Teacher education, students' autonomy and digital technologies: A case study about programming with Scratch

Ghislaine Gueudet, Sophie Joffredo-Lebrun

Teacher education, students' autonomy and digital technologies: A case study about programming with Scratch

GHISLAINE GUEUDET, SOPHIE JOFFREDO-LE BRUN

CREAD, INSPE de Bretagne
University of Brest
France
ghislaine.gueudet@inspe-bretagne.fr

CREAD/LIRFE
Catholic University of the West
France
sjoffred@uco.fr

Abstract

How can future mathematics teachers be trained to combine the use of digital technologies with student autonomy? Referring to the documentational approach to didactics, we have designed and implemented an initial training course based on collective documentation work. Here we analyze the work of a team of trainees who designed a session on programming with Scratch, in terms of the potential of the lesson designed for the use of Scratch and student autonomy, and the professional knowledge mobilised. The analysis of this case highlights possibilities and limitations of such a training.

Keywords

Digital technology, documentation work, programming, students' autonomy, teacher education

Résumé

Comment former de futurs enseignants de mathématiques à combiner usage des technologies numériques et autonomie des élèves ? En nous inscrivant dans le cadre de l'approche documentaire du didactique, nous avons conçu et implémenté une

formation initiale basée sur le travail documentaire collectif. Nous analysons ici le travail d'une équipe de stagiaires qui ont conçu une séance sur la programmation avec Scratch, en termes de potentiel de la leçon conçue pour l'emploi de Scratch et l'autonomie des élèves, et de connaissances professionnelles mobilisées. L'analyse de ce cas met en évidence des possibilités et des limites d'une telle formation.

Mots-Clés
Technologies numériques, programmation informatique, travail documentaire, autonomie des élèves, formation des enseignants

Introduction

How can a preservice teacher education program support the development of classroom uses combining digital technologies use and students' autonomy? What are the difficulties faced, when designing and implementing a preservice teacher education program with this aim?

We studied such questions within the project IDEE in France (in English, Digital Interactions for Teaching and Education). In this project we considered various kinds of technologies (we use the term ICT): videos, collaboration tools, but also mathematical software like Dynamic Geometry Software or programming software. We have designed and tested a preservice teacher education program in France; we implemented it in 2019 for prospective secondary school mathematics teachers (PMTs). In this teacher education program, teams of PMTs design and implement lessons combining ICT use and students' autonomy. In this paper we focus on the case of a team of PMTs who designed a lesson about programming with Scratch (part of the mathematics curriculum in France).

We firstly present the central theory framing our research: the documentational approach to didactics (DAD, Gueudet & Trouche 2009). We briefly synthesize research works concerning students' autonomy, which lead us to propose different categories of autonomy. Moreover we also review related works concerning the use of Scratch, and its possible links with students' autonomy. Then we present the initial teacher education program we designed, and the methods we used to follow its implementation. We describe the case studied in this paper: a team of PMTs and the lesson they designed about Scratch. We discuss our results, concerning the documentation work of these PMTs and its outcomes. Finally we present our conclusions concerning the case studied and what we learned from this case about preservice mathematics teacher education, with the aim of combining ICT use and students' autonomy.

Theoretical tools and related works

The documentational approach

The documentational approach to didactics (DAD, Gueudet & Trouche, 2009) focuses on the interactions between teachers and resources and on the outcomes of these interactions. Following Adler (2000), a resource is defined as anything likely to re-source the teacher's practice. It can be a textbook, a software package, but also something said by a student in class, for example. DAD has been developed in a context of an abundance of resources available for teachers, in particular digital resources on the internet. In this new context, a theoretical approach considering the interactions between teachers and sets of resources comprising traditional resources like textbooks and diverse kinds of digital resources (mathematical software, videos, digital platforms etc.) was needed.

The documentational approach draws on the instrumental approach, introduced in the field of cognitive ergonomics by Rabardel (1995). Like the instrumental approach, DAD is rooted in activity theory (Vygotsky, 1978) and draws on the theory of conceptual fields (Vergnaud, 1998), using in particular the concept of scheme (inherited from Piaget, 1975). It studies the teachers' goal-directed activity, focusing on the teachers' interactions with resources. A given goal (e.g., "introduce the students to the use of bounded loops") defines a class of activity situations (Rabardel & Bourmaud, 2003): a set of situations with the same goal. The activity of the teacher is mediated by sets of resources (e.g., Scratch and a textbook presenting lessons with Scratch). During the activity for different situations of the same class with a set of resources, the teacher develops a *document*. A document associates two different components: resources modified and recombined by the teacher, and a scheme of use of these resources. This process is called a documentational genesis.

The concept of scheme introduced by Vergnaud (1998) describes how subjects simultaneously act and learn, in a perspective of conceptualization-in-action. A scheme is a stable organization of the subject's activity, for a given goal. It comprises four components:

- Goals and subgoals (e.g., "Introduce grade 7 students to the use of bounded loops").
- Rules of action (e.g., "propose a situation where the same action has to be repeated").
- Operational invariants: theorems-in-action and associated concepts-in-action. A theorem-in-action is a proposition considered as true (e.g., "students will notice that using bounded loops can avoid lengthy repetitions of the same instructions"); a concept-in-action is a concept considered as relevant.
- Possibilities of inferences, to adapt to the specific features of the situation.

The operational invariants (OIs) are the most important part of the scheme; they guide

and orient the teacher's activity. Theorems-in-action (TiAs) and concepts-in-action are always associated; in our analyses we focus on TiAs.

The documentational geneses are composed of two interrelated processes: instrumentation, the features of the resources that shape the development of OIs; and instrumentalization, the already-existing OIs of the teacher lead him/her to adapt and modify the resources. Documentational geneses play a central role in teacher professional development.

This DAD theoretical perspective led us to design a particular preservice teacher education course, proposing the collective design of lessons (see § "Presentation of the teacher education program"). This collective documentation work leads to the development by the PMTs of documents, associated resources and schemes of use of these resources (in particular professional knowledge, conceptualized here as OIs). The aim of our study is to answer to the following research questions:

- RQ1. What is the potential of a lesson designed by PMTs who followed the training program, in terms of students' autonomy and ICT use?
- RQ2. What can we infer from the features of the lesson and from the presentation of this lesson, in terms of professional knowledge of the PMTs that intervened during their documentation work?

In this article, for the sake of brevity, we have chosen to focus on the case of a team of PMTs who designed a lesson about Scratch for grade 7. Thus the first research question is rephrased as:

- RQ'1. What is the potential of a lesson designed by PMTs who followed the training program, in terms of students' autonomy and programming with Scratch?

Studying these questions requires a precise view of students' autonomy, and of the possible links between programming with Scratch and students' autonomy. We present in the next subsections research literature that informed our study about these topics.

Students' autonomy

Little (1994) defines autonomy as "a capacity – for detachment, critical reflection, decision-making, and independent action" (ibid. p. 4); with this perspective, autonomy is a particular kind of relation between a learner and his/her learning processes. Yackel and Cobb (1996) propose that autonomous students refer to their own capacities, while the heteronomous ones search for an external authority. This does not mean that the autonomous work is individual: under certain conditions, students' collective work can support the development of students' autonomy (Ben-Zvi & Sfard, 2007; Wood, 2016). This socio-cultural view of autonomy is also present in the work of Brousseau (1997), who introduces the concept of an a-didactical situation, defined

as a mathematical situation where the student develops his/her knowledge through interaction with a milieu to solve a mathematical problem. The milieu is composed of mathematical objects, other students, material objects, etc. Autonomous learning in a-didactical situations results from these interactions with the milieu.

Drawing on these works, we defined students' autonomy in the IDEE project as "a process situated in a context and in a system of interactions, which permits the student to organize his/her work and to mobilize internal or external resources to achieve a given task" (Gueudet & Lebaud, 2019, p. 5).

Different dimensions of autonomy are identified in educational research (associated with different theoretical perspectives). Some of these works refer to self-regulation or self-regulated learning (SRL, see e.g. Zimmermann, 1989). They emphasize specific aspects of SRL, for example Beizhuizen and Steffens (2011) contend that: "In many models, self-regulation is depicted as a cyclic process involving three stages: (1) goal setting, (2) monitoring processes and strategies, (3) self-evaluation". We also consider these aspects as important, but introduce other categories.

We observed that, while some authors studying autonomous learning consider general elements of students' activity (like doing homework, searching information), others focus on the subject-specific activity of students (mathematics in our study, including programming). This led us to introduce a distinction between *transversal autonomy* and *mathematical autonomy*. We also stress that mathematical autonomy can concern a-didactical situations (Brousseau 1997), problem-solving and the discovery of new knowledge: we term this *mathematical discovery autonomy*. Nevertheless, it can also concern procedural fluency and the mobilization of previous knowledge: we term this *mathematical mobilization autonomy*.

Scratch, mathematics and students' autonomy

Programming was introduced in the secondary school curricula of many countries between 2010 and 2020 (see e.g. Misfeldt et al., 2020; Modeste, 2015). In some countries it has been introduced as a specific discipline and was taught by computer science education teachers; in others (France in particular) it has been inserted in the mathematics curricula and was taught by mathematics teachers. This curriculum choice raises the issue of the links between programming and mathematics. Misfeldt et al. (2020) identified four possible types of relations between mathematics and programming in the enacted curriculum of different countries: "(1) specific relations to mathematical concepts or processes [..]; (2) explicit relations to mathematics [..]; (3) implicit relations to mathematics, [..]; and (4) no or weak relations to mathematics." (ibid. p. 259).

In other countries, programming (with Scratch in particular) is taught as a discipline in its own right, and studied by computer education researchers. Meerbaum-Salant,

Armoni and Ben-Ari (2010) investigated the teaching and learning of computer science concepts with Scratch at a middle school in Israel. The curriculum mentioned in particular the following concepts: Initialization; Bounded loops; Conditional loops; Message passing; Variables; Concurrency. The authors designed and evaluated material to support what we would call with our DAD perspective the teachers' documentation work. They observed the potential of relevant classroom activities with Scratch to learn these concepts, but noticed at the same time that the concepts of variables and concurrency are especially difficult for students.

Other studies evidence the potential of Scratch for students' autonomy. The features of Scratch allow a classroom use inscribed in a constructionist perspective (Maloney, 2010; Olabe et al., 2011). We recall briefly here that the constructionist perspective suggests creating student-centered learning situations and meaningful projects (Papert & Harel, 1991), in particular through the engagement of students in programming. Scratch users without prior experience can learn basic notions of programming, since this software provides immediate feedback and makes the program execution visible (Maloney et al., 2010). The Scratch language allows overcoming obstacles and problems in an iterative process: students can implement trial-and-error strategies. The effects of a script are directly visible to students on their screen, thus they can become able to link the elements of their script with what is happening on the screen (Gaio, 2017), and become aware of their mistakes. Moreover the same program can be realized with different scripts. Thus Scratch can be used to support the development of self-directed learning (Olabe et al., 2011).

Working with Scratch in class can also provide opportunities for students' collective work (Maloney et al., 2010). Breed, Mentz and Van der Westhuizen (2014) showed that when two students work together on programming tasks, the first can code while the second observes, identifies problems and proposes improvements. The interactions within the pair would allow for better learning and greater confidence in the students' work. Thus Scratch can be used to support the collective dimension in students' autonomy: deciding collectively whether a result is right or wrong, exchanging with other students to receive help performing a task collectively, building a strategy (Gueudet & Lebaud 2019).

Following the implementation of a teacher education program: methods

Presentation of the teacher education program

In France PMTs undertake a Master's degree. During the second year of their Master's, they have two classes in a school and in parallel courses in the teacher training institute. The training was designed to be integrated in this second year of the Master's; it was

tested in 2019 with a group of around 30 PMTs. It comprised three two-hour sessions (in person), the content of which is presented in the table below (Table 1), and used a Moodle platform for offering resources and collecting the lesson plans designed.

Table 1

Description of the training

	Content/Objectives
Session 1 (2h)	Presentation of the training and of the Moodle platform. Watching (individually) videos about autonomy, differentiation, collective work, social equity. Report and debate on the videos. Presentation of an analysis tool (potential of a lesson plan in terms of digital technologies and students' autonomy), the PMTs analyze examples of lesson plans with the tool, discussion. Constitution of teams for the design of lessons, choice of a theme.
Session 2 (2h)	Design of the lesson by the teams.
Between sessions 2 and 3	The lesson plan is tested in at least one class. The lesson plan and a report about the test are uploaded on the platform. Each team reads the lesson plan of another team.
Session 3 (2h)	Report on each lesson tested, discussion, plans for an improved version.

During the first session, PMTs watched some videos (designed by the IDEE research team) about students' autonomy and digital technologies; they also received an analysis tool, designed in the IDEE project for analyzing the potential of a lesson plan in terms of students' autonomy and ICT use (Gueudet & Lebaud 2019). This analysis tool contained 4 categories (each with 6 to 10 criteria):

1. The lesson plan is clear and complete.
2. There is a rich mathematical content in the activity.
3. The use of digital technologies is relevant for the objective of the lesson.
4. The activity supports transversal autonomy.

They used it to evaluate examples of lesson plans, in terms of these four categories.

During the second session, teams of PMTs designed a lesson, which was tested as far as possible in the class of at least one member of the team. The analysis tool was then used as guidelines for this design. The lesson was presented and discussed during the third session.

A study case

In the 2019 implementation of the program, 7 teams of PMTs were formed. Five of these 7 teams tested their lesson in class. They uploaded to Moodle their lesson plan, and some elements related with the test in class. During the third and last session of the training, the teams presented the implementation of their lesson which was then discussed with the group. The different choices were debated, and possibilities for improvement were envisaged. We have chosen to focus in this article on the case of a team of PMTs (team E in what follows) who designed a lesson about Scratch for grade 7. Our analyses in terms of documentational geneses, OIs in particular, require a level of detail which prevents developing several cases. For team E we have a full set of data (some of the teams declined to be video-recorded); moreover, as presented above (§ "Scratch, mathematics and students' autonomy"), the use of Scratch to promote students' autonomy is well documented in the literature.

This team comprised 4 members, 3 males and 1 female. Two of them were teaching in grade 6 and two others in grade 7. They designed a lesson entitled "diagnostic activity of the software Scratch" for grade 7, situated in a geometrical context. This lesson was implemented in two 7th-grade classes by the same teacher (named SE1 in what follows). In the first class, in a half class with teacher SE1; and in the other class, in a full class with SE1 and a colleague. In both cases the students worked in pairs.

Data collected

We have collected several types of data for this qualitative study.

First, we have downloaded all the elements that team E uploaded to Moodle. They consist of 5 parts:

- The description and the stages of the lesson;
- Some notes about the students' work during the two sessions;
- Possible directions for improvement;
- The worksheet proposed to the students (its translation in English is in the Appendix);
- The summary of a textbook extract (Boullis, 2017) proposing progressive levels for the work with Scratch.

Second, we video-recorded and transcribed their presentation during session 3 of the training. Questions were asked by the teacher educators (authors of this paper) and other PMTs. The teacher educators' questions dealt more specifically with implementation modalities, concerning students' autonomy and ICT use. The objective was to encourage PMTs to reflect on these aspects.

Analysis of the data

For answering RQ'1 about the potential of a lesson designed by PMTs who followed the training program, in terms of students' autonomy and programming with Scratch, we coded the resources designed by the PMTs and their presentation of the implementation in class regarding the categories introduced above: mathematical mobilization autonomy, mathematical discovery autonomy and transversal autonomy. We considered both the lesson designed and the students' activity described by the PMTs when they presented the implementation of this lesson.

For answering RQ2 about professional knowledge, we coded in the PMTs' declarations (presenting the lesson of answering questions), utterances that we interpret as possible TiAs (propositions considered as true), and linked them with the resources designed by the PMTs. Indeed with a DAD perspective, the students' TiAs interact with the resources they use or design. This association of TiAs and resources also informs us about instrumentalization and instrumentation processes.

Results

In this section, we first examine the place of programming in the French curriculum at lower secondary school and we present the lesson designed by team E. Then, we analyze the potential of team E's lesson, in terms of students' autonomy and Scratch use. Third, analysing the lesson and its presentation by the PMTs, we infer professional knowledge (in terms of TiAs) of the PMTs that guided the choice of these features.

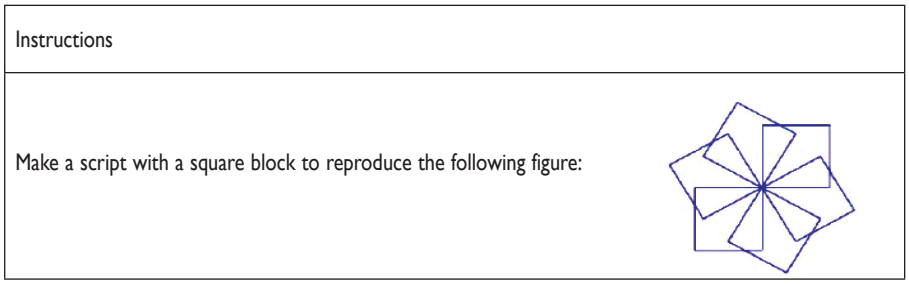
Algorithmics in the French curriculum and the lesson designed by team E

Since 2015, the French official curriculum (Gueudet et al., 2017) in mathematics includes algorithmic instruction from grade 6 to grade 9. It follows a first introduction to programming, "in particular through the programming of movements or the construction of figures" (Ministry of Education, 2018) from grade 4 to grade 5. For grade 7, "students are introduced to event programming. Gradually, they develop new skills, by programming parallel actions, using the notion of computer variables, discovering loops and conditional instructions that complete event-related control structures" (ibid.). More generally, the curriculum is organised according to six mathematical competencies: Searching, Modelling, Reasoning, Computing, Communicating, Representing.

Team E designed a lesson where Scratch is used to draw geometrical figures. This could be relevant for teaching geometry (Foerster, 2016); nevertheless, the focus here is mostly on programming. According to the classification proposed by Misfeldt et al. (2020), this lesson belongs to type 4: "no or weak relations with mathematics". The lesson was planned for 55 minutes. The students should achieve 8 consecutive and progressive tasks which are distributed as follows (see the students' sheet in the Appendix):

- Task 1 (level A): commands required for any program. Writing in Scratch a sequence of basic commands given in common language: place the sprite on a given position, orient it, put the pen in writing position, erase, show the elf and then at the end hide it.
- Tasks 2-4 (level B): programming simple algorithms. Draw a segment (task 2); draw a square using successive segments and rotations (task 3); shorten the previous program by using a loop to draw a square (task 4).
- Tasks 5-8: programming complex algorithms. Create a block to draw a square (task 5); reproduce three squares using this block (task 6); draw a six-squares rosette (task 7); draw a 24-squares rosette using different colors (see the example of task 6 on Figure 1 below).

FIGURE 1



Task 6

The PMTs indicated in the lesson plan that this activity is linked with two mathematical competencies of the official curriculum: searching and reasoning.

After launching and configuring the Scratch software and distributing the worksheets to the students, the teacher explained that each task must be validated by him/her before they can proceed to the next step. At the end of the session, the students returned to the teacher their worksheet on which they have colored the different validated steps.

Analysis of the lesson in terms of student autonomy and Scratch use

We analyze here the lesson designed by team E, regarding students' autonomy and Scratch use. We recall that programming, in the context of the French curriculum, is considered part of mathematics. Moreover this lesson includes some geometrical content; hence "mathematical" means here "concerning programming and/or geometry".

Mathematical mobilization autonomy

Concerning Scratch, the students mobilized their previously-learned knowledge on Scratch commands and simple algorithms.

Nevertheless, there are differences between the 8 tasks that students must perform. In task 1, an algorithm is written as a text; moreover all the commands are displayed next to it on the students' sheet. The students only need to order correctly the given commands in Scratch, following the order of the text. Even a student new to Scratch should be able to perform this task. For tasks 2 and 3 the students only need to add a new command in the program written for task 1. The additional command is mentioned on the students' sheet; moreover this sheet is projected on a whiteboard, evidencing the colors of the different commands. For task 2 the students have to identify that a number must be written in the white circle of the Scratch command, they must decide how much progress they need to make (70) for the command "advance from *x*". For task 3 similarly they have to indicate (90°) for the command "turn by *x* degrees".

FIGURE 2

Instructions	New command required
Complete the previous program to draw a square with side 70	

Task 3

Concerning geometry, the students have to mobilize previous knowledge concerning the square (in particular the 90-degree angles for the square). This could correspond to mobilization autonomy; nevertheless, in the first part of the session, the lesson plan indicated that the teacher write on the board reminders about the angles. Thus this autonomy is limited; nevertheless, if a student chose a wrong angle (60 degrees, for example), testing his/her program on Scratch will evidence that the result is not a square: the software could support the autonomy.

Then in task 4, the required autonomy increases with an introduction of loop: the students must decide how many times to repeat "drawing a segment, then rotating" to obtain a square. They have probably already met loops, so we consider this as mobilization autonomy. They can make several attempts in their program and observe the result; thus in this case the software supports this autonomy.

Mathematical discovery autonomy

Concerning geometry, for tasks 7 and 8, students have to compute the appropriate rotation angle for inserting it in the Scratch command to draw rings of squares. Also, they have to learn new knowledge about rotations: the lesson requires mathematical discovery autonomy. Once again, if a student chooses a wrong angle, the figure displayed can contribute to an autonomous correction of the value to be inserted in the Scratch command.

Concerning programming with Scratch, task 5 (Figure 3) requires discovery autonomy to create and use a block. Scratch is a blocked-based software (Weintrop & Wilensky 2015). The available blocks can support students' autonomy (in particular for properly associating several blocks); the ability to design personalized blocks can be considered as a new step in students' autonomy with Scratch, but the notion of block is very abstract. Here the block would more or less encapsulate the loop previously written to draw a square. The usefulness of this encapsulation is unclear, since the whole program remains quite short. SE1 explained during the presentation that the students encountered difficulties, thus he had been required at this moment to give explanations to the whole class.

FIGURE 3

Instructions	New command required
Create a «square» block that will shorten the previous program	

Task 5

In task 6, students must write a script with a square block to produce three successive squares (see Figure 1 above). The lesson plan specified that no new commands are required. Students were asked to reason and find out how to arrange the commands they have seen previously. SE1 explained that "the idea is to let them do as they want at first by multiplying the commands if they want". But during the validation, the teacher explained the need to use a loop and a block: students were not encouraged to make different attempts.

Transversal autonomy

Concerning transversal autonomy and particularly the students' collective work, the lesson plan indicated "Autonomous work by pairs on the computers". Working in pairs on the computers using the activity sheet could promote students' autonomy but PMTs explained that this choice was made to reduce the time needed for the validation by the teacher. If their program does not produce the expected result, the students do not have access to specific help. Even when the program produces the expected result, they cannot work at their own pace since they have to call the teacher each time they complete one task.

For each category of autonomy, we observed a real potential of the lesson designed. Nevertheless this potential is limited by choices made by the PMTs: adding some

indications; asking the students to call the teacher for validation; under-exploiting the potential of Scratch for collective work.

These features of the lesson are outcomes of team E's documentation work. We consider them as consequences and indicators of instrumentalization and instrumentation processes.

Documentation work and professional knowledge

During the presentation of their lesson plan, team E presented a resource which played a central role in designing the content of this lesson. This resource is an algorithmics textbook "Myriade" (Boullis, 2017) which offers 5 sequences to learn algorithms with 3 levels of difficulty. SE1 explained that their project was to design a differentiated algorithmics teaching based on "Myriade" and that this lesson was designed as a diagnostic evaluation. As a consequence, the lesson plan as described above presented different levels that the students must carry out successively as suggested in this textbook.

As mentioned above, in task 1 the students only need to write in Scratch basic commands, presented on their sheet as images in a random order. Then from tasks 2 to 5, one command must be added to the construction program. The students' worksheet clearly indicates the new command to be used. For example, in task 3 (Figure 2) students must draw a square. The instruction is as follows: "complete the previous program to draw a square of 70". It then indicates the command to be inserted in the program. In their presentation, the PMTs justified their choice: "we tell them to use just one new command (…). Each time use an additional command that is quite simple". Moreover, in task 4 a loop is introduced; in task 5 the students have to create a block. This corresponds to the learning trajectory proposed by the textbook. For tasks 6 to 8, SE1 said that new commands were not introduced "We're going to ask them to "reason" by trying to find out how to arrange the different commands introduced in the previous tasks".

We can say that the characteristics of the "Myriade" textbook influenced the team's choices in the design of their lesson on Scratch. It proposed a learning trajectory in terms of successive tasks (Confrey, Maloney, & Corley, 2014) and inspired their lesson. Firstly ordering given commands; then adding one command; programming a loop; programming a block; finally reasoning on the possible use of the previous commands. We hypothesize that members of this team developed these TiAs (propositions considered as true, Vergnaud, 1998) concerning the learning trajectory with Scratch because "students have to start with elementary commands, then loops, then blocks" and "learning programming follows a global aim of writing the shortest possible program" in an instrumentation process.

Secondly, team E has chosen to design this lesson as a diagnostic evaluation. They

intended after this lesson to propose problem-solving sessions with Scratch. During this lesson, their aim was to observe each student's achievement with Scratch, in order to incorporate later differentiation in their problem-solving sessions. For this reason, the members emphasized the need for the teacher's validation of each step. The importance of this validation is reiterated in the instructions to be given to students in the lesson plan. Moreover, the PMTs highlighted several times during their presentation the need for this validation by the teacher. They said, for example, that "students' programs must be audited very regularly in the classroom" and "it is not possible to evaluate them outside the classroom". We interpret this as TiAs and hypothesize that they developed these TiAs during previous lessons with ICT. The team members explained that the organization of the students in pairs facilitates the validation of the programs by the teacher, adding that "the organization was planned to make pairs because we had anticipated that it would be very time-consuming for the teacher". Assessing the students' achievement with Scratch is for them a central objective here. They rejected the possibility of self-evaluation, declaring that "self-evaluation is not possible, because the students would be cheating". For this reason, while the students were working in pairs, they did not exploit the potential of Scratch for collaboration. Moreover, the validation by the teacher hindered the possibility for the students to work at their own pace.

They insisted that this validation by the teacher allowed them to help the students but also to guide them in the design of their program. SE1 said, "during the validation you tell them no, now you have to do it with a loop".

In this example, we observe an instrumentalization process: the PMTs' already-existing TiAs guided their documentation work. They wanted to assess the students' achievement with Scratch. They rejected self-evaluation, because they believed that the students would cheat. As a consequence, the potential of Scratch for self-evaluation (Maloney et al., 2010) was under-exploited in this lesson. They also considered that examining all the programs out of class would be too time-consuming. So they chose to organize the students by pairs and to ask them to call the teacher before each change of task.

Thirdly, we infer that PMTs developed a new TiA about Scratch. Indeed, for task 5, all pairs of students encountered difficulties for creating a block. SE1 was forced to intervene for the whole class because the pairs were not able to discover autonomously this new command. During session 3 of the training, the PMTs seemed to agree that "the teacher has to help the students to discover how to define and use a block on Scratch".

Conclusions

In this conclusion we return to our research questions; then we discuss how the results go beyond the case of team E and raise prospects for future work.

Our first research question was:

RQ'1. What is the potential of a lesson designed by PMTs who followed the training program, in terms of students' autonomy and programming with Scratch?

Concerning the features of the lesson designed by team E in terms of Scratch use and students' autonomy, we observed some possibilities but also limitations.

Team E designed a learning path of increasing difficulty in terms of programming with Scratch. The sequence of tasks they proposed corresponded indeed to increasing difficulty of Scratch commands (Meerbaum-Salant et al., 2010). It also led students to mobilize previous geometrical knowledge and to develop new knowledge about rotations.

Nevertheless, it did not fully exploit the potential of Scratch for students' autonomy. Their lesson did not correspond to a problem-solving perspective of Scratch use (Olabe et al., 2011). The students can have feedback from Scratch (Maloney et al., 2010), but they were required to ask the teacher to validate each task, so they could work at their own pace. They worked in pairs, but the potential of Scratch for collaboration was not exploited in this lesson (Breed et al., 2014). Moreover they did not anticipate the difficulty in creating a block.

Our second research question was: RQ2. What can we infer from the features of the lesson and from the presentation of this lesson, in terms of professional knowledge of the PMTs that intervened during their documentation work?

Concerning professional knowledge, the training allowed the PMTs to reflect on the notion of autonomy and to consider different categories and criteria to be taken into account in their lesson plan (for example, the category "transversal autonomy" present in the analysis grid). We observed that some TiAs were developed during the training, in particular about the learning trajectory and the difficulty of creating a block. Some already-developed TiAs constituted an obstacle for the aim of the training. For example, the TiA, 'the students will cheat if we propose self-evaluation', hindered the possibility for students to work at their own pace, since they have to call the teacher for validating the tasks.

What do we learn from this case study, in terms of preservice teacher education programs supporting ICT use and student autonomy? Some of the features of team E's lesson that hindered students' autonomy, and the potential of Scratch for this autonomy, are likely to be present for all PMTs (and were actually present in the other lessons, though these are not analyzed here). These features include: the guidance proposed by the teacher, preventing the proposition of multiple solutions; the choice to avoid self-evaluation; and the under-exploitation of collective work come from OIs which are likely to be shared by all PMTs. Nevertheless, we noted some positive aspects which we

consider signs of an increased awareness concerning students' autonomy: students can implement trial-and-error strategies, for example to find the correct angles, thanks to Scratch feedback. The design and test of the lesson contributed to team E's professional development: they chose a textbook proposing progressive levels (corresponding to one of the criteria of mathematical autonomy, in the grid provided during the training), and used it to design progressive tasks. They observed that asking to design a block was a difficult task, and that they needed to plan some hints in advance for this task, but also more generally.

Because of these positive aspects, we consider that this case study confirms the potential of a teacher education program proposing PMTs' collective documentation work, and offering resources to support this documentation work (Joffredo-Lebrun & Gueudet 2021). This training could certainly be improved: the time allocated to it is short, the potential of students' collective work, for example, could be the focus of a full two-hour session, not only one of many aspects. Letting the PMTs select the software they want to use in their lesson can support their engagement in the training, but does not enable deepening the reflection about the specific potential of a precise software (e.g. Scratch) for student autonomy.

In 2020, because of the COVID-19 pandemic the training had to be distant and the PMTs were only asked to evaluate lesson plans, using the analysis grid - nevertheless we received enthusiastic feedback about this activity. In 2021, we hope a new complete version of the training will be implemented, allowing us to collect and analyze new data. In the COVID-19 pandemic context, combining ICT use and students' autonomy is more crucial than ever, and research in mathematics education has to produce results supporting the design of relevant teacher education programs.

References

Adler, J. (2000). Conceptualising resources as a theme for teacher education. *Journal of Mathematics Teacher Education, 3*, 205-224.

Beizhuizen, J., & Steffens, K. (2011). A conceptual framework for research on self-regulated learning. In R. Carneiro, P. Lefrere, K. Steffens & J. Underwood (Eds.), *Self-regulated learning in technology enhanced learning environments* (pp. 3-20). Rotterdam: Sense Publishers.

Ben-Zvi, D., & Sfard, A. (2007). Ariadne's thread, Daedalus' wings and the learner's autonomy. *Éducation & Didactique, 1*(3), 117-134.

Boullis, M. (2017). *Myriade Cahier d'algorithmique Cycle 4*. Paris: Bordas.

Breed, B., Mentz, E., & Van der Westhuizen, G. J. (2014). A metacognitive approach to pair programming: Influence on metacognitive awareness. *Electronic Journal of Research in Educational Psychology, 12*, 33-60.

Brousseau, G. (1997). *Theory of Didactical Situations in Mathematics.* Dordrecht: Kluwer Academic Publishers.

Confrey, J., Maloney, A. P., & Corley, A. K. (2014). Learning trajectories: A framework for connecting standards with curriculum. *ZDM, 46*(5), 719733.

Foerster, K.-T. (2016). Integrating programming into the Mathematics Curriculum: Combining Scratch and Geometry in Grades 6 and 7. In *Proceedings of the 17th Annual Conference on Information Technology Education – SIGITE, 16* (pp. 91-96). New York, United States: Association for Computing Machinery.

Gaio, A. (2017). Programming for 3rd graders, scratch-bases or unplugged? *Paper presented at the Congress of European Research in Mathematics Education (CERME)*, Dublin, 1-5 February.

Gueudet, G., Bueno-Ravel, L., Modeste, S., & Trouche, L. (2017). Curriculum in France: A national frame in transition. In D. Thompson, M. A. Huntley & C. Suurtamm (Eds.), *International Perspectives on Mathematics Curriculum*, (pp. 41-70). Charlotte: International Age Publishing.

Gueudet, G., & Lebaud, M.-P. (2019). Développer l'autonomie des élèves en mathématiques grâce au numérique. 2. Analyser le potentiel de ressources pour les professeurs. *Petit x, 110*, 85-102.

Gueudet, G., & Trouche, L. (2009). Towards new documentation systems for teachers? *Educational Studies in Mathematics, 71*(3), 199-218.

Joffredo-Lebrun, S., & Gueudet, G. (2021). *Students' autonomy and digital technologies: collective documentation work in pre-service teacher education*. Long communication at the ICME14 conference, Shangaï.

Little, D. (1994). Learner autonomy: A theoretical construct and its practical application. *Die Neueren Sprachen, 93*(5), 430-442.

Maloney, J., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. (2010). The Scratch programming language and environment. *ACM, Transactions on Computing Education, 10*(4), 1-15.

Meerbaum-Salant, O., Armoni, M., & Ben-Ari, M. (2010). Learning computer science concepts with Scratch. In *Proceedings of the Sixth International Workshop on Computing Education Research - ICER '10*, (pp. 69-76). New York, United States: Association for Computing Machinery.

Ministry of Education, (2018). *Programme du cycle 4*. Retrieved from https://cache.media.eduscol.education.fr/file/programmes_2018/20/4/Cycle_4_programme_consolide_1038204.pdf.

Misfeldt, M., Jankvist, U. T., Gerianou, E., & Bråting, K. (2020). *Relations between mathematics and programming in school: Juxtaposing three different cases*. In A. Donevska-Todorova, E. Faggiano, J. Trgalova, Z. Lavicza, R. Weinhandl, A. Clark-Wilson & H.-G. Weigand (Eds.), *Proceedings of the MEDA2020 Conference* (pp. 255-262). Austria: Linz University.

Modeste, S. (2015). Impact of informatics on mathematics and its teaching. On the importance of epistemological analysis to feed didactical research. In F. Gadducci & M. Tavosanis (Eds.), *History and Philosophy of Computing, Series: IFIP Advances in Information and Communication Technology* (Vol. 487, pp. 243-255). Cham, Switzerland: Springer.

Olabe, J. C., Basogain, X., Olabe, M. A., Maiz, I., & Castaño, C. (2011). Programming and robotics with scratch in primary education. In A. Méndez-Vilas (Ed.), *Education in a technological world: Communicating current and emerging research and technological efforts* (pp. 356- 363). Badajoz, Spain: Formatex Research Center.

Papert, S., & Harel, I. (1991). Situating constructionism. In I. Harel & S. Papert (Eds.), *Constructionism* (pp. 1-11). New York, NY: Ablex. Retrieved from http://www.papert.org/articles/Situating Constructionism.html

Piaget J. (1975). *L'Équilibration des structures cognitives: Problème central du développement*. Paris: Presses Universitaires de France.

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

Rabardel, P., & Bourmaud, G. (2003). From computer to instrument system: A developmental perspective. In P. Rabardel & Y. Waern (Eds.), *Special Issue "From Computer Artifact to Mediated Activity", Part 1: Organisational Issues, Interacting With Computers, 15*(5), 665-691.

Vergnaud, G. (1998). Toward a cognitive theory of practice. In A. Sierpinska & J. Kilpatrick (Eds.), *Mathematics education as a research domain: A search for identity* (pp. 227-241). Dordrecht: Kluwer Academic Publisher.

Vygotsky, L. (1978). *Mind in society*. Cambridge: Harvard University Press.

Weintrop, D., & Wilensky, U. (2015). The challenges of studying Blocks-based Programming Environments. In *2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond)*, 57. Retrieved from https://ccl.northwestern.edu/2015/p005-weintrop-2.pdf.

Wood, M. B. (2016). Rituals and right answers: Barriers and supports to autonomous activity. *Educational Studies in Mathematics*, *91*(3), 327348.

Yackel, E., & Cobb, P. (1996). Sociomathematical norms, argumentation, and autonomy in mathematics. *Journal for Research in Mathematics Education*, *27*(4), 58-477.

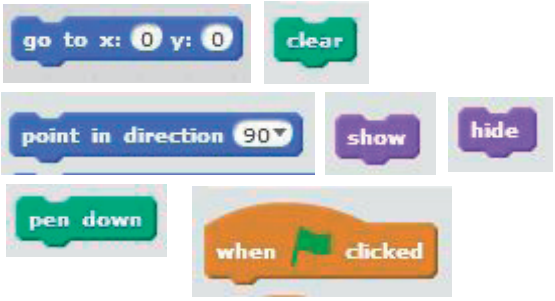
Zimmerman, B. J. (1989). A social cognitive view of self-regulated academic learning. *Journal of Educational Psychology, 81*(3), 329-339.

Appendix. Students' sheet proposed by team E

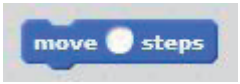
Programming with Scratch

Novice	Beginner	Apprentice	Interme-diate	Confirmed	Advanced	Expert	Master

Task 1, novice level: I know how to start a Scratch construction program

Instructions	New commands required
Write the following script in Scratch - When the green flag is clicked - Go to 0 ; 0 - Orient yourself at 90°. - Pen in writing position - Delete all - Show -..... (Instructions) - Hide	

Task 2, beginner level: I know how to draw a segment

Instructions	New command required
Complete the previous program to draw a segment of length 70	

Task 3, apprentice level: I know how to draw a square

Instructions	New command required
Complete the previous program to draw a square with side 70	

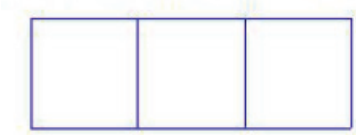
Task 4, intermediate Level: I know how to use a loop

Instructions	New command required
Modify and shorten the previous program using a loop.	

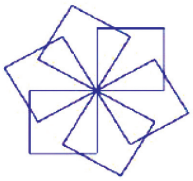
Task 5, confirmed level: I know how to create a block

Instructions	New command required
Create a "square" block that will shorten the previous program	 

Task 6, advanced level: I can reproduce 3 squares consecutively

Instructions	New command required
Make a script with a square block to reproduce the following figure: 	(No new commands)

Task 7, expert level: I can reproduce a 6-square rosette.

Instructions	New command required
Make a script with a square block to reproduce the following figure: 	(No new commands)

Task 8, master level: I can reproduce a 24 square rosette by changing the color between each square.

Instructions	New command required
Make a script with a square block to reproduce the following figure: (change the color of the pen at each iteration).