CJLT RCAT
Canadian Journal of Learning and Technology
La Revue canadienne de l'apprentissage et de la technologie

Volume 48 (1) Winter / Hiver 2022

Proposition d'une typologie des pratiques effectives de programmation visuelle

A Typology Proposition of Effective Visual Programming Practices

***Simon Parent**, Université de Montréal*

Résumé

Cet article présente les résultats d'une étude de cas multiples menée auprès de 18 élèves du primaire au Québec, Canada. L'objectif de celle-ci était de proposer une typologie des pratiques effectives de programmation visuelle d'élèves du primaire. En plus d'offrir un portrait détaillé des pratiques mobilisées par les élèves dans le cadre de cette recherche, nous présentons une typologie des tâches de programmation visuelle pour des élèves du primaire en nous appuyant d'une part sur la littérature, et d'autre part sur les données empiriques de l'utilisation d'un scénario pédagogique qui permet aux élèves de mobiliser leurs habiletés en programmant un robot humanoïde appelé NAO. Cette proposition de typologie compréhensive et adaptée offre un potentiel pédagogique non négligeable, que ce soit quant à la conception de scénarios pédagogiques mobilisant la programmation visuelle à l'enseignement primaire, ou au développement de manuels ou guides pédagogiques destinés aux élèves ou aux enseignants du primaire.

Mots-clés : programmation ; typologie ; robotique ; primaire ; pratiques effectives

Abstract

This article presents the results of a multiple-case study conducted with 18 primary school students in Quebec, Canada. The objective of this study was to propose a typology of effective visual programming practices of primary school students. In addition to offering a detailed portrait of the practices mobilized by the students in this research, we present a typology of visual programming tasks for primary school students based on the literature and on empirical data from the use of a pedagogical scenario which allows students to mobilize their skills by programming a humanoid robot called NAO. This proposal for a comprehensive and adapted typology offers a significant pedagogical potential, whether for the design of pedagogical scenarios mobilizing visual programming in primary education, or for the development of textbooks or pedagogical guides for primary school students or teachers.

Keywords: programming; typology; robotics; primary school; effective practices

Contexte

Les compétences du 21e siècle sont considérées comme étant l'une des façons d'approcher l'omniprésence technologique (Forum économique mondial, 2015). Parmi ces compétences, on retrouve la collaboration, la résolution de problèmes, la créativité, la pensée critique et plusieurs autres (Chalkiadaki, 2018). En filigrane de ces nombreuses compétences, on retrouve la programmation (Romero, 2017). L'apprentissage de la programmation et la connaissance du fonctionnement des appareils technologiques offrent aux apprenants un bagage de connaissances pertinentes pour le marché du travail, qui a vu se déployer ce que plusieurs appellent déjà la quatrième Révolution industrielle, où la technologie (intelligence artificielle, robotique, etc.) est omniprésente (Lee et al., 2018). Le Forum économique mondial, dans un rapport prospectif, affirme que les individus qui auront du succès dans l'économie du futur seront ceux qui peuvent complémenter le travail des algorithmes, autrement dit, de travailler avec « les machines » (2018, p. 3). Cela étant dit, notre intérêt pour l'utilisation de la programmation à l'école va bien au-delà de l'impératif économique ou des préoccupations relatives au marché du travail.

Au Québec, trois documents officiels récents démontrent la volonté du gouvernement d'intégrer la programmation informatique dans le cursus scolaire. En effet, dès 2018, le Plan d'action numérique en éducation et en enseignement supérieur (Ministère de l'Éducation et de l'Enseignement supérieur, 2018) présentait le souhait d'« accroître l'usage pédagogique de la programmation informatique » (p. 27). En 2019, le Cadre de référence de la compétence numérique (Ministère de l'Éducation et de l'Enseignement supérieur, 2019), dernière politique en matière de numérique pour notre système d'éducation, y faisait aussi référence : « [D]évelopper sa pensée informatique, notamment par le développement de sa compréhension et de ses habiletés à l'égard de la programmation informatique » (p. 14). Enfin, plus récemment a été publié le guide *L'usage pédagogique de la programmation informatique*, dans lequel on souligne le grand potentiel de la programmation informatique, notamment pour « structurer sa pensée », c'est-à-dire pour développer son « raisonnement logique et l'esprit critique » (Ministère de l'Éducation, 2020, p. 7).

Plusieurs auteurs se sont intéressés à l'utilisation de différents dispositifs avec des élèves du primaire. Par exemple, certains ont documenté les représentations d'élèves de maternelle d'opérations élémentaires de programmation pour faire déplacer le robot BeeBot (Komis & Misirli, 2011), et d'autres ont étudié empiriquement le lien entre la pensée informatique d'élèves et la programmation (Noh & Lee, 2020). Plus récemment, des auteurs ont observé les habiletés associées à la pensée informatique lors d'activités de programmation en dyades au primaire (Wei et al., 2021).

Cet article présente l'un des objectifs spécifiques d'une recherche dont l'objectif général était de décrire et comprendre les effets de la programmation informatique sur la mobilisation de compétences de résolution de problèmes et de collaboration d'élèves du primaire. Nous avons choisi d'observer les pratiques effectives de programmation en tant que contexte dans lequel étaient mobilisées ces

compétences. Outre la taxonomie des types de tâches de programmation proposée par Bower (2008), qui offre une perspective théorique et macroscopique peu adéquate pour l'enseignement primaire, il a été difficile de trouver des études proposant des classifications ou des catégorisations des pratiques associées à la programmation au primaire. L'objectif au cœur de cet article est donc la proposition, en s'appuyant sur la littérature, d'une typologie des pratiques effectives de programmation visuelle d'élèves du primaire lors d'activités où la collaboration était centrale. Bien que les analyses relatives à la résolution de problèmes et à la collaboration ne soient pas abordées dans cet article, les éléments conceptuels et théoriques associés à ces compétences permettent une compréhension du contexte d'observation des pratiques effectives de programmation.

Cadre de référence

L'approche socioconstructiviste offre des assises théoriques fort utiles à notre étude des pratiques effectives de programmation d'élèves du primaire. Elle met en lumière les dynamiques inhérentes à la collaboration et est à l'origine de certains de nos choix méthodologiques. Par exemple, Wood et al. (1976) avancent que, malgré les capacités naturelles de résolution de problèmes d'un enfant, il convient de tenir compte de l'apport d'autres pairs plus compétents pour l'assister dans le processus. D'ailleurs, cette disparité des compétences individuelles amène plusieurs points de vue différents pour la résolution d'un même problème, se traduisant à terme par un apprentissage par la voie de l'autre (ou des autres). Notons ici que l'intérêt principal de cette confrontation des idées est l'effet sur l'apprentissage, puisque selon Vygotsky, « l'enfant peut toujours faire plus et résoudre des problèmes plus difficiles que lorsqu'il agit tout seul » (1997, p.182).

La programmation, fortement associée au processus de résolution de problèmes, est définie comme étant l'action d'écrire, à l'aide d'un langage informatique, une série d'actions qui sont interprétées puis exécutées par un ordinateur (Blackwell, 2002). Le programme permet donc de médiatiser les interactions entre l'humain et l'ordinateur. Considérant la nature de l'agent de traitement de l'information (l'ordinateur), il est essentiel que les informations transmises soient dépourvues de toute équivoque ou ambiguïté (Turski, 1978). La programmation est une activité des plus propices à la mobilisation, voire au développement, de nombreuses compétences et processus cognitifs comme la résolution de problèmes (Lai & Yang, 2011) et la collaboration (Nugent et al., 2009). Avec l'avènement d'une multitude d'applications et de sites internet voués à l'apprentissage ludique de la programmation (p. ex. : Scratch, Code.org, Swift Playgrounds), nous pouvons sans doute parler d'une démocratisation de cette activité, à travers les âges. La programmation visuelle est l'une des adaptations pédagogiques de la programmation : elle permet de représenter les lignes de codes par des boîtes unies à l'aide de liens (Green & Petre, 1996).

Les pratiques de programmation dans la littérature

Les travaux de Ruf et al. (2015) offrent des pistes intéressantes propices à une transposition dans le contexte de l'enseignement primaire. En effet, ils ont recensé puis analysé des manuels ou cours d'apprentissage de la programmation pour l'enseignement secondaire et universitaire. Les tâches

identifiées étaient destinées à des programmeurs novices, ce qui facilite l'adaptation pour des élèves plus jeunes. Parmi les 1 098 « tâches » recensées, les auteurs ont été en mesure d'établir 11 types différents. Il s'avère que par sa constitution, et le fait qu'elle résulte de l'analyse d'ouvrages pédagogiques destinés à des programmeurs novices, cette classification de Ruff et ses collègues est très pertinente à la réalisation de nos travaux avec des élèves au primaire.

Méthodologie

Dans cette étude de cas multiples (Karsenti & Demers, 2018; Stake, 1995), nous avons observé les pratiques effectives de programmation visuelle d'élèves du primaire. Pour ce faire, nous avons constitué un échantillon par contraste-approfondissement (Pires, 1997), nous permettant d'aller plus en profondeur dans l'explication du phénomène observé par la juxtaposition des cas. Cet échantillon intentionnel (Fortin & Gagnon, 2016) est composé de six groupes-classes de trois écoles du Québec, chaque cas étant associé à un groupe-classe (Tableau 1). Ce dernier s'avère être un groupe naturel existant à l'extérieur du contexte de la recherche (LeCompte & Preissle, 1993). Aucun critère d'exclusion n'a été appliqué au sein des groupes. Nous avons néanmoins accordé une importance au caractère diversifié des différents groupes composant l'échantillon, tant au niveau du statut socioéconomique, du milieu – rural ou urbain – et du type d'école, qu'elle soit alternative ou traditionnelle, privée ou publique. Cela répond au critère de diversification externe des échantillons par cas multiples (Pires, 1997).

L'école A, située à Montréal, est une école alternative[1] accueillant des élèves du primaire et du secondaire. Les groupes n'étant pas formés en fonction de l'âge[2], les participants du groupe A1 (n = 13) sont de différents niveaux scolaires, bien que les élèves du primaire (3^{e} cycle) soient plus fortement représentés. L'école B, aussi située à Montréal, a un indice du seuil de faible revenu de même qu'un indice de milieu socioéconomique au 10^{e} rang décile, les niveaux maximums pour ces deux indices de dé favorisation du ministère de l'Éducation et de l'Enseignement supérieur du Québec (2020). Les participants de cette école, c'est-à-dire le groupe B4 (n = 3), doivent faire face à différents défis inhérents au contexte socioéconomique. Ils sont au troisième cycle du primaire, c'est-à-dire en 6^{e} année. Enfin, l'école C se situe dans la grande région de Québec, en milieu rural[3]. Les élèves du groupe C1 (n = 2) sont en 5^{e} année et sont dans un groupe-classe réduit puisque plusieurs enfants rencontrent des difficultés d'apprentissage variées.

[1] Selon le Réseau des écoles publiques alternatives du Québec, « l'école alternative est un milieu éducatif dynamique, prônant une approche participative, communautaire et humaniste dans laquelle chaque intervenant (équipe de direction, enseignants, parents) joue un rôle actif dans l'épanouissement de l'élève » (RÉPAQ, 2020).
[2] Les groupes ont été formés par le personnel enseignant.
[3] Selon les données de l'Institut national de santé publique du Québec (https://inspq.qc.ca/santescope/milieux-ruraux-urbains).

Tableau 1

Participants à l'étude

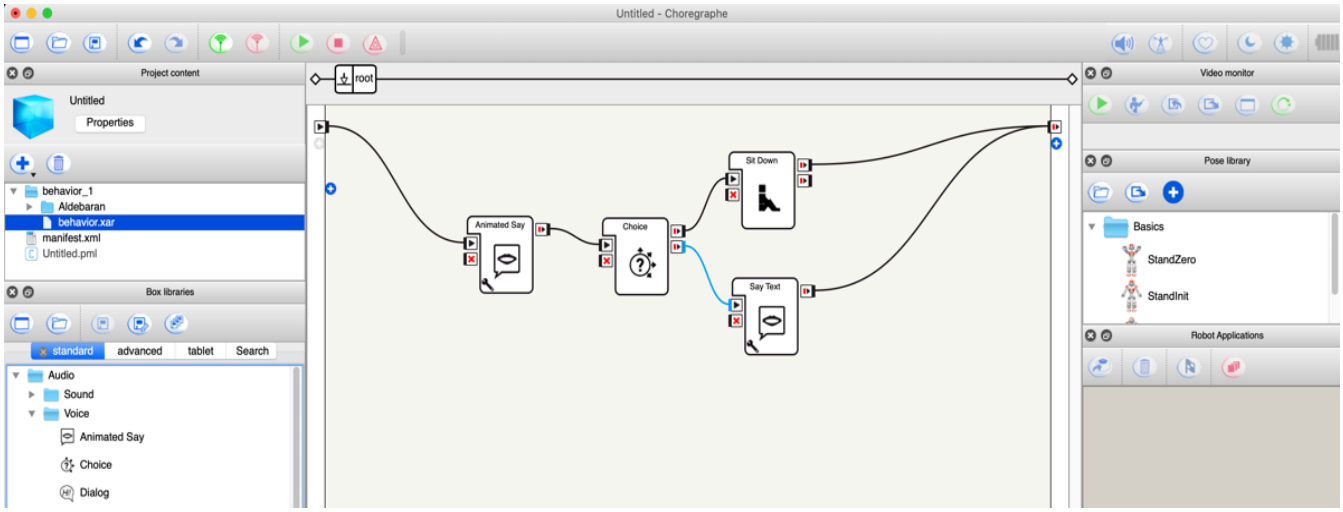
École	Groupe	Année	Milieu[4]	Filles (n)	Garçons (n)	Élèves (n total)
École A	A1	Multi	Urbain	9	4	13
École B	B4	6e année	Urbain	3		3
École C	C1	5e année	Rural		2	2
				12	6	18

Collecte de données

Nous avons implémenté un scénario pédagogique intitulé Deviens un maître NAO (Karsenti et al., 2019a), qui présente une série de tâches (réparties en 20 niveaux) visant à animer le robot NAO à l'aide de la programmation (Annexe 1). Numérotés de 1 à 20, ces niveaux induisent une progression permettant à l'élève de se familiariser avec le fonctionnement du robot et de réaliser des opérations de programmation de plus en plus complexes et variées. La programmation du robot NAO s'effectue avec le logiciel Chorégraphe (Aldebaran Robotics, 2014), dont l'interface est en anglais. Les pictogrammes et le guide d'accompagnement offert (Karsenti et al., 2019b) ont atténué significativement la barrière linguistique pour les élèves francophones. Ce logiciel repose sur le principe de programmation visuelle, c'est-à-dire que la principale façon d'animer le robot est de glisser des boîtes d'actions dans l'espace de travail, puis de les associer à l'aide de liens pour créer une séquence.

Figure 1

Interface de programmation visuelle du logiciel Choregraphe



La bibliothèque de boîtes (*box library*) du logiciel propose différentes fonctions qui permettent d'activer les fonctions du robot, allant des capteurs tactiles à la synthèse vocale, en passant par la reconnaissance

[4] Voir la note 3.

visuelle et le mouvement de ses bras, de ses jambes ainsi que de sa tête.

Les participants ont été placés en équipes de deux à cinq élèves pour réaliser les tâches du scénario pédagogique. Cette décision est motivée par le potentiel socioconstructiviste d'activités de cocréation à l'aide de la programmation (Romero et al., 2017) et, par extension, par la possibilité que les élèves puissent réinvestir seuls les apprentissages qu'ils auront réalisés en équipe (Vygotsky, 1934). Le nombre d'élèves dans chaque équipe variait selon le contexte du groupe. Chaque groupe se voyait remettre une valise contenant un robot NAO, un ordinateur portable doté du logiciel Chorégraphe, plusieurs copies imprimées des tâches du scénario pédagogique Deviens un maître NAO ainsi qu'un guide d'accompagnement en version électronique (Karsenti et al., 2019b), sur une tablette.

Afin d'observer les opérations de programmation accomplies par les équipes, nous avons utilisé deux instruments, le premier étant l'enregistrement des écrans des ordinateurs utilisés par les élèves, et le second étant l'observation vidéographiée. Des caméras ont été installées devant chaque équipe afin d'offrir une vue générale des actions à l'extérieur de l'interface Chorégraphe. Comme il s'agit de programmation impliquant l'animation d'un robot, ce second instrument s'avère utile pour compléter les données des enregistrements d'écran.

Traitement et analyse des données

L'analyse systématique des données a été effectuée sur les enregistrements d'écran afin d'observer les pratiques effectives de programmation dans l'interface du logiciel Chorégraphe. Il est à noter que les données analysées en lien avec le sous-objectif présenté dans cet article représentent une portion du corpus total. Ainsi, dans le but de proposer une typologie des pratiques effectives de programmation visuelle d'élèves du primaire, nous avons analysé les séances d'une équipe par école, et ce, pour trois visites, soit un total de neuf séances. Cela nous a permis d'analyser des données diversifiées tout en obtenant un portait complet, puisque toutes les équipes réalisaient les mêmes tâches présentées dans le scénario pédagogique commun. L'analyse des données a été réalisée avec le logiciel NVivo 12 (QSR International, 2020), où les catégories – et les codes associés – (Paillé & Mucchielli, 2005) ont été obtenus de façon inductive, tout en s'appuyant sur la classification proposée par Ruf et al. (2015) et au niveau du scénario pédagogique Deviens un maître NAO.

Résultats

En procédant, en amont, à l'analyse des notes de terrain, puis à l'analyse du corpus de données, nous proposons la typologie suivante (Tableau 2), inspirée de travaux de Ruf et ses collègues.

Le principal changement est lié au terme « écrire », qui n'est pas cohérent avec la programmation visuelle. Nous avons donc opté pour le terme « assembler », que nous avons subdivisé en opérations secondaires : chercher les boîtes, sélectionner les boîtes, lier les boîtes entre elles et enfin paramétrer ces boîtes. Certaines des pratiques de la classification de Ruf et ses collègues n'ont pu être observées : nous aborderons cela dans la discussion.

Tableau 2

Typologie inspirée de Ruf et al. (2015)

Pratiques de programmation
Assembler (opérations fondamentales) - Chercher - Sélectionner - Lier - Paramétrer
Assembler à partir d'une sélection préétablie
Ajuster, étendre ou compléter
Optimiser
Déboguer
Tester

Analyse des pratiques de programmation visuelle observées

Nous présentons ici chacune des pratiques de programmation observées en offrant des exemples concrets tirés du corpus de données.

Assembler

L'assemblage représente une agglomération des pratiques que nous qualifions de fondamentales pour la programmation visuelle. Leur caractère fondamental se manifeste par leur nécessité et leur utilisation, ce qui fut le cas dans notre corpus de données où trois pratiques observées sur quatre (76 %) relevaient de l'assemblage. Ainsi, la pratique d'assemblage devient en quelque sorte une métapratique incluant la recherche, la sélection, la liaison et le paramétrage. Le fait de décliner l'assemblage en différentes pratiques permet d'obtenir un portrait beaucoup plus précis de l'activité de programmation.

Assembler - Chercher

Contrairement à la programmation traditionnelle (écrite), la programmation visuelle nécessite une recherche afin de repérer la boîte, c'est-à-dire le code ou la fonction, qui sera utilisée dans le programme. Le logiciel Chorégraphe offre une bibliothèque de boîtes parmi lesquelles il est possible de chercher soit en naviguant dans l'arborescence de plusieurs dossiers thématiques, soit en utilisant un moteur de recherche. Ainsi, la pratique de programmation Chercher le code correspond au moment pendant lequel l'élève consulte la bibliothèque.

Assembler - Sélectionner

Une fois la boîte repérée dans la bibliothèque, l'utilisateur doit sélectionner la boîte avec le

curseur et la déplacer dans l'espace de travail. Il est également possible de créer des boîtes à partir de formats préétablis, ne laissant à l'élève que la tâche de la paramétrer. Par ailleurs, certaines fonctionnalités du robot exigent l'activation d'un bouton. C'est notamment le cas du mode Animation, qui est utilisé afin d'animer les différents membres du robot NAO à l'aide d'une technique reposant sur le principe d'animation en volume (*stop motion*). La pratique Sélectionner le code représente donc les instances où l'élève a déplacé une boîte provenant de la bibliothèque dans l'espace de travail, a créé une nouvelle boîte dans l'espace de travail, ou encore a activé différentes fonctionnalités, comme le mode Animation.

Assembler - Lier

Cette pratique est propre à la programmation visuelle. L'utilisation de boîtes de code distinctes induit *ipso facto* la nécessité de lier ces boîtes d'une façon ou d'une autre. Dans notre cas, le logiciel permet de lier les boîtes entre elles à l'aide d'un fil noir, un système comparable à un réseau électrique, où l'impulsion du départ, c'est-à-dire au lancement du programme, parcourt un fil avant de traverser (activer) chacune des boîtes. Cet influx est d'ailleurs rendu visible par un point vert parcourant les fils dans l'interface de programmation du logiciel, ce qui permet à l'utilisateur de suivre la progression du programme en temps réel. La pratique de liaison renvoie donc à l'action d'associer les boîtes entre elles, de même qu'avec les points de départ et de fin du programme. Il est à noter ici que certains élèves éprouvaient parfois des difficultés à tracer ces liens, notamment associées à la motricité fine.

Assembler - Paramétrer

Le paramétrage fait référence à toute action impliquant d'entrer dans une boîte pour ajouter, modifier ou supprimer du contenu. Par exemple, lorsqu'il était question de faire parler NAO, les élèves utilisaient la boîte *Say* ou *Animated Say*, dont le paramétrage consiste notamment à écrire le texte que le robot devra réciter. Il est également possible de paramétrer la vitesse et la tonalité de la voix du dispositif. Un autre exemple de paramétrage est la création de boîtes *Timeline*, qui sont utilisées avec le mode Animation. Comme il s'agit d'animation de volume (*stop motion*), il est nécessaire de bouger le robot dans une position, d'enregistrer cette position, puis de le mettre dans une autre position, l'enregistrer, et ainsi de suite. L'enregistrement des positions sur une ligne du temps représente aussi une action de paramétrage. La pratique de paramétrage consiste donc à insérer des valeurs numériques ou alphabétiques, à ajouter ou déplacer des indicateurs sur des échelles ou à modifier des valeurs à l'aide de boutons.

Assembler à partir d'une sélection préétablie

L'assemblage à partir d'une sélection préétablie a été distingué des opérations fondamentales d'assemblage en raison de son caractère particulier. La particularité réside dans le fait que cette pratique ne peut être réalisée que dans un contexte où une certaine forme d'encadrement est offerte, et non en pratique autonome. En effet, cette pratique implique que l'usager se soit vu offrir une sélection de boîtes et que ce dernier choisisse l'une d'elles pour composer son programme. Par exemple, dans le scénario pédagogique que nous avons proposé, au niveau 11, les élèves sont appelés à réaliser une tâche pour laquelle quatre codes sont suggérés : « Voici certaines des boîtes à utiliser » (Karsenti et al., 2019a). Ces

boîtes sont proposées à l'extérieur de l'espace de travail, ce qui implique que l'élève doit mobiliser les pratiques fondamentales d'assemblage afin d'utiliser ces boîtes suggérées (sélection préétablies).

Ajuster, étendre ou compléter

Similaire à l'assemblage à partir d'une sélection préétablie, cette pratique consiste à utiliser un code déjà présent dans l'espace de travail. La distinction est donc la présence des boîtes dans l'espace de travail, ce qui n'était pas le cas pour la pratique précédente. Dans notre corpus de données, nous avons été en mesure d'observer cette pratique au sein d'un même niveau, puisque chaque niveau de 1 à 9 est séparé en trois tâches. Ainsi, considérant la progression d'une tâche à l'autre, il est possible pour l'élève d'ajouter des boîtes ou de paramétrer à nouveau des boîtes utilisées antérieurement. Ces pratiques impliquent donc la modification d'un code déjà présent : il peut avoir été généré automatiquement dans l'interface, ou alors avoir été assemblé par l'élève précédemment.

Tester

Cette pratique concerne la vérification des programmes réalisés en les lançant, puis en observant le résultat de l'implémentation. Ce résultat peut prendre plusieurs formes selon le type de programmation. Dans le cas de la programmation d'un dispositif robotique, la vérification peut s'effectuer soit en analysant le déroulement du programme dans l'interface du logiciel, soit en observant le comportement du robot. Cette pratique permet donc de prouver le bon fonctionnement du programme conçu. Les données de notre corpus démontrent que les élèves ont parfois lancé le programme simplement pour voir le robot s'animer à nouveau, ayant préalablement confirmé, lors du premier test, que le programme fonctionnait adéquatement. Il est donc important de ne pas se limiter à la fréquence de cette pratique lors de l'interprétation des résultats de la grille.

Déboguer

Subséquente à la pratique de vérification (tester), la pratique de débogage survient lorsqu'il y a discordance entre le résultat obtenu et le résultat attendu. Le débogage s'impose alors comme un processus dont le but est d'identifier l'erreur (le bogue) et se conclut par une vérification réussie. Cette pratique est en fait une transposition du processus de résolution de problèmes, mobilisant ainsi plusieurs autres pratiques en vue de résoudre la situation. Le débogage est donc une pratique d'ordre général que l'on pourrait aussi qualifier de métapratique, comme l'assemblage. Or, ici les pratiques sous-jacentes ne sont pas propres au débogage en soi, mais bien aux pratiques d'assemblage fondamentales et de vérification notamment. En procédant de façon itérative, les élèves cernent le problème, tentent de le régler en modifiant le programme, effectuent un test, et ce, de façon itérative jusqu'à ce que le problème (bogue) soit résolu.

Mobilisation des pratiques de programmation visuelle

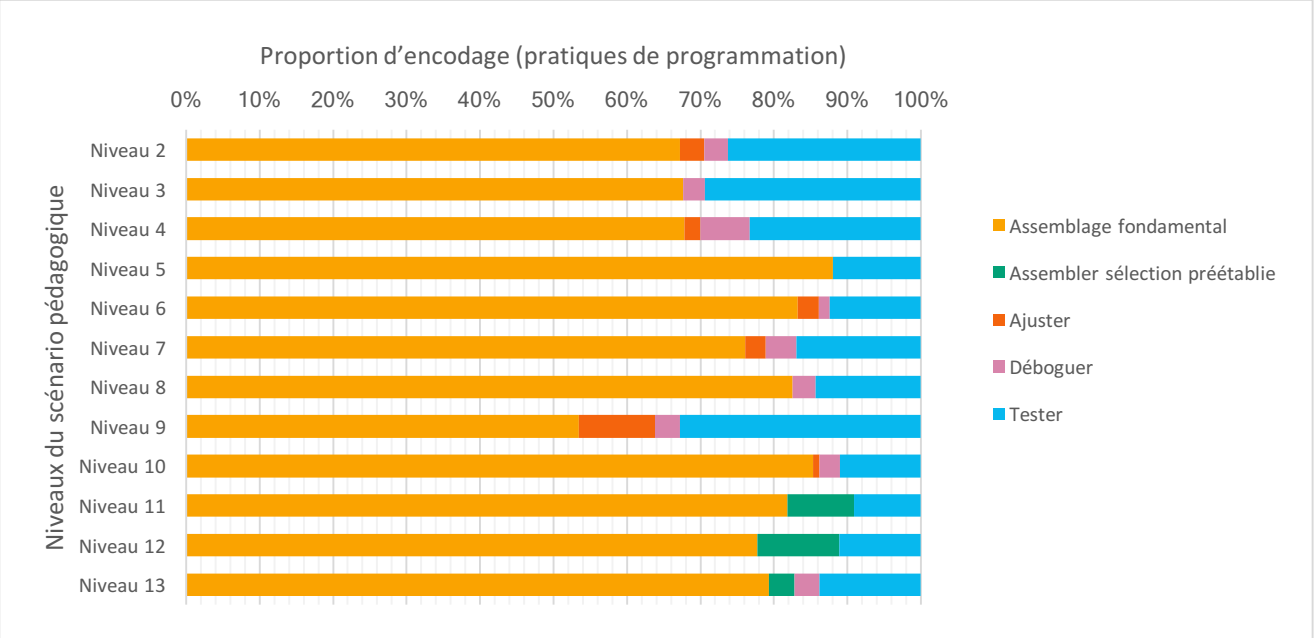
L'analyse des données a permis de démontrer dans quelles proportions les opérations de programmation ont été utilisées par les élèves (fréquence). La figure 2 présente donc les proportions

d'encodage[5] des pratiques de programmation à chaque niveau du scénario pédagogique. Le niveau 1 n'a permis l'observation d'aucune pratique de programmation puisque ce niveau amène l'élève à interagir avec le robot à l'aide de ses capteurs tactiles et de la synthèse vocale : ces fonctions font partie de la vie autonome du robot, c'est-à-dire qu'elles sont actives sans programmation. Il est également à noter que le niveau maximal atteint par les participants est le niveau 13 en raison en raison du temps limité de collecte, ce qui explique l'absence d'encodage pour les niveaux 14 à 20.

On constate que l'assemblage fondamental représente une proportion importante des pratiques de programmation effectuées par les élèves lors des activités. En effet, bien qu'étant moins utilisé au niveau 9, il représente néanmoins une majorité des pratiques à chaque niveau (M = 75,8 %). La figure 3 illustre la proportion attribuée à chaque pratique d'assemblage fondamentale (fréquence), par niveau.

Figure 2

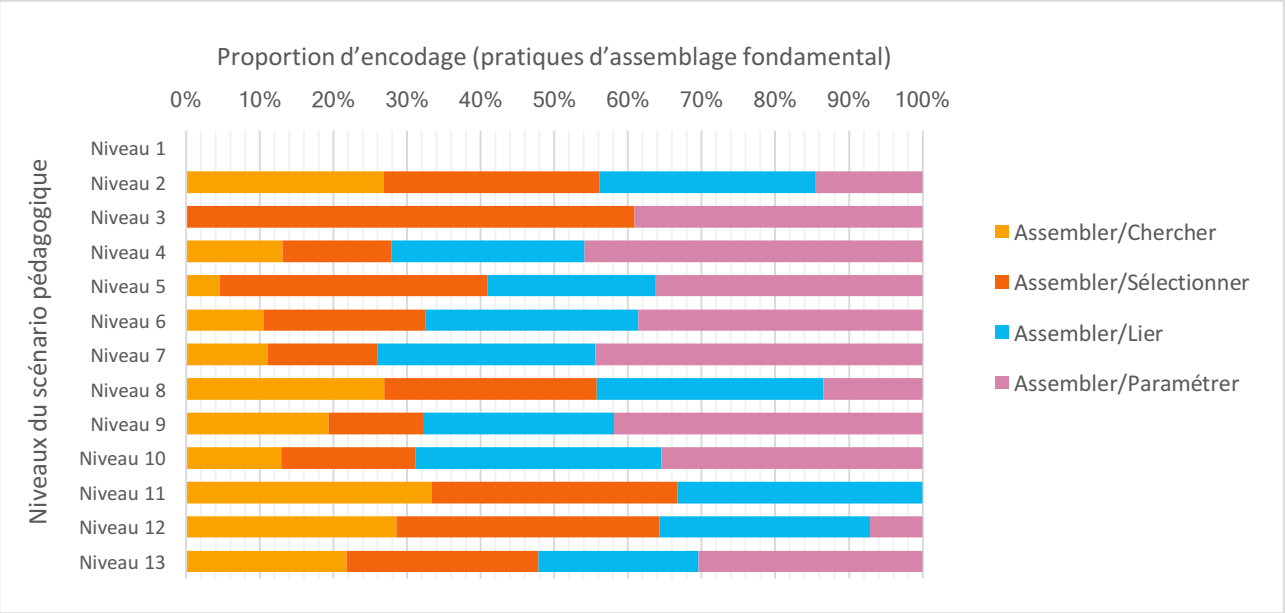
Pratiques de programmation par niveau du scénario pédagogique



[5] Le verbe *encoder* réfère à l'action d'attribuer un code à un segment du corpus de données vidéos (van der Maren, 2004). Nous utilisons le terme *encodage*, dans son emploi nominal, pour désigner l'ensemble de codes attribués.

Figure 3

Pratiques d'assemblage fondamentales (4) par niveau



Outre les proportions associées à la fréquence des opérations de programmation, nous avons cherché à savoir si le temps accordé à chaque opération était similaire. Le tableau 3 indique : (a) le nombre de minutes passées par les élèves à effectuer chacune des pratiques de programmation observées ; puis (b) compare les proportions de durée et de fréquence.

Tableau 3

Comparaison des proportions de durée et de fréquence d'encodage

Pratiques de programmation	Durée d'encodage (minutes)	Proportion (durée)	Fréquence	Proportion (fréquence)
Assembler (opérations fondamentales)				
Chercher	33	6,5 %	82	11,6 %
Sélectionner	13	2,6 %	126	17,8 %
Lier	28	5,5 %	149	21,1 %
Paramétrer	215	42,5 %	180	25,5 %
Assembler à partir d'une sélection préétablie	5	1,0 %	4	0,6 %
Ajuster, étendre ou compléter	25	4,9 %	17	2,4 %
Déboguer	119	23,5 %	22	3,1 %
Tester	68	13,4 %	126	17,8 %
	506	100,0 %	706	100,0 %

Les données obtenues indiquent une disparité entre la fréquence de certaines pratiques et leur durée. Cette disparité souligne une proportion considérablement plus importante tantôt pour la durée,

tantôt pour la fréquence. Parmi les pratiques dont la fréquence surpasse la durée, on retrouve notamment la sélection, dont la proportion de fréquence est 6,8 fois plus élevée, et la liaison, dont la proportion de fréquence est 3,8 fois plus élevée. Inversement, la proportion de temps accordé au débogage est 7,6 fois plus élevée que la proportion de fréquence de cette pratique, ce qui signifie qu'elle est moins fréquente, mais qu'elle dure longtemps.

Discussion

Les données font état d'un phénomène intéressant : alors que certaines pratiques sont très fréquentes et de courte durée, d'autres sont peu fréquentes et de très longue durée. Lorsque l'on s'intéresse à la fréquence, on remarque que les pratiques d'assemblage fondamentales dominent à ce chapitre, suivies de la pratique de vérification (tester). Lorsqu'il est question de la durée des pratiques, c'est le paramétrage et le débogage qui se démarquent. Comme le paramétrage représente une partie cruciale du processus de programmation, il semble logique de constater qu'il occupe 42,5 % du temps observé pour l'ensemble des pratiques. Cette pratique recèle une importante partie de la complexité inhérente à l'activité de programmation puisqu'elle suscite la manipulation de variables et de données qui déterminent la logique interne de chaque code. Le paramétrage est d'ailleurs une pratique souvent utilisée dans le processus de débogage. Cela dit, pour le débogage, c'est le rapport entre la durée et la fréquence qui est digne de mention : la proportion de durée est 7,6 fois plus élevée que la proportion de fréquence. Ainsi, le processus de débogage peut s'avérer chronophage.

Les figures 2 et 3 démontrent que la mobilisation des pratiques de programmation dans chaque niveau du scénario pédagogique Deviens un maître NAO est similaire. Bien que des différences subsistent, expliquées notamment par la nature plurielle des défis proposés aux élèves, il demeure que les apprenants ont été en mesure de mettre en œuvre les pratiques de façon régulière au fil de leur progression dans les niveaux du scénario.

Typologie des pratiques de programmation

Cette étude empirique a mené, dans un processus itératif, à la déclinaison d'un ensemble de pratiques effectives de programmation d'élèves du primaire. Le tableau 4 présente la typologie à l'aide de descripteurs.

Tableau 4

Typologie des pratiques effectives de programmation observées

Pratiques de programmation	Descripteur
1. Assembler (fondamental) 1.1 Chercher	L'élève cherche dans l'interface de programmation, à l'aide d'une bibliothèque ou d'un moteur de recherche, un code donné. Dans le cas des bibliothèques, il peut s'agir d'une liste du nom des codes, ou encore des boîtes elles-mêmes.
1.2 Sélectionner	À partir de la bibliothèque ou de tout autre emplacement dans

Pratiques de programmation	Descripteur
	l'interface, l'élève sélectionne ou génère une boîte, puis la déplace dans l'espace de travail, le cas échéant.
1.3 Lier	L'élève associe les codes (boîtes) entre eux et, s'il y a lieu, aux points de départ et de fin du programme. Cette pratique pourrait être sous-utilisée, voire inutilisée, dans le cas d'interfaces où le lien entre les boîtes est effectué par la juxtaposition de ces dernières (à la manière d'aimants). La pratique Sélectionner possède alors une double finalité qui inclut la liaison.
1.4 Paramétrer	L'élève ajoute des valeurs ou modifie et retire des valeurs préexistantes (par défaut). Ce paramétrage peut être effectué tant en surface, lorsque le paramétrage apparaît sur la boîte, qu'en entrant dans une boîte.
2. Assembler à partir d'une sélection préétablie	L'élève conçoit un programme incluant un ou plusieurs codes suggérés par un agent réel ou virtuel.
3. Ajuster, étendre ou compléter	L'élève utilise un ou plusieurs codes liés, déjà présents dans l'espace de travail, afin de créer un programme. Contrairement à l'assemblage à partir d'une sélection préétablie, où le code était suggéré, la pratique d'ajustement s'effectue sur des éléments déjà présents dans l'espace de travail (qu'ils y aient été placés par un agent réel, virtuel ou par l'élève lui-même, pour une autre tâche antérieure).
4. Tester	L'élève lance le programme et en vérifie le résultat. Cette pratique peut amener à améliorer (optimiser), déboguer le programme. La vérification peut aussi servir d'étape intermédiaire dans le processus de programmation, c'est-à-dire pour valider un code ou une portion de programme.
5. Déboguer	Après avoir constaté que le programme n'a pas donné le résultat escompté, l'élève part à la recherche du problème (bogue) dans l'implémentation du programme. Il s'agit d'un processus itératif dans lequel sont utilisées de nombreuses autres pratiques ; surtout la vérification (tester) et les pratiques d'assemblage fondamentales.

Pertinence de la typologie proposée

Cette proposition de typologie apporte un regard nouveau sur l'utilisation de la programmation au primaire. Considérant la démocratisation de l'usage pédagogique de la programmation informatique, il semble nécessaire d'offrir plusieurs balises afin d'encadrer cette activité à l'école. Le risque de limiter le potentiel de la programmation à son aspect ludique et motivant est bien réel. Ainsi, il est souhaitable que soient développés des scénarios pédagogiques ou des activités, comme celles de Romero et

Vallerand (2016), mobilisant la programmation de façon réfléchie, sans toutefois prétendre à l'exhaustivité de toutes ses facettes. Cette typologie offre à la fois des repères théoriques et empiriques pour la conception ou l'adaptation d'activités visant à mobiliser et à développer les compétences d'élèves du primaire par la programmation.

Limites

L'une des principales limites de notre recherche est le fait de n'avoir pu observer empiriquement que cinq types de programmation, et ce, en raison du scénario pédagogique que nous avons utilisé. Il y aurait certainement avantage à reproduire cette étude dans un contexte authentique de programmation où le scénario pédagogique permettrait d'observer d'autres pratiques.

Conclusion

L'objectif de cette étude de cas multiples était de proposer une typologie compréhensive et adaptée des pratiques effectives de programmation d'élèves du primaire. Après avoir décliné l'ensemble des pratiques observées auprès des apprenants de notre échantillon, il est devenu évident que les typologies et taxonomies tirées de la littérature n'étaient pas applicables au primaire. Cela a d'ailleurs confirmé la pertinence de notre proposition de typologie. Ainsi, les données empiriques nous ont permis de définir cinq types de pratiques. D'un point de vue scientifique, cet article pose les fondations d'un travail d'élaboration d'une typologie complète des pratiques effectives de programmation allant au-delà des limites inhérentes à un scénario pédagogique donné. Une telle typologie issue d'observations empiriques pourrait avoir des répercussions pratiques, par exemple en orientant la conception de matériel pédagogique de programmation. Notre travail ouvre également sur de nombreuses pistes d'approfondissements potentielles que nous souhaiterions explorer lors de recherches futures, notamment la validation de cette typologie auprès d'élèves du premier cycle du primaire, voire du préscolaire, ainsi que l'application de la typologie dans le cadre d'un autre scénario pédagogique.

Remerciements

Cet article présente de façon partielle les résultats d'une recherche doctorale (Parent, 2021) financée par le Conseil de recherche en sciences humaines du Canada (CRSH), le Fonds de recherche du Québec Société et Culture (FRQSC), et ayant bénéficié de l'appui de la Chaire de recherche du Canada sur le numérique en éducation.

Nous exprimons notre grande reconnaissance aux enseignant·e·s et aux élèves qui ont accepté de participer à cette étude. Nous tenons également à remercier Nicolas Kerbrat, Bong Sou Moulinet, Pierre-Luc Trahan et Simon Gosselin, qui ont contribué à la collecte et à l'analyse des données.

Références

Aldebaran Robotics. (2014). Choregraphe (version 2.1.4) [logiciel]. Softbank Group.

Blackwell, A. F. (2002, juin). What is programming? Dans J. Kuljis, L. Baldwin et R. Scoble (Eds). *Proceedings PPIG 14. 14th Workshop of the Psychology of Programming Interest Group,* Londres, Angleterre. https://ppig.org/files/2002-PPIG-14th-blackwell.pdf

Bower, M. (2008, juin). A taxonomy of task types in computing. Dans J. Amillo et C. Laxr, *Proceedings of the 13th annual conference on Innovation and technology in computer science education*. Madrid, Espagne. https://doi.org/10.1145/1384271.1384346

Chalkiadaki, A. (2018). A systematic literature review of 21st century skills and competencies in primary education. *International Journal of Instruction*, *11*(3), 1-16. https://doi.org/10.12973/iji.2018.1131a

Fortin, F., & Gagnon, J. (2016). *Fondements et étapes du processus de recherche. Méthodes quantitatives et qualitatives* (3e éd.). Chenelière éducation.

Forum économique mondial. (2015). *New vision for education : Unlocking the potential of technology*. https://www3.weforum.org/docs/WEFUSA_NewVisionforEducation_Report2015.pdf

Forum économique mondial. (2018). *Towards a reskilling revolution : A future of jobs for all*. http://www3.weforum.org/docs/WEF_FOW_Reskilling_Revolution.pdf

Green, T. R. G., & Petre, M. (1996). Usability analysis of visual programming environments : A « cognitive dimensions » framework. *Journal of Visual Languages & Computing*, *7*(2), 131-174. https://doi.org/10.1006/jvlc.1996.0009

Karsenti, T., & Demers, S. (2018). L'étude de cas. Dans T. Karsenti et L. Savoie-Zajc (dir.), *La recherche en éducation. Étapes et approches* (4e éd., p. 289-316). Presses de l'Université de Montréal.

Karsenti, T., Parent, S., Kerbrat, N., & Bugmann, J. (2019a). *Le robot NAO en éducation. Deviens un maître NAO* (2e éd.). CRIFPE.

Karsenti, T., Parent, S., Kerbrat, N., & Bugmann, J. (2019b). *Le robot NAO en éducation. Guide de l'élève* (3e éd.). CRIFPE.

Komis, V., & Misirli, A. (2011, octobre). Robotique pédagogique et concepts préliminaires de la programmation à l'école maternelle: une étude de cas basée sur le jouet programmable Bee-Bot. Dans G.-L. Baron, É. Bruillard et V. Komis, *Actes du quatrième colloque international DIDAPRO*. Colloque international DIDAPRO 4, Patras, Grèce. https://edutice.archives-ouvertes.fr/edutice-00676143/

Lai, A.-F., & Yang, S.-M. (2011, septembre). The learning effect of visualized programming learning on 6th graders' problem solving and logical reasoning abilities. Dans F. Dong, Proceedings - *2011 International Conference on Electrical and Control Engineering*. International Conference on Electrical and Control Engineering (ICECE), Yichang, Chine. https://doi.org/10.1109/ICECENG.2011.6056908

LeCompte, M. D., & Preissle, J. (1993). *Ethnography and qualitative design in educational research*. Academic Press.

Lee, M., Yun, J. J., Pyka, A., Won, D., Kodama, F., Schiuma, G., Park, H., Jeon, J., Park, K., Jung, K., Yan, M.-R., Lee, S., & Zhao, X. (2018). How to respond to the Fourth Industrial Revolution, or the Second Information Technology Revolution? Dynamic new combinations between technology, market, and society through open innovation. *Journal of Open Innovation. Technology, Market, and Complexity*, *4*(21). https://doi.org/10.3390/joitmc4030021

Ministère de l'Éducation. (2020). *L'usage pédagogique de la programmation informatique*. http://www.education.gouv.qc.ca/fileadmin/site_web/documents/ministere/Usage-pedagogique-programmation-informatique.pdf

Ministère de l'Éducation et de l'Enseignement supérieur. (2018). *Plan d'action numérique en éducation et en enseignement supérieur*. http://www.education.gouv.qc.ca/fileadmin/site_web/documents/ministere/PAN_Plan_action_VF.pdf

Ministère de l'Éducation et de l'Enseignement supérieur. (2019). *Cadre de référence de la compétence numérique*. http://www.education.gouv.qc.ca/fileadmin/site_web/documents/ministere/continuum-cadre-reference-num.pdf

Ministère de l'Éducation et de l'Enseignement supérieur. (2020). *Indices de défavorisation des écoles publiques*. http://www.education.gouv.qc.ca/references/indicateurs-et-statistiques/indices-de-defavorisation/

Noh, J., & Lee, J. (2020). Effects of robotics programming on the computational thinking and creativity of elementary school students. *Educational Technology Research and Development*, *68*(1), 463-484. https://doi.org/https://doi.org/10.1007/s11423-019-09708-w

Nugent, G., Barker, B., Grandgenett, N., & Adamchuk, V. (2009, 18-21 octobre). The use of digital manipulatives in k-12: robotics, GPS/GIS and programming. *39th ASEE/IEEE Frontiers in education conference,* Texas, États-Unis.

Paillé, P., & Mucchielli, A. (2005). *L'analyse qualitative en sciences humaines et sociales*. Armand Colin.

Parent, S. (2021). *La programmation informatique à l'école primaire : pratiques effectives de programmation et mobilisation d'habiletés de résolution collaborative de problèmes (RCP)* [thèse de doctorat, Université de Montréal]. Papyrus. http://hdl.handle.net/1866/25874

Pires, A. P. (1997). Échantillonnage de recherche qualitative : essai théorique et méthodologique. Dans J. Poupart, L.-H. Groulx, J.-P. Deslauriers, A. Laperrière, R. Mayer et A. P. Pires (dir.), *La recherche qualitative. Enjeux épistémologiques et méthodologiques* (p. 113-167). Gaëtan Morin.

QSR International. (2020). NVivo 12 (version 12.6.0) [logiciel]. QSR International.

RÉPAQ. (2020). *Portrait de l'école alternative*. Réseau des écoles publiques alternatives du Québec. https://repaq.org/portrait/

Romero, M. (2017). Les compétences pour le XXI^e^ siècle. Dans M. Romero, B. Lille et A. Patiño (dir.), *Usages créatifs du numérique pour l'apprentissage au XXI^e^ siècle*. Presses de l'Université du Québec.

Romero, M., Lepage, A., & Lille, B. (2017). Computational thinking development through creative programming in higher education. *International Journal of Educational Technology in Higher Education*, *14*(1), art. 42. https://doi.org/10.1186/s41239-017-0080-z

Romero, M., & Vallerand, V. (2016). *Guide d'activités technocréatives pour les enfants du 21^e^ siècle*. CoCreaTIC.

Ruf, A., Berges, M., & Hubwieser, P. (2015, septembre). Classification of Programming Tasks According to Required Skills and Knowledge Representation. Dans A. Brodnik et J. Vahrenhold, *Informatics in Schools. Curricula, Competences, and Competitions. Lecture Notes in Computer Science*. 8th International Conference on Informatics in Schools: Situation, Evolution, and Perspectives (ISSEP), Ljubljana, Slovénie. https://doi.org/10.1007/978-3-319-25396-1_6

Stake, R. E. (1995). *The art of case study research*. Sage Publications.

Turski, W. M. (1978). *Computer programming methodology*. London.

van der Maren, J.-M. (2004). *Méthodes de recherche pour l'éducation* (2e éd.). Presses de l'Université de Montréal.

Vygotsky, L. S. (1934). *Thought and language*. M.I.T. Press.

Vygotsky, L. S. (1997). *Pensée et langage*. La Dispute.

Wei, X., Lin, L., Meng, N., Tan, W., & Kong, S.-C. (2021). The effectiveness of partial pair programming on elementary school students' computational thinking skills and self-efficacy. *Computers & Education*, *160*, art. 104023. https://doi.org/https://doi.org/10.1016/j.compedu.2020.104023

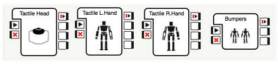
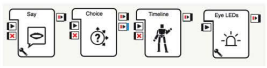
Wood, D., Bruner, J. S., & Ross, G. (1976). The role of tutoring in problem solving. *Journal of child psychology psychiatry*, *17*(2), 89-100. https://doi.org/10.1111/j.1469-7610.1976.tb00381.x

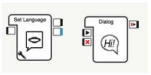
Annexe

Deviens un MAÎTRE NAO

Niveaux	Défis
1	◆ Vous devez doucement caresser le dessus de la tête de NAO pour qu'il réagisse. ◆◆ Vous devez dire « Salut » ou « Bonjour » à NAO jusqu'à ce qu'il vous comprenne et vous réponde. Demandez-lui, ensuite, « comment ça va ? ». ◆◆◆ Vous devez demander à NAO de se coucher sur le dos ou sur le ventre.
2	◆ Vous devez faire s'asseoir NAO en le programmant. ◆◆ Vous devez faire dire « Bonjour » à NAO en le programmant. ◆◆◆ Vous devez lui faire faire un *Bonjour animé* dans lequel il dira « Bonjour mon ami » en le programmant.
3	Utiliser la fenêtre *Robot View* pour ce niveau ◆ Vous devez faire tourner la tête de NAO à gauche. ◆◆ Vous devez lui faire lever le bras droit. ◆◆◆ Vous devez lui faire bouger les deux bras en même temps.
4	◆ Vous devez faire avancer NAO d'un mètre en le programmant. ◆◆ Vous devez faire reculer NAO d'un mètre en le programmant. ◆◆◆ Vous devez faire avancer NAO de 0,5 mètre vers la gauche et lui faire faire un «*Bonjour animé* » où il dit «Content de vous rencontrer !» en le programmant.
5	Utiliser le mode *animation* pour ce niveau ◆ Vous devez faire lever le bras gauche de NAO, remplacer le nom de la boîte *Timeline* par « Bras gauche levé » et changer l'image en choisissant une proposée par le logiciel. ◆◆ Vous devez faire lever les deux bras de NAO. ◆◆◆ Vous devez donner une nouvelle position à NAO sans qu'il ne tombe.
6	◆ Vous devez faire poser la question suivante à NAO : « Aimes-tu les desserts ?» en utilisant la boîte *Choice*. Il doit vous comprendre et vous répondre « Moi, j'adore ça ! » ◆◆ Vous devez faire poser la question suivante à NAO : « Quelle est la capitale du Canada ?». Vous devrez proposer 3 choix de réponse, faire en sorte qu'il reconnaisse et félicite la bonne réponse ou demande de réessayer en cas de mauvaise réponse. ◆◆◆ Vous devez faire poser la question suivante à NAO : « Quelle est la capitale des États-Unis ? ». Veillez à proposer 3 choix de réponse. NAO devra lever la main droite en cas de bonne réponse et lever la main gauche en cas de mauvaise réponse.
7 BRONZE	◆ Vous devez faire en sorte que NAO reconnaisse la balle rouge et la suive avec la tête. ◆◆ Vous devez faire en sorte que NAO reconnaisse la balle rouge, se dirige vers elle, et s'arrête à 0,2 mètre d'elle. ◆◆◆ Vous devez faire en sorte que NAO reconnaisse la balle rouge, se dirige vers elle en levant le bras droit, s'arrête à 0,4 mètre et dise la phrase : « Ma balle est ici, je la cherchais justement !».
8 ARGENT	◆ Vous devez faire apprendre à NAO deux visages. ◆◆ Vous devez faire en sorte que NAO reconnaisse votre visage et dise votre nom lorsque vous serez devant lui. ◆◆◆ Vous devez faire en sorte que NAO reconnaisse deux visages et dise un message personnalisé différent en identifiant chaque personne.
9 OR	Utiliser le mode *animation* pour ce niveau ◆ Vous devez faire jouer une musique de votre choix à NAO. ◆◆ Vous devez faire jouer cette musique à NAO et lui faire faire une chorégraphie avec les bras (*boîte Timeline*). ◆◆◆ Vous devez lui faire jouer la musique, puis lui faire faire une chorégraphie avec les bras, la tête et les jambes (boîte *Timeline*). Enregistrez la chorégraphie dans la librairie.
10 PLATINE	Utiliser le mode *animation* pour ce niveau Vous devez faire demander à NAO : « *Veux-tu que je te montre mon super parcours d'activité physique ?* ». A la réponse « oui », une musique commencera et NAO débutera sa série d'exercices que vous aurez programmé : ▪ NAO devra faire un trajet en forme de rectangle (1 mètre en avant, 0,3 m à droite, 1 m en arrière et 0,3 m à gauche) ; ▪ Il devra faire un exercice de votre choix au début, au milieu et à la fin du parcours en *mode animation*. Vous avez la possibilité de reprendre la chorégraphie du niveau 9 pour un des exercices. ▪ Quand il aura terminé il dira : « Ouf, ça fait du bien !» et s'assiéra sur le sol.

Deviens un MAÎTRE NAO

Niveaux	Défis
11	Vous devez programmer des actions différentes en fonction des capteurs sensoriels qui sont touchés. Voici certaines des boîtes à utiliser :  Vous avez le choix : Demander à NAO de dire quelque chose, de danser, de bouger, de poser une question, etc.
12	Vous devez programmer NAO pour lui faire dire quelque chose uniquement après avoir appuyé sur deux de ses capteurs, et cela l'un après l'autre. Voici certaines des boîtes à utiliser pour réaliser ce programme : 
13	Vous devez faire un diagramme qui comporte 4 phrases que NAO devra dire au hasard. Voici certaines des boîtes à utiliser pour réaliser ce programme :  Ces 4 phrases doivent toutes être différentes et parler des robots.
14	Vous devez faire faire à NAO des mouvements (Boîtes *Timeline*) qui représentent des émotions (la tristesse, la peur, la joie, la colère et l'amour) et lui faire demander à quelqu'un de quelle émotion il s'agit. Il félicite la bonne réponse et enchaîne avec une nouvelle émotion à deviner jusqu'à ce que toutes les émotions soient correctement identifiées. Voici certaines des boîtes à utiliser pour réaliser ce programme : 
15	Vous devez construire une suite d'actions qui contient 10 boîtes différentes. Toutes les boîtes doivent avoir une utilité. **Attention**, il n'est pas possible de retrouver deux fois la même boîte.

Niveaux	Défis
16	Vous devez demander à NAO de reconnaître visuellement un ballon bleu, rouge ou vert. Il doit répondre correctement en parlant et en indiquant la couleur avec ses yeux. Vous pouvez lui apprendre les couleurs ou alors utiliser les NaoMarks. Voici certaines des boites dont vous pourriez avoir besoin : 
17	Vous devez faire dire à NAO « Bonjour jeunes humains, je suis content de vous voir après tout ce temps passé dans ma boîte », avec le mode *Python Script*. Voici certaines des boîtes à utiliser pour réaliser ce programme : 
18	À l'aide d'une boîte *Timeline*, vous devez faire s'agenouiller NAO sans qu'il ne tombe. Ensuite, vous demanderez à NAO d'énoncer un problème mathématique que vous aurez créé, qui demande de multiplier deux nombres entiers entre eux. NAO devra y répondre. Voici certaines des boîtes à utiliser pour réaliser ce programme : 
19	À l'aide d'une boîte *Dialog*, vous devez créer un programme qui permet à NAO de répondre à vos questions sur différentes capitales du monde. Voici certaines des boîtes à utiliser : 
	En utilisant les 3 types de boîtes *Dialog*, *Timeline et Python*, vous devez donner des consignes de vie de classe à NAO en anglais (Assoie-toi, lève-toi, écoute, va au tableau, lève la main, excuse-toi…). Il doit les exécuter.

Auteur

Simon Parent, Ph.D., est conseiller principal de recherche à l'Université de Montréal. Il s'intéresse notamment à la compétence numérique et à l'utilisation pédagogique de la programmation au primaire. Il collabore activement à des projets de recherche portant sur l'enseignement comodal, la formation à distance et la compétence numérique en général. Email : simon.parent.2@umontreal.ca.