Intelligent tutoring systems: an ontology-based approach

Emeka Oguejiofor[1], Rafal Kicinger[2], Elena Popovici[3], Tomasz Arciszewski[4] and Kenneth De Jong[5]

ABSTRACT | A novel methodology for building tutoring system is proposed. It includes the integration of state of the art computer science methods and tools and the use of an ontology for the core knowledge representation. First, the paper presents the ongoing Information Technology revolution in engineering and the related paradigm changes in education. Next, an overview of the concept of an ontology and its various definitions are provided, along with available ontology development tools. In the following section, an architecture of an ontology-based tutoring system is proposed. As a proof of concept, the proposed architecture was used in building the GMU Educator, an intelligent tutoring system developed in the School of Information Technology and Engineering at George Mason University. A detailed description of the GMU Educator is then presented with examples. Finally, conclusions and plans for further research are provided in the last section of the paper.

KEYWORDS | intelligent tutoring system, ontology, Protégé-2000, GMU educator

1 Introduction

The main objective of this paper is to propose a new concept of an intelligent tutoring system based on the use of an ontology for knowledge representation. An ontology provides a specific body of knowledge in a formal, structured and precise form such that knowledge can be interpreted unambiguously and, more importantly, used by computers. As such, it seems to be a perfect match to the engineering education needs. Therefore, the authors decided to investigate the use of an ontology as the core knowledge representation scheme in their intelligent tutoring system. This has resulted in the development of a novel ontology-based intelligent tutoring system for learning about personal air vehicles. The conducted project has produced, in addition to building a specific educational tool, various methodological and software integration results. These results could enable and simplify the development of subsequent intelligent tutoring systems based on the use of an ontology for knowledge representation.

The paper provides a brief historical perspective on the role of Information Technology in engineering

1. Associate Professor, St. Francis Xavier University, Antigonish, Nova Scotia, Canada B2G 2W5 and Visiting Scholar, Civil, Environmental and Infrastructure Engineering Department, George Mason University, Fairfax, VA 22030
2. Ph.D. student, Civil, Environmental and Infrastructure Engineering Department, School of Information Technology and Engineering, George Mason University, Fairfax, VA 22030
3. Ph.D. student, Computer Science Department, School of Information Technology and Engineering, George Mason University, Fairfax, VA 22030
4. Professor, Civil, Environmental and Infrastructure Engineering Department, School of Information Technology and Engineering, George Mason University, Fairfax, VA 22030
5. Professor, Computer Science Department, School of Information Technology and Engineering, George Mason University, Fairfax, VA 22030

and this is followed by a description of the ontology development and implementation. Next, it proposes an architecture for a tutoring system based on an ontology. Finally, the developed system, called "GMU Educator" is described with the initial experience of its use. The paper ends with conclusions and an outline of the future possible research issues.

2 Paradigm changes

The beginning of the 21st century is marked by the ongoing Information Technology (IT) revolution in engineering. What is usually understood by this term is a complex process of changes in all engineering domains, including education, which is centered on computing. The process is mostly driven by progress in computer science and computer engineering and causes in return the ever-decreasing costs of computing, and its rapidly growing speed and availability. Also, the process is strongly affected by improvements of our understanding of such fundamental concepts as knowledge, its acquisition, representation and use, and human and machine intelligence, particularly in the computational context. The synergistic nature of developments in computing and Artificial Intelligence (AI) have already created a new situation for engineering educators who are interested in the utilization of IT, and in the development of various advanced educational computer-based tools.

Grierson has traced the origin and development of IT through the history of our civilization starting with the first cave paintings and writings, followed by the invention of printing in the 1500s, and by the development of electronic computers and communication systems in the latter half of the twentieth century [1]. The IT revolution has been facilitated by a variety of inventions and is continuously driven by the complex integration of various computer science and engineering domains, methods and tools [2]. Some of these inventions include the telegraph in the 1830s, the telephone in the 1870s, the emergence of the electronics industry in the early twentieth century, with subsequent inventions of radio (1920s), radar and television (1930s), and most importantly the electronic computer in the 1940s with the completion of the ENIAC in 1946. This was followed shortly by the invention of the transistor (1947) and the integrated circuit (1957) and later by the microprocessor in the early 1970s, which enabled the development of personal computers. Also, noteworthy are earlier developments in information transmission systems such as the microwave for long-distance telecommunications in the 1940s, satellite communication (1950s) and optical fibre transmissions in the 1970s. Continuing improvements in microprocessor speed and various other aspects of computing technology have enabled the building of more powerful machines that are able to undertake complex computations and manage complex tasks.

From the perspective of engineering educators, two paradigm changes, both caused by the IT revolution, have had a tremendous impact on the development of educational tools. The first one is the change in the character of tools that provide purely quantitative numerical analyses, to decision support tools with both quantitative and qualitative components. The second paradigm change is the transition from tools intended for a single computer/user to tools for multiple computers/users, which are integrated over the Internet. Engineering educators have been interested in the classroom use of computer tools for a long time. Over the last 30 years, an interesting line of evolution of the use of computers in engineering education can be observed, reflecting both paradigm changes and driven by progress in the state of the art in computing and in AI.

The first computer tools used in education were exclusively intended for analytical purposes to deal with the quantitative aspects of a numerical simulation of behavior of engineering systems as described in [3]. Next, a generation of computer tools for detailed structural design emerged, which was

followed by integrated systems for the analysis, design and optimization of engineering systems. The best example of such systems is SODA, which is intended for the analysis, dimensioning, and optimization of steel structural systems. It was initially developed in an academic environment at the University of Waterloo in Canada, and later commercialized by Acronym Software Inc.[4]. It has been used for educational purposes since the mid–80's.

The first paradigm shift occurred around the mid-80's when the nature of engineering computer tools changed from purely quantitative/analytical to a combination of both quantitative and qualitative/non-numerical. This significant change meant the emergence of a new generation of computer tools, called knowledge-based systems. These tools were intended to address the qualitative, non-numerical aspects of engineering decision making, particularly in the context of conceptual design [5]. Knowledge-based systems contain, in addition to various analytical components, formal knowledge in the form of a collection of heuristic decision and inference rules. The use of formal knowledge representation is a significant progress in the direction of building intelligent systems, but unfortunately such rules are still a relatively restrictive form of knowledge representation, insufficient for educational purposes. Fortunately, over the last several years, a lot of research [6, 7, 8, 9, 10] has been conducted on knowledge representation and the concept of an ontology, which is discussed in the following section, has emerged. The representation of knowledge in the form of an ontology in educational tools represents a significant advancement and a chance to expand the use of such tools, at the same time making them more universal, and easier to develop and use.

In the early 90's, the progress in network computing led to the second paradigm shift and the emergence of a new generation of Web-based educational tools. For example, Arciszewski [11] at George Mason University developed Dr. Structure, a tool for teaching fundamentals and conceptual structural design of steel structures and Lakmazaheri [12] at the Catholic University of America built a complex system OleSteel for teaching the detailed structural design of steel members and connections allowing students to take customized tests and quizzes.

3 Ontology and its development

The IT revolution has also had an impact on the information and knowledge management. Prior to the advent of computers, information was catalogued and stored in paper form. When desktop computers became easily available, information could be entered into individual machines but could only be accessed and manipulated on the machine with the information. Thus the same information had to be duplicated on other machines to make it available to more than one individual at the same time.

The advent and subsequent popularity of the Internet has resulted in a vast amount of information being readily available. With the transcription of information into electronic format, the ability of anyone with a desktop to make information readily available on the World Wide Web and the ability and ease to access such information from a desktop computer that is located anywhere in the world, the question of how to manage information has become a major challenge [6]. Advances in the areas of parallel, distributed and network computing have contributed towards the goal of intelligent and efficient management of information that reside at diverse locations. Furthermore, the growth of disciplines such as Artificial Intelligence and Machine Learning has resulted in the need to structure knowledge with sufficient clarity that people and machines can share common understanding of the terms or concepts in the knowledge domain. Specifically, the development of ontologies has been driven by the need to develop a framework for a common vocabulary that would enable people and software agents to share common understanding of the meaning of the basic concepts and terms in a given domain of knowledge as

well as the relationships between them [13]. According to [14], an ontology provides a way to carve up reality in order to understand and process it.

Software agents that use knowledge stored in ontologies are frequently utilized on the Web to categorize products and Web sites [15]. Ontologies are being applied in the traditional field of Artificial Intelligence in agent technologies that engage processes which operate on information artifacts, with agents acting on behalf of humans and also interacting with humans and other agents [14, 16]. In addition, ontologies are being applied in the fields of Knowledge Management and Knowledge Engineering. Gruninger and Lee [17] discuss the emerging discipline of ontological engineering whose objective is the effective support of ontology development and use throughout its entire lifecycle. In this context, the science of ontology may be described as the study of the general concepts and abstractions that make up the fundamental aspects of our world. Milton et al. [6] advocate the development of Knowledge Technology to address activities related to knowledge creation, acquisition, mapping, retrieval use and re-use.

The term ontology has many definitions, depending on the discipline of interest. For example, in philosophy, ontology is viewed as the study of the kinds of things that exist, while within the Artificial Intelligence and Knowledge Management community, some of the many, and at times contradictory, definitions of ontology include:

- a knowledge representation vocabulary [18] in which the terminologies have been structured to capture the concepts being represented precisely enough to be processed and interpreted by people and machines without any ambiguity [19].
- a formal explicit description of concepts in a domain of discourse, properties of each concept describing various features and attributes of the concepts, along with all relevant restrictions [15].
- knowledge representation, using representation vocabulary, of a body of knowledge that describes some knowledge domain [18]. The authors further explained that "representation vocabulary provides a set of terms with which to describe the facts in some domain, while the body of knowledge using that vocabulary is a collection of facts about the domain."

In ontologies, informational structure, represented as terminology and driven by a set of rules that govern the interpretation of definitions and constraints, is delineated from informational content [19]. An ontology typically consists of a hierarchical arrangements of the classes (which describe the major concepts in the domain) and subclasses (which are the more specific concepts under a particular class). Furthermore, the properties of the classes and subclasses (description of the features and attributes) are specified as well as the relevant restrictions. Noy and McGuinness [15] make a distinction between an ontology and a knowledge base, stating that what constitutes a knowledge base is the addition of individual instances into the ontology, i.e. the population or instantiation of the ontology. The advantages of an ontology may be summarized as follows:

- It abstracts the essence of a concept [20].
- It catalogues and distinguishes various types of objects and their relationships [13, 20, 21].
- It facilitates the seamless communication, sharing and reuse of domain knowledge [15].
- It provides efficient means of structuring concepts to achieve effective computation [19].
- It enables explicit specifications of domain assumptions [15].
- It provides a convenient means for separating operational knowledge from domain knowledge [22, 23].

A number of ontologies have been developed and deployed on the Web, the more popular ones being the Yahoo! site, which provides taxonomic categorizations of Web sites and the Amazon.com site, which provides products categorizations. The CYC project [24], began

in 1984, represent an ambitious attempt to build a large reusable ontology. The Ontolingua project at Stanford University [25] resulted in the development of the Ontolingua ontology-editing environment. Other ontology development tools that have resulted from the research at the Knowledge Systems Laboratory at Stanford University include Chimaera and Protégé-2000 [15].

After investigating a number of knowledge representation and management packages, the Stanford University ontology editor, Protégé-2000 [26], was adopted in the reported research for capturing and structuring the personal air vehicle domain knowledge. This tool comes with an easy to use set of tools for building, editing and visualizing ontologies, as well as capabilities for importing from and exporting to different ontology formats. Its open-source Java API allows dynamic ontology access. Also, its architecture is extensible, and various new plug-ins are continuously being developed by a large and active user community.

The first task in building an ontology within the Protégé-2000 environment is to define the concepts, which are treated as classes, and to arrange these in a taxonomic, superclass-subclass hierarchy. The creation of classes within the Protégé-2000 environment is easily accomplished and does not require any skill beyond that needed to use the Microsoft Windows Operating System. Once created, classes can be rearranged simply by "grabbing-and-dropping" at the location desired. It is also possible to define concepts that belong to multiple superclasses. Figure 1 shows part of the taxonomic hierarchical arrangement of the classes and subclasses in the ontology developed in the reported research. In this figure, all the concepts with the superscript ^{M} belong to more than one superclass. When a class is selected, a list of the superclasses it belongs to is displayed, e.g. the PersonalAirVehicle concept shown highlighted in Figure 1 has as its superclasses both the HeavierThanAirCraft and PrivateAirplane concepts. The naming convention adopted for the concepts is to capitalize the first character of the word(s) that make up the class name.

Following the establishment of the taxonomic hierarchy for the concepts, the properties are defined. In Protégé-2000, this is done through the creation of slots and defining the allowed values for the slots. Different

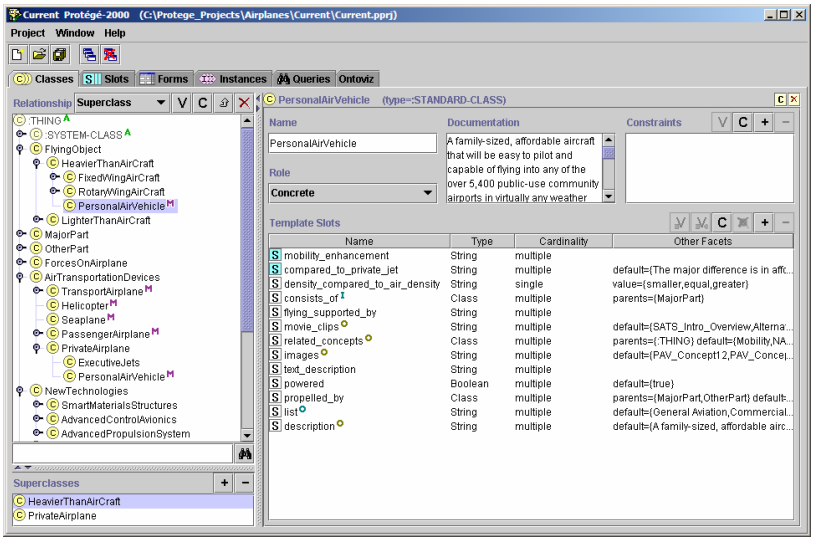

Figure 1. Hierarchical Arrangement of Classes and Subclasses in Ontology

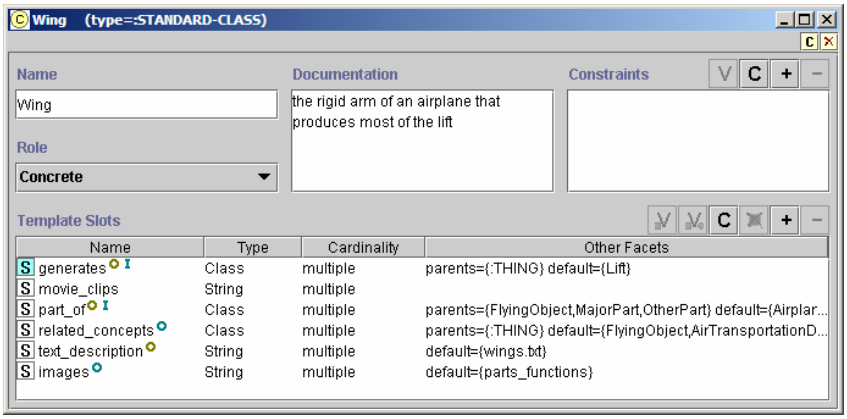

Figure 2. List of slots for the class "Wing"

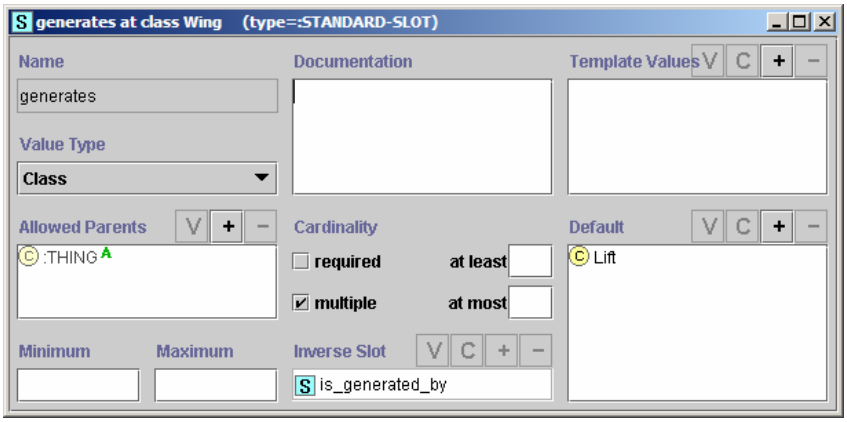

Figure 3. Details of the slot "generates" for the class "Wing"

concepts could share the same slot name that has unique slot values associated with the concepts, for example the slot "part_of", is associated with the concepts "Wing", "FlyingObject", "MajorPart and "OtherPart", as shown in Figure 2. Slot values could be of type Class, String, Instance, Integer, Boolean, Symbol, Float, or Any. Depending on the slot type, it could have multiple cardinality, which would allow for the specification of more than one value for the slot. Figure 2 shows, as an example, a list of the slots that are associated with a given concept, in this case "Wing" while Figure 3 shows detailed information for the slot "generates" of the same concept, with default slot value "Lift". In this manner, the knowledge that Wing generates Lift is encoded in the ontology. Because the property "generates," which is a slot of type Class and has multiple cardinality, has ":THING" as its allowed superclass (or parent), the same slot can be attached to the concept "Engine," but it would have a default value of "power" to specify that the engine generates power.

The final step in the development of an ontology and turning it into a knowledge base, involves the creation of instances and entering slot values for the instances. Fig. 4 shows a partial listing of the instances which have been created for the concept "Topics" that describes teaching scenarios for learning about the domain of personal air vehicles.

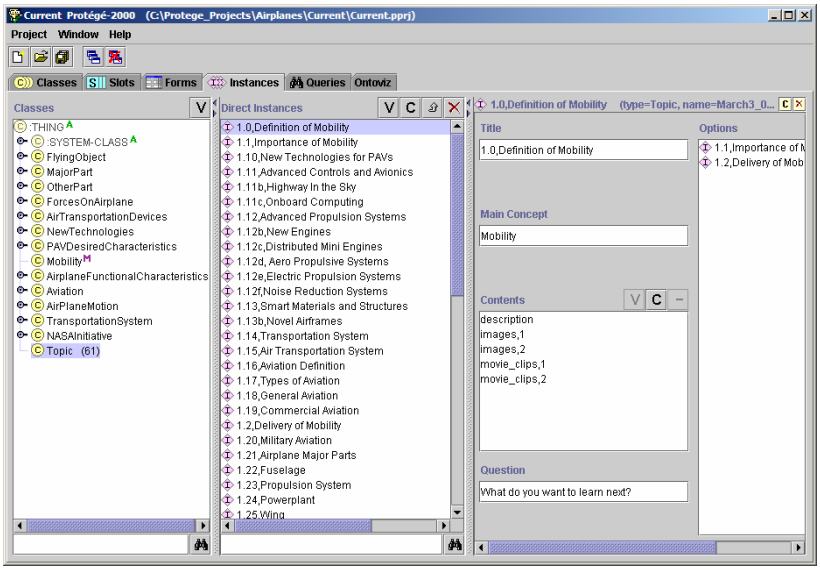

Figure 4. List of Classes and Instances

At the present time, all the information in the ontology is derived from available literature in the area of the domain knowledge. It is possible that in the future research the developed ontology will be tested and evaluated by the NASA experts who in the process will transfer their unique knowledge into the knowledge system.

Prior to the choice of Protégé-2000, the ConceptMap software [27], developed at the Institute for Human and Machine Cognition, University of West Florida, was used to lay out ideas and concepts pictorially and show how they are interconnected. For example, Figure 5 shows a concept map that was developed in the process of structuring the domain knowledge. This map depicts the arrangement of the classes and subclasses for a part of the ontology. Multimedia resources (text files, HTML files, video clips and graphic files) were used as values for some of the slots. However, unlike Protégé-2000, the concept maps cannot be used for dynamic access to the knowledge or for reasoning. A similar idea to the ConceptMap, called Think-maps, was proposed by Oxman [28] and used as the framework for structuring knowledge in the domain of design education.

4 Architecture of an Ontology-Based Tutoring System

One of the goals of the research presented in this paper was the development of a novel methodology for building intelligent tutoring systems for engineering education. This was achieved by integrating state-of-the-art computer science techniques as presented below.

Building an intelligent tutoring system for engineering education involves tackling tasks such as: acquiring and representing knowledge; visualizing and updating knowledge; integrating knowledge; constructing a graphical user interface that is friendly, intuitive and uses multimedia for a more effective learning experience; and integrating everything into an intelligent tutoring system. An extensive research effort has been conducted to analyze existing tools and technologies in order to find the best ones for these tasks, while keeping in mind goals such as platform independence, modularity and web-based access.

As already discussed earlier in the paper, an ontology has been chosen for knowledge representation and, an

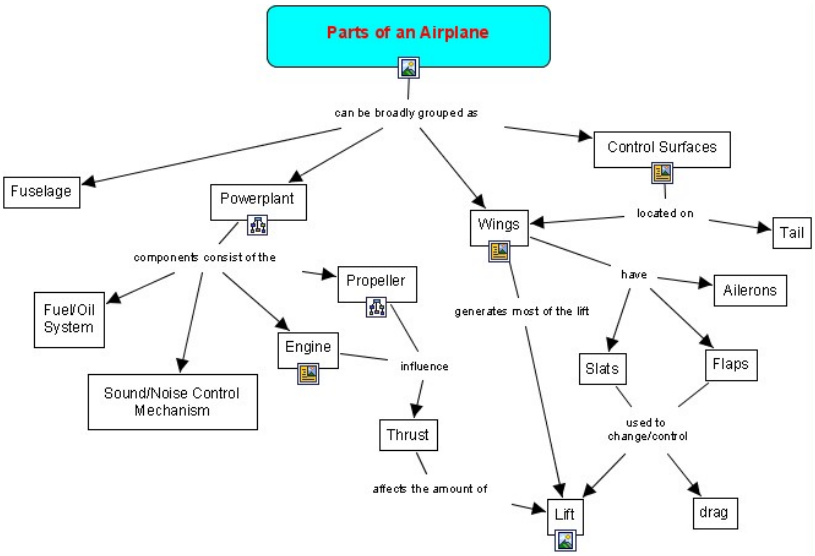

Figure 5. Concept Map of Part of the Knowledge Base

ontology management software, called "Protégé-2000", was used. The package provides sufficient visualization capabilities in order to be used by a knowledge engineer for knowledge acquisition. However, the envisioned system was targeted at an end-user trying to learn a domain. This required the development of a user-friendly graphical user interface that would seamlessly present the domain knowledge to the end-user. The development of such an interface implied the need for integrating the knowledge base produced by Protégé-2000 with visualization software.

A set of Macromedia products turned out to be the solution for knowledge integration and visualization. Macromedia Flash [29] was used for developing the graphical user interface (GUI) front end. The Macromedia JRun [30] application server was chosen as the middleware that connects the GUI with the back-end knowledge base, the main reason being its smooth integration with both Action Script and Java (Flash's Scripting language). The dynamic communication between the Flash graphical interface and the server is facilitated by the Flash Remoting [31] technology. Flash Remoting is basically an Action Script application programming interface (API) for transparent remote method invocation (RMI). On the other side of the middleware, the JRun server accesses the knowledge base by forwarding the calls to a Java object built on top of the Protégé-2000 API.

Web-based access to the system was provided by embedding the Flash movie in an html web page, which was then deployed on a web server and from which it is available to anyone with access to the internet. The user interacts with the Flash movie through the browser; the Flash movie makes calls to the backend through the JRun server. Information is extracted from the knowledge base and sent back to the Flash movie which dynamically displays it in the form of text, images and movies. Support for Microsoft Agent technology has been added for Windows platform users in order to enhance the learning environment with speech interaction.

The schematic architecture of the George Mason University Intelligent Educator is presented in Figure 6. In an offline process, a Knowledge Engineer works

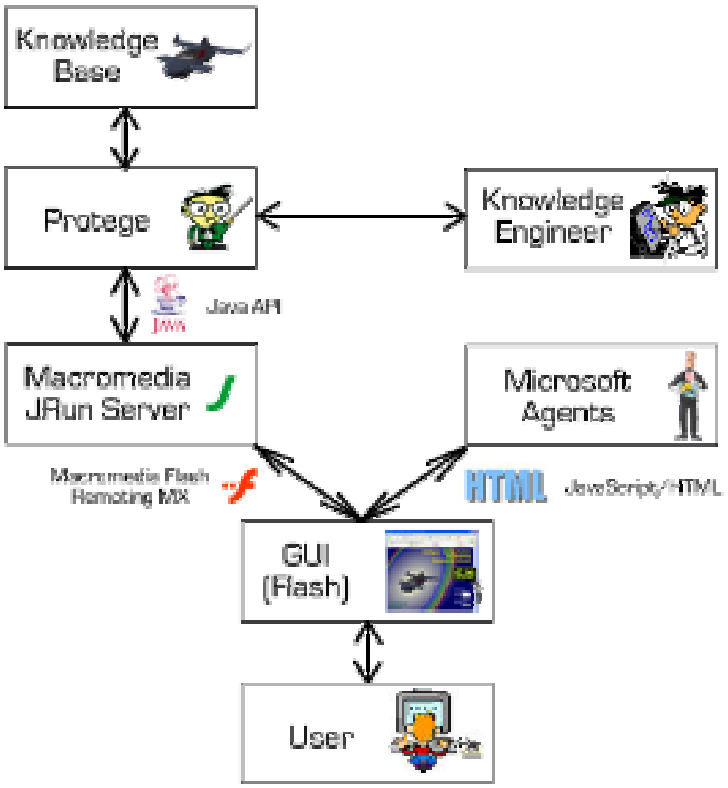

Figure 6. Architecture of the George Mason University Intelligent Educator

with the Protégé-2000 tools to build an ontology and subsequently update it. The ontology can then be accessed by the end user through the system and its contents displayed to the user. The system is independent of the actual contents of the knowledge base and this fact guarantees the independence of the application domain.

The developed system is modular, which allows for future replacement of various components as long as they comply with the current interfaces. Platform independence is provided at various levels. On the back-end, ontology building is platform independent because the Protégé-2000 software is a Java application; so is ontology access, as it is done through a Java API. For the middleware, the JRun is available for several platforms or can be replaced with a different compatible application server such as, for example, Macromedia ColdFusion. The front-end is web-accessible and Flash Player plug-ins are available for a multitude of browsers and platforms.

5 Proof of concept: GMU Educator

Using the presented methodology and architecture, an actual tutoring system has been built as a "proof of concept" demonstration for the domain of Personal Air Vehicles (PAVs). Effort has been put into the study of this domain, resulting in a preliminary ontology that is continuously being refined.

Two modes of operation have been envisioned for the system: guided (through tutoring scenarios) and autonomous (independent browsing). In the browsing mode, the users can navigate on their own through the concepts present in the ontology, either by typing

Figure 7. Multimedia Learning Environment – Independent Browsing Mode

concept names or by clicking on the boxes showing concepts other than the current one (see Fig. 7). Small snapshots of the hierarchical structure can be seen as each concept (central) is shown together with its super-concepts (above) and sub-concepts (below). Other related concepts are also shown to the right, together with a text description. The bottom part displays images and movies.

The guided, or tutoring mode, gives the user the option of learning in a predetermined way. In an offline process, a domain expert can create teaching scenarios. Subsequently, students can autonomously use the system and learn, following such scenarios. A scenario is a set of topics. Each topic refers to a concept from the ontology of the domain. The teacher can however choose which parts of all the information associated in the ontology with a given concept are to be used in that particular topic. Obviously, there can be multiple topics associated with the same concept. The teacher also defines the order in which it is recommended that the student goes through the topics. At each step however, several alternatives are allowed, in addition to the recommended/default one, as shown in Figure 8.

The approach implemented is flexible from two perspectives. First, it can accommodate different styles of teaching and learning. At one end of the spectrum, there is the very strict style of having a linear sequence of topics that the user must follow in order, disallowing any choices other than to go backward or forward. The other extreme is where at every step the student is given multiple choices of where to go next. Somewhere in between is a teaching style where at the beginning there is a linear sequence of topics and as the student gets enough information in the field, choices are then presented. Except for the very strict teaching style, the approach can accommodate different learning styles, too. Some students may prefer to always pick the recommended/default option, while more independent learners can make their own choices. So various teachers can present the same ontology contents in different ways and various students can browse the same tutorial by following their own paths.

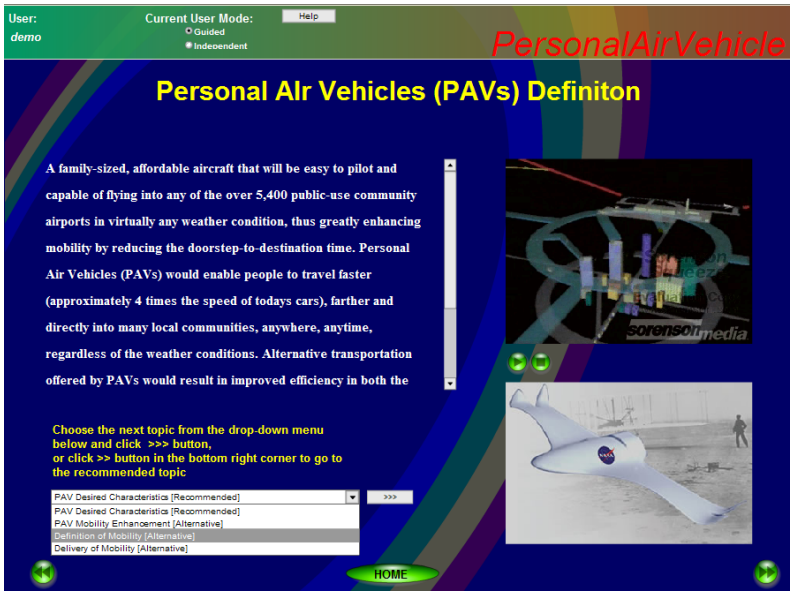

Figure 8. Multimedia Learning Environment – Guided Mode

For maximum convenience, we chose to store the tutorials in the ontology too. Teachers work offline with the Protégé-2000 tool to define topics, group them in scenarios and define the sequence of topics and the alternatives at every step. Each topic is an instance of a general concept called Topic. The properties of such instances point to concepts from the domain specific part of the ontology and their properties.

An additional feature is the possibility of switching back and forth between the guided and independent browsing modes of the system. If the user is viewing a topic in the guided mode and switches to the independent mode, he or she will be taken to the concept associated with that topic. The user can further browse independently and upon switching back to the guided mode the system returns to the point from where it had left.

For now, constructing scenarios requires that the teacher is first familiarized with the use of the Protégé-2000 software. In the future, a more friendly user interface could be developed to allow for easier tutorial building.

6 Conclusions

Our initial research has clearly demonstrated that building intelligent tutoring systems based on ontologies is feasible. Unfortunately, the development process is still much more complicated, difficult, and time consuming than initially anticipated. At present, there is not a single development tool that could be used for building an entire intelligent tutoring system. Therefore, at this time such a system can be developed only by a team comprised of engineers (domain experts), knowledge engineers, and computer scientists working with various computer tools. Unfortunately, the optimal use of a combination of various tools and the integration of their products require deep understanding of computer science and extensive programming. This leads to a long, costly and complicated development process in which careful coordination of the efforts of the individual team members is required. However, all these reported problems could be overcome if a significant effort is undertaken in building a tool for the development of intelligent tutoring systems. The development of such a tool could significantly reduce the costs of

building intelligent tutoring systems and could lead to the expansion of their use. Facilitating the user's ability to update the knowledge base through some form of software-assisted knowledge acquisition (SAKA) would have a significant impact on reducing development costs and encourage more people to use the system

The research team intends to continue the project with the goal of further addressing the challenges listed above. During the second year of the project, the developed intelligent tutoring system will be thoroughly tested in cooperation with the NASA experts. Also, the content of the system will be expanded and modified, if necessary. More importantly, the system will be integrated with Inventor 2003/G, an experimental design tool, developed in the School of Information Technology and Engineering at George Mason University [32]. Inventor 2003/G is a multi-population evolutionary design system developed in Java and intended for use over the Internet. It is a generic system that can be integrated with various simulation packages to be used for the evaluation of the generated designs. The integration of Inventor 2003/G with the George Mason University Intelligent Educator will allow the user not only to learn the conceptual knowledge about PAVs, but also to develop some understanding of the design concepts and various notions related to the domain being studied in the context of an entire design process.

The reported research has clearly revealed many developmental challenges related to intelligent tutoring systems. On the other hand, it has demonstrated the feasibility of building such systems and has shown the potential educational benefits of their use in an engineering environment. Most likely we will witness the emergence of various intelligent tutoring systems in the near future and that will mark a paradigm change in engineering education.

Acknowledgements

The authors gratefully acknowledge the support for their research from the NASA Langley Research Center under the Grant 01–1231.

REFERENCES

[1] Grierson, D.E. *Information Technology in Civil and Structural Engineering Design in the Twentieth Century*. Computers adnd Structures, 1998. **67:** p. 291–298.

[2] Arciszewski, T. and S. Lakmazaheri, *Designing Structural Design Education*. Proceedings of the Second Mudd Design Workshop "Designing Design Education for the 21st Century, 1999. California.

[3] Fenves, S.J. and W.J. Rasdorf, *Role of ASCE in the Advancement of Computing in Civil Engineering*. Journal of Computing in Civil Engineering, 2001. **15**(4): p. 239–247.

[4] Grierson, D.E. *Computer-Automated Optimal Design for Structural Steel Frameworks*. Proceedings of the *NATO ASI Conference on Optimization and Decision Support Systems in Civil Engineering*. 1989. Edinburgh.

[5] Maher, M.L., D. Sriram, and S.J. Fenves, *Tools and Techniques for Knowledge-Based Expert Systems for Engineering Design*. Advances in Engineering Software, 1984. **6**(4).

[6] Milton, N., N. Shadbolt, H. Cottam and M. Hammersley, *Towards a Knowledge Technology for Knowledge Management*. International Journal of Human-Computer Studies, 1999. **51:** p. 615–641.

[7] Chan, C.W., L. Chen and L. Geng, *Knowledge Engineering for Intelligent Case-Based System for Help Desk Operations*. Expert Systems with Applications, 2000. **18:** p. 125–132.

[8] Mili, F., W. Shen, I. Martinez, P. Noel, M. Ram and E. Zouras, *Knowledge Modeling for Design Decisions*. Artificial Intelligence in Engineering, 2001. **15:** p. 153–164.

[9] Richards, D. and S.J. Simoff, *Design Ontology in Context – A Situated Cognition Approach to Conceptual Modelling*. Intelligence in Engineering, 2001. **15:** p. 121–136.

[10] Ahmed, S. and K.M. Wallace, *Understanding the Knowledge Needs of Novice Designers in the Aerospace Industry*. Design Studies, 2004. **25**(2): p. 155–173.

[11] Arciszewski, T. *Internet-Based Teaching/Learning Tools in Structural Engineering Education*. Proceedings of the *NASA Workshop on Advanced Technology for Engineering Education and Training*. 1998. Hampton, VA.

[12] Lakmazaheri, S. and T. Arciszewski, *Towards Self-Directed Computer-Mediated Learning for Undergraduate Structural Engineering Education*. Proceedings of the *Third International Symposium on Civil Engineering Learning Technology*. 1999. Cardiff University, Wales, UK.

[13] Gruber, T.R. *A Translation Approach to Portable Ontology Specification*. Knowledge Acquisition, 1993. **5:** p. 199–220.

[14] Castel, F. *Ontological Computing*. Communications of the ACM, 2002. **45**(2): p. 29–30.

[15] Noy, N.F. and D.L. McGuinness, *Ontology Development 101: A Guide to Creating Your First Ontology*. 2001, Knowledge Systems Laboratory/Stanford Medical Informatics, Stanford University.

[16] Skolicki, Z. and R. Kicinger, *Intelligent Agent for Designing Steel Skeleton Structures of Tall Buildings*. Proceedings of the *International Workshop on Information Technology in Civil Engineering*. 2002. Washington DC: American Society of Civil Engineers.

[17] Gruninger, M. and J. Lee, *Ontology Applications and Design*. Communications of the ACM, 2002. **45**(2): p. 39 – 41.

[18] Chandrasekaran, B., J.R. Josephson, and V.R. Benjamins, *What Are Ontologies, and Why Do We Need Them*? IEEE Intelligent Systems, 1999. **14**(1): p. 20–26.

[19] Kim, H. *Predicting How Ontologies for the Semantic Web Will Evolve*. Communications of the ACM, 2002. **45**(2): p. 48 – 54.

[20] Crawford, D., *Editorial Pointers*. Communications of the ACM, 2002. **45**(2): p. 5.

[21] Musen, M.A., *Dimensions of Knowledge Sharing and Reuse*. Computers and Biomedical Research, 1992. **25:** p. 435 – 467.

[22] McGuinness, D.L. and J. Wright, *Conceptual Modeling for Configuration: A Description Logic-Based Approach*. Artificial Intelligence for Engineering Design, Analysis and Manufacturing. Special Issue on Configuration, 1998.

[23] Rothenfluh, T.R., J.H. Gennari, H. Eriksson, A.R. Puerta, S.W. Tu, and M.A. Musen, *Reusable Ontologies, Knowledge-Acquisition Tools, and Performance Systems: Protégé-Ii Solutions to Sisyphus–2*. International Journal of Human-Computer Studies, 1996. **44:** p. 303 – 332.

[24] Lenat, D.B. Cyc: *A Large-Scale Investment in Knowledge Infrastructure*. Communications Of The ACM, 1995. **38**(11).

[25] Farquhar, A., R. Fikes, and J. Rice, *The Ontolingua Server: A Tool for Collaborative Ontology Construction*. Proceedings of the *Tenth Knowledge Acquisition for Knowledge-Based Systems Workshop*. 1996. Banff, Canada.

[26] Noy, N.F., R.W. Fergerson, and M.A. Musen, *The Knowledge Model of Protege–2000: Combining Interoperability and Flexibility*. Proceedings of the *2nd International Conference on Knowledge Engineering and Knowledge Management (EKAW'2000)*. 2000. Juan-les-Pins, France.

[27] Coffey, J.W., R. Hoffman, A.J. Cañas, and K.M. Ford, *A Concept Map-Based Knowledge Modeling Approach to Expert Knowledge Sharing*. Proceedings of the *IASTED International Conference on Information and Knowledge Sharing*. 2002. Virgin Islands.

[28] Oxman, R. *Think-Maps: Teaching Design Thinking in Design Education*. Design Studies, 2004. **25**(1): p. 63–91.

[29] Macromedia Flash Mx: Product Overview. 2003, http://www.macromedia.com/software/flash/productinfo/product_overview/.

[30] Macromedia Jrun: Product Overview. 2003, http://www.macromedia.com/software/jrun/productinfo/product_overview/.

[31] Macromedia Flash Remoting Mx: Product Overview. 2003, http://www.macromedia.com/software/flashremoting/productinfo/product_overview/.

[32] Pullmann, T., Z. Skolicki, M. Freischlad, T. Arciszewski, K.A. De Jong, and M. Schnellenbach-Held. *Structural Design of Reinforced Concrete Tall Buildings: Evolutionary Computation Approach Using Fuzzy Sets*. Proceedings of the *10th European Group for Intelligent Computing in Engineering EG-ICE*. 2003. Delft, The Netherlands: SIECA.