Model of Articulation between aLDEAS Assistance Rules

Le Vinh Thai, Stéphanie Jean-Daubias, Marie Lefevre, Blandine Ginon
Université de Lyon, CNRS, Villeurbanne, France
Université Lyon 1, LIRIS, UMR5205, F-69622, Villeurbanne, France
{le-vinh.thai, stephanie.jean-daubias, marie.lefevre, blandine.ginon }@liris.cnrs.fr

1 INTRODUCTION

More and more applications are used in different contexts: professional, personal and educational. However, because of technical difficulties of handling or use of these applications, their users can abandon or under-exploit them and lose motivation (Gapenne et al., 2002). In the educational context, learners use various applications (pedagogical or non-pedagogical applications) to acquire knowledge (Ginon et al., 2014c). These applications must not only resolve technical difficulties as applications used in other interactive environments, but also give learners the pedagogical feedback and guidance which meet the pedagogical goals of teachers. For example, a given hint as a pedagogical feedback helps learners to do an exercise when they meet difficulties. Or, a pedagogical guidance guides learners to choose activities to work. However, these pedagogical feedback and guidance are not always supported by applications, especially non-pedagogical ones used in the educational context. Therefore, adding an assistance system is considered as a solution for both technical and pedagogical problems of an existing application. In details, pedagogical assistance systems can meet both technical (handling, use of applications) and pedagogical assistance needs (hint, explanation, guidance, etc.). However, the existing pedagogical assistance is varied and complex. It can be a complex pedagogical guidance to propose learning activities suitable to learners (Antoniadis et al. 2004). For example, the remediation activities are proposed depending on the progression of learners. In a pedagogical activity, the pedagogical assistance can have different modes to sequence of assistance events (Melis et al., 2001), (Winke and MacGregor, 2001). Theses modes describe the **articulation between assistance elements**. For instance, a successive assistance gives one message after another in order to guide learners. As part of my thesis, we identified two research issues: "How to help teachers to define the pedagogical guidance?" and "How to help teachers to define the articulation between assistance elements?" In this paper, we present our answer to second issue.

The AGATE project proposes the SEPIA system (Ginon et al., 2014a) that allows assistance designers (teachers) to add an assistance system in the existing ILE (Interactive Learning Enviroment) by creating and executing the aLDEAS rules (Ginon et al., 2014b). SEPIA supports various types of applications (windows, java, web and MacOS applications), assistance techniques (textual, vocal, enhancing, automatic actions, etc.) and it is the independent of application domains. SEPIA is a full solution to create rich assistance systems. However, definition of the articulation between assistance elements is still implicit and difficult. So, this paper presents the evolution made to SEPIA to overcome these limitations.

In this paper, first we present the AGATE project and its major results: the SEPIA system and the aLDEAS language. We also show that expression of the articulation between assistance elements of an assistance system in ILEs is a complex task. Then, we present the different existing modes of articulation through examples of assistance (section 3.1). We too confront these modes to tools that aim at the definition of assistance as well as modes of articulation (section 3.2). These studies allow us to propose a model of articulation between aLDEAS assistance rules (section 4). Then, the implementation of this model is presented (section 5). To validate our approach, we present some results from our evaluation (section 6). Finally, we give the general conclusions and the issues that motivate the future works (section 7).

2 SEPIA SYSTEM

The AGATE (Approach for Genericity in Assistance To complEx tasks) (AGATE, 2015) project aims at proposing generic models and unified tools to enable the setup of assistance systems in various existing applications, that we call target-applications, by applying a generic and epiphytic

approach. Epiphytic application is the application that is able to perform actions in another application without requiring any change to it. Thus, the functioning of an epiphytic assistance system added in the target-application doesn't disturb the functioning of this application (Paquette et al., 1996). The models and tools proposed are specific neither to an application nor to a domain. For that reason, we previously proposed an adjunction process of epi-assistance systems to a given target-application (Ginon et al., 2014b). This process (Figure 1) consists of two phases: the assistance specification and the assistance execution in an epiphytic way.

The **assistance specification** is performed by an expert of the target-application, called the *assistance designer*. This preparatory phase enables the designer to specify the assistance that he wishes for a given target-application. The **assistance execution** concerns end-users of the target-application. It is the execution of the assistance designed by the designer; it occurs at any use of the target-application by an end-user. The **epi-detector**s make possible the monitoring of the target-application. They detect the events related to interactions between the user and the target-application by exploiting the accessibility libraries compatible with a type of applications (windows, java, web applications…). Finally, the **epi-assistants** handle the elaboration of the answer to provide assistance to the end-user by pop-ups windows, speech or animated agent as well as highlighting a component on interfaces (for instance, by colouring a component).

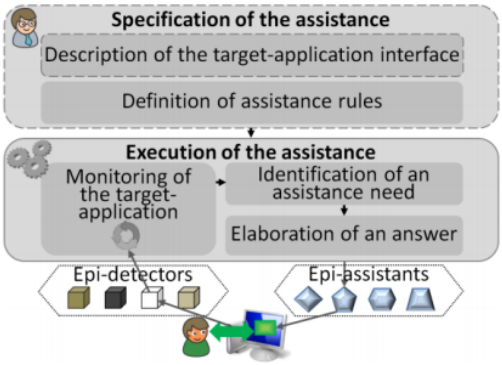


Figure 1: Adjunction process of epi-assistance systems (Ginon et al., 2014b).

The **aLDEAS language** (a Language to Define Epi-Assistance Systems) (Ginon et al., 2014b) is proposed in order to connect the two phases of this process. aLDEAS consists of three principal elements: event wait (click on a button…), consultation (of profile, of states of application,…), assistance action (message, enhancing,…). This language is completed by a **rules pattern** (Figure 2) and also by other patterns facilitating the definition of assistance actions (for example, step by step pattern). A **rule** begins with event wait called **trigger event**. When this event occurs, the launch of **assistance actions** is immediate (upper path in Figure 2), or is constrained by a **condition** (lower path in Figure 2). This condition takes the form of a consultation with the alternatives each associated with one of these actions. Finally, the rule can be terminated by **end event** that ends all elementary actions launched by this rule. For example, the example in Figure 2 shows a rule among many rules which define an assistance system. This rule waits a click on the button 'help' in order to verify the answer of the learner and to provide an error message when this answer is not correct (text written by the learner is not equal to 1). This message is closed after 10 secs.

aLDEAS and its patterns are implemented in the **SEPIA system** (Ginon et al., 2014a) that consists of two tools: an assistance editor and an assistance engine. The **assistance editor** operationalizes the assistance specification phase (upper part in Figure 1). It provides an interface that allows the assistance designer to define an assistance system by creating a set of aLDEAS assistance rules. The **assistance engine** operationalizes the assistance execution phase (lower part in Figure 1). It executes the assistance system created in the previous phase by executing its set of aLDEAS assistance rules.

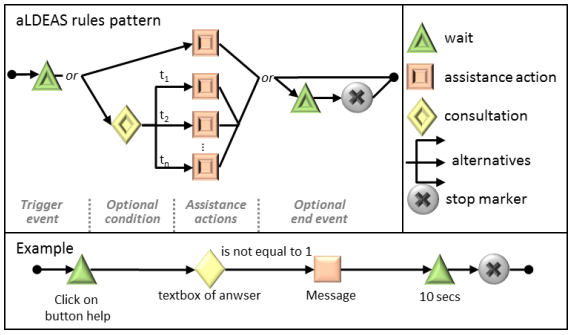


Figure 2: aLDEAS rules pattern.

SEPIA and aLDEAS allow creating useful assistance systems in various domains, among which ILE. However, the creation of assistance systems which can be found in ILEs is complex. Thus, the principal objective of target-applications in ILE is learning. Assistance systems for these applications must be effective, suited to learners, to their tasks in the target-application and to their progression. Additionally, they must meet the pedagogical

objectives and strategies of teachers. In SEPIA, such assistance systems require the definition of a lot of rules which need to be articulated with different modes. For instance, an assistance system provides the learner with progressive assistance to solve an exercise when he asks for assistance. For the first time, this system gives an explanation, then a hint and finally a solution. An explanation can be given by showing the messages step by step. So the rules defining this assistance system are articulated in both progressive and successive modes. However, aLDEAS and SEPIA have not yet provided an explicit and easy way of definition of modes of articulation between rules. To tackle this problematic, we firstly made a state of the art in order to identify existing modes of articulation between assistance elements. We sum up this work through the examples (section 3.1) and present the support of these modes in some tools (section 3.2). Then, in order to allow aLDEAS and SEPIA to support these modes, we proposed a model of articulation between rules (section 4) and its implementation (section 5).We conclude this paper by the presentation of the evaluations that we defined for this research.

3 STATE OF THE ART

3.1 Modes of Articulation between Assistance Elements

Currently, pedagogical assistance is found in some applications. This assistance can be executed according to different modes to sequence of assistance events. These modes describe articulation between assistance elements.

In many applications, an assistance element is given independently from another. There are not constraints between assistance elements. We can easily find this mode in most applications with the tooltips. A tooltip appears when the user hovers a component on application interface. So, each tooltip is independently showed. We take a concrete example of IXL learning (IXL Learning, 2015) that provides comprehensive, curriculum-aligned mathematics and English content for preschool to grade 12. It shows overviews of course through a sequence of independent pop-ups (A in Figure 3) when learners hovers links of course. We call this mode of articulation *independent mode*.

The tutorials integrated in some applications provide step by step assistance in order to guide users. For instance, Connectify (B in Figure 3) (Connectify, 2015) provides the messages one after the others which allow user to learn the use of this application. Such messages may also explain step by step user errors. Each assistance element is constraint by the end of the previous assistance element. We call this mode of articulation *successive mode*.

Hot Potatoes (Winke and MacGregor, 2001) allows teachers to create different types of exercises. It is especially useful for creating online, interactive language learning exercises and for providing pedagogical assistance such as diagnosis to verify the answers of the learner. The diagnosis can be given for a several parts at the same time. For instance, an exercise created by EOLF in Franche-Comté university (EOLF, 2016) (C in Figure 3), shows correct or incorrect answers for an English exercise at the same time. We can get the assistance in the forms frequently offered on web where the diagnosis on different user inputs can be simultaneously displayed. These examples show that all assistance elements can be simultaneously given. We call this mode of articulation *simultaneous mode*.

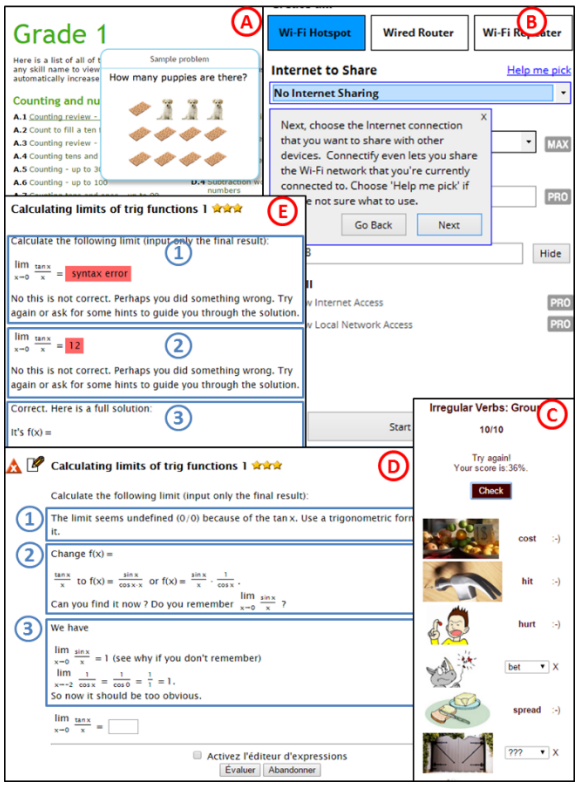


Figure 3: Examples of modes of articulation between assistance elements.

ActiveMaths (Melis et al., 2001) offers to learners a mathematical web application with pedagogical assistance. The learner can ask for assistance in progressive way (D in Figure 3). At the first request, a hint is given to the learner. Then, at the next request, the given hint is more detailed. At the last request, the solution is given in order to avoid blocking the learner in working on problems. The assistance is more and more detailed and concrete. We call this mode of articulation *progressive mode*.

Some applications consult information sources to provide suitable assistance such as the application state, the user profile or the user's choice. Thus, both ActiveMaths and Hot Potatoes allow consulting the state of the application such as a text filled by the learner. For instance, ActiveMaths (E in Figure 3) verifies answers of the learner and shows the result (syntax error, incorrect/correct answer). Consultations are therefore essential to provide a suitable assistance. We call this mode of articulation *interactive mode*.

These modes of articulation can also coexist or be combined in assistance systems. For instance, tutorials provide step by step assistance, but in one step, a textual message and an enhancing can be simultaneously performed. This is a combination of successive and simultaneous modes.

3.2 Related Works

The modes of articulation presented in the previous section exist in many applications. However, some tools allow also defining these modes explicitly or implicitly. So, we confronted these modes to tools related to our approach.

In Marco advisor systems (Richard and Tchounikine, 2004), the advices are represented as a graph. The subset of graph represents the assisted website. There is no explicit articulation between advices because each advice is represented independently. However, the articulation between the elements of a same advice can be conditioned by the navigation history of the user. This is a case of interactive articulation of assistance but implicitly represented. In the Astus platform (Paquette et al., 2014), educational interventions are represented as rules organized in a graph. The conditions of the rule make explicit interactive articulation between interventions. In the same way, the Epitalk system (Paquette et al., 1996) explicitly represents the advices through a tasks graph. These works concentrate on the definition of assistance but do not explicitly address the aspect of articulation. Therefore, the articulation may be defined explicitly or implicitly by their tools and we can't find the presence of progressive and successive modes of articulation.

In another way, the Grafcet graphical language (David, 1995) is proposed in order to represent the sequential automation in systems decomposable into steps. Although Grafcet is not specific to assistance systems, the expression of sequence of steps can inspire our work. In Grafcet, a step can be an active step, initial step, macro-step, etc. Actions are associated with a step. The transition between two steps is done through a transition. A transition is one or more logical condition (boolean). With Grafcet, we can describe explicitly the independent, successive, interactive, simultaneous modes but only implicitly the progressive mode. In addition, some elements of Grafcet are not suitable with the ones of aLDEAS. For instance, Grafcet doesn't distinguish events from conditions as aLDEAS does. It contains also useless information in our context to the phase of specification of an assistance system such as active state on step.

To overcome these limitations in the literature, we proposed a model of articulation between aLDEAS rules and implemented in SEPIA. This model and its implementation are presented in the following.

4 MODEL OF ARTICULATION BETWEEN ALDEAS RULES

If aLDEAS and its implementation in SEPIA already allow the definition of the articulation between assistance elements such as those presented in the section 3 with aLDEAS rules, the expression of the articulation between the rules is implicit and can be complex to define for the assistance designer. To more effectively operationalize these modes of articulation in our propositions, it is necessary to allow defining explicitly the articulation between aLDEAS rules.

Thus, an assistance system is currently defined in the AGATE project by a set of aLDEAS rules always at the same level. In the aLDEAS rules pattern (Fig. 2), the trigger event, the end event and the trigger condition are central elements to form the articulation between rules. For instance, we defined two rules R_1 and R_2 which describe two successive steps in the tutorial of Connectify. So, these rules are articulated in successive mode. It means that R_2 is launched at the end of R_1. For this order of launch, the trigger event of R_2 must be the event "end of R_1".

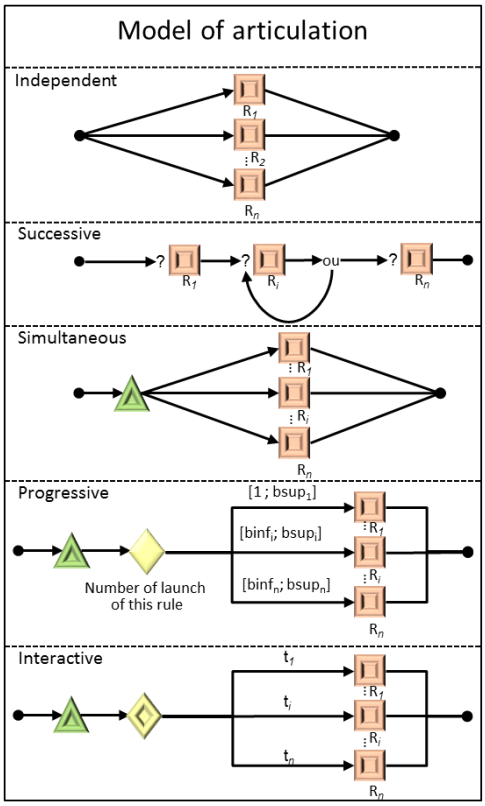


Figure 4: model of articulation between aLDEAS assistance rules

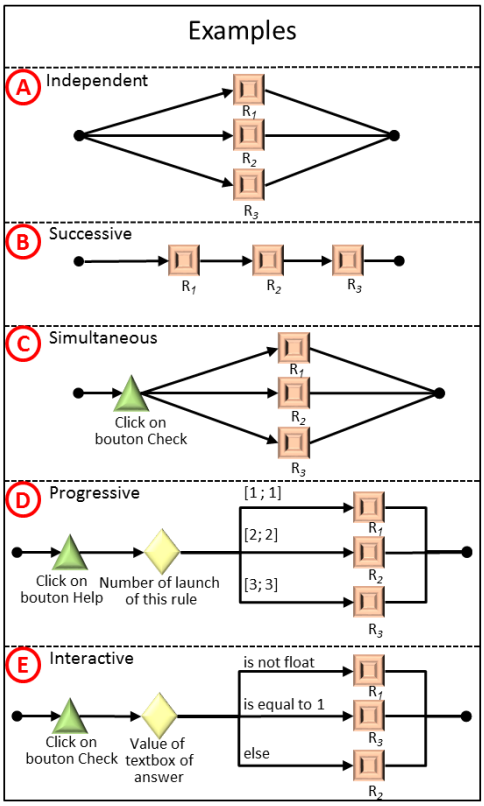


Figure 5: examples of model

On the one hand, we must carefully define elements in the rules in order to ensure correct articulation between them. On the other hand, we must examine them in order to understand which mode of articulation to choose. Therefore, this articulation between rules is implicitly expressed and is complexly defined with aLDEAS.

For these reasons, we propose to complete our language by a model of articulation between assistance rules. To simplify the representation of the model, we note that rules between which we want to make an articulation are named R_i with i $\in[1, n]$, ($n \geq 2$). The representation of our model is given in Figure 4. It gives an overview of the five modes of articulation that we identified from a study of existing works: independent, successive, simultaneous, progressive and interactive.

In each mode of articulation, there are constraints that rules must respect to ensure the correct articulation between them (for instance, for successive mode, each rule should be launched by the end of the previous rule). The constraints of each mode of articulation are shown in the next section with examples of assistance. These examples of assistance are inspired by examples presented in section 3. To simplify, we describe only three rules articulated for each example.

4.1 Independent Mode of Articulation

In the independent mode of articulation (Figure 4) the rules R_i are launched by their own trigger events. This mode doesn't impose any constraint. The definition of rules articulated in independent mode reflects the classical definition with aLDEAS. Obviously, the other modes presented thereafter are specific cases of this mode with specific constraints on rules.

Example A in Figure 5 is a case of assistance to IXL Learning (section 3.1). Here, we present a similar but simpler assistance written in aLDEAS which takes only the three first overviews corresponding to the three first courses. This assistance is created with 3 rules articulated in independent mode. The rules R_1, R_2, R_3 are respectively corresponding to the three courses "Counting review - 0 to 10", "Count to fill a ten frame", "Counting review - up to 20". Each rule begins with its own trigger event "hover on link of course" in order to show a message which presents overview of this course.

4.2 Successive Mode of Articulation:

In the successive mode of articulation (Figure 4), the rules are launched one after the other, it means that at the end of the rule R_i, the rule R_{i+1} is launched.

In the detailed definition of this mode of articulation (Figure 6), we can see that the rule R_i is forced to have at least an end event and R_{i+1} is forced to have a trigger event "end of R_i." This constraint is applied to all rules except the first and last ones. The first rule R_1 can begin with any trigger event(s) and the last rule R_n may end with none, one or several end events. In this mode of articulation, the rule R_1 is an entry point of the rules R_i. So, the trigger events of the rule R_1 launch this set of rules in successive mode.

However, a rule R_{i+1} cannot be launched until the end of its preceding rule R_i. Consequently, if R_i has a trigger condition that is not validated at the time of the assistance execution, the rule and its assistance actions will not be executed until its end events, and the following rule will therefore not be launched. So, the whole sequence of rules is interrupted. This requires a rule R_i to contain a condition to have an alternative "else". This alternative ensures that the condition is always valid.

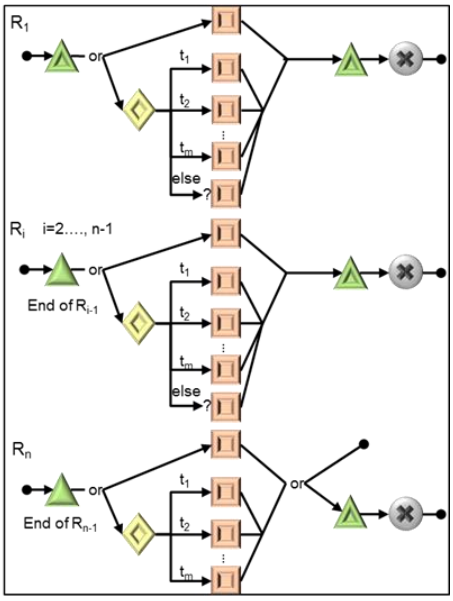


Figure 6: Constraints on rules of successive mode of articulation in aLDEAS

Let's take the example of the assistance to Connectify (section 3.1). Here, we present an assistance similar but simpler which takes only the three first steps of the tutorial of Connectify. This assistance is created with three rules articulated in successive mode. These three rules (defined in more detail in Figure 7) must respect the constraints of the successive mode (Figure 6). Thus, the first rule R_1 waits until a user's click on bouton "Tutorial" in order to show a message of welcome and closes this message after 10 seconds. Then, the rule R_2 that waits until the end of R_1 shows a message of internet connection check and closes this message after 10 seconds. Finally, the rule R_3 that waits until the end of R_2 shows a message of choice of an internet connection as well as highlight the related combo box. These message and highlight is also closed after 10 seconds.

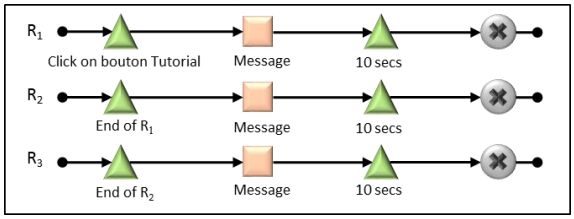


Figure 7: Detail of three rules articulated in successive mode in aLDEAS

4.3 Simultaneous Mode of Articulation

The simultaneous mode of articulation (Figure 4) allows executing several assistance rules simultaneously.

In the simultaneous mode of articulation, the rule R_i must begin the same trigger events as ones defined for this mode. When these events occur, all rules are launched at the same time.

Let's take the example of the assistance to an exercise created with Hot Potatoes that shows simultaneously all errors of answers. Here, we present an assistance similar but simpler which takes only the three first answers corresponding to three first user inputs. This assistance is created with three rules R_1, R_2 and R_3 articulated in simultaneous mode (Figure 5). These three rules must begin with trigger event "click on bouton Check". When the learner clicks on this button, they are launched. They verify learner's answers with the consultation of the user inputs on the application and shows the result (correct or incorrect) by adding a text (OK if correct, X else) near to these user inputs.

4.4 Progressive Mode of Articulation

In the progressive mode of articulation (Figure 4), the launch of assistance rules depends on the number of times that the learner is in a same situation. In particular, this mode allows to provide the user with

the assistance more and more detailed and concrete to meet a repeated request of assistance.

In the successive mode of articulation, the rule R_1 is the entry point of assistance to successively start the rules R_i. In the simultaneous mode, all rules R_i start with same trigger events. However, in this progressive mode, there must be an additional rule as an entry point to constraint the launch of the rules R_i. We call this rule R'. R' launches one rule among the rules R_i, according to the number of launches of the rule R'. In the rule R', each rule R_i is associated with an interval [$left_i$, $right_i$]. It means that R_i is launched for one or several times between $left_i$ and $right_i$. For the first times [$left_1$, $right_1$], R' launches R_1 and for the next times [$left_2$, $right_2$] R' launches R_2. In this mode of articulation, the rules R_i must begin with a trigger event "launch by a rule (R')".

Let's take the example of the assistance to ActiveMaths (section 3.1) that gives at first a hint, then a more detailed hint and finally the solution when the user repeatedly asks for assistance. The assistance is created with three rules articulated in progressive mode (Figure 5). R' begins with trigger event "click on bouton Help" and launches the rules R_1, R_2, R_3 which show respectively a hint, another more detailed hint and the solution. This launch is constrained by the number of launches of R'. It means that the number of clicks on button "Help" is counted. To be launched, these three rules R_1, R_2 and R_3 must begin with a trigger event "launch by a rule (R')". So, R_1 is launched by R' for the first click on bouton "Help", R_2 for the second click and R_3 for the third click.

4.5 Interactive Mode of Articulation

In the interactive mode of articulation (Figure 4), one of the rules R_i is launched according to a consultation of the user profile, of the application state, of the history of the assistance, of the trace and / or of the user.

Again, a rule R' is used as an entry point for the launch of the assistance. Each rule R_i is associated with an alternative of the trigger condition of R'. R_i must begin with the trigger event "launch by a rule (R')." The progressive mode of articulation (see section 5.3) is a special case of the interactive mode, frequently encountered in the existing assistance systems and in which the trigger condition of R' is exclusively a number of launches of R'.

Let's take the example of the assistance to ActiveMath (section 3.1) which shows the diagnosis by consulting the learner's user. This assistance is created with three rules articulated in the interactive mode. The representation of the additional rule R' is the same as the representation of interactive articulation. R' begins with the trigger event "click on bouton Check" and launches the rules R_1, R_2 and R_3 which show respectively a syntax error, a calculation error and success. This launch is constrained by the learner's answer: the value of the text box entered by the learner. To be launched, these three rules R_1, R_2 and R_3 must begin with a trigger event "launch by a rule (R')". When the learner clicks on bouton "Check", one rule among the three rules is launched by R'. If the entered value of the text box does not belong to float type, R_1 is launched, if this value is equal to 1, R_3 is launched and elsewhere, R_2 is launched.

5 IMPLEMENTATION OF OUR MODEL OF ARTICULATION BETWEEN RULES

We implemented this model of articulation between rules in SEPIA that haven't supported the explicit expression of the articulation until now (Figure 8). More concretely, we enriched the SEPIA assistance editor to support designers to define explicitly the five modes of articulation as well as to facilitate their definition. In order to facilitate the comprehension of designers, we adopt the notion of *bloc* that regroups the rules articulated in a given mode among these five modes.

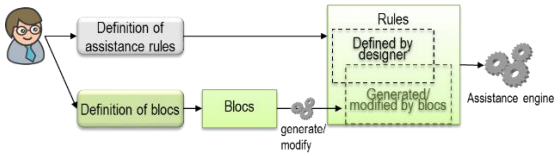


Figure 8: SEPIA completed with the model of articulation between rules

Thus, SEPIA allows assistance designers to define and view graphically blocs of rules articulated in the wished mode. Each mode has constraints on rules to ensure a correct articulation. The automatic application of these constraints facilitates the definition of assistance systems. It allows automatically generating or modifying aLDEAS rules as well as eventually verifying designer's definition. For instance, to define a bloc of rules articulated in the successive mode, the next rules must have a trigger event "end of previous rule". The definition of the bloc will add automatically this event in these rules. Otherwise, the previous rules

must have at least an end event which must be determined by the designers. Therefore, SEPIA will check this constraint and show a message if it is not satisfied.

An assistance system is now represented in SEPIA by rules not only defined by designers but also generated and modified by the application of constraints thanks to blocs (Figure 8). This allows keeping the current operation of the SEPIA engine which doesn't need to change because of the implementation of model of articulation. This implementation allowed us to create and executes with SEPIA examples of assistance similar to assistance presented in the section 3. Figure 9 shows the execution of these examples.

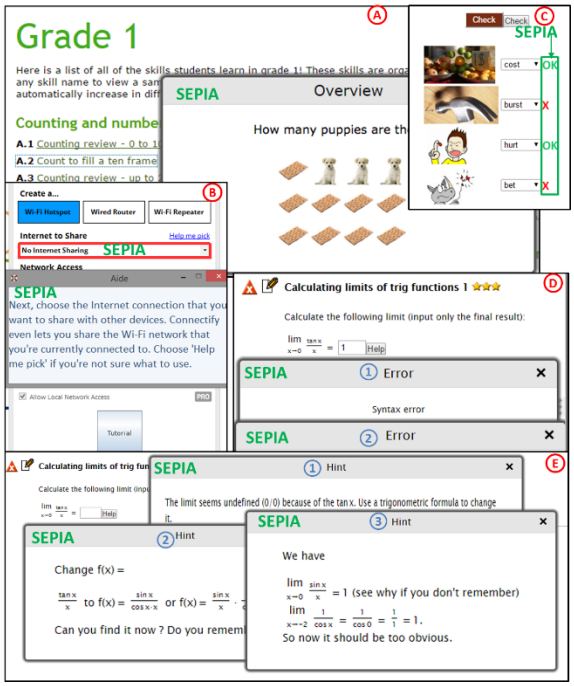


Figure 9: Execution of examples of assistance created by SEPIA with the implementation of our model

6 EVALUATION

The aLDEAS language and its implementation in the SEPIA system were previously evaluated. The usability of aLDEAS and the usability of the SEPIA were evaluated in (Ginon et al., 2014b). The execution of assistance systems with the SEPIA in ILEs was evaluated in (Ginon et al., 2014c).

In this paper, we propose a model of articulation between assistance rules. Therefore, in this section, we focus on the evaluation of this model of articulation. Regarding the feasibility of model, it is demonstrated by the implementation of our model in SEPIA. The examples in Figure 9 show the possibility of our model. Thus, the model of articulation allows us to define the assistance systems similar to ones presented in section 3. In addition, we made an experiment of our model with few users and will make another with more users in spring 2016. The objective of these experiments is to evaluate: (C1) the capacity of comprehension of an assistance system created by using blocs, (C2) the capacity of use of blocs, (C3) the benefices of use of blocs, (C4) the coverage of 5 modes relative to expectation of designers. The designers must work with three ways of definition of assistance systems: definition without bloc by using the textual interface, definition without bloc by using graphical interface and definition with bloc by using the graphical interface. With each way, they will execute 3 steps: comprehension, modification, and completion of an assistance system. Next, they must create an assistance system with a preferred way of definition. Each user uses one mode of articulation.

We started the first experiment with students in France in order to observe and improve the future experiments. In this experiment, there are 5 master students. We only observed their tasks without considering the result. However, the majority of them answered that they understood how an assistance system operates and define an assistance system with bloc. Then, we improved the documents for the experiment with 6 Vietnamese students in the course HCI (Human Computer Interaction) in Vietnam. We summarized results, which is presented through the above figures. Figure 10 shows the number of users who succeeded the comprehension, modification and completion by three ways (for evaluation of C1, C2). There are no major difference between them. However, most users (5 out of 6 users) prefer to use the bloc in order to define an asked assistance system (Figure 11). In more detail, these users indicate that the comprehension and the definition of an assistance system with bloc are easier than others (Figure 12) (for evaluation of C3). The five modes are indicated enough for the definition of an assistance system because the users didn't give any other mode existing in other applications or in reality (for evaluation of C4). Through this experiment, we can think that our model of articulation facilitates the comprehension, the definition of an assistance system. The modes of articulation deducted from bibliographical studies (cf. section 3) can define various assistance systems.

However, the number of users who participated

in the above experiment is low. Therefore, we will make another experiment in spring 2016 with 30 French master students with the same objectives. For the evaluation of coverage of model, we will improve this experiment by asking students to imagine a pedagogical assistance system. They must show whether it can be defined by using blocs with one among the five modes of articulation or with a non-existing mode of articulation.

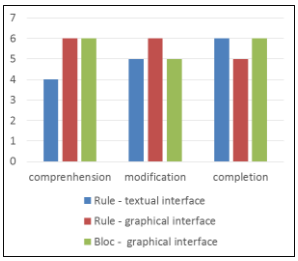


Figure 10: number of success for the realization of tasks in our experiment

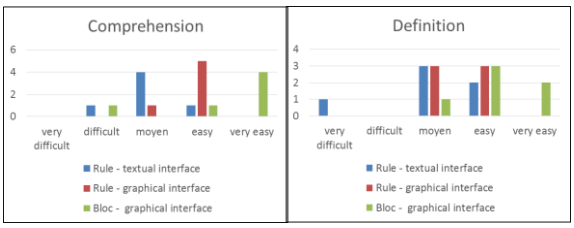


Figure 11: levels of comprehension and definition of an assistance with three ways of definition

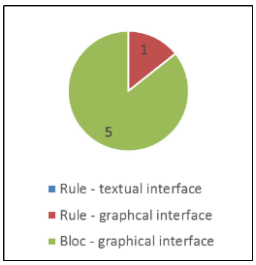


Figure 12: number of users for their preferred way of definition out of 6 tests

7 CONCLUSION AND FUTURE WORK

In this article, we presented the model of articulation between aLDEAS assistance rules which completes the aLDEAS language. This model explicitly expresses the notion of articulation between rules of an assistance system. It offers five modes of articulation corresponding to those we have identified in our bibliographical study: independent, successive, simultaneous, progressive, and interactive. We implemented this model in the SEPIA system by adding the notion of bloc of rules articulated in a mode. This implementation have two main advantages: it makes explicit the definition of blocs of rules with graphical interface and it applies semi-automatically constraints on rules. With the introduction of this model in our approach, an assistance system is defined not only by a set of rules, but also by a set of blocs that explain the articulation between these rules. It allows teachers to view more explicitly as well as define more easily a complex assistance system. We evaluated our propositions by the experiment which showed some big potentials.

However, an assistance system can be described by many blocs of rules articulated in different modes. SEPIA just shows the graphical representation of a bloc but not the global graphical representation of all the blocs. The blocs are listed in a table that limits designer's view of a whole assistance system. Therefore, in the future, we will aim at a global graphical representation of assistance systems which will be more intuitive.

As part of thesis, we continue to evolve SEPIA which will facilitate the definition of pedagogical guidance. We will find out how existing applications or systems propose pedagogical activities suitable to learners. For example, the activities can be temporally planned or the proposition of activities can be constraint by states of previous activities (e.g. remediation activities). Then, with SEPIA, we try to define assistance systems which can also propose these activities in order to identify difficulties. Thus, SEPIA has not yet supported the concepts "pedagogical guidance" and "learning activity". It's difficult and complex for assistance designers who wish to define a pedagogical guidance. So, we aim to propose these concepts in SEPIA. Through our state of the art, we will enrich these concepts in SEPIA (for example, temporal attribute in pedagogical guidance, output states in pedagogical activity).

REFERENCES

Antoniadis, G., Echinard, S., Kraif, O., Lebarbé, T., Loiseau, M., & Ponton, C., 2004. NLP-based scripting for CALL activities. In Proceedings of the Workshop on eLearning for Computational Linguistics and Computational Linguistics for eLearning pp. 18–25.

AGATE. http://liris.cnrs.fr/agate/ [retrieved 2015]

Connectify. http://www.connectify.me/ [retrieved 10 December 2015]

David, R., 1995. Grafcet: A powerful tool for specification of logic controllers. In *Control Systems Technology*, IEEE Transactions on, vol. 3, pp. 253–268.

EOLF. http://eolf.univ-fcomte.fr/wp-content/uploads/grammar/verbs/irregular/10.htm [retrieved 10 December 2015]

Gapenne, O., Lenay, C., Boulier, D., 2002. Defining categories of the human/technology coupling: theroretical and methodological issues. In *workshop ERCIM on User Interface for All*, pp. 187-198. France.

Ginon, B., Jean-Daubias, S., Champin, P.-A., & Lefevre, M., 2014a. Setup of epiphytic assistance systems with SEPIA. In: *EKAW*, pp. 1-4. Linkoping.

Ginon, B., Jean-Daubias, S., Champin, P.-A., & Lefevre, M., 2014b. aLDEAS: a Language to Define Epiphytic Assistance Systems. In: *EKAW*, pp. 153-164. Linkoping.

Ginon, B., Thai, L.-V., Jean-Daubias, S., Lefèvre, M., & Pierre-Antoine, C., 2014c. Adding epiphytic assistance systems in learning applications using the SEPIA system. In *EC-TEL*, pp. 138-151. Graz.

IXL learning, https://eu.ixl.com/math/grade-1 [retrieved 10 December 2015]

Melis, E., Andres, E., Budenbender, J., Frischauf, A., Goduadze, G., Libbrecht, P., Pollet, M. & Ullrich, C., 2001. ActiveMath: A Generic and Adaptive Web-Based Learning Environment. In: *IJAIED*, vol. 12, pp. 385-407.

Paquette, G., Pachet, F., Giroux, S., Girard, J., 1996. Epitalk, a generic tool for the development of advisor systems. In *IJAIED*, p. 349-370.

Paquette, L., Lebeau, J.-F., Beaulieu, G., & Mayers, A., 2014. Designing a Knowledge Representation Approach for the Generation of Pedagogical Interventions by MTTs. In *International Journal of Artificial Intelligence in Education*, p. 1-39.

Richard, B., & Tchounikine, P., 2004. Enhancing the adaptivity of an existing website with an epiphyte recommender system. In *New review of hypermedia and multimedia*, vol. 10, p. 31-52.

Winke, P., & MacGregor, D., 2001. Hot Potatoes version 5. In: *Language Learning Journal*, vol. 24, pp. 30–33.