About including a virtual teacher in a competitive or collaborative context in AlgoPath

Estelle Perrin*, Sébastien Linck*and Dimitry Zekrouf*
*CReSTIC
IFTS - University of Reims Champagne-Ardenne
Charleville-Mézières, France
Email: estelle.perrin@univ-reims.fr, sebastien.linck@univ-reims.fr, dimitry.zekrouf@etudiant.univ-reims.fr

***Abstract*—AlgoPath is an entertainment program to help students with no classical computer science background understand the main concepts of algorithmics. It looks like a serious game in which the player builds a world of concrete, paths and grass. We show in this paper that the inner model of AlgoPath is based on the Model-View-Controller architecture (MVC). We intend to add a virtual teacher in this architecture, who oversees the players' interactions and intervenes when the interactions do not correspond to what is expected by a real teacher. We explain how the model of the virtual teacher interacts with the Controller component of the architecture. We describe the model that lets a real teacher add his particular comments for the various mistakes that can be made by a player. We explore the solutions to implement two multiplayer gameplay types: a competitive one, in which two or several players fight against one another to be the first to find the best algorithm, and a collaborative one, in which they have to find the best algorithm as a team. Finally, we explain our choice to develop the distributed version of AlgoPath with Unity.**

I. Introduction

Today's students are not the people they used to be. They live in a world surrounded by electronics: smart phones, tablets and computers (preferably connected to the Internet), ... provide them with video games, chats, e-books, information at any time. Such students, born after the eighties, are called digital natives as opposed to digital immigrants [1]. They live in a world full of multimedia but sometimes multimedia seem to remain away from their teaching space. It is particularly the case with algorithmics courses where the teacher asks them to think with a pen and a sheet of paper. This technique may seem odd for a course dedicated to computer science.

Students may want to graduate in computer science without having a full computer science background. For example, we offer in our university under-graduate courses dedicated to web-sites design. The courses include design, project management, communication, HTML and CSS languages but also PHP language. The latter is much related to algorithmics and object-oriented programming concepts. This leaves us with students having difficulties to catch a good mental representation of an algorithm and not being motivated to do so. Variables, conditional instructions, loops, parameters, functions, objects, polymorphism, etc will remain meaningless words however hard teachers can try to explain. Nick Ellis pointed out in [2] that in the matter of word meaning, "the eyes see vividly, but ears only faintly hear, fingers barely feel and the nose doesn't know". Virtual environments like computer games can motivate students to engage in learning activities [3]. So to help students conceptualize and learn, we created the virtual world of AlgoPath. In this world, variables are 3D figures born in huts and carrying backpacks. Statements are made of grass and stones. Functions and procedures are forests - for more details see [4] and [5] and for a blink of it see Fig.1.

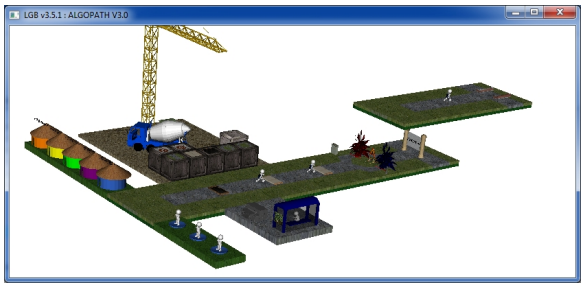


Fig. 1. This is the world of AlgoPath.

If AlgoPath can currently help students avoid common mistakes (such as using a variable in an expression before it has been set or forgetting to assign a parameter passed by reference), it cannot - in its previous versions at least - help them find the flow of statements for a given problem. In this paper, we discuss the inner model of AlgoPath that is based on the Model-View-Controller architecture (MVC) in order to include a virtual teacher. The role of this virtual teacher is to reproduce the advice, hints and tips a real teacher would give in a 'I-don't-know-what-kind-of-statement-I-should-add' situation. We also want this virtual teacher to be there helping students whether they are discovering AlgoPath for the first time or they are playing against one another to find the best algorithm or they are looking for the best algorithm as a team.

This paper is sectioned as follows: section II reviews the different models of a virtual teacher and the impact of gameplays on learning activities. Section III shows the inner model of AlgoPath. Section IV discusses the introduction of a virtual teacher in the architecture of AlgoPath. Section V describes the technology that will be used in AlgoPath to implement the virtual teacher and section VI concludes this paper and proposes future work.

978-1-4799-5350-9/14/$31.00

II. State of the art

We review the impact of gameplays on the learning process in section II-A, the ways a virtual teacher is included in a virtual environment in section II-B and how it is possible to create a virtual teacher in section II-C.

A. Multi-player interactions

Numerous studies dealing with human relations have been conducted. We focused on papers analyzing competitive or cooperative aspects of relations between individuals or groups. The first studies were conducted in 1949 (see for example Deutsch [6] [7]). Competitive or cooperative relations initiated in classrooms obey the same principles that exist among humans in the context of everyday life. Relations between students involved in the learning of problem-solving are studied in [8]. It shows that cooperative learning is a better way to solve non linguistic problems such as mathematics. Thanks to the exchange of information and recommendations between students, more iingn are shared and students solve problems more rapidly than alone. The usefulness of cooperative learning in the context of scientific courses is shown in [9]. It states that cooperative students significantly outperform conventional students.

Today these relations are studied within a computer-made environment often named virtual learning environment or serious game. Tom Weneck, a games designer and critic, pointed out in [10] that games are emblematic of the time they come from. Social media are embedded in our lives. So, games, learning environments and serious games naturally appeared, taking into account relations between players. For example, in some games, players can only succeed if they cooperate with one another.

Not all serious games engage students in committed learning. [11] shows that learning activities have to be integrated insiin the game story but also outsiin of the game story when the computer is off. The game story is important to keep learners in the process but the gameplay is essential too. Players engaged in a collaborative gameplay spend more time in the game and, by doing so, get a higher quality result [12] because the efforts made to achieve the objectives are more important. Similarly, playing with friends leads to greater efforts and to a better commitment [13]. So we can assume that playing with other classmates can have an interesting influence on learning.

Collaborative gameplay seems to benefit to non-linguistic learning [11]. Collaborative and competitive gameplays can be mixed in games as described in [14]. The game "Escape from Wilson Island" described in [15] involves teams. The gameplay focuses on collaborative tasks so that players need to coordinate their actions in order to succeed.

B. Role of a virtual teacher

A serious game or a virtual learning environment remains a tool for learning activities. Learning activities may involve a teacher. That is the reason why it may be interesting to include a teacher in the game. In a virtual environment, the teacher acts as a gamemaster: the system determines the learning style of an individual and proposes the most appropriate content based on her/his errors or on her/his progress during the process of learning. Studies focus on the structure of the courses and the way the student learns [16] to adapt the chronology of the ongoing needs of the student. The e-teacher of [17] assists students in the choice of his/her courses. The presentation of the content is adapted to her/his way of learning [18] [19]. The intelligent tutoring system described in [20] helps students during their learning progress. The content changes according to the students' skills and knowledge.

In [21], the virtual teacher of the virtual learning environment is an expert system. The latter can be decomposed into two parts. The first detects and iinntifies the errors made by students. The second is a pedagogical model that offers an assistance to students. How to achieve this role (providing an assistance) is the topic of the next section.

C. Intelligent agents

Having a teacher by her/his siin would bn beneficitial for the user if he/she would get lost when he/she would discover AlgoPath by her/his own for example. Such a good Samaritan could be a patient and merciful teacher. A teacher can be virtualized by an intelligent agent. An intelligent agent is an nntity in a software or in a part of one supposedly able to act by itself [22]. This nntity is situated in an environment, is rather autonomous in the sense that it can react to changes in the environment, and is able to perform reasoning and by doing so to determine actions. Intelligent agents are and have been used in games, simulation games or simulators. See for example the serious game Roma Nova whose goal is to teach history to young individuals and, for that purpose uses pedagogical conversational agents [23]. The goal of an

Fig. 2. Serious game Roma Nova developed at the Serious Games Institute.

intelligent agent is to balance its external environment and its internal environment in order to make them meet [24]. In a learning environment such as AlgoPath it would mean that the external environment would be the user, the internal environment would be the game and the goal of the virtual

teacher would be to help the user changes the game in a way the virtual world would represent the algorithm to be created.

When the virtual teacher is represented by an intelligent agent, the latter can answer questions asked by the user. In AlgoPath, the first available question would be: "What can I do?". To answer such a question, predicate logic is used to formalize knowledge [25]. Formulas of predicate logic contain variables which can be quantified. They let us express natural language sentences that often contain a conditional statement and a hypothesis. For example, the sentence "Every Bourgueil wine has a red hue" - which is quite true but not entirely because 2% of Bourgueil wine has a less red shade - can be expressed by a formula similar to $8xBOURGUEIL(x) \;) \; HUE(x;RED)$.

The state of the environment is expressed by a set of facts. Knowledge is a set of formulas commonly named rules. Reasoning is achieved when new facts are discovered after having considered every possible rules.

Reasoning can be performed forward starting from a fact in order to discover new ones or backward when a specific goal is to be reached. A fact can also be proved by adding its negation to the knowledge base and by reaching a contradiction. Inference can be done by using several normalized rules:

- the "Modus Ponens" rule $(P \; ^ \; (P \;\;) \;\; Q)) \;\;) \;\; Q$ that stands for if P is asserted to be true and P implies Q, so therefore Q must be true;
- the "Modus Tollens" rule $((P \;\;) \;\; Q) \; ^ \; :Q) \;\;) \;\; :P$ that stands for if P implies Q and Q is false then P must be false;
- the material implication rule $(P \;\;) \;\; Q) \;\; , \;\; (:P \_ Q)$ that states that P implies Q is true is equivalent to the negation of P or Q is true.

III. Model

In this section, we want to show how the inner model of AlgoPath is structured in order to understand how it is possible (1) to develop AlgoPath in a distributed context and (2) to include a virtual teacher in it.

AlgoPath is based on the Model-View-Controller architecture [26] (see Fig. 3). MVC was conceived as a general solution to the problem of users controlling a large and complex data set, which is usually the case with games. MVC consists of three kinds of objects. The Model is the application object, the View is its screen presentation, and the Controller defines the way the user interface reacts to user input.

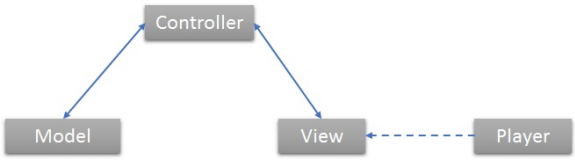
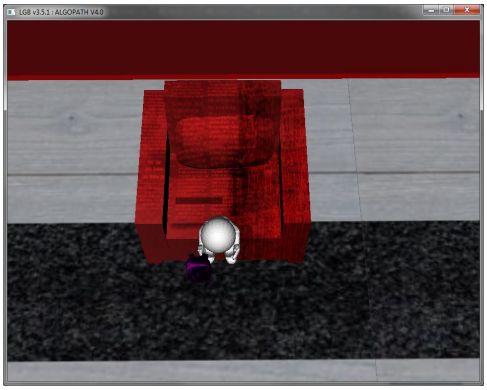


Fig. 3. The MVC architecture.

MVC is known to increase flexibility and reuse. A View must ensure that its appearance reflects the state of the Model. Whenever the Models data change (generally in response to a user interaction), the Model - through the Controller - notifies Views that depend on it. In response, each View can update itself. MVC is also known to handle pretty well multiple Views to a Model to provide different presentations.

Each non-abstract class in Fig. 4 has a visual representation in the world of AlgoPath. For example, the class "Integer" is represented by a blue hut in AlgoPath. As a consequence, we have a class diagram of the same kind for the visualization of AlgoPath (the class at the top of the hierarchy in the visualization class diagram is called "Visualization"). An "assignment" is represented by a 3D figure with a backpack running on a straight forward stone path when it is viewed in an imperative programming context (see Fig. 5(b)), but it is represented by a 3D figure sitting on a couch on a flooring carpeted with baize in an object-oriented programming context (see Fig. 5(a)). Obviously, assignments are not the only statements that can be viewed differently depending on the context. The choice of a MVC architecture was also driven by these situations because we knew that an "ElementAlgoPath" might need to be represented differently according to the context in use.



(a) View of an assignment in object-oriented language context.

(b) View of an assignment in imperative language context.

Fig. 5. Different views of an assignment in AlgoPath.

As the visualization is totally independent from the model, this architecture is also suitable if we may decide to change the world theme.

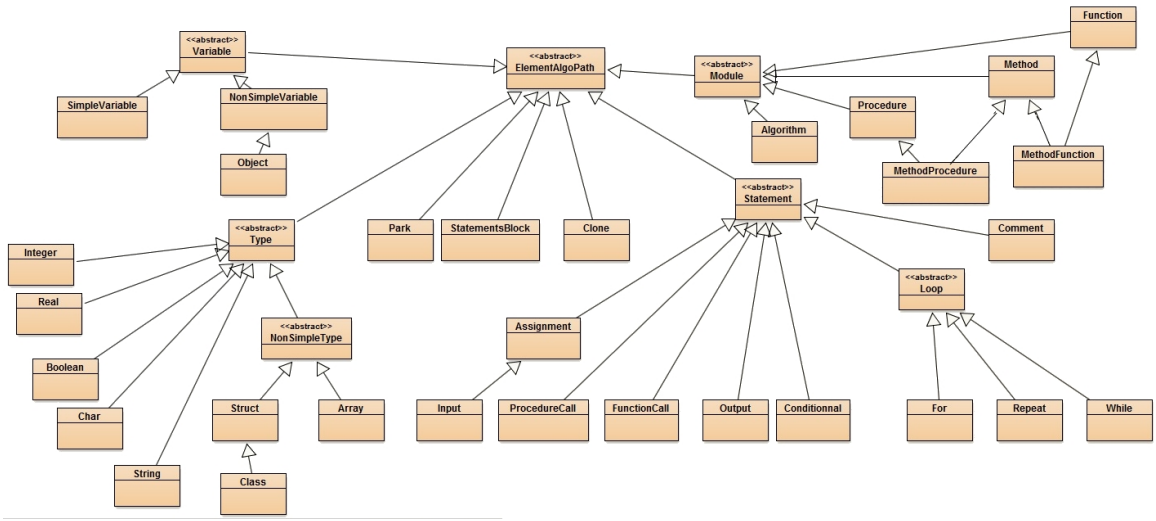


Fig. 4. Class diagram of the model of AlgoPath.

Thanks to object-oriented programming, the Controller part is very synthetic (see Fig. 6). It only has to determine which visualization part the user clicked onto, find the corresponding ElementAlgoPath data and launch the appropriate interaction. Within each "visualization" class, several interactions are stored (when the user clicks on the left button, when the user clicks on the right button and so on if necessary). Consequently, we have a third class diagram dedicated to interactions.

```
186     VisualisationAlgorithme* va = (VisualisationAlgorithme*)(A.AccedeVisualisation ()) ;
187
188     int id = recupererDessin3DDepuisCoordonneesEcran(x, y);
189     if (id != -1)
190     {
191         Visualisation * v = va -> AccedeLienIdentifiantsVisualisation()->AccedeIemeVisualisation (id) ;
192         ElementAlgoPath * el = va -> AccedeLienIdentifiantsVisualisation()->AccedeIemeElement (id) ;
193         if (v != NULL)
194             cout << id << " a une visualisation associee de type " << v -> typevisualisation () << "." << endl ;
195         if (el != NULL)
196             cout << id << " a un element AlgoPath associe de type " << el -> typeElementAlgoPath () << "." << endl ;
197         if (v != NULL)
198         {
199                 v -> AccedeInteractionSourisClicGauche () -> queFaireQuandOnClicGauche (v, el, &A) ;
200                 v_sauv = v ;
201                 el_sauv = el ;
202                 rafraichir () ;
203         }
204     }
```

Fig. 6. A portion of the code of the Controller dedicated to left clicks in AlgoPath.

IV. Virtual teacher

The previous versions of AlgoPath ([4] and [5]) were developed for a single player. AlgoPath can be use for personal entertainment or in a classroom. When used for personal entertainment, the user can click here and there but if she/he has not a goal in mind, she/he can be bored pretty quickly. That is the reason why we want to add a presence by implementing a virtual teacher helping her/him to take the game in hand. When used in classrooms, a teacher needs to explain to her/his students the algorithm to be achieved before the session starts. The dialog between students and teacher is necessary because even if AlgoPath was able to help students go through conceptual errors, it couldn't help them find the algorithm the teacher thought of. Because of that, AlgoPath couldn't be seen as a real serious game in which the goal to be achieved by the player is encoded (and not specified by a human). In this section, we present the new features we are currently implementing into AlgoPath in order to create a virtual teacher (see section IV-A). Then we explain how we modified the inner model of AlgoPath to add the virtual teacher (see section IV-B) and how AlgoPath is suitable in a single player mode (see section IV-C) and in a multi-player mode (see section IV-D).

A. New features of AlgoPath

In this section, we explain the new features we want to implement within AlgoPath:

- The teacher can design her/his own algorithms. A database of algorithms stores each algorithm the teacher wants to store. In that way at the beginning of a session, the teacher can pick a precise algorithm or AlgoPath can pick one randomly.
- A virtual teacher is designed within AlgoPath. This virtual teacher is only available if the player chooses to take it with her/him along the game. The virtual teacher can help the student if she/he discovers AlgoPath for the first time or if she/he chooses to be challenged. In both cases, the virtual teacher must oversee the interactions of the player. So it is in direct connection with the Controller part of the MVC architecture of AlgoPath presented in section III. The virtual teacher can be seen as a super controller. It can change the course of events within AlgoPath and can propose advice and hints in order to help the player find the best interaction. If necessary, at any time, the user can ask the virtual teacher questions such as "What can I do now?". Then, knowing the logic of designing algorithms the virtual teacher can propose things to do to the student. For example, it can answer "You can declare a variable. A variable stores values. The declaration of a variable is performed by clicking on huts. There are 5 huts. The one with the blue roof declares a variable that stores an integer, the one with the purple roof declares a variable that stores a real, [...] that stores a string. Go knock on a door!". As answers

like this is totally related to logic, the real teacher cannot sway the sayings of the virtual teacher. But it is totally different when the user makes a mistake when designing an algorithm chosen by the real teacher. Logic can lead to a logical algorithm. The methodology is correct and the syntax is correct but what the algorithm does nobody knows. Algorithms have a purpose. They are designed for a precise reason. Therefore the logic of any algorithms does not suffice anymore. The goal has to be taken into account. Then when a mistake occurs, it is important to explain to the user why there is a mistake considering the algorithm in progress. That is the reason why we added the feature explained in the next item.

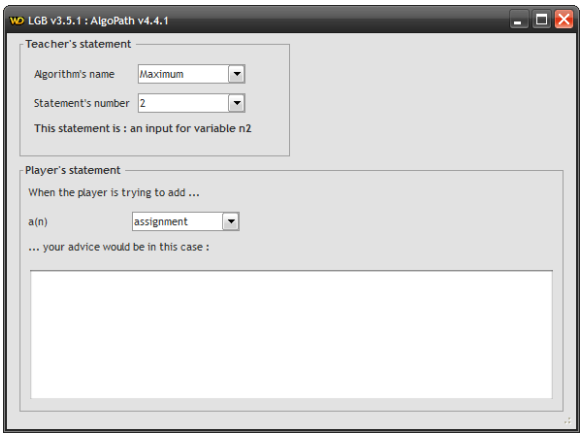


Fig. 7. The interface to set the virtual teacher for an algorithm.

- Each teacher has her/his own way of teaching. We do not want AlgoPath to be the reflection of only one way of teaching. So AlgoPath proposes a tool in which the teacher can express her/his own words (see Fig. 7). Before the player can be challenged to find an algorithm, the teacher has to fill a database of rules for the algorithm if the player is wrong. The system reviews each interaction made by the teacher during the designing of the algorithm leading to an ElementAlgoPath. It recalls each one of them to the teacher. It asks her/him what should be said if the player made the wrong interaction. There are only six possible wrong interactions. As the answer is probably linked to the wrong interaction, the teacher has to find a sentence she/he would say for every wrong interaction (pretty much all interactions but the right one). This sentence is either a tip leading to the right interaction or a specific comment regarding the wrong interaction. When these steps are fulfilled, the virtual teacher can give the comments a real teacher would give whatever the couple (wrong interaction, right interaction) is. The process of finding the algorithm must be seen in a linear way by the virtual teacher even if designing an algorithm is not necessarily sequential and continuous. For this first version, we do not want to take into account every possible order leading to the right algorithm. But this assumption is moderated for the interactions leading to the creation of variables. The variables of an algorithm belong to three sets: the input, the output and the others. Before expressing the statements of an algorithm, the player has to answer two questions: (1) what are the variables needed for input? (2) what are the variables needed for output? Once she/he has answered these questions, the player can create the variables in any order she/he wants. During the design of the statements of the algorithm, the player can also create a variable at any time. In the end, the virtual teacher only checks that the player created the right number of every type of variables for the output, the input and the algorithm.
  The real teacher has some preparatory work to do but once the game is on, the Virtual Teacher acts on its own and the interactions between the real teacher and the Virtual Teacher are no longer necessary. Let's take an example: the two first statements leading to the full algorithm of the computation of the maximum of two values are inputs. Players may interact with the wrong statements boxes. The real teacher has two choices: either she/he can write the same advice for all the wrong interactions (for example, something like "Think of asking the user the values first") or she/he can write a specific comment (for example, she/he can write "It is good to think of setting variables but ask the user instead" if the player interacts with the assignment box instead of the input box). The way teachers teach is very personal and we thought that letting them express their own words was the best option.
- The evaluation of a player is performed at the end of a game. The score is calculated according to the number of errors the player made - obviously, the fewer she/he made, the better the score is. This score is added to her/his previous scores in order to get a global evaluation. Only the teacher and the player can see the scores.

B. Model

In the next two sections, we explain how we modified the general architecture of AlgoPath and what we added to it to fully implement the features described in the previous section (see section IV-A). Two events must be considered:

- when the virtual teacher realizes the user has made a mistake. The user has just interacted but it is not the right interaction. The hints and tips given by the virtual teacher are directly linked to the algorithm in progress and the pedagogy of the real teacher.
- when the user asks a question to the virtual teacher. The user does not know what to do and ask a question prior to any interaction. The hints and tips given by the virtual teacher are linked to logic.

1) The virtual teacher realizes an error has been made: The real teacher creates algorithms and states sentences the system stores into the Model. The player tries to find the algorithm the teacher or AlgoPath selected. The Controller catches interactions. If necessary, it passes them to the Virtual Teacher. The Virtual Teacher accesses the real teacher's sentences. It is

also an engine following simple rules to oversee the Controller. The Virtual Teacher computes if the interaction is correct or not. If it is, the whole system acts as if the Virtual Teacher was not there. If it is not, the Virtual Teacher stops the interaction, disrupts the system and takes over the actions of the View. The View displays the specific guidance in correlation with the stopped interaction. Another interaction may occur again and the system iterates. See Fig. 8 for an overview of this process.

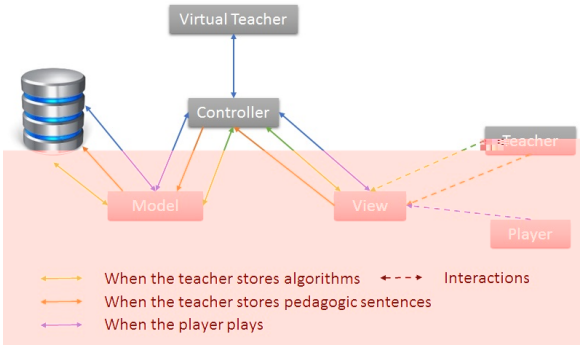


Fig. 8. A possible integration of a virtual teacher inside the MVC architecture.

The database shown in Fig. 8 can be split into two independent parts: one for data of algorithms (the ones the teacher designs and the ones the player designs) and one for data of the virtual teacher (rules and sentences it must access).

Fig. 8 also summarizes the three modes:

- When the teacher stores algorithms to be further used in the classroom or at home. In that case, the Virtual Teacher is off.
- When the player plays. In that case, the Virtual Teacher is on.
- When the teacher stores pedagogic sentences she/he would say in the classroom if errors were to be made by the player. In that case, the Virtual Player is off.

2) The user asks a question: Logic of any algorithms is stored within the virtual teacher of AlgoPath using first order logic like we explained it can be done in section II-C. Let us show you some of the rules we created to formalize the logic of algorithms when the user can create declarations, assignments, conditional statements and loops (obviously it goes on with functions and procedures and structures but is not mentioned here):

- Rule number 1 is $) DECLARATIONS\_ARE\_ALLOWED$. It means that when nothing has been done, the user can declare variables.
- Rule number 2 is $8v : IS\_DECLARED(v)) CAN\_BE\_DECLARED(v)$. It means that if a variable named v is not declared yet then it is possible to declare a variable named v.
- Rule number 3 is $8v IS\_DECLARED(v)) CAN\_BE\_ASSIGNED(v)$. It means that if a variable named v is declared then it is possible to create assignments on it.
- Rule number 4 is $8v IS\_DECLARED(v)) CAN\_BE\_INPUT(v)$. It means that if a variable named v is declared then it is possible to create inputs on it.
- Rule number 5 is $9v IS\_ASSIGNED(v)) OUTPUTS\_ARE\_ALLOWED \wedge CONDITIONALS\_ARE\_ALLOWED \wedge LOOPS\_ARE\_ALLOWED$. It means that if there exists a variable that is assigned a value then the user can create outputs, conditional statements and loops. Obviously, at first anyway, only statements on v will be allowed. This is controlled by rules numbered 7, 10 and 11.
- Rule number 6 is $9v IS\_DECLARED(v)) ASSIGNMENTS\_ARE\_ALLOWED \wedge INPUTS\_ARE\_ALLOWED$. It means that if somehow the user succeeded in creating the declaration of a variable, then assignments and inputs are now possible. Obviously, at first anyway, only statements on v will be allowed. This is controlled by rules numbered 3 and 4.
- Rule number 7 is $8v IS\_ASSIGNED(v)) CAN\_BE\_OUPUT(v)$. It means that if the variable v is assigned a value then the user can create an output on it.
- Rule number 8 is $8v9x(CAN\_BE\_ASSIGNED(v) \wedge IS\_ASSIGNED(x))) CAN\_BE\_USED\_IN\_RVALUE(v; x)$. It means that if the variable x is assigned and if an assignment on v can be performed then the variable x can be used in the right value of the assignment of v.
- Rule number 9 is $9v(IS\_DECLARED(v) \wedge INTEGER(v))) COUNT\_LOOPS(v)$. It means that if there exists a variable named v that is declared and is an integer then the user can create count loops whose exit condition is based on v.
- Rule number 10 is $9v IS\_ASSIGNED(v)) CONDITIONAL\_LOOPS(v)$. It means that if there exists a variable named v that is assigned a value then the user can create conditional loops whose exit condition is based on v.
- Rule number 11 is $9v IS\_ASSIGNED(v)) CONDITIONAL\_STATEMENTS(v)$. It means that if there exists a variable named v that is assigned a value then the user can create conditional statements whose conditions are based on v.

Within this new version of AlgoPath, the user can only ask one question that is "What can I do?". When this question is asked, the virtual teacher launches the inference engine. The inference engine acts as follows:

1) It detects all the rules that match the knowledge base.
2) It selects everyone of them and executes them. The order it applies them is the order of the numbers of the rules.

It adds new facts in the knowledge base.

3) It iterates until no new rules are a match.
4) It gives new facts to the virtual teacher.
5) The virtual teacher expresses the answers to the user.

Table I shows the answers the virtual teacher can say to the user when a new fact is discovered.

Let us see how it goes with a few examples:

C. Single Player

When this tool is fully developed, we will be able to use Algopath in either single or multi-player modes. This section explains how it can be used by a single player. There are two cases:

1) The player is self-thinking in a classroom. Before the course, the teacher has prepared her/his exercise within AlgoPath. She/he can explain the subject of the exercise and afterwards gives them the name of the algorithm. Instead of taking a pen and a paper, each student begins to play with AlgoPath. They choose the algorithm thanks to the name the teacher gave them and try to find the algorithm by themselves. The Virtual Teacher helps them go through the process at any time if necessary.
2) The player is self-thinking at home. After school, a student can try again the exercise the netwousolseayerVirject y TdedalgorThis section

New Fact	Sentence
DECLARATIONS_ARE_ALLOWED	You can declare variables. Variables store values. The declaration of a variable is performed by clicking on huts. There are 5 huts. The one with the blue roof declares a variable that stores an integer, the one with the purple roof declares a variable that stores a real, [...] that stores a string. Go knock on some doors!
ASSIGNMENTS_ARE_ALLOWED	You can assign values to variables. Two how the computer store values. Values must be of the same type as variables. The only exception is for real variables that can store integers or reals. Assignments are created by clicking on the box (image of the top of the box here). Go click on that box!
INPUTS_ARE_ALLOWED	When you do not know what values you can assign to variables, you can ask the future user of your algorithm. Asking the user to give you a value is called an input. Inputs are created by clicking on the box (image of the top of the box here). Go click on that box!
OUTPUTS_ARE_ALLOWED	Now, you can show values to the future user of your algorithm. Showing her/him values is called outputs. Ouputs are created by clicking on the box (image of the top of the box here). Go click on that box!
CONDITIONALS_ARE_ALLOWED	Great, you can create conditional statements from now on. Conditional statements allow your algorithm to execute certain statements based on a decision. Conditional statements are created by clicking on the box (image of the top of the box here). Go click on that box!
LOOPS_ARE_ALLOWED	You can also create loops. Loops repeat continually some statements until a certain condition is reached. Loops are created by clicking on the box (image of the top of the box here). Go click on that box!
CAN_BE_DECLARED(v)	Fact not available since the only question is "What can I do?".
CAN_BE_ASSIGNED(v)	If you don't know what assignment create, you can assign a value to the variable named v.
CAN_BE_INPUT(v)	If you don't know what input create, you can ask the future user of your algorithm the value he wants to assign to the variable v.
CAN_BE_OUTPUT(v)	If you don't know what output create, you can show the future user of your algorithm the value stored in the variable v.
CAN_BE_USED_IN_RVALUE($v;x$)	You can use the variable named x to assign a value to the variable named v.
CONDITIONAL_STATEMENTS(v)	You can create a conditional statement whose condition involves the variable named v.
COUNT_LOOPS(v)	You can create a count loop whose counter is the variable named v.
CONDITIONAL_LOOPS(v)	You can create a conditional loop whose condition involves the variable named v.

TABLE I
SENTENCE SAID BY THE VIRTUAL TEACHER WHEN A NEW FACT IS DISCOVERED.

- XML3D is very similar to X3Dom. It proposes to extend the existing properties of HTML wherever possible, embedding 3D content into a web page, with the maximum reuse of existing features.
- Xflow is a solution to process intensive data on the web. It is integrated into XML3D. It provides character animation, simulations of various types (fire, smoke, air movements, ...).
- CSS 3D enables to format elements using 3D transforms (rotations, scales, translations). But CSS 3D cannot perform true light and shadow.
- Google O3D is an open-source JavaScript API for creating interactive 3D applications in the browser. It is viewed as bridging the gap between desktop based 3D accelerated graphics applications and HTML based web browsers.
- WebGL (Web Graphics Library) is a JavaScript API for rendering interactive 3D graphics within any compatible web browser without the use of plug-ins. It is intended to be used by experienced graphics programmers. Several libraries originated from WebGL such as SpideGL or LightGL or Three.js.
- Java3D features a full scene graph and is able to render using Direct3D or OpenGL. JOGL (Java OpenGL) and LWJGL (LightWeight Java Game Library - it featured Minecraft, the popular multiplatform game) are libraries that provide 3D graphics, 3D sound and controllers (joysticks, ...) to applications.

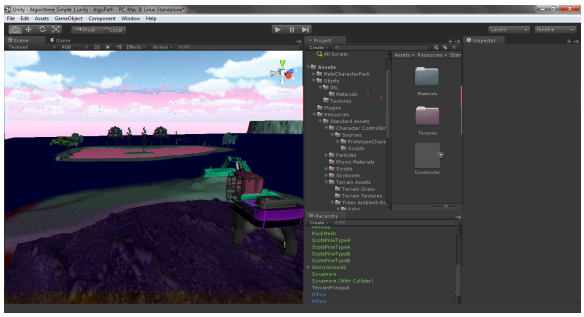


Fig. 9. Distributed version of AlgoPath in progress using Unity.

The selection of the appropriate technology depends on the needs and requirements of the application developed. Unity [33] is a proprietary videogame engine which, compared to the technologies listed above, is highly superior, aiming to be an application to create video games. It is powerful and fully integrated with a complete set of tools and rapid workflows to create interactive 3D and 2D contents. So we chose Unity to develop AlgoPath in a distributed way. The implementation is at its beginning (see Fig. 9). For us, one of the main interests of Unity is that a Unity scene can be created entirely in code. Unity accepts C# or JavaScript code but it is also possible to import our own library. The latter lets us import the M-part

and the C-part of the MVC model of AlgoPath, the V-part being totally handled by Unity.

VI. Conclusion and future work

In this paper, we explained how we made the model of AlgoPath evolved in order to add new features, turning it into a distributed virtual environment in which students can play in a collaborative or competitive way. The main feature is a Virtual Teacher. The Virtual Teacher intervenes whenever necessary to give the player hints and tips. This advice comes from a real teacher who has to specify her/his own words thanks to a tool embedded in AlgoPath. Consequently, her/his pedagogy goes beyond the classroom. This new version of AlgoPath will be implemented using Unity, a videogame engine.

Future work will focus on realizing focus groups so that AlgoPath can be properly evaluated. Some have already begun with the University of Lorraine in France. Results will show if the new features are appreciated by teachers and students or not. Moreover, we want to improve the quality of the virtual teacher. We stated that the virtual teacher has to follow the flow of statements and therefore, so have the players. When in practice programmers do not think straight but go back and forth. More thoughts will also have to be given to implement a collaboration during oriented-object programming.

References

[1] M. Prensky, "Digital Natives, Digital Immigrants," *On the Horizon*, vol. 9, no. 5, Oct. 2001.

[2] N. Ellis, "Chapter 21 word meaning and the links between the verbal system and modalities of perception and imagery or in verbal memory the eyes see vividly, but ears only faintly hear, fingers barely feel and the nose doesn't know," in *Mental Images in Human Cognition*, ser. Advances in Psychology, R. H. Logie and M. Denis, Eds. North-Holland, Jan. 1991, vol. 80, pp. 313 – 329.

[3] M. Chau, A. Wong, M. Wang, S. Lai, K. W. Chan, T. M. Li, D. Chu, I. K. Chan, and W. ki Sung, "Using 3d virtual environments to facilitate students in constructivist learning," *Decision Support Systems*, vol. 56, no. 0, pp. 115 – 121, 2013.

[4] E. Perrin, S. Linck, and F. Danesi, "Algopath: A new way of learning algorithmic," in *The Fifth International Conference on Advances in Computer-Human Interactions*, Valencia, Spain, Jan. 2012, pp. 291–296.

[5] E. Perrin and S. Linck, "Algopaths new interface helps you find your way through common algorithmic mistakes," in *The Sixth International Conference on Advances in Computer-Human Interactions*, Nice, France, Feb. 2013, pp. 188–193.

[6] M. Deutsch, "A Theory of Co-operation and Competition," *Human Relations*, vol. 2, no. 2, pp. 129–152, Apr. 1949.

[7] ——, "An experimental study of the effects of co-operation and competition upon group process." *Human relations*, 1949.

[8] Z. Qin, D. W. Johnson, and R. T. Johnson, "Cooperative versus competitive efforts and problem solving," *Review of educational Research*, vol. 65, no. 2, pp. 129–143, 1995.

[9] Z. Aziz and M. A. Hossain, "A comparison of cooperative learning and conventional teaching on students achievement in secondary mathematics," *Procedia - Social and Behavioral Sciences*, vol. 9, no. 0, pp. 53 – 62, 2010, world Conference on Learning, Teaching and Administration Papers.

[10] V. K. Kuntz, "Bretter, die die welt bedeuten," Nov. 2012.

[11] F. Ke, "A case study of computer gaming for math: Engaged learning from gameplay?" *Computers & Education*, vol. 51, no. 4, pp. 1609 – 1620, 2008.

[12] R. Hamalainen, "Designing and evaluating collaboration in a virtual game environment for vocational learning," *Computers & Education*, vol. 50, no. 1, pp. 98 – 109, 2008.

[13] W. Peng and G. Hsieh, "The influence of competition, cooperation, and player relationship in a motor performance centered computer game," *Computers in Human Behavior*, vol. 28, no. 6, pp. 2100 – 2106, 2012.

[14] V. Wendel, S. Göbel, and R. Steinmetz, "Collaborative learning in multiplayer serious games," in *Clash of Realities Proceedings 2012*, W. Kaminski, Ed. München: KoPäd Verlag, Sep 2012.

[15] V. Wendel, M. Gutjahr, S. Göbel, and R. Steinmetz, "Designing collaborative multiplayer serious games for collaborative learning," *Proceedings of the CSEDU*, vol. 2012, 2012.

[16] B. Fernndez-Gallego, M. Lama, J. C. Vidal, and M. Mucientes, "Learning analytics framework for educational virtual worlds," *Procedia Computer Science*, vol. 25, no. 0, pp. 443 – 447, 2013, 2013 International Conference on Virtual and Augmented Reality in Education.

[17] S. Schiaffino, P. Garcia, and A. Amandi, "eteacher: Providing personalized assistance to e-learning students," *Computers & Education*, vol. 51, no. 4, pp. 1744 – 1754, 2008.

[18] zcan zyurt, H. zyurt, A. Baki, and B. Gven, "Integration into mathematics classrooms of an adaptive and intelligent individualized e-learning environment: Implementation and evaluation of {UZWEBMAT}," *Computers in Human Behavior*, vol. 29, no. 3, pp. 726 – 738, 2013.

[19] D. Xu and H. Wang, "Intelligent agent supported personalization for virtual learning environments," *Decision Support Systems*, vol. 42, no. 2, pp. 825 – 843, 2006.

[20] F. A. M. Fonte, J. C. Burguillo, and M. L. Nistal, "An intelligent tutoring module controlled by {BDI} agents for an e-learning platform," *Expert Systems with Applications*, vol. 39, no. 8, pp. 7546 – 7554, 2012.

[21] C. Buche and R. Querrec, "An expert system manipulating knowledge to help human learners into virtual environment," *Expert Systems with Applications*, vol. 38, no. 7, pp. 8446 – 8457, 2011.

[22] R. Schleiffer, "An intelligent agent model," *European Journal of Operational Research*, vol. 166, no. 3, pp. 666–693, Nov. 2005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0377221704004096

[23] S. d. Freitas and F. Liarokapis, "Serious games: A new paradigm for education?" in *Serious Games and Edutainment Applications*, M. Ma, A. Oikonomou, and L. C. Jain, Eds. Springer London, Jan. 2011, pp. 9–23. [Online]. Available: http://link.springer.com/chapter/10.1007/978-1-4471-2161-9_2

[24] V. Y. Terziyan and S. Puuronen, "Knowledge acquisition based on semantic balance of internal and external knowledge," in *Multiple Approaches to Intelligent Systems*, ser. Lecture Notes in Computer Science, I. Imam, Y. Kodratoff, A. El-Dessouki, and M. Ali, Eds. Springer Berlin Heidelberg, Jan. 1999, no. 1611, pp. 353–361.

[25] F. Nassiri-Mofakham, "How does an intelligent agent infer and translate?" *Computers in Human Behavior*, vol. 38, pp. 196–200, Sep. 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0747563214002970

[26] T. M. H. Reenskaug, "The original MVC reports," Feb. 2007.

[27] S. Singhal and M. Zyda, *Networked Virtual Environments: Design and Implementation*. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1999.

[28] Z. M. B. D. McGregor D., Kapolka A., "Requirements for large-scale networked virtual environments," *Telecommunications, 2003. ConTEL 2003. Proceedings of the 7th International Conference on*, vol. 1, pp. 353–358, Jun. 2003.

[29] D. N. B. Ta and S. Zhou, "A two-phase approach to interactivity enhancement for large-scale distributed virtual environments," *Computer Networks*, vol. 51, no. 14, pp. 4131 – 4152, 2007.

[30] S. Zhou, W. Cai, B.-S. Lee, and S. J. Turner, "Time-space consistency in large-scale distributed virtual environments," *ACM Trans. Model. Comput. Simul.*, vol. 14, no. 1, pp. 31–47, Jan. 2004.

[31] "VTK - The Visualization Toolkit." [Online]. Available: http://www.vtk.org

[32] A. Evans, M. Romeo, A. Bahrehmand, J. Agenjo, and J. Blat, "3d graphics on the web: A survey," *Computers & Graphics*, vol. 41, no. 0, pp. 43 – 61, 2014.

[33] "Unity - Game Engine." [Online]. Available: http://unity3d.com