RELATIONSHIP BETWEEN DESIGN AND USAGE OF EDUCATIONAL SOFTWARE: THE CASE OF APLUSIX

Jana Trgalova*, Hamid Chaachoua**
*National Institute for Pedagogical Research, Lyon and LIG, Grenoble, France
**Joseph Fourier University and LIG, Grenoble, France

In this contribution, we are interested in the design process of Aplusix, a microworld for the learning of algebra and in the impact of usages on this process. In the first part, we present general principles that seem to be guiding the overall design process of the system and the development of tree representation of algebraic expressions, which has been added recently. The second part is devoted to a design and implementation of a learning scenario involving Aplusix. Examples of impact of this empirical study on the software design choices are discussed.

***Key words**: Aplusix, algebra, tree representation, pedagogical scenario*

INTRODUCTION

The research reported in this paper is carried out in the framework of the ReMath project (http://remath.cti.gr) addressing the issue of using technologies in mathematics classes "*taking a 'learning through representing' approach and focusing on the didactical functionality of digital media*". The digital media at the core of this research is Aplusix, software designed to help students learn algebra. The work has been developed in three phases:

(1) *Design and implementation of a new representation of algebraic expressions.* During this phase, fundamental choices for a representation of expressions in a form of a tree were made collaboratively through interactions between computer scientists and didacticians of mathematics: on the one hand, computer scientists make sure that the new developments comply with general principles of the software, on the other hand, didacticians ensure that these choices are based on didactical and epistemological hypotheses. The choice of theoretical frameworks in both domains has an impact on functionalities of the tree representation. This design phase is presented in the following section.

(2) *Design of a pedagogical scenario*. Based on the choices made in the design phase, didacticians designed a pedagogical scenario to explore possible contributions of this new representation to the learning of algebra. The scenario has to take account of institutional constraints in order to implement it in ordinary classes. The design of scenario may lead to reconsidering certain choices concerning the new representation, or suggesting other. Such cases will be presented further in the paper.

(3) *Experimentation*. The scenario has been experimented in three different classes, which allowed validating underlying didactical hypotheses, as well as assessing the way students manipulate this new representation. This phase is discussed in the last part of the paper.

DESIGN AND DEVELOPMENT OF APLUSIX

When developing computer-based learning environments, designers need to make choices at the interface level and thus at the level of the internal universe of the environment. Thus pieces of knowledge implemented in such an environment will live not only under constraints of the didactical transposition (Chevallard 1985), but also under other constraints proper to the environment resulting from what Balacheff (1994) calls *computational transposition*. Thus, designers of computer-based learning environments have to respond to at least two types of requirements. First, they need to respect basic principles that are characteristic of the environment. The second type is related to the practice of the piece of knowledge in the institution in which it will be used.

Principles governing a design of software are not always made explicit and choices made are rarely explicitly linked to these principles. In what follows, we present a study carried out in an attempt to make explicit principles and choices that were guiding designers of Aplusix (aplusix.imag.fr), software for learning algebra, when they were developing tree representation of algebraic expressions.

General design principles of Aplusix

Aplusix software (Nicaud et al., 2003, 2004) has been developed since 1980s. A new mode of representation of algebraic expressions, a tree representation, is being added to this software. As was already mentioned above, the new developments must not affect the coherence of the whole software and thus have to comply with fundamental principles that guide the design and development of Aplusix. Three main design principles have been identified:

(1) *The student is free to write algebraic expressions*. This principle, influenced by research in the domain of interactive learning environments, considering mainly microworlds, resulted in the development of an editor of algebraic expressions and in the necessity to consider and deal with students' errors.

However, freedom in manipulating algebraic expressions is limited by constraining the selection of sub-expressions, based on the syntactic and semantic dimensions of expressions, which seems to be another important design principle and that can be formulated as follows:

(2) *In manipulating algebraic expressions, their syntactic and semantic dimensions are taken into account*. For example, given the expression 2+3x, it is not possible to select 2+3 as a sub-expression. This principle brings the idea of scaffolding since this choice aims at helping understand algebraic expressions and make their manipulation easier.

As regards the interaction between a student and a system, there are two modes of interaction: (1) a test mode in which the student does not get any feedback from the system, and (2) a training mode, in which a feedback is provided both in terms of

equivalence of a student's expression and the given one, and in terms of the correct end of the exercise. Thus the third principle is:

(3) *In a training mode, scaffolding should be provided by the system.* Scaffolding in the training mode requires taking decisions about validation of student's answers. It is important to clarify at this point that Aplusix recognizes 4 basic types of exercises: calculate, expand and simplify, factor and solve (equation, inequality or system of equations or inequalities). For these types of exercises, these decisions have been implemented. For example, for the "solve equation" exercise, it has been decided that the expression x = 2/4 will not be accepted as it is written in a non-simplified form, but will not be rejected either as it is not incorrect. Therefore a feedback message is sent to the student saying that the equation is almost solved.

Design and development of tree representation in Aplusix

The decision to implement a new representation system into the existing Aplusix software was taken in relation with the ReMath project focusing on representations of mathematical concepts in educational software. Two possibilities were considered: tree and graphical representations. The reasons for choosing the development of tree representation system are numerous (Bouhineau et al. 2007): (1) from an epistemological point of view, trees are natural representations of algebraic expressions; (2) from a didactical point of view, the introduction of a new register of representation would allow creating activities requiring an interplay between registers, which would enhance learning of algebraic expressions (Duval 1993); (3) from a point of view of computer science, trees are fundamental objects used to define data structures. Indeed, internal objects used in Aplusix to represent algebraic expressions and their visual properties are trees; (4) graphical representation of algebraic expressions is available in a few educational systems, while tree representation is scarcer.

Let us note first that the fundamental choices related to the tree representation were discussed during several meetings among developers (computer scientists and engineers) and didacticians.

Different modes of tree representation

The first idea was to develop the tree representation in a way that the student can see the articulation between the usual representation of an expression and a tree representing it: given an expression in a usual representation, a tree representation is provided progressively by the system, according to the student's command. A "mixed representation" mode has thus been designed where each leaf of a tree is a usual representation of an expression that can be expanded in a tree by clicking at the "+" button that appears when the mouse cursor is near a node; a tree, or a part of a tree, can be collapsed into a usual representation by clicking at the "-" button that appears when the mouse cursor is near a node. The developers considered this idea interesting from the learning point of view. However, it was in contradiction with the principle 1, according to which it was necessary to let the student edit freely a tree. The

development of a “free tree representation” mode, where the student can freely built trees, brought new difficulties the developers had to face: notion of erroneous operator, representation of parentheses, difficulties related to the “minus” sign, to the square root… These difficulties and the ways the developers have coped with them are described elsewhere (Trgalova and Chaachoua 2008).

Based on the principle 3, the developers wished to implement an editing mode providing scaffolding to the student. Design and implementation of scaffolding requires to define new kinds of exercises that would be recognized by the system and the means of validation of these exercises. We will discuss some of these choices below. It led also to the implementation of a “controlled tree representation” mode with constraints and scaffolding when a tree is edited: internal nodes must be operators and leaves must be numbers or variables. The arity of operators must be correct. In the current prototype of Aplusix, 3 modes of editing trees are thus available: free, controlled and mixed representations.

Choices of criteria for validating a student’s answer

According to the principle 3, when the student builds a tree in the free tree representation mode, the system should provide her/him with a feedback. Decisions about the conditions for a tree to be accepted as correct had to be taken and implemented. The student’s tree is compared with the expected one: (1) when, after normalisation of the minus signs (transformation of all minus signs in opposite), the trees are identical, then the student tree is accepted; (2) when the two trees differ only by commutation, the student’s tree is not accepted, but a specific message indicates that there is a problem with order; (3) when there is neither identity between the trees (case 1) nor commutation (case 2) but the two trees represent equivalent expressions, a message is generated indicating that the student’s tree is equivalent but not the expected one; (4) when there is no equivalence between expressions represented by the trees, another message is generated indicating that the answer is not correct.

These choices were made by one of the developers based on *fundamental issues* present in Aplusix such as *the notion of equivalence*, *the notion of commutation* and *of associativity*. They are considered as *a first stage choices* that can be discussed and analysed from the didactical point of view, both in terms of messages to be generated and of considering different cases of behaviour.

PEDAGOGICAL SCENARIO

Before presenting a pedagogical scenario we designed in order to validate design choices for the tree representation of expressions in Aplusix, we discuss some theoretical considerations that underpin the scenario.

According to Sfard (1991), mathematical notions can be conceived in two different ways: structurally as objects, and operationally as processes. An object conception of a notion focuses on its form while a process conception focuses on the dynamics of the notion. Algebraic expression, when conceived operationally, refers to a computational process. For example, the expression 5x-2 denotes a computational

process “multiply a number by 5, and then subtract 2”, which can be applied to numerical values. When an expression is conceived structurally, it refers to a set of objects on which operations can be performed. For example, 5x-2 denotes the result of the computational process applied to a number x. It also denotes a function that assigns the value 5x-2 to a variable x. Yet, in the French high school, the operational conception of algebraic expressions prevails in the teaching of algebra. Specific activities are needed to favour the distinction between these two conceptions of an algebraic expression. Examples of such activities are describing the expression in natural language, which requires considering the structure of the expression, or using tree representation of an expression, which highlights its form.

Semiotic representation is of major importance in any mathematical activity since mathematical concepts are accessible only by means of their representations. Duval (1995) calls “*register of representation*” any semiotic system allowing to perform three cognitive activities inherent to any representation: formation, treatment and conversion. These activities correspond to different cognitive processes and cause numerous difficulties in learning mathematics. Duval (2006) claims that while treatment tasks are more important from the mathematical point of view, conversion tasks are critical for the learning. Consequently, conceptualisation of mathematical notions requires manipulating of several registers for the same notion allowing to distinguish between a notion and its representations. As Duval (1993) says, the conceptualisation relies upon the articulation of at least two registers of representation, and this articulation manifests itself by rapidity and spontaneity of the cognitive activity of conversion between registers. Yet, school mathematics gives priority to teaching rules concerning both formation of semiotic representations and their treatment. The amount of activities of conversion between registers is negligible, although they represent cognitive activities that are the most difficult to grasp by students.

Motivated by these considerations, in the design of our pedagogical scenario, we decided to take into account three semiotic registers of representation of algebraic expressions: natural language register (NLR), usual register (UR) and tree register (TR) and to design activities of formation, treatment and conversion between these registers. The pedagogical scenario thus aims at helping the students grasp the structure of algebraic expressions by means of introducing TR and articulating it with UR and NLR. The following hypothesis underpins the scenario: the introduction of TR and its articulation with NLR and UR will have a positive impact on students’ mastering of the usual register of representation of algebraic expressions, which is the one taught in school algebra. The scenario is composed from 4 units: *pre-test*, *learning*, *assessing*, and *post-test* (cf. Table 1). The *pre-test* aimed at diagnosing students’ difficulties in algebra, especially those related to the structural aspect of expressions. On the other hand, the results of the pre-test compared to those of the post-test should provide us with evidence about the efficiency of the pedagogical scenario. Two kinds of activities are proposed in the pre-test: (1) classical school

algebra exercises (calculate, expand and simplify, factor), which are, in Duval's terms, treatment tasks in the register of usual representation, and (2) communication games between students proposing, in Duval's terms, activities of conversion between UR and NLR. The aim of the *learning* unit is to introduce the students to TR, a new register of representation of expressions, as well as to articulate it with the already familiar registers, namely NLR and UR. Then, conversion activities between TR and NLR and UR respectively are proposed. Most of the activities are to be done in a computer lab with Aplusix in the training mode. Eventually, simple tasks of treatment in TR are proposed to assess the mastery of the new register of representation by students. The unit called *assessing* aims at evaluating to what extent TR and conversion tasks between the registers are mastered by the students after having done activities of the *learning* unit. The evaluation is organized in the form of communication games between students similar to those from the pre-test, but this time, TR is involved in the tasks. In the *post-test*, tasks similar to those from the pre-test are proposed in order to enable a comparison of results. Confronting results obtained at the two tests should provide us with evidence confirming or not the underlying hypothesis.

	Activities	*Description*	*Environment*	*Duration*
Pre-test	Treatment in UR	Calculate, Factor Expand and simplify	Aplusix	50 min
	Conversion NLR ↔ UR	Communication games	Paper & pencil	30 min
Learning	Introduction to TR	Scenario TR introduction	Aplusix in video projection	55 min
	Conversion NLR ↔ TR	Conversion NLR → TR Conversion TR → NLR	Aplusix: controlled then free mode Paper & pencil	90 min
	Conversion UR ↔ TR	Conversion UR → TR Conversion TR → UR	Aplusix: controlled then free mode	80 min
	Treatment in TR	Calculate in TR Simplify in TR	Aplusix with second view	20 min
Ass.	Formation TR Conversion TR ↔ NLR (UR)	Communication games	Aplusix: free mode Paper & pencil	55 min
Post-test	Treatment in RU	Calculate, Factor Expand and simplify	Aplusix	30 min
	Conversion NLR ↔ UR	Communication games	Paper & pencil	20 min

Table 1. Structure of the pedagogical scenario.

EXPERIMENTATION

The scenario was proposed to 3 teachers with a possibility to adapt it to the constraints of their class. In this section, we present one of the experiments that took place in a Grade 10 class (15 years old students) in November 2007.

The pre-test revealed expected errors in treatment tasks within UR, in particular errors showing difficulties to take account of the structure of algebraic expressions, e.g., transforming 2+3x in 5x, and errors with handling powers and minus sign, e.g., transforming $3(-5)^2$ in -3×5^2 or in $\pm3^2\times5^2$. On the other hand, we were surprised by the results obtained in communication games. Algebraic expressions given in UR were described in NLR by the students, but with characteristics of an oral register, i.e., the students described actions allowing to obtain the initial expression (cf. Table 2). This register is based on language structure used to "read" an expression in UR. It presents two specificities: left-to-right reading and presence of implicit elements.

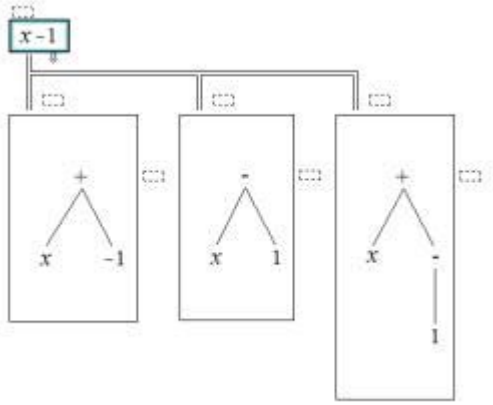
Expression given in UR	***Student emitting a message***		***Student receiving a message***	
	Register	*Examples of messages*	*Correct in UR*	*Wrong in RU*
$2x - y$	Oral (left-to-right)	"2 x minus y"	14	0
$2x - y^2$	Oral with ambiguity	"2 x minus y squared"	22	4
$\frac{(3x+2)(3x-1)}{a-(x+2)}$	Oral with brackets explicitly stated	"open a bracket, 3 x plus 2, close the bracket, open a bracket, 3 x minus 1, close the bracket, all this over a minus, open a bracket, x plus 2, close the bracket"	7	1
	Oral with brackets explicitly stated and with ambiguity	"open a bracket, 3 x plus 2, close the bracket, open a bracket, 3 x minus 1, close the bracket, over a minus, open a bracket, x plus 2, close the bracket"	19	1
Total			62	6

Table 2. Conversion from UR into NLR.

All messages result from the oral register and they accentuate operational aspect of the expressions rather than structural one. Moreover, more than 66% of messages are ambiguous. Despite of the ambiguities, most of pairs succeeded the game thanks to implicit codes of the oral register the students share and understand and which result from didactical contract (Brousseau 1997). Thus, the goal we assigned to the communication games, namely to lead students to become aware of the limits of the oral register they use in algebra, which does not take into account the structural aspect of expressions, was not achieved.

The learning unit started by an introductory session aiming at introducing tree representation to the students. The teacher asked one of the designers of the

pedagogical scenario to manage this session since he did not feel comfortable enough with the new representation implemented in the software although he uses Aplusix on a regular basis with his students. This introductory session allowed discussing with the students specificities of the tree representation of expressions and introducing vocabulary related to this new register (branch, leave, operator, argument…). Particular attention was paid to reading the expressions. Thus for example, the expression x+2y was read as "the sum of x and of the product of 2 by y", which accentuates the structure of the expression, instead of "x plus 2 y" highlighting its operational aspect. A particularity of the tree register residing in the fact that several different trees can represent a same algebraic expression was also discussed with the students based on the following example showing different meanings of "minus" sign (Fig. 1):



In the expression x-1, the minus sign can be conceived in three different ways leading to three different trees (this difference is hardly visible in UR):
- Sign of a negative number (tree on the left);
- Binary operator "difference" (tree in the middle);
- Unary operator "opposite number" (tree on the right).

Figure 1. Three different meanings of minus sign.

The rest of the scenario was shortened in order for the teacher to be in line with the global pedagogical program shared by all Grade 10 classes in the school. The teacher decided to individualize the implementation of the scenario according to the students in the following way: conversion NLR→TR and UR→TR in controlled mode only (only one group, denoted G1); conversion TR→NLR assigned as homework (whole class); treatment in TR optional (a few students with severe difficulties in algebra).

The G1 group was formed from rather low attaining students. The results obtained in the conversion tasks TR→NLR showed a significant difference between the two groups (cf. Table 3). These results can be considered as evidence proving efficiency of the work on conversion tasks NLR/UR→TR.

	Answer in NLR with structural aspect	*Answer in NLR with operational aspect*
G1 15 students having worked on conversion tasks with Aplusix in controlled mode	10	5
G2 15 students who have not benefited from the work on conversion tasks	3	12

Table 3. Students' answers to the conversion tasks TR→NLR.

As we mentioned above, the scenario, and thus the new prototype of Aplusix, had been tested in three classes. Feedbacks from students and teachers led the developers to re-examine some choices, which allowed some adaptations and improvements at the interface of Aplusix. Let us take the example of the "second view" functionality that enables visualizing a given algebraic expression represented in two registers at the same time. Initially, the second view displayed only a current step of the transformation. Observing the students using this functionality, we realized that when a student performs the next transformation step, the representation in the second view is updated and the student cannot observe the effects of the transformation in the second register. For this reason, the developers were asked to redesign this functionality in a way for the student to be able to observe the transformation s/he has performed in both registers. At present, the second view displays both current and previous steps.

CONCLUSION

The example of the design and implementation of tree representation of algebraic expressions presented in this contribution shows that the decision to introduce a new register of representation has been motivated by the didactical considerations about the necessity of being able to represent mathematical notions in at least two different registers. Considerations of different nature had an impact on the development of the new register: (1) taking account of a didactical dimension led to make choices allowing the implementation of tasks of conversion between registers, which seem to be essential for conceptual understanding of mathematical notions (Duval 1993); (2) taking account of users' feedback allowed to make some improvements at the interface level. An example was presented in the previous section; (3) respecting the general principles of the development of Aplusix guarantees the coherence of the system after the introduction of the new register of representation of algebraic expressions. As regards the choices made in the design of the Aplusix tree module, it seems that most of them were made internally, i.e., by the developers themselves, and sometimes even individually, i.e., by one of the developers. Decisions are driven by the fundamental design principles in a way that a coherence of the whole system is preserved. Although it seems that the decisions are taken regardless the school

context, both teachers and students are taken into account in the system design. The principles 1 and 3 concern especially students and their interactions with the system. Moreover, the developers are respectful towards the students' ways of editing expressions, which is shown by the decision to make it possible to recover an expression in exactly the same way as the student has edited it, even if the implementation of such a decision was difficult (Trgalova and Chaachoua 2008).

The example of the development of Aplusix illustrates a way the synergy between computer scientists, researchers in math education and users can serve a project of development of educational software.

REFERENCES

Balacheff N. (1994), La transposition informatique, un nouveau problème pour la didactique des mathématiques, In Artigue et al. (eds.), *Vingt ans de didactique des mathématiques en France*, La pensée sauvage éditions, Grenoble, 364-370.

Bouhineau D., Chaachoua H., Nicaud J.-F., Viudez C. (2007), Adding new Representations of Mathematical Objects to Aplusix. In *Proceedings of the 8th International Conference on Technology in Mathematics Teaching*, Hradec Králové, Czech Republic, 1-4 July 2007.

Brousseau G. (1997). *Theory of Didactical Situations in Mathematics*. *Didactique des mathématiques*, *1970-1990*. Mathematics Education Library Series 19.

Chevallard Y. (1985), *La transposition didactique*, La Pensée sauvage, Grenoble.

Duval, R. (1993), Registres de représentation sémiotique et fonctionnement cognitif de la pensée, *Annales de Didactique et de Sciences Cognitives* 5, IREM de Strasbourg.

Duval R. (1995), *Sémiosis et pensée humaine*. Ginevra: Peter Lang.

Duval R. (2006), A cognitive analysis of problems of comprehension in a learning of mathematics. *Educ. Studies in Mathematics* 61(1-2), 103-131.

Nicaud, J.-F., Bouhineau, D., Chaachoua, H., Huguet, T., Bronner A. (2003), A computer program for the learning of algebra: description and first experiment. In Proceedings of the PEG 2003 conference, St. Petersburg, Russia, June 2003.

Nicaud J.-F., Bouhineau D., Chaachoua H. (2004). Mixing Microworld and CAS Features in Building Computer Systems that Help Students Learn Algebra. International Journal of Computers for Mathematical Learning 9 (2), 169-211.

Sfard A. (1991). On the dual Nature of Mathematical Conceptions: Reflections on Processes and Objects as Different Sides of the Same Coin, *Educational Studies in Mathematics* 22(1), 1-36.

Trgalova J., Chaachoua H. (2008), *Development of Aplusix software*. Paper presented at the 11th International Congress on Mathematics Education, Monterrey (Mexico), 6-13 July 2008. (Available online http://tsg.icme11.org/tsg/show/23).