Personalised e-learning through an Educational Virtual Reality Game using Web Services

George Katsionis & Maria Virvou
Department of Informatics, University of Piraeus, 80 Karaoli & Dimitriou Str.,
18534 Piraeus, Greece
{gkatsion@kman.gr, mvirvou@unipi.gr}

Abstract. This paper describes how a personalised educational game architecture has been used in conjunction with Web services to provide remote access to the system. The educational game is a Virtual Reality adventure game that performs affective user modelling by measuring emotional characteristics of users. Virtual Reality (VR) games are so popular among children and adolescents that can be used for the purposes of educational software to render it more attractive and motivating. The benefits of such an application can be maximised if it is available over the Web. Software applications that operate over the Web are targeted to a wide range of users. Hence they need a high degree of adaptivity and dynamic individualisation to each user that interacts with the application. This should include the students' emotional state that affects their learning. However, the environment of a Web-based VR-game that performs user modelling is so demanding that the technology of Web Services is necessary for its effective operation and interoperability. Moreover, reusability may be achieved for the user modelling component.

Keywords: Web services, Intelligent Tutoring Systems, Educational game, Virtual reality, Affective user modelling.

1 Introduction

Software games are mainly created for entertainment. However, their great popularity among children and adolescents may also be exploited for the purpose of education. Papert [36] notes that software games teach children that some forms of learning are fast-paced, immensely compelling and rewarding whereas by comparison, school strikes many young people as slow and boring. As a result, many researchers have developed games for educational purposes (e.g. [4], [13], and [26]). Most of these software games operate as standalone applications, which do not operate over the Web. On the other hand, the effects of educational games could be maximised if they were available over the Web. In such a case, educational games would be widely available for students to use them from school at school-time or home at their leisure time and learn while being amused. Indeed, the benefits of the Web for educational software are indisputable, since it renders tutoring systems accessible by anyone, at any time, from any place. Thus, during the last decade, a lot of research effort has been put in Web-based education (e.g. [2], [29], [33]).

In the case of Web-based education, software applications are targeted to a wide audience of students of various backgrounds and needs. Thus, the need for personalisation of the software is vital. To this end, it is very appropriate to use the technology of Intelligent Tutoring

Systems (ITSs), which are computer-based systems that aim primarily at adapting dynamically to the needs of individual students. ITSs provide individualised instruction by maintaining detailed models of their students [21]. Student modelling is of primary importance in ITSs as has been acknowledged by many ITS researchers (e.g. [14], [28] and [32]).

However, one aspect of students that plays an important role in each student's learning and has been overlooked so far is affect. How people feel has been argued to be of great importance during their cognitive processes [19]. Indeed, the existence of affective information in the students' models improves the intelligent tutoring system, by allowing it to provide more adequate help for the student [44]. Especially for the case of Web-based educational software that aims at being used by students without the physical presence of a human instructor, ITSs have to incorporate as many affective features as possible. This should be done to compensate for the emotional human-human interaction between human teachers and students that is missing from the asynchronous Web-based instruction. In the case of a Web-based ITS that operates as a VR adventure game, student-players are bound to experience many different emotions while the plot of the game is taking place in parallel with the presentation of lessons and assessment of students' knowledge.

A quite recent review [8] has shown that all well-known technologies from the areas of ITS have already been re-implemented for the Web. Usually, in Web-based ITSs, information about the learner is no longer stored locally on each learner's computer but in a central repository that can be accessed by any client of the application that requests it. This is usually implemented by using client-server architectures.

However, so far, there have not been Web-based ITSs that operate as virtual reality educational games. Such applications are quite demanding and need special emphasis on design issues to be able to operate through the Web. Moreover, developing new Web-based educational systems still starts from scratch. Therefore development of Web-based ITSs requires enormous time and effort. This problem is partly due to the fact that there are very few educational components available on the Web that can be shared and reused [50]. Thus, there exists a need for a standard mechanism that supports all aspects of the web based learning process. This includes finding suitable learning components or learning management services, getting information about their services and invoking their services. It is also wise to separate learning objects from the educational applications [50]. By doing this, new Web based educational systems can use previously created learning objects and achieve knowledge sharing and reducing the heterogeneity of Web based distributed systems [41], [52].

A solution to the above problems may be given by the technology of Web services. As Chen [50] points out, using Web Services for the educational components can increase the interoperability and reusability of a Web-based educational system, and can greatly reduce the time and effort spent on building it. In this way the development of such applications becomes less tedious and more efficient.

In view of the above, this paper describes and discusses the use of Web services for the development of a virtual reality educational game that can operate as a Web-based ITS. The educational game that has been developed for the purposes of our research is called VIRGE. VIRGE works as an adventure VR-game but it has educational content as well and supports personalised learning based on a student modelling component. The student modelling component focuses on perceiving the users' cognitive and emotional state while they play the educational VR-game and while they answer questions concerning the lessons taught. Affective and cognitive user modelling is achieved by combining evidence from students' errors

and from detectable user behaviour characteristics. In this way the system can dynamically generate personalised advice based on the student's affective and cognitive needs.

The fact that the gaming environment that cooperates with the underlying reasoning mechanisms of the ITS is very demanding and needs special attention for its efficient implementation over the Web, has rendered the technology of Web services necessary. Moreover, the use of Web Services allows the student modelling component to be accessible to other educational applications. An earlier preliminary version of VIRGE has been briefly described in [47].

2 Related work

2.1 Web services

Recently, the technology of Web services has attracted a lot of attention within the computer research and industrial community. The concept of Web services is widely considered as one of the next steps in the evolution of application integration among other steps such as, recent developments in the fields of software architecture, design and architectural patterns and component-based engineering.

Web services can be defined as self-contained, self-describing, modular applications, generally independent, that can be published, discovered and invoked across the Internet or an institution intranet [38]. A Web service is a collection of functions that operate as a single entity and are published to the network for use by other programs. Web services are building blocks for creating open distributed systems [24], [15]. There is another, somewhat broader, view of Web services in which anything available over the network, with some associated description of its functional aspects, is considered a Web service.

The main purpose of Web services is to allow services to work together seamlessly because they are developed using the same standards for self-description, publication, location, communication, invocation, and data exchange capabilities [39]. Thus, by using Internet protocols, such as HTTP (HyperText Transfer Protocol) and XML technology, Web services contribute to the interoperability of the components on different platforms and are implemented in different programming languages. The components use SOAP (Simple Object Access Protocol) [7] as a message-based communication protocol to talk to each other. WSDL (Web Service Description Language) [11] is used to define and describe the interfaces of the components. UDDI (Universal Description, Discovery and Integration) [43] is used for service discovery and integration. All of these specifications are W3C (World Wide Web Consortium) [20] standards and are widely supported by industrial companies, such as IBM [23], Microsoft [30], BEA and others. The development and deployment complexity are greatly reduced. In simple terms, Web services are basically designed to allow loose coupling between client and server, and they do not require clients to use a specific platform or language. For these reasons among others, these services are becoming very popular. Today, the majority of software companies are implementing tools based on these new standards [23], [30]. Their use in many different kinds of applications is increasing rapidly.

Additionally to the interoperability, one important advantage of Web Services is the potential reusability of components. The technologies chosen for Web services are inherently neutral to compatibility issues that exist between programming languages, middleware solutions, and operating platforms. As a result, applications using Web services can dynamically locate and use necessary functionality, whether available locally or from across the Internet. As Tsalgatidou and Pilioura [42] point out, the Web Service paradigm motivates developers to build applications by locating and using existing Web Services rather than building the required functionality from scratch. Also, as Ciganek, Haines and Haseman [12] point out, key benefits that are frequently associated with Web services are that they offer easier and cheaper connectivity, and can be leveraged to create a more flexible IT infrastructure, and possibly generate new revenue streams.

However, these benefits of Web Services are not by default available to applications if these applications are not designed appropriately. Web Services provide the technology to be used by application developers and thus it is the application developers' problem to create applications that make use of the features of this technology. This is not trivial, especially in cases of complex applications such as Intelligent Tutoring Systems that operate as virtual reality games over the Internet.

2.2 Web services for Web based education

One important application area that may benefit from the advantages of the technology of Web Services is the field of education, where there exist very demanding applications such as ITSs with individualised user models that have to be transferred over the Web. So far, most of the implementations of ITSs over the Web that have been described in the literature were not based on Web services. For example, a method used for the deployment of an ITS over the Web [2] was to take all the parts of the client program and provide them through the Internet, in the form of an applet running on a web page. In that way, the client-server program worked by having the client as Web pages on the Internet and the server located at the Web server where the Web pages existed. The server was receiving information from the applet of the Web page. Examples of other intelligent learning environments that are based on client-server architectures but did not use Web services are WITS [33] and ILESA [29].

The kind of architecture described above has many advantages due to the fact that the client runs on a web browser and is not installed on its client machine. However, some tutoring systems like VIRGE are too demanding on resources to be implemented in this kind of architecture. The 3D environment of VIRGE in which the students can play and learn renders the application very attractive but at the same time requires a lot of memory and CPU usage. This is certainly expected if one takes into consideration the requirements of most of the commercial games. Thus, the virtual game was too heavy to work through Web pages and would take over 15 minutes just to load each of the Web pages. Navigating through the virtual worlds would not be possible.

One additional limitation of educational applications over the Internet that have been developed so far is that reusability of functionalities has not been achieved by any of the previous Web-technologies that were used in Web-based education. For example, Mizoguchi and his colleagues [31] enumerate many drawbacks of current intelligent Web-based educational systems. One of these drawbacks is that building educational systems requires a lot of work

because it is always built from scratch and that it is not easy to specify functionalities of components in educational systems. Devedzic [17] as well, discussed the limitations of current Web based educational applications and proposed possible solutions. He pointed out that the next-generation Web-based applications should pay more attention to knowledge sharing issues and standardisation efforts in Web development. However, there was not any running educational application introduced in that paper. Current limitations of Web-based tutoring systems have led many other researchers as well to propose preliminary solutions for Web based applications that make use of Web Services. For example, Wen and Jesshope [51] have proposed a Web Service management architecture and a model for constructing decentralised Virtual Learning Environments. However, their approach has not yet been implemented. Similarly, Xu, Yin and Saddik [52] have proposed a flexible integration model for dynamic e-learning systems where all the learning components and applications are defined and loosely connected using Web Services. However, their model too has not been implemented yet.

As it appears from current research in the area of Web-based educational applications, the benefits of Web Services seem very promising to researchers. As a result, there have already been preliminary proposals on how Web Services can be used effectively in Web-based applications but there have not yet been many implementations that show clearly what an educational Web-based architecture can be in an actual running system.

In view of the above, in this paper, we give an educational architecture that is based on Web Services for a demanding running application that uses a Virtual Reality gaming environment and performs user modelling. Our architecture and the allocation of the components of VIRGE have been fully implemented and show how this system can operate effectively over the Web using Web Services. By separating the student modelling and educational components, in the form of Web services, from the other parts of the ITS we can have these components work through the Internet while the other parts of the application are installed locally. Furthermore, these modelling components constitute the essential part of the reasoning of an ITS and might be reused by other educational applications. In this way, Web service implementations of ITSs can contribute to the reusability of their key functionalities if they are designed appropriately.

2.3 ITSs and affective computing

The latest scientific findings indicate that emotions play an essential role in decision-making, perception and learning. They influence the very mechanisms of rational thinking. According to Picard [37], if we want computers to be genuinely intelligent and to interact naturally with us, we must give computers the ability to recognise, understand, and even to have and express emotions.

To support this need of educational games in VIRGE, there are agents that, as part of game playing, generate tailored interventions aimed at stimulating the student to learn from the game. The interventions are based on information that has been silently collected about the users' current emotional and cognitive state. The interventions are kept to the minimum at the occasions when the system judges it is necesserary to intervene. This is done to avoid too much interference while users are playing the game.

Toward this direction, we have devised a mechanism for the automatic collection of probabilistic information of student affective states that the agents of VIRGE can use. This infor-

mation is used in conjunction with information on student answers' results, to generate interventions that improve learning without compromising engagement. The affective information includes many characteristics of each student's behaviour and possible emotional state, and can provide useful guidance to the agents' provision of help and advice.

Our main objective is to find out when a characteristic of a student's behaviour is out of its usual range. Something like that would either have positive or negative consequences. That is because the sum of such negative or positive evidence can lead to the underlying emotional states of the student. For the purpose of implementing such an environment we are using parts of the Ortony, Clore and Collins (OCC) cognitive theory of emotions [35]; a brief description of the OCC theory can be found in the Appendix of this paper. This theory is based on grouping emotions by their eliciting conditions: reactions to events and their consequences, reactions to agents and their actions, and reactions to objects. The authors of OCC believe that cooperative problem-solving systems must be able to reason about emotions and they provide suggestions for rendering computational implementation easier.

In this paper, we are mostly interested in the reactions to events and the group of emotions that may be inferred. We are focusing on the part of the OCC cognitive theory of emotions that suggests computational methods about measuring the intensity of an emotion. Our aim is to find out the occurrence and intensity of possible experienced emotions in our application, by measuring the value of their intensity variables just like the OCC theory suggests. What we are doing is: First, we define some behavioural characteristics of the students that are the intensity variables of their affective states, and can reveal the student's affective state. Secondly, we calculate the intensity values of these characteristics. Finally by using the OCC theory we try to estimate the emotional significance of these values.

Affective user modelling is a rather new research area, therefore only a few computational modes of user affect have been devised to date. A model that uses fuzzy rules to assess anxiety in combat pilots is described in [22]. Due to the specificity of the modelling task, the model does not need to deal with the high level of uncertainty involved in modelling affect in less constraining interactions, like those generated by educational games. There are other educational applications that are based on the OCC theory and are discussed in [1], [13], and [18]. More specifically, the model of Conati and Zhou [13] relies on a Dynamic Decision Network (DDN) to probabilistically integrate information on both the possible causes of affective reaction and its observable effects. It does so by representing how game events relate to the students' goals, and is trying to find possible affective states that depend on the satisfaction or not of these goals. The overall goal of [1] is to create a framework that allows the user to define the characteristics that the emotions agents can display for a given interaction. In addition they define how these characteristics affect their actions, and hence the interactions. However, unlike our research, none of the above applications refers to virtual reality games, which typically provoke a wealth of emotions to users.

3 Operation of the virtual reality game

The virtual reality game that we have developed, is an ITS for teaching English orthography and grammatical rules. This ITS-game is called VIRGE (Virtual Reality Game for English). In particular, VIRGE integrates the VR-game with the reasoning of an ITS. Students

have the opportunity to play a 3D game, similar to the commercial ones, which enables them to learn while playing.

More specifically, VIRGE has features that are quite common in virtual reality adventure games [25]. Such features include dungeons, dragons, castles, keys etc. The difference of VIRGE, which is an educational game, is that one must fight one's way through by using one's knowledge. In VIRGE, the student-player tries to reach the "land of knowledge" and find the treasure, which is hidden there. However, to achieve this, the player has to obtain a good score, which is accumulated while the player navigates through the virtual world and answers questions concerning English spelling.

VIRGE constructs a student-model for each individual student by taking into account the history of answers of students. VIRGE also monitors closely the actions of students while they play the game, and it updates the individual student models. As a result, it generates individualised instruction and advice for students based on their student models. The student characteristics that are being modelled concern the knowledge level of students (answers' results - errors) as well as their behaviour while they play and learn (users' actions/ characteristics), which can be connected to their emotions.

Agents have been quite successful at observing users' behaviour and, therefore, they have been widely used in learning environments in order to capture the users' characteristics and perform user-modelling tasks [6], [34] and [46]. Software agents play an important role in human-computer interaction and in the coordination of the internal processes of the system [5]. VIRGE is based on a multi-agent architecture. There exist three types of agents. The first type of agent is the user interface agent. User interface agents are the animated agents that interact with the user. The two other types of agents are not visible to the user. The second type of agent is the speech synthesising agent of the game. Finally, the third type of agent is the student modeller agent, which is responsible for forming and updating the long term student-model, which consists of cognitive and behavioural characteristics of the student. Among those agents, the student modelling agent is an intelligent agent that uses reasoning mechanisms to make inferences about students. The rest of the agents are software components that interoperate.

More specifically the animated agents are visible to the user and are responsible for asking questions, forming and giving advice and interacting with the student-players. There are three different animated agents, the guard of a passage, the advisor and the student's companion. Guards of passages ask questions to players in order to let them continue their way into the passage and receive more points for their total score. Animated agents, who act as advisors, lead the student to lessons that s/he has to read or repeat, and make some general remarks about the student's behavioural characteristics. Finally, companions are responsible for showing empathy to the students and helping them when they make mistakes. Moreover, they help the students in managing their emotions while they play and answer questions.

The speech synthesising agent is responsible for giving voice to the animated agents described above. The student modeller passes information about the student to the advisor and the companion agents so that they may give proper and individualised advice to each student.

From the above five agents in total, of three different types , only one of them is responsible for interacting with the Web. That is the student modeller agent. This works through the Internet and either updates the model of the student that is kept on the Web server, or provides the advisor and companion agents with useful information.

The architecture of the multi agent part of the system is illustrated in Figure 1. In this figure, each of the five agents is illustrated in an ellipsis. Their functions are shaped in rectangles. The student interacts with the guard agent who poses the questions, using the speech synthesising agent. The student's answers' characteristics and results are taken to the student modeller agent, which updates the student model at the Web server. The student modeller provides the advisor and companion agent with the information presented to the student.

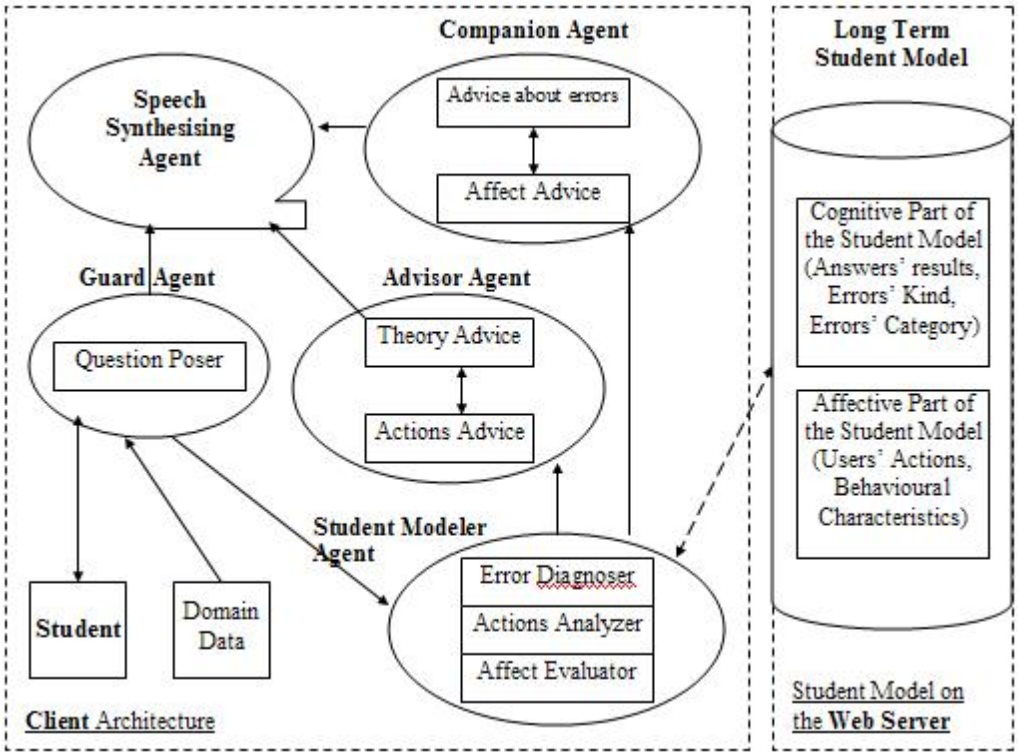

Figure 1: Multi agent system architecture.

More specifically, the animated agent who acts as a guard of a passage is a dragon that appears outside every door in the virtual worlds of the game, as illustrated in Figure 2. This animated agent is responsible for posing questions to students about the domain being taught. Every time a student comes across this particular agent then the guard agent randomly retrieves a question from the domain knowledge data and poses it to the student. The student gives an answer in the appropriate dialog box. If the answer is correct then the door opens and the student is allowed to continue his/her navigation. Otherwise the student can try again, or just accept his/her mistake, see the correct answer and move on. The student modeller performs error diagnosis and affect evaluation every time there is an interaction of the student with the guard agent and updates the student model.

The questions posed to the student constitute tests relating to the factual knowledge of the domain. Tests are part of the domain data and consist of questions about the English grammar of the following types:

1. Spelling questions.
2. Plural form of nouns.
3. Comparative and superlative form of adjectives.

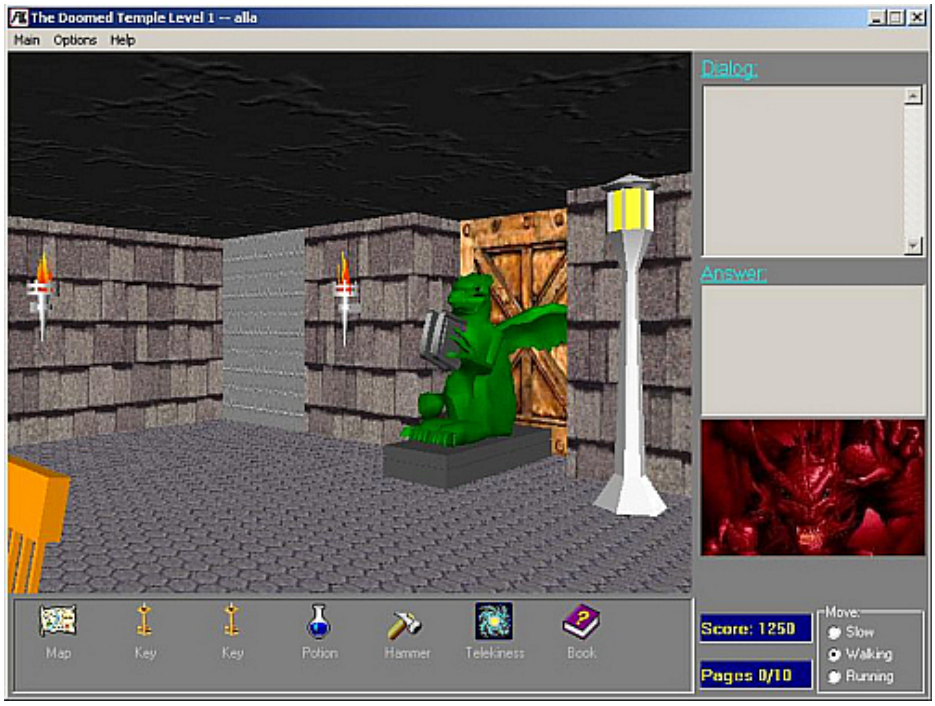

Figure 2: The virtual guard agent.

The animated agent who acts as a virtual companion has the form of an elf (Figure 3). It appears in cases when the student has repeatedly made the same kind of domain mistake while answering to the dragon. Such cases imply that the student has a persistent misconception about a part of the domain being taught. Also the virtual companion appears in cases when the student has given an answer, either correct or wrong, and the student's behaviour before or after giving the answer is different from his/hers usual behaviour. Such cases may imply that the student may be experiencing an emotion, which is not typical of him/her, and the companion provides affect advice. The companion agent obtains all relevant information about the student from the student modeller agent.

For example, supposedly a student has made an error in a question concerning the plural form of nouns, and specifically the rule concerning the nouns that end in "-f". Such nouns have their plural formed by converting the "-f" to "-ves" (e.g. calf-calves, knife-knives). Then the student modeller checks the recent previous questions asked concerning the same rule, that are located at the long term student model, and examines whether the student has given wrong answers in those too. If this is so then the student modeller informs the companion agent, which creates advice about the specific error and provides it to the student. As a result, the

virtual companion appears, and gives advice in a casual way as if a friend was talking to the student.

There are times that the student is prompted to comment on these remarks. The existence of the virtual companion has been considered quite important by many researchers for the purpose of improving the educational benefit of tutoring systems and promoting the student's sense of collaboration. Figure 3 illustrates a screenshot of the game where a window with the virtual companion (elf) agent has opened to give advice to the student.

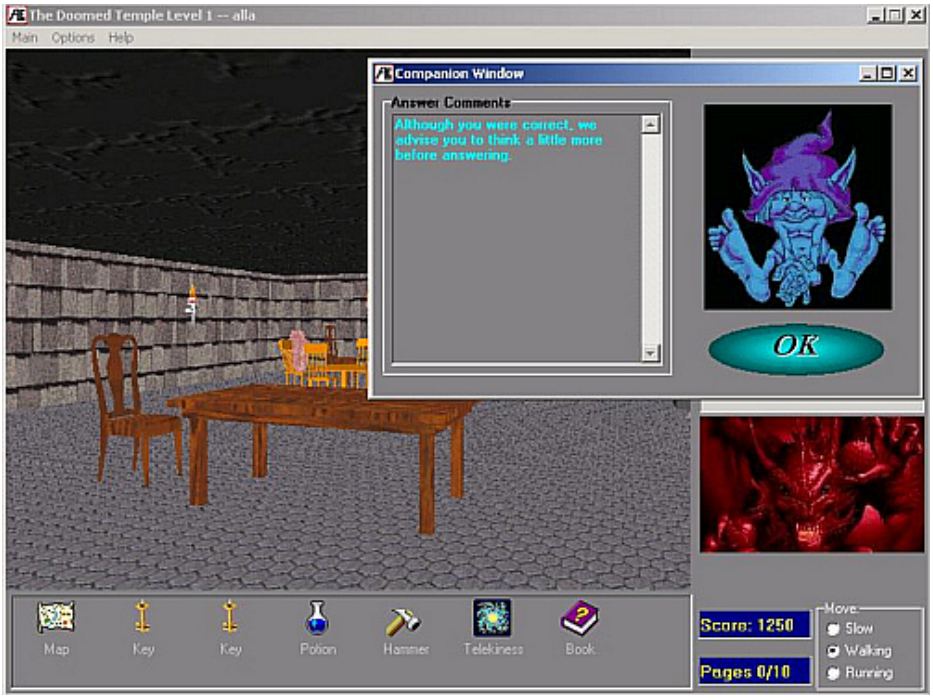

Figure 3: The virtual companion agent.

The virtual advisor agent is located at places, which are important for the plot of the virtual reality adventure game. Every time a student comes across an advisor agent s/he is free to use it. The virtual advisor agent, who has the form of a female angel, provides the student with important information about new parts of the theory that s/he has to read, or about parts of the theory that s/he appears not to know well and s/he has to repeat. In addition, virtual advisors are responsible for making some general remarks to the students about their observable behavioural characteristics while using the game, and giving advice to the students about avoiding problematic situations.

Similarly to the companion agent, the advisor agent also obtains information from the student modeller agent. When an advisor agent is activated from a student then the student modeller agent acquires data from the long-term student model, and generates information about the student's most common error types, and the student's possible problematic behavioural characteristics. Then the student modeller grants this information to the advisor agent, which

generates advice about the domain theory and the behavioural actions of the student. For example if the student modeller has diagnosed many mistakes concerning a rule about the plural form of nouns the advisor would advise the student to repeat the theory of the specific rule. Also if the student has been repeatedly observed to wander around the worlds of the game without answering questions and moving on, the advisor agent would advise the student to be more concentrated and try to finish the test. The virtual advisor agent is illustrated in Figure 4.

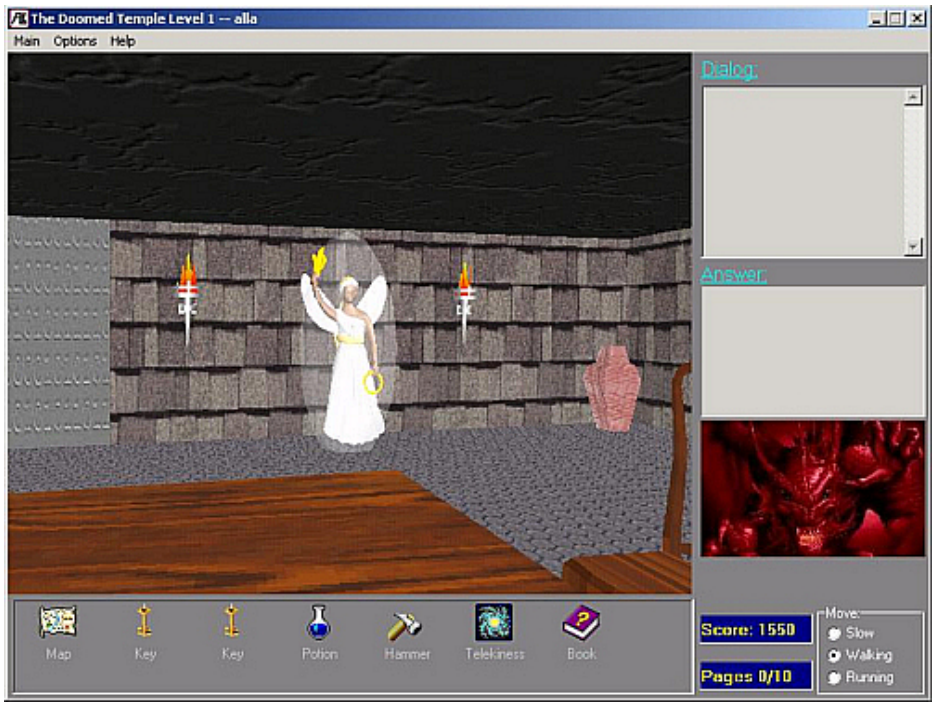

Figure 4: The virtual advisor agent.

The kind of advice offered by the advisor and companion agents to the student, usually have common parts but there exists a basic difference between them. The virtual companion is self-triggered, and gives advice about a specific answer, either about the existence of a repeated mistake in the case of a wrong answer, or about the experience of a possible affective state derived from the student behaviour before or after the answer to the question. On the other hand, the virtual advisor has to be triggered by the student like a help file, and it provides general remarks about theory parts that have to be repeated and about the user's behavioural characteristics. These remarks concern the whole session that the student is using the educational application and not a specific answer. The existence of both the virtual companion and advisor is very important, because the game itself may motivate students but it may also cause disappointment and frustration each time a student does not perform as well as s/he would like or expect. Moreover, the testing process where students have to answer questions may cause them anxiety (as exams always do) and thus they may perform worse than they could, if they let their anxiety take over them.

It is very important that the animated agents may be able to speak to the students. In most of the commercial games there are sound effects that a player hears during the game. However, the effect that makes games more human-like is that of the agents of the game speaking to the player. This feature gives the player a feeling of interplay with the agents, something like the player having a companion in his/her journey. This feature provides a great asset to the educational game, because the students do not become bored easily.

All three animated agents of the virtual game, the Guard, the Companion and the Advisor Agent send their results to the Speech-Synthesising Agent. The Speech-Synthesising Agent is responsible for giving voice to the animated agents in order to make the interaction more natural and enjoyable. The particular agent does not contain any further reasoning mechanisms. So each particular animated agent speaks with a specific voice. Such characters provide an entertaining and emotional function, which may help novice learners of computer applications to become more easily familiar with them [46]. In addition, such characters improve the effectiveness of the system by engaging and motivating learners [27].

As we have mentioned the student modeller agent is responsible for forming and updating the long-term user model located on the Web server. When a question posed by the guard agent is being answered the student modeller agent reasons about the results (correct, wrong, error type) of the answer, and any other user's actions and characteristics that might have taken place before or after giving the answer. Such actions include the speed by which the student has typed the answer, keyboard actions, mouse movements etc. The results of the reasoning are used to form and update the student model. After that, the student modeller agent acquires data from the student model, which in combination with the results of the specific answer are used to provide information to the companion agent either for a repeated mistake or an experienced affective state. When an advisor agent is activated from a student then the student modeller agent acquires data from the long-term student model, and generates information about the student's most common error types, and the student's possible problematic behavioural characteristics.

Examples about error diagnosis and actions analysis, were given in the description of the companion and advisor agents, that use the student modeller. Detailed information about affective student modelling will be given in the last section of the paper.

4 System's architecture over the Web

4.1 The Web-based architecture and examples

The Web-based architecture of VIRGE is implemented as illustrated in Figure 5. The student modeller is installed on each client and interacts through the Internet with the Web server (at the other end of the architecture). The Web services that are known both to the client and the server exist in WSDL (Web Service Description Language) documents. The way that the student modeller of the client communicates with the Web server is by calling a Web service using its WSDL description. In order to call the service it sends a SOAP (Simple Object Access Protocol) message containing the WSDL description and the XML metadata of the service. In our case, the UDDI (Universal Description, Discovery and Integration) registry is the

Internet Information Server (IIS) of the Web server and is responsible for publishing the interface of the Web service in order to be known both to the client and the server. The UDDI gets the SOAP message, translates it, discovers the specific service of the Web server that is registered on it, and prompts the Web server to execute it. After the execution of the service, if necessary, the Web server sends a corresponding SOAP message, through the IIS, to the client containing the results of the function.

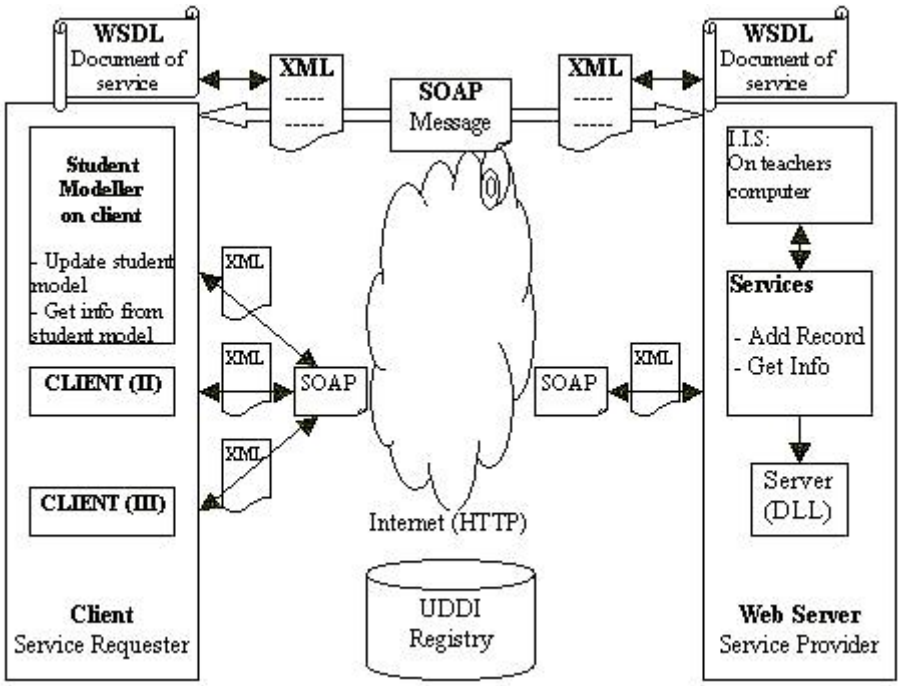

Figure 5: The Web based architecture.

More specifically, the Web services are registered on the Internet Information Server (IIS) of the main computer of the teacher's computer. The services are implemented on the main computer in a server program that can be another executable application (.EXE) or even a DLL (Dynamic Link Library). The virtual game is installed in various computers of the computer lab that are the clients. Each of these clients uses the client server architecture above to perform the tasks that they are responsible for.

One such task is the addition of a student's answer from the student modeller to the database, which is located at the main computer. The characteristics of the student's answer (e.g. whether it was correct, what kind of mistake occurred, what actions took place before or after the answer) are also stored in the database along with the answer. Other tasks include questions regarding the information that exists in the database that keeps the student model. For example the companion agent frequently asks the student modeller about which specific rule of the domain knowledge the student is frequently making mistakes on, as to provide help for the student.

An example of the operation of the above architecture of the system is the following. Every client, located in each student's computer can add an answer of that student to the database located at the main computer. Thus the client calls the specific Web service using its interface described in the respective WSDL document, by sending a SOAP message to the Web server through the Web server's IIS. Every call provides the Web service with the declaration and some variables, which contain the information about the student's answer. The Web server executes the service by using the server program (.DLL in our example) that implements this interface and store the answer to the database.

4.2 Technical discussion of the use of the architecture.

To create the 3D environment of the educational game, we have used the Virtual Reality Modelling Language (VRML) [3]. VRML was originally designed to allow 3D worlds to be delivered over the World Wide Web. Using a VRML browser the user can explore this world, zooming in and out, moving around and interacting with the virtual environment. This allows fairly complex 3D graphics to be transmitted across networks without the very high bandwidth that would be necessary if the files were transmitted as standard graphic files. VRML can also include multimedia elements, such as texture images, video and sounds.

In the past years, some researchers have used VRML language for creating simple educational applications that work through the Internet ([49], [10]). In these cases the clients of the educational applications included the virtual worlds of the applications because they were not so complex as the worlds of a virtual game. As a consequence, they could be displayed through an Internet browser. A VRML world is made up of lots of simple shapes, such as cones and spheres, grouped together to form objects. The more shapes in the file, the more detailed the world, but at a cost of increasing the file size and the time taken by the browser to display the world.

In our application there exist *complex graphical objects* that consist of hundreds of simple shapes. Thus, it is made clear that a virtual reality game like our educational application, which has numerous objects inside its virtual worlds that are highly detailed, would take too much time to load and run on a web browser. Additionally, we cannot decrease the detail of our virtual worlds or try to make them smaller in order to create thinner virtual scenes, because then there will not exist a virtual game for educational purposes, that aims at the motivation of the students by simulating the commercial games.

As a result, taking all the parts of the client program, including the VRML virtual worlds, and providing them through the Internet was not an option for the case of VIRGE. Thus, what was best was to have the client installed at each student's computer, and by some way to make the student modelling component, which is the student modeller agent, communicate with the server at the teacher's computer through the Internet. The server would store affective and cognitive information about the detailed student models. To achieve this goal we are using Web services.

Our aim was to have all services that would concern functions and methods of the student modelling component to be implemented at the server as Web services so that the clients would be able to call them through the Internet. Web services are interfaces that describe a collection of operations that are network-accessible through standardised XML messaging. The Service interface description contains information such as the operations of the service,

the input and output parameters of the operations, the protocol, data format, security and other attributes and is written in a WSDL document. The WSDL document contains sufficient information to describe to the service requestor (client) how to invoke and interact with the Web Service. Nothing is implemented in the interface part of the service. The service provider (Web server), which is also aware of the interface of the service, when receiving such form of request from the student modelling agent located on the client, uses the implementation of the service to execute it.

After the service description has been defined, it is published. The service provider first registers its service in a registry server like the UDDI, in our case the IIS, in which service providers and service requesters can publish and find learning services respectively. Once the service provider has published the service description, service requesters are able to find and invoke it. After a service description is acquired, the service requester uses the service description to generate SOAP requests. The WSDL document and XML metadata of the services are wrapped in a SOAP message that will be sent over HTTP to the UDDI.

5 Affective student modelling

The game itself may motivate students but it may also cause disappointment and frustration each time a student does not perform as well as s/he would like. Moreover, the testing process where students have to answer questions may cause them anxiety (as exams always do) and thus they may perform worse than they could, if they let their anxiety take over them. On the other hand there are students that may be quite confident and efficient in which case they may only need affective help when they face exceptionally disappointing situations for them (e.g. when they do not remember something correctly and they are not allowed to continue the game).

5.1 Evidence and inferences

For the purposes of finding out which aspects of the students' emotional state in relation to their performance in the educational game could be modelled, we conducted an empirical study. In this study, computer logging was used to record students' actions while they interacted with the application in a similar way as in [45] and [16] so that they could be analysed by human experts. [45] and [16] describe empirical studies in which human tutors are asked to infer issues about a user's cognitive and motivational state respectively. In both of these empirical studies, the only information that was given to human experts was the pre-recorded screen interaction of a user with the respective systems.

Indeed, through computer logging the system may continuously collect objective data for further analysis and interpretation without interfering with users during their interactions with the system [9]. In the case of VIRGE, the collected user protocols were passed on to 5 human experts, teachers with experience in pedagogic matters for many years, who were asked to observe students' actions while they played the game and to note down what the students were likely to have felt. As a result, the experts had distinguished between different characters of students and assessed their emotions in relation to the students' characters and correctness of their knowledge.

Taking into account the results of the empirical study, the educational game uses as evidence on students' characters and emotions several actions that relate to typing and mouse movements. Time has played a very important role in our measurements. There are many inferences that can be drawn for the students' feelings and reactions depending on the time they spend before and after making some actions. Some examples of inferences based on observations on time spent for various activities are the following:

- *The time that it takes to the student to answer a question.* This measures the *degree of speed* of the student.
- *Pausing time after a system's response.* The time the computer is left idle after a response to the student is used to measure the *degree of surprise* that the response may have caused to the student.

In addition, certain patterns of actions are used to show aspects of the students' cognitive and emotional state. Some examples of students' actions that are used as evidence are the following:

- *The number of times that a student presses the "backspace" and "delete" button while forming an answer*. This evidence is used to measure the *degree of certainty* of the student concerning a particular answer; the more times the student presses "backspace" and "delete" the less certain s/he is about the answer.
- *Mouse movements without any obvious intent in the virtual reality space of the game.* This evidence is mainly connected to the *degree of concentration* or *frustration or intimidation* of the student; the more mouse movements without any obvious intent, the less concentrated or the more frustrated or intimidated the student is. The exact interpretation depends on the context. For example if the mouse movement without any obvious intent occurs some time after the student has been asked a question then it shows frustration since the student does not probably know how to answer.

In some cases, inferences are drawn from the combination of two different categories of evidence. For example, the *degree of confidence* is calculated as the mean value of the degree of speed and the degree of certainty of a student.

The above kind of evidence based on a student's actions is combined with evidence on the student's degree and quality of knowledge of the parts of the lessons that are examined during the game. Therefore for each question asked, the system examines the correctness of the student's answer and if the answer is incorrect it performs error diagnosis. The system also tries to estimate the severity of an error (i.e. whether it was an accidental slip or whether it was due to a persistent misconception).

Examples of some rules that show such kind of combination are the following:

- If a student repeatedly answers questions with a high degree of speed and s/he produces a high degree of incorrectness then this may show *anxiety.*
- If a student repeatedly shows a high degree of confidence irrespective of correctness of his/her answers then this may show *determination* (the student does not give up).

All kinds of evidence are used by the system to adapt its interaction with the user. Moreover a user may see a report of the inferences drawn by the system. This report is used to show the user a more personal interaction since the system tries to know him/her better. An example of such a report is illustrated in Figure 6.

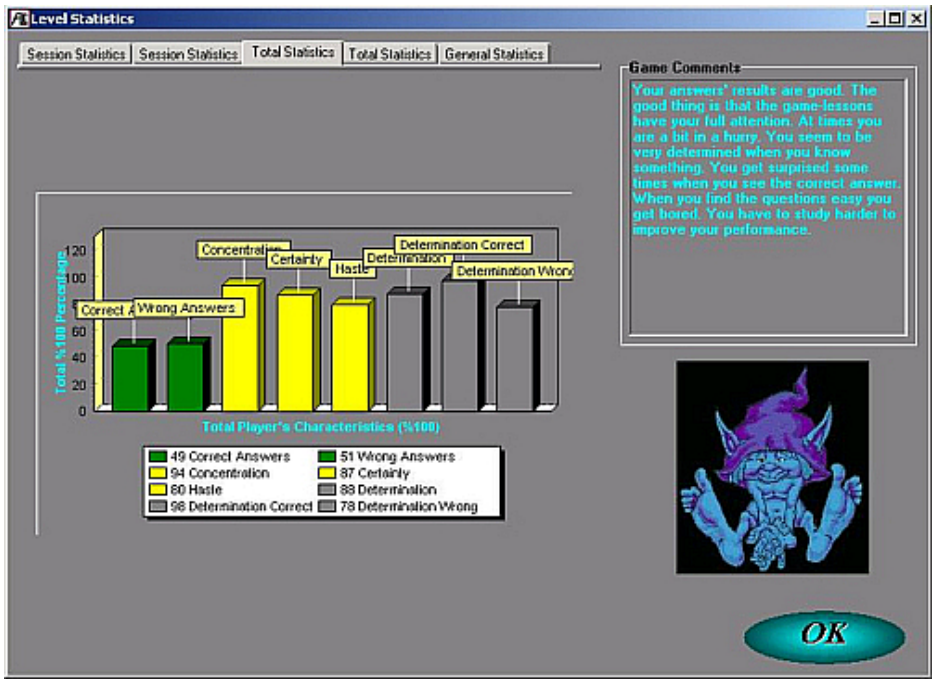

Figure 6: A report of inferences drawn by the system.

In such a report there are mentioned some observations and suggestions of the system. There exists information about the correct, the wrong answers of the student and the categories of his mistakes. For example if the student answers very quickly the questions there should be a message like “At times you seem to be in hurry, take your time while answering”.

5.2 Discussing the adaptation of the OCC theory

As we have mentioned above there are some characteristics of the student game players that are observed and recorded during their interaction with the game. It is our aim to know when the existence of some of these characteristics results in the student’s experience of a good or bad mood. Thus, we have adapted the OCC theory of emotions into the VR-game in order to obtain a measure for the significance of each of these characteristics. The significance of each of these characteristics allows the system to know its importance each time it is measured.

To be more specific, as we have already mentioned, the system tries to give to each independent student advice tailored to his or her needs. By summing evidence and inferences for some of the players’ characteristics the system tries to make some remarks and provide useful information. What we seek is to know when a remark is really worthy for the system to trigger a reaction. Thus, for every characteristic of each student there is a significant value that if it is exceeded the system can assume that it has a value worthy of being seriously taken into account. In such cases the system should generate advice. For example if a student shows hesita-

tion by pressing the delete and backspace key many times, it is very useful to know if this behaviour is typical of this student in all of his or her interactions or whether it has occurred to a great extend in the current interaction.

The OCC theory of emotions provides a way for finding out if an emotion has really occurred to an individual. Then, if an emotion has really occurred, the OCC theory provides a way for measuring its intensity. For these purposes, the theory requires the specification of a "threshold" value for each emotion, which is considered as a significant value. If the potential of an emotion is lower than the threshold value then the individual does not experience the emotion. Otherwise the intensity of the emotion experienced is the difference between the potential of the emotion and the threshold value. For example in order to determine when a joy emotion has occurred the OCC theory suggests:

IF (JOY-POTENTIAL) > (JOY-THRESHOLD) **THEN**
  SET (JOY-INTENSITY) = (JOY-POTENTIAL) – (JOY-THRESHOLD)
ELSE
  SET (JOY-INTENSITY) = 0;

As we have explained above the VR-game tries to interpret evidence about some mood and character features of the student players. The first thing that we needed to do is to calculate a threshold value for every characteristic and then find the significance either positive or negative that it has.

During the game the system keep track of characteristics of students by storing information about each player answers. Hence, we decided that the way we would calculate the threshold value of each characteristic in a specific moment was to calculate the mean value of this characteristic and add or subtract its standard deviation value. By performing these calculations we have two threshold values (High & Low) about each characteristic, one upper bound and one lower bound.

1) T. High = Mean Value + Standard Deviation
2) T. Low = Mean Value - Standard Deviation

So if the value of a characteristic is greater than the T. High value or lower than the T. Low value the system has a good reason to make some remarks about it. The significance of the characteristic either positive or negative could result from the value of the following operations.

IF (VALUE) > (T. HIGH) **THEN**
  SET (SIGNIFICANCE) = (VALUE) – (T. HIGH)
ELSE IF (VALUE) < (T. HIGH) **THEN**
  SET (SIGNIFICANCE) = (T. LOW) – (VALUE)
ELSE
  SET (SIGNIFICANCE) = 0;

For example as mentioned earlier by watching the student's mouse moves and keyboard actions the system makes some assumptions. Also we have accepted a scale from 0 to 1 for every characteristic in order to classify it for every student's answer, depending on the mouse moves and keyboard actions. So if for the characteristic of concentration, which depends on the mouse movements, the mean value is 0.75 for a student with a standard deviation of 0.12, then the T. High value would be 0.87 and the T. Low value 0.63. So if the student during his or her answer had a degree of concentration of 0.45 that according to the above type would

mean a significance value of 0.63-0.45=0.18. The fact that a student is 18% less concentrated than usual gives the system a good reason to assume that s/he is not concentrated enough or is having navigation problems. As a result the companion agent should be notified about this possible affective state, in order to provide appropriate help or advice.

In a similar way the system collects information for all of the characteristics of the student that have been included in the affective state representation. This information might either be positive or negative but either way can give evidence for the system to provide more detailed assistance. This information in conjunction with the learning model of the student that keeps track of his or her mistakes and answers is a useful guide for the advice model incorporated in the game.

6 Evaluation

The user modelling aspect and the web-based operation of VIRGE was evaluated by 50 students of 11-12 years of an elementary school. The students were given two versions of the educational game to work with. One version was VIRGE as has been described above whereas the other version was a similar educational VR game that did not have any user modelling aspect. Thus, it was not personalised and did not have a student modelling component that operated through the Web. After all the children had used VIRGE and the non-personalised educational game application for 2 hours respectively, they were given the choice to use for a maximum of 1 hour, either of the applications to make revisions on the lessons that they had been taught. Moreover, after they had completed the use of both applications the students were interviewed concerning the likeability and usability of the programs.

6.1 Analysis of observed students' behaviour

The results during the one-hour of free time in class showed a significant preference of the student-users for VIRGE in comparison with the non-personalised educational game software. In total, the students spent 54% of their time using VIRGE and 41% of their time using the non-personalised game application. The reason that the sums of the respective percentages are not 100% is because there was a small amount of time for some students spent in neither of the two applications. The total results showed that VIRGE had achieved its aim of being more attractive and motivating, for the students than the non-personalised educational game software.

In more detail, the total time in minutes that was available for the students was 50x60 minutes = 3000 minutes. In total, the students used the VIRGE application for 1616 of the 3000 minutes available, while using the non-personalised game application for 1219 minutes. The statistical analysis, which took place after the 1-hour use of either of the two applications, concerned the significance of the difference on the time spent in each of the two applications for all the students. There was a t-test performed to compare the time spent on VIRGE and the time spent on the non-personalised educational game application.

The null hypothesis, $H_{0,}$ was that there was no difference between the time spent for each of the two applications. The research hypothesis, $H_{1,}$ was that there was significant difference

between the time spent for each of the two applications. The t-value result of 2.12 for all the students was significantly greater than its critical value of 1.68. This showed that the difference in the time spent by the students, between the two applications was statistically significant. Thus, we can reject hypothesis H_0 and accept H_1. The time spent on the VIRGE was significantly greater than the time spent on the non-personalised educational game.

The results of the above t-test are summarised in Table 1. In particular, Table 1 illustrates the mean value of the differences in the time spent for all the students in using VIRGE and the non-personalised game application. Additionally it includes the results of the t-test about significant difference in the time spent between the two applications. These results involve the standard error of the differences, the T value and the Critical value of the t-test.

Variable	**VIRGE**	**Non-personalised educational game application**			**T_Value, C_Value**
	Mean Value of minutes played (0-60)	*Mean Value of minutes played (0-60)*	*Mean of Differences*	*Standard error of Differences*	
Students results	32.32	24.38	7.94	3.74	Tv=2.12 Cv=1.68

Table 1: Results of the analysis of the students' time spent either on VIRGE or the non-personalised educational game application.

In the above t-test, the t-value of the t-test is calculated by performing a t-test for correlated samples for the time spent on each of the applications [48]. The critical value for each t-test is the value taken from Table T for a one-tailed research hypothesis depending on the sample number. The t-test result shows that there is a statistically significant difference for the two samples in favour of the time spent using VIRGE for the students. More specifically the t value (2.12) is significantly greater than the corresponding critical value (1.68).

Therefore the students preferred to play VIRGE that includes cognitive and affective student modelling and provides adequate help and advice for the students, rather than use the non-personalised educational game application during their free time in class, or to repeat lessons at the school classroom. Thus, the evaluation result revealed that the students found useful the personalisation of the system.

6.2 Analysis of students' answers to interviews concerning the usability and likeability of the two applications

After the one-hour use of either of the applications to make revisions on the lessons that the students had been taught, they were interviewed about the two applications used, the VIRGE educational game and the non-personalised educational game application. These interviews included questions about the usability and the likeability of the two applications.
Among the questions asked during the interviews were the following:

1. Which one appeared to be more easy to use?
2. Which one was more interesting or motivating?
3. Did you find the advice given by VIRGE annoying?
4. What do you think of the applications as a game?
5. Would you use it, through the web, at your leisure time from home?
6. What do you think of learning while playing in class?
7. Did you have problems using VIRGE through the Web?

Figure 7 illustrates important parts of the interviews information concerning the opinions of students about which application between VIRGE and the non-personalised educational game application was more easy to use, interesting or motivating. VIRGE was reported as more easy to use by 64% of the students. This shows that the interaction with the system helped most of the students to play the educational game. The 57% of the students selected VIRGE as to be more motivating and 78% as to be more interesting.

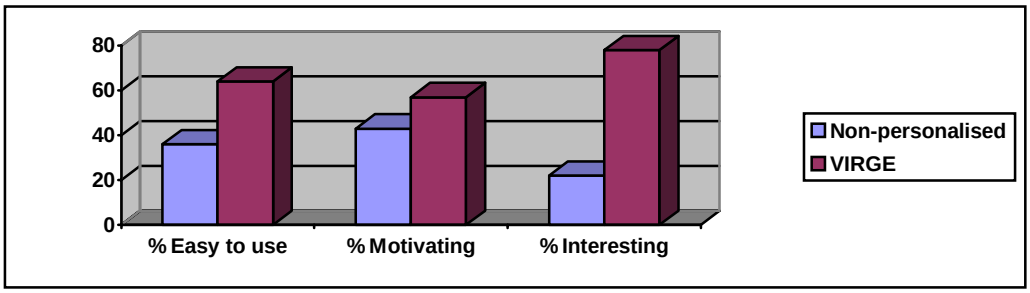

Figure 7: Which one between VIRGE and the non-personalised educational game application was more easy to use, motivating, interesting.

Concerning the annoyance of the students by VIRGE's advice, the likeability to learn while playing VIRGE in class, and the probability of using VIRGE to repeat lessons at home the students reported the information gathered in figure 8 below. 18% of the students thought of VIRGE to be annoying when giving advice, this might be due to the fact that the personalisation of a student requires some interaction with the student-user that some times is time consuming and a bit interrupting. 64% of the students are enthusiastic with the idea of learning in class while playing, and 55% noted that they would use VIRGE during their leisure time at home.

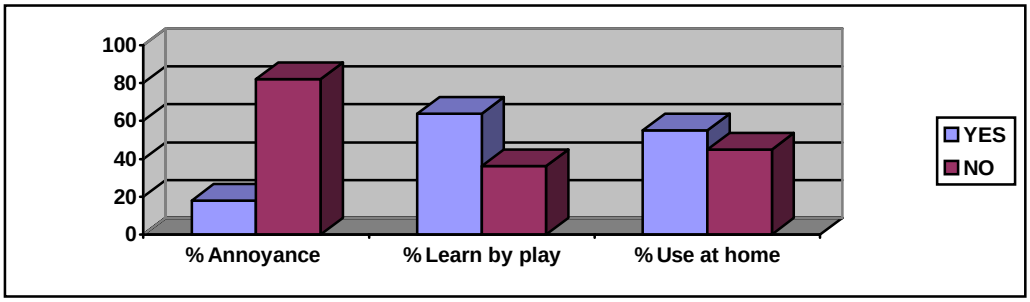

Figure 8: Students' preferences about VIRGE.

Additionally, almost all the students reported that they did not face any extra difficulties, on using VIRGE educational application, concerning the fact that its student modelling component worked through the Web. They noted though, that the use of the VIRGE web-based application was similar to the non-personalised educational application that worked completely locally.

7 Discussion and conclusions

This paper has presented and discussed a Web-based educational virtual reality game that uses the technology of ITSs. Technically to achieve the availability and efficient maintenance of the VR-game over the Web we have used the technology of Web services. Moreover, the Web-based educational game performs affective student modelling. This is done so that the necessary personalisation is achieved and the distance learning is enriched with emotional features that would otherwise be missing from the application due to the absence of a human teacher. Finally, the fact that a VR-game provides an environment for a high degree of generation of users' emotions is exploited for the adaptation of tutoring to each individual student's emotions while playing and learning.

The advantage of using Web services is that it renders the application usable in the same way as the standalone version despite its very high demands on computer resources. Through Web services we have managed to make this demanding application work over the Internet. Additionally it is possible to make our student modelling component available through the Internet for other educational applications and ITSs to use, if we register its Web services to a UDDI server as to make them accessible to other applications.

As a result, all students may access the ITS-game through the Internet and their human tutors may have access to these students' progress reports and user models without any problem. The user modelling component of the system exploits evidence from the students' actions using the keyboard and the mouse that is combined with evidence on the student's knowledge of the domain being taught, for drawing inferences for the student's emotional and cognitive state while interacting with the educational application. The affective inferences are drawn using an adaptation of the OCC theory. Then these inferences are used for the generation of the system's advice for each individual student, taking into account his/her knowledge state, character and mood. In this way the system achieves a high degree of dynamic personalisation of the tutoring that it provides.

The evaluation of VIRGE in comparison with a non-personalised version of the game revealed that the students found useful the help and advice that is offered by the personalisation of the system, and preferred the educational application to be provided with it. Moreover, the interference of the user modeller with the students proved to be tolerable, and the students were positive to the idea of using VIRGE in class or at home. Additionally, the students did not face extra difficulties due to the fact that the student modelling component worked through the Web. These results were very encouraging due to the fact that an educational game application primarily aims at motivating and entertaining the students. So, students' disruption or annoyance, and any extra difficulties faced by the students, would constitute significant drawbacks to our cause.

The architecture that we have presented in this paper, is fully implemented and has been evaluated by end-users. This is an important step for showing how Web services can be used by any system to enhance the performance of an Intelligent Tutoring System that operates as a virtual reality game. However, the benefit of the reusability and knowledge sharing that can be achieved in other applications has not been fully exploited and evaluated yet. This is due to the fact that we have not yet developed other applications that may use the Web service components of the current version of VIRGE. It is within our future plans to develop other applications so that we can explore the capabilities of Web services in the reusability and knowledge sharing in the context of personalised virtual reality games in education. The main components that could be reused are the user modeling component and the Virtual Reality game platforms. The result of this advancement would be a Web services' user model server and Virtual Reality game server that may be used by many applications independent of each other.

The personalisation approach that we have shown in this paper makes inferences about the users' cognitive and affective state. In this way, we achieve to follow the user's cognitive and affective state in the context of the educational game. Both the educational context and the gaming environment are bound to create feelings to the user depending on how well the user understands and learns the educational content and how well s/he performs in the game. The clues that we have used about the user are all implicit, meaning that the user is monitored by the application while s/he works unobtrusively with the educational game. Thus, the system monitors the user through the keyboard and mouse input channels that are currently available in VIRGE. In the near future we plan to add more modalities, such as audio and visual, so that the inferences drawn about the affective state of users could be made more accurate. For example, if there was visual-facial affect recognition, the system could "see" a user smile with joy or yawn with boredom and have more complete idea about the user's affective state. Currently, there is ongoing research about visual-facial affect recognition [40] in parallel with the work that we have presented. It is within our plans, to integrate the modalities and create a multi-modal affective user-interface for the VR-game.

References

1. Adamatti, D.F., Bazzan, A.L. (2002). "*A Framework for Simulation of Agents with Emotions*," http://www.inf.ufrgs.br/~adamatti/pag/ingles/Workcomp02.pdf
2. Alpert S. R., Singley M.K. & Fairweather P.G. (1999) "*Deploying Intelligent Tutors on the Web: An Architecture and an Example*", International Journal of Artificial Intelligence in Education, 10, 183-197.
3. Ames A., Nadeau D., Moreland J. "VRML 2.0 Sourcebook", *2nd Edition*.
4. Amory, A., Naicker, K., Vincent, J. & Claudia, A. (1998). "*Computer Games as a Learning Resource*." Proceedings of ED-MEDIA, ED-TELECOM 98, World Conference on Education Multimedia and Educational Telecommunications, 1, pp. 50-55.
5. Aroyo L. and P. Kommers, "Preface-Intelligent Agents for Educational Computer-Aided Systems", Journal of Interactive Learning Research, Special Issue on Intelligent Agents for Educational Computer-Aided Systems, 1999 Vol. 10, No 3/4, pp. 235-242.
6. Belkada S., A.I. Cristea & T. Okamoto, "Measuring Knowledge Transfer Skills by Using Constrained-Student Modeler Autonomous Agent", Proceedings of the IEEE International Conference on Advanced Learning Technologies (ICALT 2001), IEEE Computer Society, 2001, pp. 375-378.

7. Box D., D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. Nielsen, S. Thatte, D. Winer, "*Simple Object Access Protocol (SOAP) 1.1*", W3C Note, May 2000, "http://www.w3.org/TR/SOAP".
8. Brusilovsky, P. (1999) "Adaptive and Intelligent Technologies for Web-based Education." In C. Rollinger and C. Peylo (eds.), Künstliche Intelligenz (4), Special Issue on Intelligent Systems and Teleteaching, 19-25, http://www2.sis.pitt.edu/~peterb/papers/KI-review.html.
9. Chou, C. (1999). *Developing CLUE: A Formative Evaluation System for Computer Network Learning Courseware*, Journal of Interactive Learning Research, 10(2).
10. Chover M., Ó. Belmonte, I. Remolar, R. Quirós, J. Ribelles. "Web-based Virtual Environments for Teaching", Proceedings of the Eurographics/ACM SIGGRAPH *Workshop on Computer Graphics Education*, 2002.
11. Christensen E., F. Curbera, G. Meredith, S. Weerawarana (eds.) "*Web Services Description Language (WSDL) 1.1*", W3C Note, March 2001, "http://www.w3.org/2001/NOTE-wsdl-20010315".
12. Ciganek A., Haines M., Haseman W. (2005). 'Challenges of Adopting Web Services: Experiences from the Financial Industry'. In Proceedings of the 38th Hawaii International Conference on System Sciences (HICSS'05)**,** January 03 - 06, 2005, Big Island, Hawaii**.**
13. Conati, C. & Zhou, X. (2002) " *Modeling students' emotions from cognitive appraisal in educational games*". In S. A. Cerri, G. Gouarderes and F. Paraguacu (Eds.) : Intelligent Tutoring Systems 2002, LNCS, 2363, pp. 944-954, Springer-Verlag Berlin Heidelberg 2002.
14. Cumming G. & McDougall A.*: Mainstreaming AIED into Education?* International Journal of Artificial Intelligence in Education, Vol. 11, (2000), 197-207.
15. Curbera F., Nagy W. and Weerawarana S (2001). 'Web services: Why and how'. In Workshop on Object-Oriented Web Services OOPSLA, October 2001 - Tampa, Florida, USA.
16. De Vincente A. & Pain H. (2002). "*Informing the Detection of the Students' Motivational State: An Empirical Study*". In S.A. Cerri, G. Gouarderes and F. Paraguacu (Eds.): Intelligent Tutoring Systems 2002, LNCS 2363, pp. 933-943, Springer-Verlag 2002.
17. Devedzic, V. (2001) 'A Pattern Language for Architectures of Intelligent Tutors'. In Proceedings of the International conference on Artificial intelligence AIED 2001, San Antonio, Texas, pp 542-544.
18. Elliott, C., J. Rickel, and J. Lester,. *Lifelike Pedagogical Agents and Affective Computing: An Exploratory Synthesis.* Artificial Intelligence Today , Lecture Notes in Computer Science 1600, M. Wooldridge and M. Veloso (eds.), 1999. Springer Verlag. 195-212.
19. Goleman D. (1995). "*Emotional Intelligence*". Bantam Books: New York.
20. Haas H. and Brown A. (2004). 'Web Services Glossary', http://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/.
21. Hartley D. & Mitrovic (2002): *"Supporting learning by opening the student model"* In S.A Cerri, G. Gouarderes and F. Paraguacu (Eob.) : ITS 2002, LNCS 2363, pp 453-462, 2002, Springer-Verlag.
22. Hudlicka, E. and D. McNeese. *Assessment of User Affective and Belief States for Interface Adaptation: Application to an Air Force Pilot Task.* Journal of User Modeling and User-Adapted Interaction, 2002. 12(1): p. 1-47.
23. IBM White Paper (2000) "*The IBM WebSphere software platform and patterns for e-business invaluable tools for IT architects of the new economy*".
24. IBM: "*The Web services (r)evolution*, Part 1" ,http://www106.ibm.com/developeworks/ webservices/library/ws-peerl.html.
25. Id Software (1993), Doom - Virtual Reality Computer Game, Id Software Company Texas.
26. Inkpen, K., Upitis, R., Klawe, M., Lawry, J., Anderson, A., Mutindi, N., Sedighian, K., Leroux, S. & Hsu, D. (1994). "*We Have Never-Forgetful Flowers In Our Garden: Girl's Responses to Electronic Games*". Journal of Computers in Math and Science Teaching, 13(4), pp. 383-403.
27.Johnson, W. L., Rickel, J. W., & Lester, J. C. (2000). 'Animated Pedagogical Agents: Face-to-Face Interaction in Interactive Learning Environments'. *International Journal of Artificial Intelligence in Education, 1*1, pp 47-78.

28. Kay J. (2000). "*Accretion Representation for scrutable student modeling*". Proceedings of ITS 2000, pp 514-523.
29. Lopez J.M., E. Millan, J.L. Perez-de-la-Cruz, F. Triguero, "*ILESA: a Web-based Intelligent Learning Environment for the Simplex Algorithm*", in Alvegard C. (ed.): Proceedings of CALISCE'98, 4th International Conference on Computer Aided Learning and Instruction in Science and Engineering, 1998, pp. 399-406.
30. Microsoft White Paper (2000) "*A Blueprint for Building Web Sites Using the Microsoft Windows DNA Platform*".
31. Mizoguchi, R. and Bourdeau, J. (2000). 'Using Ontological Engineering to Overcome Common AI-ED Problems', International journal AIED, vol. 11(2), pp. 107-121.
32. Mc Calla G., Vassileva, J., Greer, J. & Bull, S. (2000). "*Active Learner Modelling*". Proceedings of ITS 2000, pp 53-62.
33. Okazaki Y., K. Watanabe, H. Kondo, "*An Implementation of an intelligent tutoring system on the World-Wide Web*", Educational Technology Research, 1996 Vol. 19, No. 1, pp. 35-44.
34. O'Riordan and J. Griffith, "A Multi-Agent System for Intelligent Online Education", Journal of Interactive Learning Research, Special Issue on Intelligent Agents for Educational Computer-Aided Systems, 1999, Vol. 10, No 3/4, pp. 263-274.
35. Ortony, A.; Clore, G. L. and Collins, A.. *The Cognitive Structure of Emotions*. Cambridge University Press, Cambridge, UK, 1988.
36. Papert, S. (1993). "*The Children's Machine: Rethinking School in the Age of the Computers*". Basic Books, New York, 1993.
37. Picard, R. W. *Affective Computing*. The MIT Press, Cambridge, MA, 1998.
38. Pires P.F., Benevides M. & Mattoso M. (2002) "*Building Reliable Web Services Compositions*", Net.Object Days - WS-RSD'02, 551-562.
39. Pullen M., Brunton R., Brutzman D., Drake D., Hieb M., Morse K., Tolk A. (2005). 'Using Web services to integrate heterogeneous simulations in a grid environment'. Proceedings of the International Conference on Computational Science 2004, Krakow, Poland, June 2004.
40. Stathopoulou I.O & Tsihrintzis G.A (2006). "Facial Expression Classification: Specifying Requirements for an Automated System", In B. Gabrys, R. J. Howlett, L. C. Jain (Eds.): Knowledge-Based Intelligent Information and Engineering Systems, 10th International Conference, KES 2006, Bournemouth, UK, October 9-11, 2006, Proceedings, Part II. Lecture Notes in Computer Science 4252 Springer 2006, pp. 1128-1135.
41. Torres J., Dodero J., Padron C. (2004). 'A Framework Based on Web Services Composition for the Adaptibility of Complex and Dynamic Learning Processes'. In Learning Technology newsletter, Vol. 6, Issue 1, January 2004.
42. Tsalgatidou, A., & Pilioura, T. (2002). An Overview of Standards and Related Technology in Web Services. *Distributed and Parallel Databases, 12*, 135-162.
43. *Universal Description, Discovery and Integration (UDDI) Version 2.0 Specification*, June 2001, "http:/uddi.org/specification.html".
44. Vicari R.M (2002): "*ITS, Agents, BDI and affection: Trying to make a plan come together"*, In S.A Cerri, G. Gouarderes and F. Paraguacu (Eob.) : ITS 2002, LNCS 2363, pp 8-9, 2002, Springer-Verlag.
45. Virvou M. & Kabassi K., "*An Empirical Study Concerning Graphical User Interfaces that Manipulate Files*", Proceedings of ED-MEDIA 2000. World Conference on Educational Multimedia, Hypermedia & Telecommunications, AACE, Charlottesville VA, 2000a. pp. 1724-1726.
46. Virvou M. & Kabassi, K. (2002). *A Multi-Agent System for Intelligent Assistance in a GUI*. In Proceedings of the 3rd International NAISO Symposium on Engineering of Intelligent Systems, NAISO Academic Press, Canada/The Netherlands.
47. Virvou M. & Katsionis, G. (2003). '*VIRGE: Tutoring English over the Web through a Game'*. In Proceedings of the IEEE International Conference on Advanced Learning Technologies (ICALT 2003, Greece, Athens).

48. Voelker, D.(2001): Statistics, Wiley Publishing, Inc. New York 2001.
49. Warrick, P.A. Funnell, W.R.J.:" A VRML-based anatomical visualisation tool for medical education", *Information Technology in Biomedicine*, Vol. 2, (1998), 55-61.
50. Weiqin Chen (2002). 'Web Services -- What Do They Mean to Web-based Education?'. In Proceedings of the International Conference on Computers in Education (ICCE'02), December, 2002, Auckland, New Zealand.
51. Wen L., Jesshope C. (2003). 'Web Services Technology and Learning Technology - A Web-services Model for Constructing Virtual Learning Environments'. In The First International Conference on Web Services (ICWS'03), Las Vegas, Nevada, USA.
52. Zhengfang Xu, Zheng Yin, and Abdulmotaleb El Saddik (2003). 'A Web Services Oriented Framework for Dynamic E-Learning Systems'. In Systems CCECE 2003 – CCGEI 2003, Montreal, May 2003.

Appendix

A. The OCC cognitive theory of emotions

The OCC (Ortony, Clore, & Collins, 1988) model has established itself as a very popular model for emotion synthesis. A large number of studies employed the OCC model to generate emotions for their embodied characters, and lately there are some studies, like our case, that are using it to model user emotional states. This model specifies 22 emotion categories based on valenced reactions to situations constructed either as being goals of relevant events, as standards of actions of accountable agents (including itself), or as attitudes of attractive or unattractive objects (see Figure 9). It also offers a structure for central intensity variables, such as the desirability of an event, the praise-worthiness of an action of an agent and the appealing-ness of an attitude of an object, which determine the intensity of the emotion types. It contains a sufficient level of complexity and detail to cover most situations that an emotional interface character might have to deal with.

Goal-based emotions: In order to determine the intensities of emotions pertaining to the success or failure of goals, the OCC model uses several variables depending upon the context of the situation. Specifically, the variables used depend on whether the event is unconfirmed, confirmed or disconfirmed, whether the event was anticipated, and whether it happened to the agent itself or someone else. Under the OCC model, unanticipated confirmed goal successes and failures for one's self generate the "Well-Being" emotions category of *joy* and *distress*. Anticipated goal effects for our-self generate the "Prospect-Based" emotions category. In an unconfirmed state they generate *hope* and *fear*. In a confirmed state, hope and fear will turn to *satisfaction* or *disappointment*, respectively, and in the disconfirmed state fear and hope become *relief* and, for lack of a better term, *fears-confirmed*. When evaluating how the goals of others have been affected the "Fortune-Of-Others" emotions category is triggered. Goal successes will generate *happy-for* or *resentment*, and goal failures will generate *gloating* or *pity*, depending on whether the agent in question is liked or disliked by the agent experiencing the emotion.

Standards-based emotions: The degree of judged praiseworthiness or blameworthiness of the action of an agent is implemented as the result for an effected standard. Standards are

responsible for what the OCC model terms "Attribution" emotions. When responsibility for an action is attributed to one's self, *pride* or *shame* will result. When attributed to an external agent, these turn to *admiration* or *reproach*.

Attitudes-*based emotions:* The degree to which an object is considered appealing or unappealing is modeled. Attitudes are responsible for what the OCC model terms "Attraction" emotions resulting from effects upon preferences. They come only in two varieties under the OCC model, *love* and *hate*.

There are also 4 other emotion types that occur from the combination of the "Well-Being" and the "Attribution" emotions. The entire schema of the OCC theory is illustrated in Figure 9.

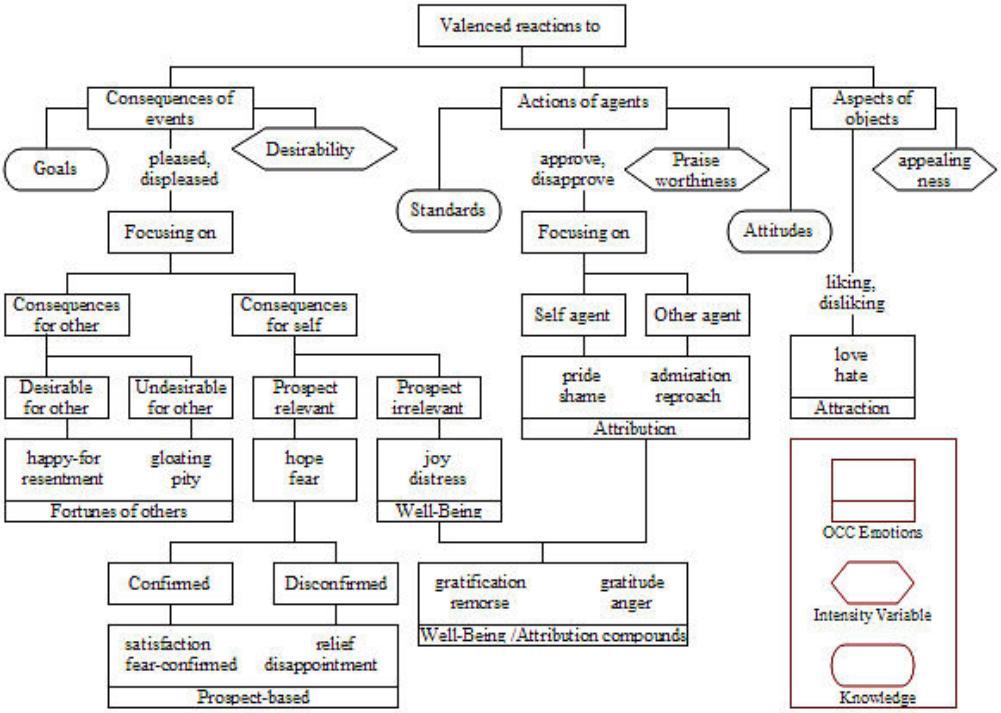

Figure 9: The OCC cognitive theory of emotions model.