PENSÉE INFORMATIQUE EN ENSEIGNEMENT DES MATHEMATIQUES POST-SECONDAIRES : UN CADRE THÉORIQUE

BUTEAU[*] Chantal – MULLER[*] Éric – SACRISTÁN[**] Ana Isabel – MGOMBELO[*] Joyce

Résumé – Nous voyons un besoin crucial de comprendre comment habiliter les étudiants en mathématiques à participer à la pensée informatique qui devient une partie intégrale des mathématiques. Dans notre recherche, nous explorons comment les étudiants en mathématiques de niveau universitaire peuvent s'approprier la programmation et s'engager dans la pensée informatique pour les mathématiques, comme le feraient des mathématiciens. Cet article présente le cadre théorique de nos recherches.

Mots clefs : pensée informatique, pensée computationnelle, genèse instrumentale, programmation, mathématiques postsecondaires, 3e pilier de la recherche scientifique

Abstract –We see a crucial need in mathematics education to understand how students could be empowered to participate in the computational thinking that is now becoming an integral part of the mathematics community. In our research, we are interested in examining how university mathematics students may appropriate programming and engage in computational thinking for mathematics, as mathematicians would do. In this paper, we present the theoretical framework that grounds our research.

Keywords: computational thinking, instrumental genesis, programming, third pillar of scientific inquiry, undergraduate mathematics

I. INTRODUCTION

Avant même de l'arrivée des ordinateurs personnels, Papert (1971) imaginait un monde dans lequel les enfants programmaient naturellement l'ordinateur, les utilisant comme un outil pour agir en tant que jeunes mathématiciens. Près de 50 ans plus tard, nous assistons à un regain répandu de l'intérêt porté à cette vision, qui se manifeste par plusieurs réformes éducatives, en mathématiques et ailleurs (p. ex. En Europe : Bocconi et al., 2016), sous le couvert de la pensée informatique, aujourd'hui considérée comme une compétence du 21e siècle.

Wing (2008) décrit la pensée informatique comme un type de pensée critique qui rejoint la pensée mathématique de par les approches générales par lesquelles on peut aborder la résolution de problèmes. Mais on observe aussi un apport de l'informatique à une approche mathématique traditionnelle (sans technologie numérique). Gadanidis et al. (2016) discutent d'affordances qui, lorsque combinées, mettent en avant une dimension expérimentale à l'activité mathématique, notamment par la modélisation dynamique et l'automatisation (p. ex. simulation), des surprises conceptuelles (p. ex. utiliser la programmation pour explorer de « nouvelles » mathématiques), d'entrées multiples (p. ex. plusieurs angles d'entrées possibles pour explorer et résoudre un problème mathématiques), et de la sensation tangible (p. ex. avec diverses visualisations dynamiques de concepts mathématiques). En somme, nous pouvons décrire celles-ci comme caractérisant, selon les termes de Papert (1980b), un « objet-avec-lequel-penser » reprenant selon nous l'essence de l'utilisation de la programmation par les mathématiciens dans leur travail de recherche (Broley, 2015).

Nous voyons un besoin important de comprendre comment on peut habiliter les étudiants en mathématiques à faire de la programmation, cet objet-avec-lequel-penser qui occupe aujourd'hui une place si importante dans la communauté mathématique. On s'intéresse donc à la programmation en tant qu'instrument d'investigation et d'application mathématique plûtot qu'à la nature mathématique de la programmation. Cet article se concentre sur le cadre

[*] Brock University – Canada – {cbuteau,emuller,jmgombelo}@brocku.ca
[**] Cinvestav-Matemática Educativa – Mexico – asacrist@cinvestav.mx

conceptuel qui sous-tend notre projet de recherche en cours (2017-2022), financé par le Conseil de recherche en sciences humaines du Canada (CRSH). Il vise à explorer comment les étudiants de mathématiques de niveau postsecondaire apprennent à utiliser la programmation comme un instrument de pensée informatique pour les mathématiques.

Il s'agit d'une étude non interventionnelle qui s'inscrit dans une séquence de trois cours de mathématiques axés sur la programmation à l'université Brock (Canada). Dans ces cours, en place depuis 2001 et destinés aux étudiants en mathématiques et aux futurs enseignants de mathématiques, les étudiants apprennent à concevoir, programmer, et utiliser des environnements informatiques interactifs pour l'investigation de conjectures, concepts et théorèmes mathématiques et d'applications de situations réelles (Buteau et al., 2015; Muller et al., 2009). Les objectifs de notre recherche incluent : (a) décrire la genèse instrumentale des étudiants de l'utilisation de la programmation en tant qu'instrument de pensée informatique pour les mathématiques; (b) déterminer s'il y a appropriation et si celle-ci est maintenue au-delà des exigences du cours; (c) identifier comment les enseignants créent un environnement d'apprentissage pour soutenir les genèses instrumentales des élèves. Cette étude s'appuie sur nos recherches passées et en cours (p. ex. Buteau et Muller, 2014; Buteau et al., 2016).

II. CADRE CONCEPTUEL

Nous commençons par discuter plus largement de la pensée informatique et de la programmation, en nous appuyant notamment sur les travaux de Wing (2008). Nous portons ensuite notre attention sur la pensée informatique en mathématiques telle que décrite par les pratiques de recherche des mathématiciens. Cela conduit à une discussion sur la pensée informatique dans l'enseignement des mathématiques, qui à son tour est éclairée par les travaux de Weintrop et al. (2016) et par le paradigme constructionniste (Papert et Harel, 1991). Ensuite, nous développons notre vision de l'apprentissage des mathématiques par une entrée informatique en nous inspirant des travaux de Lave et Wenger (1991). Enfin, nous présentons l'approche instrumentale de Guin et Trouche (1999) afin d'éclairer notre compréhension de l'intégration de la technologie dans l'enseignement des mathématiques.

1. La pensée informatique

Wing (2014) décrit ainsi la pensée informatique « les processus de pensée impliqués dans la formulation d'un problème et de ses résolutions de telle sorte qu'un ordinateur—humain ou machine—puisse l'effectuer efficacement. » (para. 5)[1]. Ainsi, la pensée informatique est un processus sous-jacent à la programmation informatique. Et comme le déclarent Grover et Pea (2013), la programmation informatique « n'est pas seulement une compétence fondamentale de [l'informatique] et un outil clé pour soutenir les tâches cognitives impliquées dans [la pensée informatique], c'est aussi une démonstration des compétences informatiques » (p. 40). De plus, Wing (2008) explique que l'essence de la pensée informatique se trouve dans l'abstraction.

La relation qu'entretiennent la programmation informatique et la pensée informatique avec la pensée et l'apprentissage mathématiques et scientifiques est reconnue depuis le développement du langage de programmation Logo (voir Papert, 1980a). Cette relation est mise en évidence dans le cadre tridimensionnel proposé par Brennan et Resnick (2012) qui caractérise la « pensée informatique » en termes de

[1] Toutes les citations d'origine anglaise sont des traductions libres.

> *les concepts informatiques* (les concepts avec lesquelles les concepteurs interagissent dans la programmation, comme l'itération, le parallélisme, etc.), *les pratiques informatiques* (les pratiques que développent les concepteurs alors qu'ils interagissent avec les concepts, comme le débogage ou le remixage du travail d'un autre), et *les perspectives informatiques* (les perspectives que les concepteurs se forment du monde qui les entoure et d'eux-mêmes). (p. 1)

Dans les sections suivantes, nous abordons le thème de la pensée informatique en mathématiques puis celui de la pensée informatique dans l'enseignement et l'apprentissage des mathématiques.

2. La pensée informatique en mathématiques

La Société mathématique européenne (SME) (2011) a reconnu une nouvelle façon de s'engager dans la recherche en mathématiques : « Avec la théorie et l'expérimentation, un troisième pilier de l'investigation scientifique des systèmes complexes a émergé dans la combinaison de la modélisation, de la simulation, de l'optimisation et de la visualisation » (p.2). En 2016, les mathématiciens qui ont dirigé un semestre thématique de 6 mois sur les mathématiques numériques dans les applications émergentes au Centre de recherches mathématiques (CRM) à Montréal (Canada) ont indiqué que :

> Un changement fondamental est en train de se produire dans le rôle des mathématiques appliquées et computationnelles. Le rapport entre la modélisation, l'analyse et les solutions des problèmes mathématiques dans les applications a changé. […] Dans les applications émergentes, le choix des modèles va de pair avec les outils computationnels et l'analyse mathématique. (CRM, 2016, paragr. 1)

Ces pratiques émergentes en matière de mathématiques relèvent de la pensée informatique pour les mathématiques et sont enracinées dans la technologie programmable. En effet, la taxonomie de Weintrop et al. (2016) (Figure 1) offre un aperçu de l'engagement des mathématiciens dans la pensée informatique, qui englobe les activités décrites par la SME (2011) et par les organisateurs du semestre sur les mathématiques informatiques au CRM. Les travaux de Weintrop et al. sont fondés sur une revue de littérature approfondie, une analyse d'activités d'apprentissage des mathématiques et des sciences et sur des entrevues avec des scientifiques tels que biochimistes, physiciens, ingénieurs en matériaux, informaticiens et ingénieurs biomédicaux. Les auteurs décrivent également ce qu'ils croient être les pratiques intégrales de la pensée informatique pour les mathématiques et les sciences. Broley, Buteau et Muller (2017) ont illustré par des travaux de recherche de mathématiciens les différentes formes de pratiques de pensées informatiques intégrales proposées par Weintrop et al.

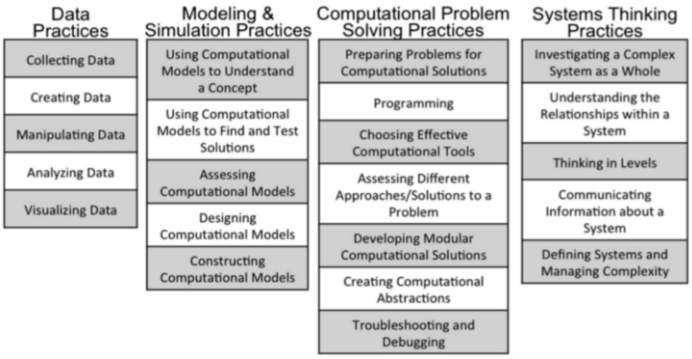

Fig. 1 – Taxonomie de la pensée informatique en mathématiques et sciences (Weintrop et al., 2016, p. 135).

En adoptant le cadre de Brennan et Resnick (2012) dans le contexte des mathématiques, le travail de Weintrop et al. (2016) fournit non seulement des détails spécifiques aux disciplines sur la dimension des pratiques informatiques, mais met également de l'avant des perspectives informatiques—c'est-à-dire des perspectives récemment développées par les mathématiciens

sur les mathématiques en tant que discipline « conformément à la nature de plus en plus informatique des mathématiques et des sciences de l'ère moderne » (Weintrop et al., 2016, p. 127).

3. La pensée informatique dans l'enseignement et l'apprentissage des mathématiques

De plus, Weintrop et al. (2016) affirment que « l'utilisation variée et appliquée de la pensée informatique par les experts du domaine fournit un plan détaillé pour l'enseignement de la pensée informatique » (p. 128). Leur taxonomie détaillée fournit donc une image de ce que ça représente pour la classe de mathématiques de s'engager dans la pensée informatique en mathématiques comme le feraient les mathématiciens.

Tel que mentionné précédemment, le langage de programmation Logo et la théorie du constructionnisme constituent un héritage vieux de 45 ans pour la pensée informatique dans l'enseignement des mathématiques (Papert & Harel, 1991). La prémisse du paradigme constructionniste est de créer des situations d'apprentissage centrées sur les élèves afin qu'ils s'engagent consciemment dans la construction (p. ex. programmer) d'objets tangibles, à travers des projets—habituellement informatiques—significatifs : « Les gens se construisent de nouvelles connaissances avec une efficacité particulière quand ils sont engagés dans la construction d'objets significatifs [...] [c'est-à-dire] quelque chose de significatif pour eux-mêmes et pour les autres autour d'eux » (Kafai & Resnick, 1996, p. 214).

Des études sur le constructionnisme dans l'enseignement supérieur des mathématiques montrent comment la programmation aide les élèves dans leur compréhension des concepts mathématiques (Leron et Dubinsky, 1995) et comment elle contribue au développement de la pensée critique (p. ex., Abrahamson et al., 2004). En fait, Noss et Hoyles (1996) soulignent qu'un apprenant, lorsqu'il entreprend de modifier un programme, articule des relations entre des concepts présents au sein d'un micromonde « et c'est dans ce processus d'articulation qu'un apprenant peut créer des mathématiques et révéler simultanément cet acte de création à un observateur » (p. 54). Dans notre travail, nous reconnaissons l'approche constructionniste pour la mise en œuvre de la programmation et de la pensée informatique qu'elle entraîne, et concevons l'apprentissage des mathématiques en prenant appui sur des aspects de la théorie de la cognition située, comme décrite ci-dessous.

4. Apprendre les mathématiques en s'engageant dans une pensée informatique

Notre vision de l'apprentissage s'inspire de quelques idées des travaux de Lave et Wenger (1991) sur les communautés de pratique. Hoadley (2012) souligne que deux définitions de communautés de pratique découlent des travaux de Lave et Wenger : (i) une définition basée sur les caractéristiques, signifiant une communauté qui partage des pratiques et (ii) une définition basée sur les processus d'apprentissage par lesquels les communautés de pratique sont considérées comme des groupes dans lesquels prend place un processus constant de « participation périphérique légitime ». Dans notre travail, on s'appuie sur cette deuxième définition. Lave et Wenger utilisent le concept de participation périphérique légitime pour décrire comment les apprenants entrent dans une communauté et adoptent progressivement ses pratiques. Nous utilisons cette idée de la participation périphérique légitime pour comprendre comment les étudiants apprennent les mathématiques en s'engageant dans la pensée informatique. Les « mathématiques » ne sont alors pas considérées comme un ensemble de connaissances à acquérir par l'étudiant, mais plutôt comme un processus de participation continuelle par lequel l'étudiant devient progressivement membre d'une communauté (de mathématiciens). En outre, nous ne voyons pas la pensée informatique d'un

point de vue cognitif (par exemple, voir un ordinateur comme un outil d'apprentissage interactif illustrant des concepts). Plutôt, nous nous concentrons sur la façon dont les étudiants créent et utilisent des outils informatiques pour s'engager dans des opportunités de participer en périphérie à des pratiques considérées comme faisant partie intégrante de la communauté mathématique, comme indiqué par Weintrop et al. (2016). En d'autres termes, notre emphase est sur la façon dont les étudiants (les novices) se servent de la pensée informatique pour les mathématiques, comme le feraient les mathématiciens (les vétérans).

Cette vision de l'apprentissage concorde avec le paradigme constructionniste. Papert (1971) a soutenu que « être mathématicien [...] comme être un poète, un compositeur ou un ingénieur, c'est *faire*, plutôt que de savoir ou de comprendre » (p.1, notre traduction), et qu'en programmant les mathématiques, les apprenants s'engagent dans des « mathématiques informatiques » (p. 25) à travers lesquelles ils mathématisent. Pour Papert (1980b), l'ordinateur fournit à l'apprenant un moyen de construire des « objets-avec-lequel-penser» et, par l'exploration et la découverte dans un certain domaine de connaissances, « permet à un apprenant de mobiliser des idées productrices ou des habilités intellectuelles » (p.204). Cela résonne avec le grand nombre de mathématiciens et de scientifiques qui utilisent l'ordinateur au 21e siècle.

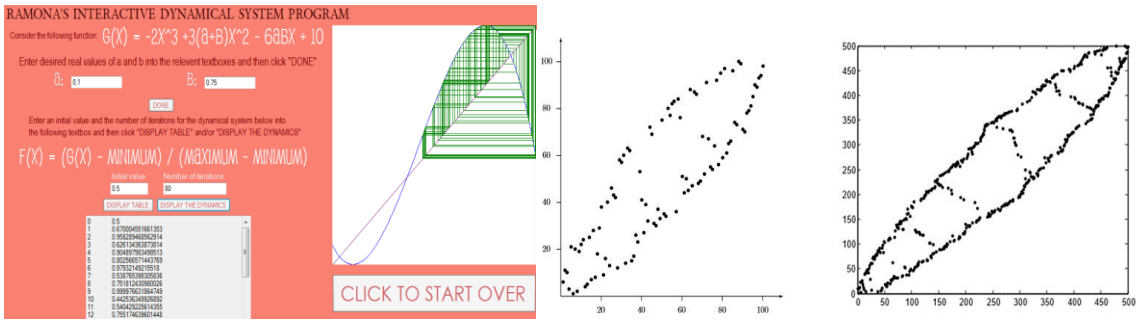

Fig. 2 – Exemples de pratiques en résolution de problèmes informatiques. À gauche : capture d'écran du travail exploratoire d'un étudiant de premier cycle d'un système dynamique À droite : capture d'écran du travail exploratoire d'un mathématicien sur une structure de permutation (Broley et al., 2017, pp. 4, 6).

Le travail de Broley et al. (2017), cité plus haut, illustre comment des étudiants du premier cycle universitaire ont appris les mathématiques à travers la construction d'objets informatiques interactifs (c'est-à-dire, « objets-avec-lequel-penser ») et comment ces pratiques s'harmonisent avec celles des mathématiciens : par exemple, l'engagement d'une étudiante dans les pratiques informatiques de résolution de problèmes—où elle a dû concevoir, programmer et utiliser un environnement interactif pour explorer, graphiquement et numériquement, le comportement d'un système dynamique basé sur une fonction cubique à deux paramètres—montre des similitudes avec l'engagement d'un mathématicien dans ses recherches sur la permutation de sous-séquences (voir Figure 2). C'est lorsque l'étudiant devient compétent à s'engager, par la programmation, dans la pensée informatique pour les mathématiques « comme les mathématiciens le feraient » que nous considérons qu'une appropriation a eu lieu. Ce qui suit s'intéresse à cette appropriation et à comment on peut en rendre compte.

5. *L'appropriation, par les étudiants, de la programmation en tant qu'instrument de pensée informatique*

Cook et al. (2002) expliquent que l'appropriation est un processus de développement impliquant des actions exprimées socialement, orientées vers un but et véhiculées par des

outils par lesquelles les apprenants adoptent activement (nous pourrions dire « font siens ») des outils conceptuels et pratiques, internalisant ainsi les modes de pensée liés aux contextes spécifiques dans lesquels l'apprentissage a lieu. L'approche instrumentale (Rabardel, 1995/2002) est un cadre utile pour l'analyse de l'intégration technologique (Artigue, 2002; Guin & Trouche, 1999) et pour mieux comprendre comment les étudiants s'approprient un outil (technologique).

L'approche instrumentale décrit comment les artefacts (matériels ou symboliques) sont appropriés lorsqu'ils sont transformés en instruments à travers des schèmes d'utilisation et d'action par ce qu'on appelle la *genèse instrumentale* (Artigue, 2002). Trouche et Drijvers (2010) suggèrent qu'un instrument a été approprié lorsqu'il existe une « relation significative entre l'artefact et l'utilisateur pour un type spécifique de tâche » (p. 673). Ainsi, afin d'évaluer l'appropriation et l'intégration technologique, il est nécessaire d'observer la genèse instrumentale, en regardant à la fois l'artefact et les schèmes qui lui sont associés. Une façon de faire cela est de regarder les traces laissées par les élèves dans leur activité et ce qu'ils font avec l'artefact (Trouche 2004). Il est également nécessaire de prendre en compte l'activité de l'enseignant : ses conceptions et ses orchestrations des ressources (Trouche, 2004) et l'*intégration instrumentale*, qui est « comment les enseignants organisent les conditions pour une genèse instrumentale de la technologie proposée aux étudiants et dans quelle mesure ils favorisent l'apprentissage des mathématiques à travers la genèse instrumentale » (Goos & Soury-Lavergne, 2010, p. 313).

L'intégration instrumentale décrit quatre étapes d'une utilisation croissante de la technologie en classe (Assude, 2007) : (a) l'initiation instrumentale (étape 1)—les étudiants s'engagent uniquement dans l'apprentissage de l'utilisation de la technologie; (b) l'exploration instrumentale (étape 2)—les problèmes mathématiques motivent les élèves à en apprendre davantage sur l'utilisation de la technologie; (c) le renforcement instrumental (étape 3)—les étudiants résolvent des problèmes mathématiques avec la technologie, mais doivent étoffer leurs compétences technologiques; et (d) la symbiose instrumentale (étape 4)—la maîtrise de la technologie par les étudiants permet d'élargir la tâche mathématiques si bien que ce sont à la fois les compétences technologiques et la compréhension mathématiques des élèves qui sont enrichies.

Nous associons ces étapes aux dimensions de développement de la pensée informatique d'un étudiant du cadre de Brennan et Resnick (2012) : les étapes 1 et 2 aux concepts informatiques, les étapes 2 à 4 aux pratiques informatiques et les étapes 3 et 4 aux perspectives informatiques. Et c'est à l'étape 4 que nous soutenons que l'étudiant s'est approprié la programmation comme un instrument pour les mathématiques « comme le feraient les mathématiciens » (à la fois en termes de pratiques et de perspectives informatiques), ce que nous nommons « la programmation en tant qu'instrument de pensée informatique pour les mathématiques ».

III. PROCHAINES ÉTAPES DE NOTRE RECHERCHE

Dans cet article, nous avons présenté le cadre théorique sur lequel s'appuie notre recherche, centré sur la façon dont les étudiants de premier cycle universitaire en mathématiques en viennent à s'approprier la programmation en tant qu'instrument de pensée informatique pour les mathématiques. Brennan et Resnick (2012) suggèrent des façons d'évaluer le développement de la pensée informatique d'un apprenant, y compris l'analyse d'entrevues et de porte-folio de projets. Par conséquent, dans le cadre de notre recherche, nous collecterons les projets de mathématiques axés sur la programmation des étudiants (14

au total sur les trois cours), ainsi que leurs journaux de bord. leur travail en cours. Des entrevues finales et des questionnaires seront utilisés à la fin des études du programme de quatre ou cinq ans des participants, afin d'examiner la durabilité de leur utilisation de la programmation. Conformément à la recommandation de Trouche (2004), des entrevues semi-structurées avec les instructeurs de cours, des notes de terrain sur les observations durant les séances en laboratoire informatique et le matériel de cours permettront de mieux comprendre les objectifs didactiques des instructeurs et l'environnement d'apprentissage des participants. Ces dernières données éclaireront également les décisions pédagogiques des enseignants et dans quelle mesure elles sont en accord avec le paradigme constructionniste.

RÉFÉRENCES

Abrahamson, D., Berland, M., Shapiro, B., Unterman, J., & Wilensky, U. (2004). Leveraging epistemological diversity through computer-based argumentation in the domain of probability. *For the Learning of Mathematics*, *26*(3), 19-45.

Artigue, M. (2002). Learning mathematics in a CAS environment: The genesis of a reflection about instrumentation and the dialectics between technical and conceptual work. *International Journal of Computers for Mathematical Learning*, *7*(3), 245-274.

Assude, T. (2007). Teachers' practices and degree of ICT integration. In D. Pitta-Pantazi & G. N. Philippou (Eds.), *Actes du 5ième congrès de l' European Society for Research in Mathematics Education* (pp. 1339-1348). Larnaka, Cyprus: Department of Education, University of Cyprus.

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., & Engelhardt, K. (2016). Developing computational thinking in compulsory education: Implications for policy and practice. *EU Science Hub*. Retrieved from https://ec.europa.eu/jrc/en/printpdf/175911

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *Actes de l'American Educational Research Association (AERA) annual conference.* Retrieved from http://web.media.mit.edu/~kbrennan/files/Brennan_Resnick_AERA2012_CT.pdf

Broley, L., Buteau, C., & Muller, E. (2017). *(Legitimate peripheral) computational thinking in mathematics*. Proceedings of the Congress of European Society for Research in Mathematics Education (CERME), Dublin (Ireland), February 2017.

Buteau, C., & Muller, E. (2014). Teaching roles in a technology intensive core undergraduate mathematics course. In A. Clark-Wilson, O. Robutti, & N. Sinclair (Eds.), *The mathematics teacher in the digital era: An international perspective on technology focused professional development* (pp. 163-185). Dordrecht, Netherlands: Springer.

Buteau, C., Muller, E., Marshall, N., Sacristán, A. I., & Mgombelo, J. (2016). Undergraduate mathematics students appropriating programming as a tool for modelling, simulation, and visualization: A case study. *Digital Experience in Mathematics Education*, *2*(2), 142-156.

Buteau, C., Muller, E., & Ralph, B. (2015, June). Integration of programming in the undergraduate mathematics program at Brock University. In *Online Proceedings of Math+Coding Symposium*, London, ON.

Centre de recherches mathématiques. (2016). *Computational mathematics in emerging applications*. Retrieved from http://www.crm.umontreal.ca/act/theme/theme_2016_1_fr/index.php

Computational Thinking in Mathematics Education. (n.d.). *About*. Retrieved from http://ctmath.ca/about/

Cook, L. S., Smagorinsky, P., Fry, P. G., Konopak, B., & Moore, C. (2002). Problems in developing a constructivist approach to teaching: One teacher's transition from teacher preparation to teaching. *The Elementary School Journal*, *102*(5), 389-413.

European Mathematical Society. (2011). *Position paper on the European Commission's contributions to European research.* Retrieved from http://ec.europa.eu/research/horizon2020/pdf/contributions/post/european_organisations/european_mathematical_society.pdf

Gadanidis, G., Hughes, J. M., Minniti, L., & White, B. J. (2017). Computational thinking, grade 1 students and the binomial theorem. *Digital Experiences in Mathematics Education*, *3*(2), 77-96.

Goos, M., & Soury-Lavergne, S. (2010). Teachers and teaching: Theoretical perspectives and classroom implementation. In C. Hoyles & J.-B. Lagrange (Eds.), *ICMI Study 17, technology revisited, ICMI study series* (pp. 311-328). New York, NY: Springer.

Grover, S., & Pea, R. (2013). Computational thinking in K-12: A review of the state of the field. *Educational Researcher*, *42*(1), 38-43. doi:10.3102/0013189X12463051

Guin, D., & Trouche, L. (1999). The complex process of converting tools into mathematical instruments. The case of calculators. *International Journal of Computers for Mathematical Learning*, *3*(3), 195-227.

Hoadley, C. (2012). What is a community of practice and how can we support it? in D. H. Jonassen & S. M. Land (Eds.), Theoretical foundations of learning environments (2nd Ed.) (287-300) New York: Routledge.

Kafai, Y. B., & Resnick, M. (1996). *Constructionism in practice: Designing, thinking, and learning in a digital world*. Mahwah, NJ: Erlbaum, Routledge.

Lave, J., & Wenger, E. (1991). *Situated learning: Legitimate peripheral participation*. New York, NY: Cambridge University Press.

Leron, U., & Dubinsky, E. (1995). An abstract algebra story. *American Mathematical Monthly*, *102*(3), 227-242.

Muller, E., Buteau, C., Ralph, B., & Mgombelo, J. (2009). Learning mathematics through the design and implementation of exploratory and learning objects. *International Journal for Technology in Mathematics Education*, *63*(2), 63-73.

Noss, R., & Hoyles, C. (1996). *Windows on mathematical meanings: Learning cultures and computers* (Vol. 17). Dordrecht, Netherlands: Kluwer.

Papert, S. (1971). Teaching children to be mathematicians vs. teaching about mathematics. *Artificial Intelligence Memo* No. 249. Retrieved from http://hdl.handle.net/1721.1/5837

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

Papert, S. (1980b). Computer-based microworlds as incubators for powerful ideas. In R. Taylor (Ed.), The computer in the school: tutor, tool, tutee (pp. 203–210). New York: Teacher's College Press.

Papert, S., & Harel, I. (1991). Situating constructionisn. In S. Papert & I. Harel (Eds.), *Constructionism* (pp. 1-12). Norwood, NJ: Ablex.

Rabardel, P. (1995/2002). *Les hommes et les technologies; approche cognitives des instruments contemporains.* Paris, France: Armand Colin.

Trouche, L. (2004). Managing complexity of human/machine interactions in computerized learning environments: Guiding students' command process through instrumental orchestrations. *International Journal of Computers for Mathematical Learning*, *9*, 281-307.

Trouche, L., & Drijvers, P. (2010). Handheld technology for mathematics education: Flashback into the future. *ZDM: The International Journal on Mathematics Education*, *42*, 667-681.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal for Science Education and Technology*, *25*, 127-147.

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A*, 366(1881), 3717-3725.

Wing, J. M. (2014, January 9). Computational thinking benefits society. *Social Issues in Computing*, 40th Anniversary Blog, University of Toronto. Retrieved from http://socialissues.cs.toronto.edu/index.html%3Fp=279.html