résultats mettent en lumière certains indices qui encouragent à poursuivre les recherches dans cette direction. Nous en relevons deux principaux. D'une part l'autoconfrontation croisée sur les vidéos tournées par les apprentis informaticiens permet de faire remonter les parties invisibles de cette activité, contribuant au développement et à l'évolution de tous les étudiants. D'autre part, au sortir de cette activité, les étudiants se disent plus en confiance pour reprendre le code d'autres développeurs. Cette sensation de capabilité peut contribuer à rendre l'entrée plus facile sur le milieu professionnel.

Ainsi, par ce dispositif, les enseignants contribuent à donner confiance aux étudiants sur leur niveau de compétence, élément déterminant à la motivation scolaire (Viau, 2009). Si les référentiels de formation pour les informaticiens prévoient nombre de compétences essentielles à l'exercice du métier (chapitre 2.2), cette approche contribue à réduire l'écart entre les concepts vus en école professionnelle et les besoins du terrain. Par ailleurs, le fait d'aborder en école les compétences de débogage permettant selon certaines études d'améliorer les compétences en développement algorithmique des étudiants (Fitzgerald et al., 2008), nos résultats suscitent des espoirs de retombées indirectes en termes de formation des apprentis à plus large échelle.

Si cet article ne revient pas sur cette considération, il faut cependant résoudre en amont la question des droits nécessaires pour pouvoir tourner la vidéo. Dans cette perspective, une information claire doit être donnée aux étudiants sur les fins utilisées à cette vidéo. Il nous apparait également comme important que la liberté soit laissée à l'étudiant de ne pas se filmer s'il ne le souhaite pas. Il conviendra dès lors de leur donner une autre consigne. En ce sens, nous pouvons évoquer comme piste le fait de filmer uniquement le flux de l'écran mais sans l'utilisation de la webcam.

Forts de cette expérimentation, nous pouvons également envisager la mise en place de dispositifs similaires pour d'autres publics que celui de la formation professionnelle initiale, parmi lesquelles : a) des dispositifs basés sur l'analyse de traces vidéos pour d'autres tâches que le débogage, telles que l'utilisation d'outils bureautiques ; b) l'utilisation à des fins de formation continue pour des professionnels confirmés ; c) d'autres domaines ou métiers que celui de l'informatique gagneraient à être expérimentés par l'utilisation de la vidéo et de l'autoconfrontation croisée. Ceux de la formation des enseignants (Neuhaus, 2024a) et de la formation des chefs d'orchestre (Neuhaus, 2024b) ont déjà été expérimentés et portent des espoirs tant en formation de base qu'en formation continue.

La montée de l'intelligence artificielle donne une perspective supplémentaire. Nous pouvons déjà noter que les étudiants utilisent l'intelligence artificielle pour pouvoir comprendre et analyser leurs programmes au détriment d'autres outils ou forums tels que Stackoverflow[3] (Ramalho, 2024). De plus en plus d'étudiants utilisent l'IA notamment pour déboguer leur programme quand une trace est disponible. Cette alternative doit également être accompagnée par les enseignants pour améliorer les automatismes liés à cette aide de l'IA. Nous pourrions aussi imaginer avoir dans les prochains mois des outils d'intelligence qui permettraient d'analyser automatiquement les manipulations faites par les étudiants sur le code à déboguer. De même l'IA pourrait contribuer tant à la création de programmes qu'à l'ajout de bugs aléatoires dans ceux-ci. Il s'agit là de perspectives qui seront à prendre en compte en fonction des apports futurs de ces outils qui arrivent en nombre tant sur le terrain professionnel des informaticiens que dans le monde de la formation.

6. Bibliographie

Clot, Y. (2018). *La fonction psychologique du travail*. Presses Universitaires de France.

Clot, Y. et Faïta, D. (2000). Genres et styles en analyse du travail, concepts et méthodes. *Travailler*, *4*.

Clot, Y., Faïta, D., Fernandez, G. et Scheller, L. (2000). Entretiens en autoconfrontation croisée : une méthode en clinique de l'activité. *Perspectives interdisciplinaires sur le travail et la santé*, (2-1). https://doi.org/10.4000/pistes.3833

Dévé, V., Gagnayre, R. et d'Ivernois, J.-F. (2009). Le portfolio : définitions et perspectives pédagogiques à partir d'une analyse de textes canadiens et européens. *Education du Patient et Enjeux de Santé*, *27*(1). https://ipcem.org/img/articles/Port_Folio_pedagogie.pdf

Fitzgerald, S., Lewandowski, G., McCauley, R., Murphy, L., Simon, B., Thomas, L. et Zander, C. (2008). Debugging: finding, fixing and flailing, a multi-institutional study of novice debuggers. *Computer Science Education*, *18*(2), 93-116. https://doi.org/10.1080/08993400802114508

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

Gauthier, P.-D. (2009, 1 janvier). *La réflexivité dans la démarche Portfolio*. https://doi.org/10.13140/2.1.4585.9844

Gautier, M. et Wrobel-Dautcourt, B. (2015). Patrie 3, chapitre 3 / artEoz, visualisation dynamique de programmes. Dans *Informatique en éducation: perspectives curriculaires et didactiques*. Presses universitaires Blaise Pascal.

Gremion, C. (2022). Usages et mésusages des pédagogies de l'alternance. *Phronesis*, *11*(1-2), 91. https://doi.org/10.7202/1087560ar

Gremion, C. et Maubant, P. (à paraître). Vers une mise en œuvre pédagogique de la didactique professionnelle par alternance et situation. Dans G. Ghisla, E. Boldrini, C. Gremion, F. Merlini et E. Wüthrich (dir.), *Didactique et situations, Approches et expériences pour la formation professionnelle*, (p. 63-73), CREME.

Hartmann, W., Näf, M. et Reichert, R. (2011). *Enseigner l'informatique*. Springer.

[3] *https://stackoverflow.com/ Stackoverflow est un forum web dédié aux développeurs sur lequel de nombreux professionnels et développeurs amateurs échangent sur des problèmes rencontrés lors de développements.*

IBM. (2024, 16 septembre). *Qu'est-ce que le débogage ? | IBM*. https://www.ibm.com/fr-fr/topics/debugging
Kernighan, B. W. et Plauger, P. J. (1995). *The elements of programming style* (2. ed., 16. print). McGraw-Hill.
Kriesi, I., Bonoli, L., Grønning, M., Hänni, M., Neumann, J. et Schweri, J. (2022). *Rapport de tendance - La formation professionnelle au niveau international et en Suisse - Tensions, Défis, Développement, Potentiels*. OBS HEFP Rapport de tendance 5. Zollikofen : Haute école fédérale en formation professionnelle HEFP. https://www.sfuvet.swiss/sites/default/files/2022-11/Trendbericht_2022_FR.pdf
McCauley, R., Fitzgerald, S., Lewandowski, G., Murphy, L., Simon, B., Thomas, L. et Zander, C. (2008). Debugging: a review of the literature from an educational perspective. *Computer Science Education*, *18*(2), 67-92. https://doi.org/10.1080/08993400802114581
Michaeli, T. et Romeike, R. (2019). Improving Debugging Skills in the Classroom – The Effects of Teaching a Systematic Debugging Process.
Muller, A. et Lussi Borer, V. (2018). Chap. 4 - Analyse de l'activité, environnements de vidéoformation et développement professionnel des enseignants. Différence intra et inter-activité: le possible, le réel et le virtuel. Dans C. Gaudin, S. Flandin, S. Moussay et S. Chaliès (dir.), *Vidéo-formation et développement de l'activité professionnelle enseignante* (p. 95 à 116). L'Harmattan.
Neuhaus, O. (2024a). *L'autoconfrontation collective entre pairs pour accompagner l'entrée dans le métier des enseignants professionnels* [thèse de doctorat]. Université de Genève. https://archive-ouverte.unige.ch/unige:178502
Neuhaus, O. (2024b, janvier). *L'évaluation croisée entre pairs pour former les chefs d'orchestres amateurs en Suisse* [présentation internationale]. Présentation internationale communication présentée au Symposium « Évaluer l'éducation artistique et culturelle, entre normalisation et différenciation » lors du 35e colloque international ADMEE, Braga, Portugal.
Nijimbere, C., Boulc'h, L. et Baron, G.-L. (2015). Patrie 3, chapitre 4 / Apprendre l'informatique par la programmation des robots. Cas de Legos Mindstorms NXT. Dans *Informatique en éducation: perspectives curriculaires et didactiques*. Presses universitaires Blaise Pascal.
Olry, P., Métral, J.-F. et Chrétien, F. (2021). L'usage des vidéos en didactique professionnelle : statuts technique, pragmatique, épistémique et acceptabilité sociale. *Savoirs*, *N° 55*(1), 59-75.
Paul, M. (2022). *Accompagner la problématisation des situations professionnelles: analyser ses pratiques autrement*. De Boeck Supérieur.
Plumettaz, M. F. N. (2024). *Institutionnalisation des savoirs informatiques lors du débriefing. Le cas de l'apprentissage avec le jeu Programming Game* [thèse de doctorat]. Université de Genève. https://archive-ouverte.unige.ch/unige:175760
Ramalho, M. (2023). *Didactique disciplinaire et de domaine professionnel - Atelier du débogueur* [Rapport de validation d'un module de formation à la HEFP]. Haute école fédérale en formation professionnelle HEFP.
Ramalho, M. (2024, 3 mai). L'utilisation de ChatGPT en formation, au-delà de l'instantanéité. *ICTJournal*. https://www.ictjournal.ch/articles/2024-05-03/lutilisation-de-chatgpt-en-formation-au-dela-de-linstantaneite
Rogalski, J. (2015). Partie 3, chapitre 5 / Psychologie de la programmation, didactique de l'informatique. Déjà une histoire… Dans *Informatique en éducation: perspectives curriculaires et didactiques*. Presses universitaires Blaise Pascal.
Romero, M. et DeBlois, L. (2022). Analyse du processus de construction de connaissances dans des activités de programmation à l'école. *Canadian Journal of Science, Mathematics and Technology Education*, *22*(2), 405-421. https://doi.org/10.1007/s42330-022-00210-9
SEFRI, S. d'État à la formation à la recherche et à l'innovation. (2022). *En 2022, près de 60% des jeunes optent pour une formation professionnelle initiale*. https://www.admin.ch/gov/fr/accueil/documentation/communiques.msg-id-89058.html
Siegmund, B., Perscheid, M., Taeumel, M. et Hirschfeld, R. (2014, 4 novembre). *Studying the Advancement in Debugging Practice of Professional Software Developers*. https://doi.org/10.13140/2.1.2073.7288
Thievenaz, J. (2019). La théorie de l'enquête de John Dewey : réexplorations pour la recherche en sciences de l'éducation et de la formation. *Recherche & formation*, *92*(3), 19-38.
Thievenaz, J. (2021). Éditorial Le recours à l'image en formation des adultes : voir, se voir, donner à voir. *Savoirs*, *N° 55*(1), 7-13.
Thomas, D. et Hunt, A. (2024). *Le programmeur pragmatique* (20 ans, édition anniversaire). Pearson France.
Viau, R. (2009). *La motivation en contexte scolaire* (2e éd). De Boeck université.
Vincent, J.-M. et Cohen, G. (2017). *L'informatique débranchée: le numérique sans ordinateur : activités de découverte du primaire au lycée*. Éditions Pole.
Vitali-Rosati, M. (2024). *Éloge du bug: être libre à l'époque du numérique*. Zones.
Yang, S., Baird, M., O'Rourke, E., Brennan, K. et Schneider, B. (2024). Decoding Debugging Instruction: A Systematic Literature Review of Debugging Interventions. *ACM Transactions on Computing Education*, *24*(4), 1-44. https://doi.org/10.1145/3690652