An Authoring Tool based on Semi-automatic Generators for Creating Self-assessment Exercises

Nathalie Guin[a] and Marie Lefevre[b]
Univ. Lyon, UCBL, CNRS, INSA Lyon, LIRIS, UMR5205, F-69622 Villeurbanne, France

Keywords: Self-assessment, Semi-automatic Generation of Exercises, Authoring Tool, University Context.

Abstract: This article presents ASKER, a tool for teachers to create and disseminate self-assessment exercises for their students. Currently used in the first year of a bachelor's degree at the University of Lyon (France), it enables students to carry out exercises in order to evaluate their acquisition of concepts considered important by the teacher. ASKER enables the creation of exercises (matching, grouping, short open-ended questions, multiple choice questions) that can be used to assess learning in many different fields. To create exercises to assess a concept, the teacher defines a model of exercises that will enable the generation of various exercises, using text or image resources. Such an exercise model is based on constraints that the exercises created from this model must comply with. Automatic generators create, from the resources defined by the teacher, many exercises respecting these constraints. The possibility for the learner to request the generation of several exercises from the same model enables her to assess herself several times on the same concept, without the teacher having to repeatedly define many exercises.

1 INTRODUCTION

The purpose of the work described in this article is to enable a learner to assess her knowledge within a learning path for which a teacher defines learning objectives. Our approach is to enable the teacher to provide the learner with exercises on the concepts to be acquired. The learner can use these exercises if she wishes to evaluate her mastery of the concepts involved in the exercises. We therefore place ourselves here in the context of a *formative* evaluation.

Self-assessment with immediate feedback requires activities or features that allow students to assess themselves against the course objectives. Exercisers (or exercise generators) are a way to quickly diagnose the skills acquired, to perform performance memorization and skill development through trial and error learning based on repetition (Mostow *et al.*, 2004). Since the exerciser environment keeps track of the learning activity and provides immediate feedback, it facilitates the learner's regulation by allowing explicit reflection on the skills worked on (Steffens, 2006).

We propose an authoring tool that gives the teacher the freedom to set the notions on which learners can assess themselves, and that enables him or her to create exercises to assess the mastery of these notions. In order to meet the needs of as many people as possible, we have chosen to consider types of exercises that can be applied to many fields, such as MCQ (Multipe Choice Questions), matching, grouping, ordering, gap texts and so on.

The following section explains why we have chosen an approach based on exercise generators to address this issue of creating self-assessment exercises. We then specify the knowledge acquisition processes necessary for this approach, before presenting the architecture of the tool we have developed: ASKER (Authoring tool for aSsessing Knowledge genErating exeRcises). Finally, after having shown how this tool can be used by both teachers and learners, we carry out an evaluation of the use of ASKER in first-year bachelor's degree courses at the University of Lyon.

[a] https://orcid.org/0000-0001-9999-9878
[b] https://orcid.org/0000-0002-2360-8727

Guin, N. and Lefevre, M.
An Authoring Tool based on Semi-automatic Generators for Creating Self-assessment Exercises.
DOI: 10.5220/0010996100003182
In *Proceedings of the 14th International Conference on Computer Supported Education (CSEDU 2022) - Volume 1*, pages 181-188
ISBN: 978-989-758-562-3; ISSN: 2184-5026

2 AN AUTHORING TOOL BASED ON EXERCISE GENERATORS

Our aim was to enable a teacher to offer learners online self-assessment exercises. Such exercises enable learners to autonomously test their level of mastery of what they have learned in the course, whether this course is online or face-to-face. Learners may fail in initial attempts to respond to exercises if knowledge is not mastered. It is therefore possible that a learner may have to answer several times to the same exercise before achieving success. In order for the learner not to be influenced by her previous resolutions, it is necessary that the self-assessment exercises be different from one time to another, while evaluating the same knowledge. However, it does not seem reasonable to ask the author to write many versions of the exercise. That's why we have chosen to use exercise generators that the author can easily use in any field. However, we designed a semi-automatic process of generating exercises, in order to let teachers take part in the choice of the criteria that the exercises will have to meet.

To meet the needs of teachers and learners, the expected properties of the authoring tool were as follows:

- The exercise is different from one time to another.
- The author is in control of the content of the exercise and is assured that it meets his or her expectations in terms of educational content.
- Exercise generators can be used in a wide range of fields and grade levels.
- The diagnosis of the response is made automatically and in real time.
- The construction of exercises with generators is a time saving for the author compared to a creation exercise by exercise.
- The creation of exercises does not require technical skills.

Many researches have studied the question of authoring tools in the field of Technology Enhanced Learning, and several literature reviews about the topic were published (Murray, 1999) (Murray, 2003) (Woolf, 2010) (Dermeval *et al*., 2016). In these works, the objective is to create Intelligent Tutoring Systems. For our part, we only want to enable the creation of training and self-assessment exercises, which is why we focus our study on exercise generators. As we also want our authoring tool to be usable in any field, we have not integrated a model of the knowledge and skills to be assessed. Indeed, our goal is that a teacher can directly use the tool to create exercises without the need for a prior knowledge modeling phase. We thus agree with Baker's approach (Baker, 2016), which, noting that ITS used at scale are not the most intelligent ones, proposes to design simpler tools that support teachers' activity by enhancing their expertise.

Existing exercise generators can be categorized into three categories: manual, automatic and semi-automatic.

Often used as part of authoring tools, **manual generators** give a great deal of freedom to the author, which precisely defines the content of the exercise and all the formatting options. Some commercially available authoring tools such as Articulate Quizmaker[1] or Hot Potatoes[2] are commonly used to create paper-based or computer-based exercises. The online learning platforms Claroline[3] and Moodle[4], commonly used in higher education, also offer their own exercise editing tools. With this type of generators, the author is guaranteed to have an exercise that precisely matches his/her expectations, which meets one of our needs. On the other hand, the author must create each instance of exercise one by one, specifying its contents. This type of generator is not able to automatically create a large number of exercises assessing the same skill.

Automatic generators do have this ability, but unfortunately leave little space for the author in the creation process. With this kind of generator, a large number of exercises are created automatically without the author being able to influence the system's choices. He or she can simply choose the category of the exercise (form, theme, knowledge addressed) but cannot act on the content or on specific constraints. Examples include the Reading Tutor generator (Mostow *et al*., 2004), or the Aplusix generator (Bouhineau *et al*., 2008).

In order to take advantage of the features of the automatic generators while leaving to the author the editorial freedom on the exercises created, we have chosen **semi-automatic generators**, which combine the advantages of the two previous categories. These generators propose that the author define a model of exercises, which is then instantiated to produce a large number of exercises (Jean-Daubias and Guin, 2009) (Delozanne *et al*., 2003). Some e-learning platforms like Moodle, Wims[5], WeBWorK[6] or (Auzende *et al*., 2007) offer exercises involving

[1] https://articulate.com/360/studio#quizmaker
[2] http://hotpot.uvic.ca/
[3] https://www.claroline.com/
[4] http://moodle.org
[5] https://wims.univ-cotedazur.fr/wims/
[6] http://webwork.maa.org/

variables that are similar to the concept of an exercise model. This type of generator partially meets our needs but is limited to the areas requiring calculation. They are used for fields such as mathematics and science, and not all of them are accessible to non-programmer authors.

In order to have semi-automatic generators adapted to many domains and including other types of exercises, we have chosen to use the GEPPETO approach (Lefevre *et al.*, 2012). This approach consists of enabling the teacher to express constraints on the exercises to be generated. To do this, it is necessary to have a model of the types of exercises that can be generated, in order to know the type of constraints that the teacher must be able to express. The following section thus specifies the models guiding the acquisition of the knowledge necessary to generate exercises according to the GEPPETO approach.

3 ACQUISITION OF KNOWLEDGE FOR GENERATING EXERCISES

Figure 1 presents the GEPPETO approach (GEneric models and processes to Personalize learners' PEdagogical activities according to Teaching Objectives). This approach enables the acquisition of knowledge at several levels, from experts and teachers, to generate exercises.

In GEPPETO, a meta-meta-model of exercises was defined by the research team (Lefevre et al., 2009). This model specifies the knowledge that an expert will need to define in order to create a meta-model of exercises of a given type (see A in Figure 1), for example a meta-model of exercises of the MCQ type, or of the matching type.

This meta-model of exercises of type X or Y then enables the teacher to specify constraints defining a model of exercises (cf. B in Figure 1). Depending on the type of exercises, the constraints will not be the same, so that's why the exercise meta-model is needed. For example, the teacher could use the meta-model of the MCQ-type exercises to define a model of MCQ-type exercises covering a given subject and containing N questions, with for each M propositions with only one correct answer.

Using such exercise models, exercise generators can construct several exercises conforming to these models (see C in Figure 1). The exercise generators able to use these models depend on the type of exercises, and therefore on the meta-model of exercises of type X or Y.

It can therefore be seen that the GEPPETO approach requires two knowledge acquisition processes:

- Acquiring the expert's knowledge for the creation of meta-models of exercises (see A in Figure 1). This acquisition is done only once for each type of exercises and is based on the meta-meta-model of exercises.
- The acquisition of the teacher's knowledge for the creation of the exercises models (see B in Figure 1). This acquisition is carried out on a regular basis for the construction of various exercise models and requires an interface that is based on the meta-model of exercises of a given type (the constraints that the teacher must define depend on the type of exercises).

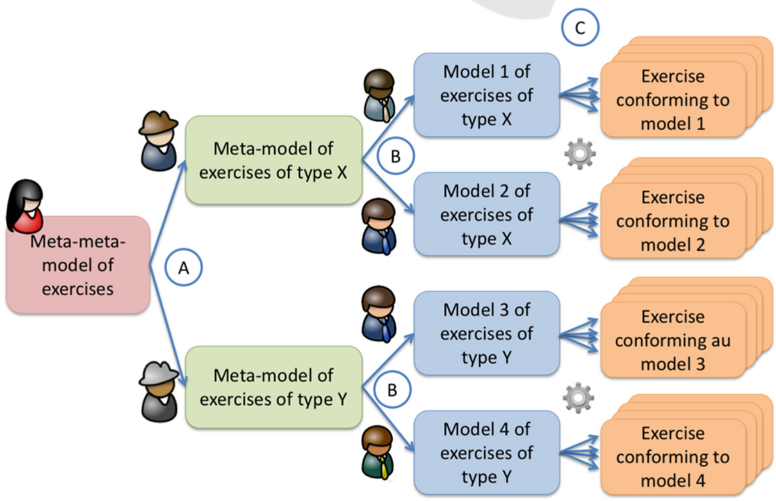

Figure 1: The GEPPETO approach.

Meta-models and the meta-meta-model are independent of the field for which an exercise will be generated. For example, the meta-model *Identification of text parts* specifies how to formulate the guidelines, how to characterize the different text-based resources, and how to describe the actions to be carried out on these resources to generate exercises.

An exercise generator is associated to each of these meta-models. An interface associated with the generator and based on the meta-model enables the teacher to define constraints on the exercises to be generated. It is at this point, when creating the model of exercises (for example, a gap text where it is necessary to put the verbs in the past), that the application to a field and a level of study will be carried out. The exercises generated from the exercise model are therefore, of course, dependent on the field.

Since the meta-models are all consistent with the meta-meta-model, all the exercise generators share a common architecture (Lefevre et al., 2009).

4 ASKER TOOL ARCHITECTURE

The GEPPETO approach was designed to create paper-pencil exercises. To design an authoring tool to provide learners with online exercises and immediate diagnosis, we chose to follow the same approach. Our research hypothesis was that using constraints on the exercises to be generated allowed both to obtain a sufficient variety of exercises for learners to train and self-assess, while requiring less work for the author teacher.

Thus, in the ASKER authoring tool, we have chosen a model inspired by the GEPPETO approach: after having chosen a type of exercise (MCQ, matching, gap text...), the author can create an exercise model that describes the content and form of the exercises he or she wants to create, but without necessarily constraining it completely. By exploiting this model, an exercise generator is able to provide the learner with a large number of different exercises evaluating the same skill. Each exercise instance generated in this way will be interactive: the learner will respond online and obtain a diagnosis of her response.

The types of exercises we selected are: identification of parts of the text (includes gap text), organization of elements (ordering, grouping, matching), annotation of illustrations, Multiple Choice Questionnaire, open and short-ended questionnaire.

Figure 2 shows the architecture of the ASKER authoring tool. The upper block is made up of the different levels of representation of the exercises, resulting from the GEPPETO approach: the meta-models of types of exercises, the models of exercises, and the instances of exercises. In the central block are the three mechanisms that manipulate these representations of exercises. The lower block contains the resources used in the exercise creation process.

Resources are the basic elements that are used to build exercises, for example texts, images or element sequences. Each resource has metadata characterizing it to facilitate searches (theme, level, etc.) as well as metadata enriching the resource to define exercises, such as image captions or annotations on image areas. For example, the author can define image type resources, with flag images, and define for each flag image its country, its capital city, and its continent.

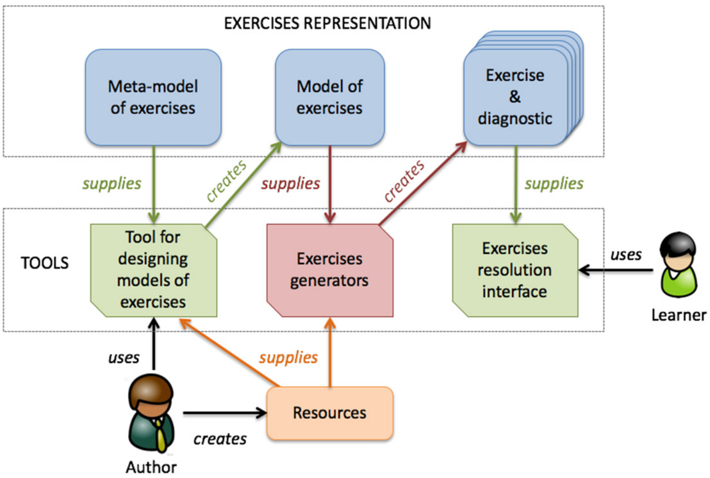

Figure 2: ASKER tool architecture.

The author chooses a type of exercises T (for example matching). Let's call MMT the meta-model of the T-type exercises described by an expert. The author creates a model of exercises (let's call it MT) compliant with MMT using a dedicated interface based on the knowledge contained in MMT about the type of exercises T. This MT model defines a set of constraints that the exercises resulting from this model must respect. For example, it is a 5 pair matching exercise, choosing flag images, and matching each image with its "country" metadata. The author can generate some instances exercises of MT (let's call them ExoT) to test if the model does give rise to the expected exercises

The T-type exercises generator associated with MMT (so here the matching exercises generator) therefore receives an input model of exercises MT that it instantiates to produce an output ExoT exercise. The ExoT exercise is consistent with the MT model and therefore with the choices of the author who created it. The generator requires no human intervention. It has all the necessary information in the MT exercise model and makes use of resources (here the flag images). The generator is used whenever you want to obtain a new ExoT instance of exercise (which contains the diagnosis) from the MT model.

The exercise is then presented to the learner via a resolution tool that formats the exercise, gathers the learner's answer and provides a diagnosis. In our example, the generated exercise will propose 5 flags to the student, and the student will have to find the country of each flag. The variety of exercises generated will therefore come in this example from the amount of flag images available in the resources. The same applies to exercises using texts. Variety can also come from the use of formulas using variables whose values must be chosen within an interval defined by the author. The resources can be used for different exercise models. For example, we could define an exercise model where the capital associated with each flag must be found. Or a categorization exercise where flags must be put in boxes corresponding to their continent.

5 THE ASKER TOOL

The ASKER (Authoring tool for aSsessing Knowledge genErating exeRcises) platform can be used on the one hand by a teacher to create models of exercises, and on the other hand by learners to carry out exercises generated from these models and to obtain a diagnosis for self-assessment.

An Authoring Tool for the Teacher. ASKER currently enables a teacher to create models of exercises of type MCQ, short open-ended questions, matching, grouping and ordering. The teacher can create resources such as texts, images, MCQ questions, short open-ended questions. On each of these resources, he or she can add meta-data that will be used by the exercise models using the resource. A resource can be used for several exercise models of different types. Thus, the same meta-data on a resource can be used for both a matching, grouping or ordering exercises.

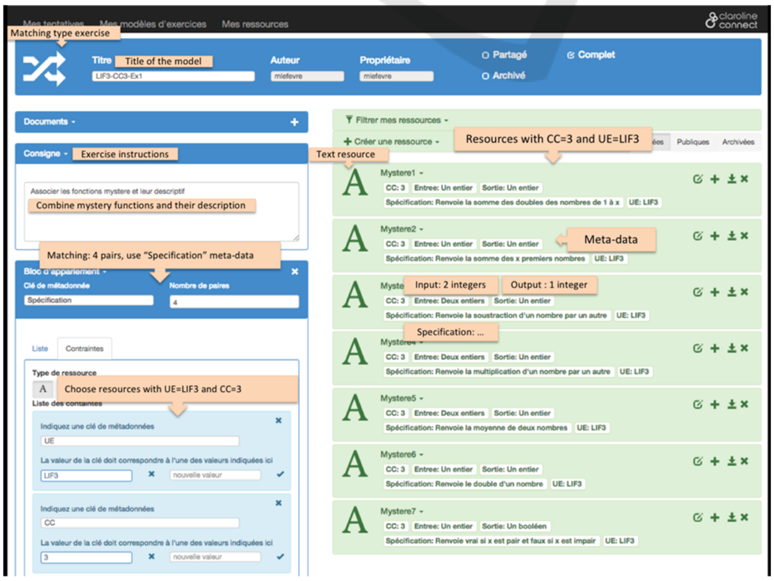

Figure 3: Author's interface for creating a model of matching exercises.

Figure 3 shows the author's interface for creating a model of matching exercises. The author selected text-based resources (right-hand side of Figure 3) and filtered out those for CC (chapter) number 3 of UE LIF3 (the name of the teaching unit). Several texts of functions in Scheme programming language can be used. On these texts, meta-data specifies the specification of each function, its input type(s) and its output type.

To create an exercise model, the author defines (left part of Figure 3) that he or she wishes to create exercises in which 4 functions must be associated with their specification. The exercises generator then uses this exercise model to create exercises that meet these constraints, using the available resources describing functions. Another model of exercises could use these same resources describing functions but asking to group them by input type or output type. In this way, the same resources can be used to produce another type of exercise and related to another notion of the course.

A Self-assessment Tool for the Learner. The teacher suggests to his or her students models of exercises corresponding to the concepts studied in class. For each exercise model, the learner can request the generation of several instances of exercises. She then resolves a first exercise derived from the exercise model, next the system diagnoses her answers and displays a feedback (in green and red) on her answers as well as the correct answer (in blue) to the exercise (see Figure 4).

A commentary prepared by the teacher explaining a common error or reminding an important concept may also be displayed. The learner can then revise the course if necessary and ask for a new exercise based on the same exercise model.

6 EVALUATION OF THE ASKER TOOL

At the University of Lyon, in the first year of a Computer Science degree, there are two initial courses in programming: one on imperative and iterative programming in C language, the other on functional and recursive programming in Scheme language.

We set up the use of the ASKER tool for the Scheme programming course several years ago. It is a use of ASKER for a teaching that is not digitized, the platform being a complementary tool to face-to-face teaching. We suggested that the students use ASKER to self-assess their understanding of the concepts presented in class before coming to the supervised works. This enables students to situate themselves in relation to the acquisition of the notions covered in courses, to prepare the assessments carried out each week in supervised works, and to diagnose their difficulties.

For this purpose, we have proposed a set of model exercises for each of the 9 lectures. By instantiating the meta-exercise models, we provided students with 18 models of matching exercises, 9 models of grouping exercises, and 8 models of MCQs. The creation of these 35 models and their 121 resources represented between 1 to 2 hours of work each week, during the 12 weeks of teaching, for the teacher in charge of the fall semester. This is a considerable workload, but it only concerned the initialization phase. These different models and their resources were then exploited and completed by the spring semester teacher in a more reasonable time: 1/2 hour per week. Since then, the use of ASKER in this course does not require any more time for the teacher.

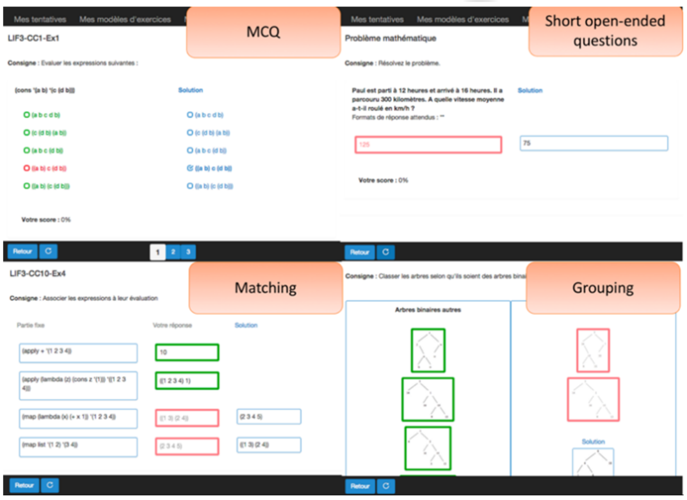

Figure 4: Feedback provided by ASKER to the learner on her resolution of several types of exercises.

We did not consider evaluating the quality of the exercises generated in terms of their impact on student learning. Indeed, our objective being to provide the teacher with a tool enabling her or him to generate varied exercises in sufficient number, we consider that the tool fulfils its objective if the exercises generated are in accordance with the teachers' expectations, which is the case with our approach of constraints defined by the teachers and satisfied by the generators.

We then introduced the use of ASKER in the other first year course, devoted to algorithms and programming in C. The teacher has created 37 exercise models, divided into 6 chapters. Of the 584 students enrolled in the course, 485 (83%) used ASKER at least once. For each model planned by the teacher, the number of students who did at least one exercise from this model was around 46%.

In order to measure the impact of the platform on students, we could not for ethical reasons conduct a comparative experiment, giving access to ASKER only to a part of the students. A questionnaire was distributed at the end of the semester to students enrolled in the course. We received 106 responses from students who used the platform as this:

- 67% of students did exercises each week, 23% every other week, 9% at the beginning of the semester but no more afterwards.
- At each use, 43% used it between 5 and 10 minutes, 43% between 10 and 20 minutes.
- 82% of students generated multiple instances of exercises from the same model.
- Students mainly used it to study before the supervised works sessions (83%). They each time made all the models of exercises proposed for a chapter (82%).
- Students reported that ASKER helped them to understand the course (58%) and to identify concepts not understood (63%). 89% of them think that using ASKER has enabled them to progress in this course (70% a little and 19% a lot).

Although this is not an *evidence* of ASKER's impact on learning, students think that this tool has had a positive impact on their learning. Students are not under any obligation to use ASKER during their first year. Their use of the tool is in no way used to evaluate their work. The tool is just available if they want to use it and many use it all year round. Usage statistics and student comments also show that this tool increases their motivation to work, which in itself is already a very satisfying result.

ASKER has also been used by 3 physics and chemistry teachers at high school. These teachers used to offer their students paper-based exercises to enable them to self-assess certain skills. They wanted to use ASKER to produce similar exercises that would give their students immediate feedback on their answers. This immediate feedback was a demand for 75% of their students.

After a while of hands-on learning, these teachers were able to use ASKER to create 44 exercise models for their students. They have made extensive use of image-type resources. They appreciated the opportunity to have a competency used in various situations (due to the variety of resources) and to create exercises involving different cognitive tasks (due to differing types of exercises). Using ASKER also gave them the idea of new exercises compared to those they used on paper. Their students loved the application, and in particular the immediate feedback. They have asked to be able to use ASKER in all chapters of their Physics and Chemistry courses.

To meet the self-assessment requirements that we formulated in Section 2, our tool had to have the following properties:

- *The exercise is different from one time to another*. This property is satisfied thanks to the generators using constraints set in the exercises models. The practice shows that students actually do several exercises for each exercise model (82% of them).
- *The author is in control of the content of the exercise and is assured that it meets his or her expectations.* This property is satisfied with the authoring tool that enables the teacher to create an exercise model specifying the constraints that the exercises must satisfy, and enable him/her to control the exercises generated from each model.
- *Exercise generators can be used in a wide range of fields and grade levels*. ASKER has been used in computer science, optics and anatomy at university, physics and chemistry at high school, but also to evaluate schoolchildrens' knowledge of countries around the world or to generate IQ tests based on logical sequences.
- *The diagnosis of the response is made automatically and in real time*. The models of exercises defined by teachers include the knowledge necessary for generators to diagnose student responses.
- *The construction of exercises with generators is a time saving for the author compared to a creation exercise by exercise*. Although the definition of exercise models takes time, especially at the beginning, teachers appreciate the variety of exercises generated by using annotated images and texts or calculation formulas. Creating such a variety of exercises without generators would take too much time.
- *The creation of exercises does not require technical skills*. In both programming courses, the teachers

who have used ASKER are computer scientists, but the use of ASKER does not require computer skills. In other uses, the authors were professors of physics, chemistry or optics. The latter has taken charge of this tool in complete autonomy.

To conclude, all the feedback from the use of ASKER in different contexts allows us to consider that the tool meets the needs of both teachers and learners.

7 CONCLUSION AND PROSPECTS

In this article we introduced ASKER, a tool that enables teachers to create self-assessment exercises for their students. This tool can be used for distance learning or as a complement to face-to-face teaching. It enables the creation of exercises (matching, groupings, short open-ended questions, MCQ) that can be used to evaluate learning in many fields. To create exercises to assess a concept, the teacher defines a model of exercises that will enable the generation of various exercises, using text or image resources. The possibility for the learner to request the generation of several exercises from the same model enables her to self-assess repeatedly on the same concepts, without the teacher having to repeatedly define many exercises.

Our research hypothesis was that using constraints on the exercises to be generated allowed both to obtain a sufficient variety of exercises for learners to train and self-assess, while requiring less work for the author teacher. The evaluation results, reported in Section 6, allow us to validate this research hypothesis.

ASKER is a tool that can be used in a variety of fields, and in a variety of learning contexts, at any level. It thus offers many possibilities of use. Its main limitation is that there is no explicit representation in ASKER of the knowledge to be learned. The acquisition of this knowledge therefore represents a major challenge. The main users of ASKER being the authors, it would be interesting for them to build the domain knowledge, as they already do for formulas. The system could assist them in this task by proposing a generalization of the information that they provide to create their models of exercises. We intend to use activity traces of teachers using ASKER to enable the system to assist them in this elicitation of domain knowledge.

We also envisage the use of a particular meta-data describing the skills mobilized by a model of exercise, so that we can propose to the learner an open profile of skills that will enable her to be more involved in her self-assessment process, for example by setting objectives to be achieved. Such skills profiles will also enable us to propose to the student a learning and training path that will enable her to achieve such objectives.

REFERENCES

Auzende, O., Giroire, H, Le Calvez, F. (2007). Extension of IMS-QTI to express constraints on template variables in mathematics exercises. In *13th International Conference on Artificial Intelligence in Education - AIED*, Los Angeles, USA, 524-526.

Baker, R.S. (2016). Stupid tutoring systems, intelligent humans. *International Journal of Artificial Intelligence in Education*, 26(2), 600–614.

Bouhineau, D., Chaachoua, H., Nicaud, J.F. (2008). Helping teachers generate exercises with random coefficients. *Int. journal of continuing engineering education and life-long learning* 18(5-6), 520–5337.

Delozanne, E., Grugeon, B., Prévit, D., Jacoboni, P. (2003). Supporting teachers when diagnosing their students in algebra. In *Workshop Advanced Technologies for Mathematics Education Artificial Intelligence in Education, AIED*, Sydney, Australia, 461–470.

Dermeval, D., Paiva, R., Bittencourt, I. et al. (2018). Authoring Tools for Designing Intelligent Tutoring Systems: a Systematic Review of the Literature. *Int. Journal of AI in Education* 28(3), 336-384.

Jean-Daubias, S., Guin, N. (2009). AMBRE-teacher: a module helping teachers to generate problems. In *2nd Workshop on Question Generation, AIED*, 43-47.

Lefevre, M., Jean-Daubias, S., Guin, N. (2009). Generation of pencil and paper exercises to personalize learners' work sequences: typology of exercises and meta-architecture for generators. *E-Learn* 2843-2848.

Lefevre, M., Guin, N., Jean-Daubias, S. (2012). A Generic Approach for Assisting Teachers During Personalization of Learners' Activities. *Workshop PALE, UMAP* 35-40

Murray, T. (1999). Authoring intelligent tutoring systems: an analysis of the state of the art. *International Journal of Artificial Intelligence in Education*, 10, 98–129.

Murray, T. (2003). An overview of intelligent tutoring system authoring tools: updated analysis of the state of the art. In *Authoring tools for advanced technology learning environments*, Springer, 491–544.

Mostow, J., Beck, J.-E., Bey, J., Cuneo, A., Sison, J., Tobin, B., Valeri, J. (2004). Using automated questions to assess reading comprehension, vocabulary and effects of tutorial interventions. *Technology, Instruction, Cognition and Learning*, Vol. 2, 97–134.

Steffens, K. (2006). Self-Regulated Learning in Technology-Enhanced Learning Environments: lessons of a European peer review. *European Journal of Education* 41, 353–379.

Woolf, B.P. (2010). *Building intelligent interactive tutors: student-centered strategies for revolutionizing e-learning*. Morgan Kaufmann.