Asian Journal of Information Technology 5 (5) : 497-503, 2006

A Cooperative Authoring of Intelligent Tutoring Systems

[1]Said Talhi, [2]Mahieddine Djoudi and [3]Mohamed Batouche
[1]Department of Computer Science, University of Batna, Algeria
[2]Laboratoire SIC and ERTe IRMA, University of Poitiers, France
[3]Department of Computer Science, University of Constantine, Algeria

Abstract: In this study, we present an authoring tool model for the cooperative design of hypermedia intelligent tutoring systems. Based on a client-server architecture, the system allows several authors geographically dispersed to collaborate to produce such tutors together. We consider, within this framework, the creation of a shared workspace gathering all tools necessary to the cooperative development task. The architecture of the authoring tool and the mechanism needed to manage the notification and group awareness is outlined

Key words: Cooperative authoring systems, intelligent tutoring systems, groupware, client-server

INTRODUCTION

Despite the efforts carried out during the last few years, the Intelligent Tutoring Systems[1] development remains a difficult task to undertake. This task often requires the constitution of interdisciplinary team. Experts from different fields such as education and psychology must cooperate with computer engineers to design such systems. In order to improve the productivity in this domain and allow a wider community to be involved, authoring systems have appeared and some of them allow the users to develop Intelligent Tutoring System (ITS), sometimes, without writing a single line of code. Thus, the task is reduced in a way that the experts need only to introduce knowledge to generic ITS predetermined by the system. Some of these authoring systems are discussed by Murray[2], however, they were all designed to work in a single-user mode.

Today, with the complexity and the interdependence of different sciences, we think that it's very difficult for an alone author to construct an ITS. The design and implementation of an ITS in a given domain, usually necessitates the cooperation of various experts who, in most time, are dispersed geographically. Consequently, it is necessary to provide them with cooperative tools that give them the possibility to communicate and coordinate their activities.

Recently, thanks to the networks and groupware, virtual meetings involving many people are made possible[3,4]. Several works in this area are already available in such domains as the cooperative writing[5-7], the multimedia, the cooperative design of objects, etc. The common point between all these systems is that they allow several participants to work together in synchronous or asynchronous manner to realize a common task.

Since the cooperative aspect, through a computer network, has been experimented successfully in a lot of domains, this leads us to think that it would be desirable that the designers of future authoring tools should integrate this cooperation functionality for ITSs production.

This is the object of this study. We investigate this idea through an authoring system called CAMITS (Cooperative Authoring Model for Intelligent Tutoring Systems).

ITS AUTHORING SYSTEMS

Single-user authoring systems: During this last decade, several works has been taken on the design and implementation of ITS authoring systems. Besides our experience with MOALIM system[2,8,9], Tom Murray listed more than twenty references in his recent paper and presented his authoring system named Eon[10]. These systems are classified in seven categories according to the type of ITSs they produce. These categories are: 1. curriculum sequencing and planning, 2. tutoring strategies, 3. device simulation and equipment training, 4. domain expert system, 5. multiple knowledge types, 6. special purpose and 7. intelligent/adaptive hypermedia.

Given that ITSs are often described as having four main components (domain module, tutoring module, student model, and student interface), the authoring systems must therefore theoretically include all the necessary tools for building these components. However,

Corresponding Author: Said Talhi, Department of Computer Science, University of Batna, Algeria

it has to be recognized that, very few systems require from the author to construct all modules, as in the case of Eon system[10] for example. The other systems are usually limited to tools for constructing one, two or at limit three components among the four. The remaining components are generally predefined in a pattern of ITS and the author is solicited only to introduce necessary parameters for their functioning.

CAMITS, the system presented in this paper, offers cooperative functionalities to the authors and generates ITSs classified as first and seventh category of the Murray classification mentioned above. These authoring systems generally structure the teaching material as a network of Learning Units (LUs) where every LU satisfied some educational objectives.

The LUs are linked together to show prerequisite-relations between them. Although these systems do not use any explicit representation of domain knowledge, they investigated nevertheless the intelligence at the sequencing process of the LUs, the manipulation of the hypertext links and the adaptation of the course according to a student level of knowledge[11,12]. The LUs to be presented to the learner are then determined dynamically based on the learner performance, the lesson learning objectives and the relations that exist between the different LUs.

The CAMITS system is among the systems that, in order to facilitate the authors' task, requires only instantiation of the LUs. He next will have only to introduce the prerequisite network and the necessary parameters for the functioning of the three other components.

Cooperative authoring systems design approaches: To develop a cooperative authoring system, several approaches can be proposed. We can classify them in two large categories[13]. A first approach, pragmatic, and more economic in implementation effort, consists to take an existing single-user authoring system and enrich it with other functionalities that makes it cooperative one. However, the rigidity induced by knowledge acquisition units of the single-user authoring systems, makes it very difficult to take into a count group awareness control and the distributed management of the knowledge base. The produced authoring tools will lack certainly effectiveness and will use cooperation mechanisms only to a limited degree. The second method, which we adopted in the design of CAMITS, consists in taking into a count the paradigm of cooperation and the needed tools to do it, at the design step of the system architecture. This approach, although expensive, allows us to apply rigorously the mechanisms of the cooperation metaphor. Though, we must provide through this software architecture, a common work-space to the authors involved in the cooperative construction of an ITS.

However, we should notice that the software does not constitute the only parameter in the success of such cooperative system. Also, we have to take into account the human factors involved due to the group activities because of their importance[14]. Thus, to avoid the inherent conflicts due to the human nature, we propose a group organization that allows an optimal way the construction of the ITS.

This organization facilitates also the manipulation of different components of the ITS during all steps of the project advancement. So, we define three roles through which the authors can participate during the ITS construction process: main author, constructor coauthor and commentator coauthor.

- The role of the main author is to coordinate the whole work and to verify that the calendar is well respected. He defines the ITS logical structure to be produced by decomposing it in several components (chapters, LU, Figures, images, etc.), then he affects the roles to different co-authors. He has free access to all ITS components.
- A constructor coauthor is authorized to create, modify or delete only the components assigned to him. On the remaining ITS components he will have only the role of commentator.
- A commentator coauthor is authorized only to read and/or comment the components assigned to him.

HYPERMEDIA ITS MODEL

The learning environment offers two learning modes to the learner: information mode (free exploration) and training mode (learning with auto-evaluation), and it organizes the teaching process around hypermedia components[15].

The teaching material is structured in three abstraction level hierarchy according to three level hierarchy of pedagogical objectives defined by Hameline[16]: parts (satisfying the general objectives), chapters (satisfying the specific objectives) and the Hypermedia Learning Units (HLU) (satisfying the operational objectives). To intelligently sequencing the curriculum and adapt it to each learner capacities, the management of these components, is ensured by a multi-expert system based on a five sets of production rules.

These rules (for which parameters can be set), called main rules (MRules), describe the different tutoring plans depending on the different learning situations. They

constitute therefore a generic knowledge base that is instantiated in a suitable way for each ITS created by CAMITS.

The instantiation process, producing "generated rules" (GRules), is carrying out automatically by the system, based on parameters delivered by authors. These ITS parameters which are represented in predicates form, describe the quantitative aspect of the teaching material (number of parts, number of chapters, number of learning units, number of questions, number of exercises, etc.).

For the goal of reusability and independence from the domains, the main rules invoke abstracted structures called HLU. These HLU have no knowledge about the ITS domain. They are supposed to receive all kind of knowledge about the domain via instantiation, under all media types that are allowed by the HTML language (text, image, sound, video, applet).

To summarize, we can consider, two levels of knowledge in the curriculum definition:

- A higher level corresponding to the tutoring plans: These plans consist of five sets of rules that invoke HLU of the lower level. Every set of rules has a specific function. These functions are the following: negotiation with the learner for the entry point in the course (where to start) and/or the objectives to reach; deduction of HLUs assumed to be understood after a negotiation phase; planning the learning session; searching and filtering the content of HLU; and auto-evaluation.
- A lower level corresponding to the HLU universe: This universe consists of a hierarchical network constituted of six HLU sub-levels where the first four sub-levels correspond to the course-type HLU (module abstract, part abstract, chapter abstract, HLU classes) and the last two sub-levels correspond to evaluation-type HLU (questions and exercises).

ITS architecture: CAMITS generated ITS architecture is similar to that of a traditional tutoring system[1], It is composed of:

A free-exploration module that allows the learner to navigate freely through the different HLUs.

Three modules representing the training mode:

- A domain-expert module using generated rules to search and filter concept-indexed HLU asked at a given moment.
- A pedagogical module that allows the negotiation of learning session objectives with the learner and generates in turn sequencing plan for the adaptive presentation of the lesson. Two sub-modules realize these two tasks: the negotiator using the negotiation generated rules and the planer using the planning generated rules.
- A diagnosis module that allows the learner evaluation and the maintenance of an overlay type learner model. This module is made up of three sub-modules: an "evaluator" using evaluation generated rules; a "deduction agent" using the deduction generated rules; and a "learner model manager" managing its persistent content.

A supervisor module that allows on one side, the communication with the learner, and on another side, the coordination between the three modules: domain-expert module, pedagogical module and diagnosis-module. This coordination is carried out via message sending.

The difference between our ITS and a classical ITS is the fact that our ITS resides on the server and can therefore be accessed distantly by learners. Its implementation in Php/MySql has certainly helped in spreading its use as a distance learner system.

ITS COOPERATIVE AUTHORING

To be efficient, collaboration requires not only a shared space, but also some support. The shared space must be structured, each user must be aware of the activity of others, some direct communication between people should be provided to allow them to discuss their common task, and coordination means should be provided.

In authoring mode, we have proposed some necessary tools for ITS cooperative authoring. From an author point of view, designing an ITS using CAMITS consists in introducing, via a cooperative editor, a set of objects that will be manipulated in the learner mode.

These objects are made up with teaching material in the form of Hypermedia Learning Units (HLU), prerequisite-network in the form of an oriented graph, ITS parameters in the form of predicates and pedagogical knowledge in the form of production rules.

The cooperation task in CAMITS is introduced at the editing level of the teaching material and at the editing level of the prerequisite-network. These two components are well structured: the teaching material is organized as parts, chapters and HLU and the prerequisite-network is organized as sub-networks form (part-prerequisite network, chapter-prerequisite network, HLU-prerequisitenetwork and concept-prerequisite network).

These structures are well convenient for the fragmentation and then constitute the basis of our cooperative editing approach as in Alliance[5,17]. The two

concept-keys on which is based the design of CAMITS are the fragmentation and edition roles. As previously said, we defined three edition roles of participation for the authors: main author, constructor coauthor and commentator coauthor.

At the beginning of the ITS construction task, a negotiation step is necessary. The main author assigned the edition roles to different co-authors around different fragments of ITS structure in accordance with their competences and availability. Five learning principles had been incorporated into the authoring process[11]. These principles were: a clear definition of pedagogical objectives, definition of pre-requisite knowledge, providing a variety of presentation styles (tell, show and do), enhanced feedback and testing, and permitting the learner to control the direction of the learning session by choosing himself the pedagogical objectives.

Adopted cooperation modes: The cooperative building process of ITS is characterized by a steps-sequence during which the authors can works either individually or collectively. We defined four cooperation modes in JamEdit[6] that we believe to be needed in CAMITS: individual responsibility, dynamic exchange, alternate version and collective responsibility

The first three modes are typically asynchronous cooperation modes. Especially, the third one is inspired from the real principle let us reflect separately on the question and then compare our results after.The fourth one is a typically synchronous mode that allow to relatively reduced number of authors chosen by main author, to finalize the ITS version when the project reaches its final phase

Software architecture: The cooperative editor is organized according to centralized client/server architecture[18]. Then, all the communications pass automatically through the central site (the server). To every client- site, we associate a client process (CPR) that accomplishes all the tasks that are processed locally (the editing tasks for example). We define a server process (SPR) that manages all the communications between the different CPRs and keeps up to date the content of the ITS central copy and the ITS logical structure. The software architecture offers several functionalities that we can decompose in three layers: server layer, editor layer and presentation layer. Every layer is structured as a collection of modules where each module consists of several objects implementing some functionalities (Fig. 1).

The need for information exchange between the two client layers on one side, and between the client and the server on the other side, implies the presence for "dialog

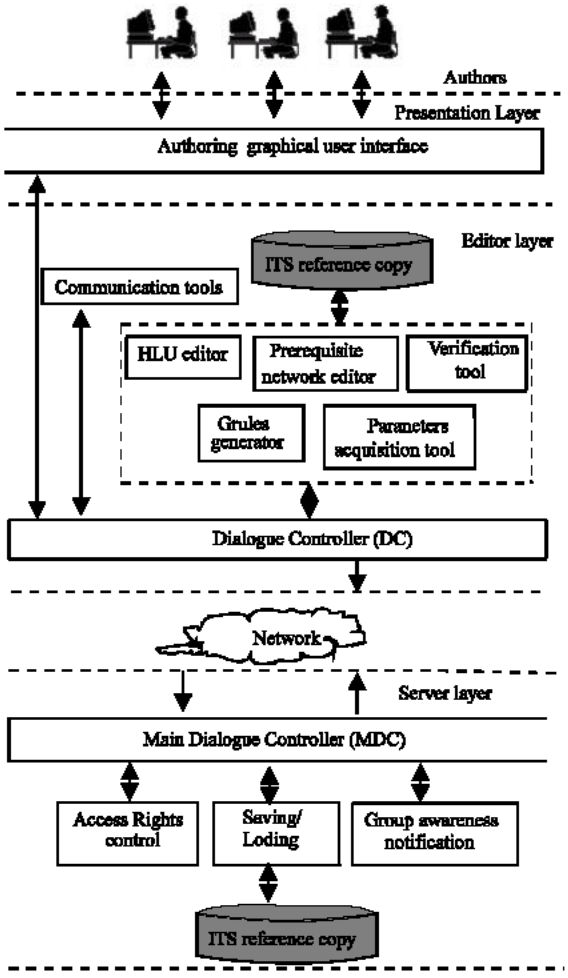

Fig. 1: Software architecture of CAMITS

controllers". We interpose therefore between every presentation layer and every editor layer a Dialog Controller (DC), and between the server layer and every editor layer a Main Dialog Controller (MDC). Messages exchanged between layers transits automatically by the dialog controllers. According to message-type, the convenient objects are then chosen among those that are defined in a layer.

Server layer: This software layer gathers several types of functionalities, among which those that concern ITS logical structure management, as well as the content of ITS components. They allow the authors to save and retrieve ITS objects whose logical structure is declared, as well, at the central level as at the local level.

This software layer is responsible for access rights control, events handling and events notification. In the case of events notification, for example, the concerned module manages a set of queues such as engagement

queue, locking-queue, etc. At every time, if an event occurs, this process identifies the concerned authors and proceeds to structure the notifications as a message form to transmit. These messages then will be made available to another sender module that sends the message.

Editor layer: This software layer gathers many types of functionalities allowing every author to manipulate the objects that constitutes the ITS. These functionalities include not only the support of individual actions, but also the sharing aspect and transparency management. For example, the access to a file in a single user editor delivers directly its content. But in our case, this process consists in several tasks such as access rights verification, locking state of the object and warning the authors working on this file in the same time.

At each author site, some associated functionalities allow the author to save locally the objects that are accessible to him. He will solicit regularly the server to update the versions of these objects. The different components of the editor layer are.

HLU/prerequisite-network-editor: Two modules are designed to implement this component software. They allow the creation task and the maintenance of different ITS objects. The first module allows the wisiwig HLU edition using HTML language. The second module allows the author to edit the prerequisite-network in a graphical form. This oriented network is made up of linked nodes where the links indicates the different possible progressions between the teaching material components. Four levels are used in the network. One level shows the concept-prerequisites, the second shows parts-prerequisites, the third one shows the chapter-prerequisites of a particular part, and the fourth level shows the HLU-prerequisites of a chapter.

Parameters acquisition module: This module allows the main author to specify the ITS parameters that indicate the manner in which the teaching material is decomposed (number of parts, number of chapters in every part, number of HLU in every chapter, etc.). These parameters are saved in the predicates form and then used to instantiate the main rules. For example the predicate nbhlu(1, 2, 4) indicates that chapter 2 in part 1 contains 4 HLUs

.

GRules generator module: This module allows the author to generate the five packages of generated rules that represent different tutoring plans. Based on the ITS parameters introduced via the previous module, this generation consists of an instantiation of the five packages of the Mrules.

Verification module: As most authoring systems, CAMITS offers a tool to help the author in the diagnosis of errors and bugs. It facilitates detection of incoherencies that can be occurred during the ITS construction. For example, at the end of the construction process of the ITS, it is necessary to check the compatibility of ITS parameters with the effective structure of the teaching material.

Presentation layer: This layer gathers an organized set of interactive objects defining the graphical user interface (buttons, icons, cursor, scrolling bar, task-progression bar, pull-down menus, etc.). Thus, for every object, modeling a part of our application domain, we associate a presentation technique accomplished by a reactive object that reacts to the different authoring actions.

Besides the pull-down menus achieving the different functions, we especially find a toolbox containing graphical icons that refer to the frequently used functions and specialized widget-based palette allowing the graphical construction of the prerequisite-networks.

Dialogue Controllers DC and MDC: Each dialog controller is composed of three independent modules performing respectively, message reception, message control and message sending (Fig. 2).

The Control module allows the coordination and synchronization of the running of the different modules within the three layers, in accordance with the actions of the different authors. At any time, it used all necessary information to determine exactly what are the functionalities to invoke within the layers for which it is responsible.

Every time that an event occurs, the associated receiver delivers the message materializing this event to the control module. The control module reacts then following three steps: analyze the event, draw up an action plan and then carry out the established plan.

Group awareness/notification: The notification and group awareness functions constitute an important point in the cooperative application design[19]. It includes all the interface functions and all systems functions that allow the users to perceive the activities of the other users, as well as to control and to act on the distributed environment.

To develop the appropriate computer support, we recognized the principal factors concerned by the transparency within the workspace. The following Table summarizes the considered elements as well as the questions that the author might asked[6].

In CAMITS, we use a notification mechanism that broadcast to the authors all different events that are

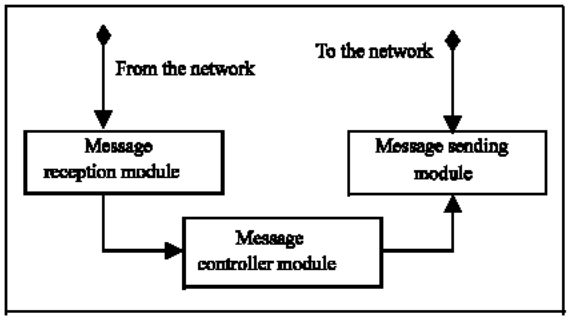

Fig. 2: Dialog controllers architecture

Table 1: Factors and authors questions

Factors	Authors questions
Identification	Who is participating in this activity?
Place	Where is he/she?
Presence	Is he/she present in this session?
Role	What's his/her role?
Modifications	What did he/she modify?
Fragments	On what object is he/she working?

occurred within the distributed workspace. During the cooperation process, the participants will know "who is doing what, where and when?". That's how CAMITS can provide to every participant information concerning his colleagues (role, identity, etc.), interactions state within the workspace (current activities, manipulated objects, realized modifications, etc.) and also the handling states of every object of the ITS (locked, free, authors handling it, locking time, etc.).

The event management process is the responsibility of the server. Since the server holds a reference copy, the notification events, linked to the different ITS fragments, is thus facilitated. Every author can request the server to inform him every time that an event is occurred within the shared space. For example, it can establish an engagement in order to receive the modifications made to a specific fragment of the ITS, which then will be automatically delivered to him.

RESULTS AND DISCUSSION

The ITS pattern is implemented in PHP/MySQL and resides on a server; it can therefore be accessed simultaneously by different distant learners.

The authoring mode software, implemented in JSP (Java Server Pages) and the Java language, is organized as centralized client-server architecture. It makes it possible to several authors to be connected to a working session characterized by a cooperative space and a control strategy.

The cooperation space is represented by a set of structured components (HLU, prerequisite-networks, tutoring parameters and tutoring rules) and tools, which

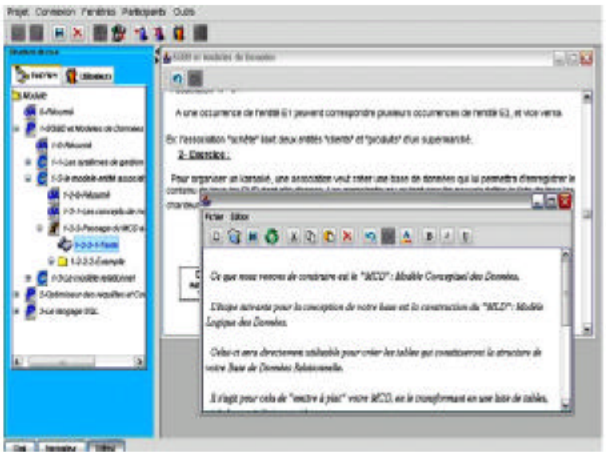

Fig. 3: Developing a data bases tutorial

make it possible the edition and communication tasks. Figure 3 shows an example of a window-space where developing a Data Bases tutorial.The control strategy manages the negotiation of the access right to a component of the ITS and then participation of users during the work session.The authors are oriented to incorporate five learning principles into the authoring process of the tutorial: a clear definition of pedagogical objectives, definition of pre-requisite knowledge, providing a variety of presentation styles (tell, show and do), enhanced feedback and testing, and permitting the learner to choose himself the pedagogical objectives of the learning session.

Two different approaches were used to test the validity that the system actually incorporated pedagogy and effective cooperative design concepts as part of the developmental process.

To evaluate the system a group of four teachers were surveyed to seek their opinion if the authoring system did incorporate the five learning principles[11] into its design. Their survey results validated that the system would prompt developers to build an ITS based on pedagogy. In addition a high agreement was noted in the self-direction of the lesson offered to the learner. In a second means to validate the system, five teachers geographically dispersed were invited to develop a data base tutorial, via local network, and were surveyed to seek their opinion if the authoring system offers all cooperative tools necessary to construct the tutorial in a synchronized manner.

CONCLUSION

The presented study is related to the design of architecture of cooperative authoring system for intelligent tutoring systems called CAMITS. Integrating

cooperation paradigm in ITS authoring systems is the original idea of this study. This system allows geographically distant authors to collaborate to produce a tutoring system according to a predefined ITS pattern. We were interested in the group interaction through accounting of various exchanges operated between the authors during a work session. Especially, we record the aspects related to notification and group awareness.

Although the system does exhibit positive results after a pilot test in the local network context, a question for future research is the experimentation of the system in the internet/web context. This research would provide evidence that the concepts incorporated into the system do impact learning in a positive manner. On the positive side the survey results from the two different experimentations provides indication that the system is a positive benefit to teachers and developers of web-based intelligent tutoring systems.

REFERENCES

1. Wenger, E., 1987. Artificial Intelligence and Tutoring Systems, Addison-Wesley Eds., USA.
2. Murray, T., 1999. Authoring Intelligent Tutoring systems: An analysis of the state of the art , Intl. J. AI in Education, 10: 98-129.
3. Attaran, M and S. Attaran, 2002. Collaborative computing technology: The hot new managing tool, Team Management: An Intl. J., 8: 13-20.
4. Mills, K.L., 2003. Computer-Supported Cooperative Work Challenges, Enclyclopedia of Library and Information Sci. (2nd Edn.), Marcel Dekker, New York, pp: 678-684.
5. Decouchant, D and A.M. Martínez, 2000. A cooperative, deductive and self-Adaptive web Authoring Environment, In Proc. MICAI-2000 Mexican Intl. Conference on Artificial Intelligence, Springer Verlag, Acapulco (Mexico), pp: 443-457.
6. Zidani, A., M. Boufaida and M.D. joudi, 2000. JamEdit un outil interactif et coopératif pour l'édition coopérative de documents, Technique et Sci. Informatiques, 19: 1-23.
7. Pacull, F., A. Sandoz and A. Schiper, 1994. Duplex: A Distributed Collaborative Editing Environment in Large Scale, Proceedings of the ACM Conference CSCW'94, ACM Press.
8. Talhi, S., M. Boufaida and Z. Boufaida, 1996. Moalim: Un système auteur pour l'EIAO , Conférence nationale en informatique, SNITO'96, Tizi Ouzou, Algérie.
9. Harous, S., L. Douidi, M. Djoudi and C. Khentout, 2004. An Authoring System for Distance Education, IADIS Intl. e-Society 2004 Conference, Avila, Spain, ISBN: 972-98947-5-2, pp: 387-394.
10. Murray, T., 1998, Authoring knowledge-based Tutors: Tools for Content, instructional strategy, student model and interface design J. Learning Sci., Vol. 7, n° 1.
11. Janicki, T and O. Jens, Liegle, 2001. Development and evaluation of a framework for creating web-based learning modules: A Pedagogical and Systems Perspective, JALN J. pp: 5
12. Nkambou, R., C. Frasson and G. Gauthier, 2003. CREAM-Tools: An Authoring Environment for Knowledge Engineering in Intelligent Tutoring Systems. In Murray, T., S. Blessing and S. Ainsworth (Eds): Authoring Tools for Advanced Technology Learning Environments: Toward cost-effective adaptative, interactive, and intelligent educational software, Kluwer Publishers, pp: 93-138.
13. Talhi, S., M. Djoudi and A. Zidani , 2001. Un système auteur de tuteurs intelligents: Evolution du mono-usager vers la coopération, Revue Techniques et Sci. Éducatives, 8 n° 1-2, ISSN 1265-1338, pp: 127-138.
14. Greenberg, S and M. Roseman, 1999. Groupware Toolkits for Synchronous Work, In: Beaudouin-Lafon, M. Ed., Computer-Supported Cooperative Work (Trends in Software 7), Chapter 6, John Wiley and Sons Ltd, ISBN 0471 96736 X, 258: 135-168,
15. Talhi, S., M. Djoudi, S. Zidat and M. Batouche, 2005. 0Un système tuteur intelligent hypermedia pour l'apprentissage à distance asynchrone, Congrès intl. en informatique appliquée, CIIA'05, Bordj Bou Arréridj, Algérie.
16. Hameline, D., 1990. Les objectifs pédagogiques en formation initiale et en formation continue", Edition ESF, 8ième édition, Paris.
17. Decouchant, D., A.M. Martínez and E.Martínez, 1999 . Documents for web cooperative authoring, In Proc.CRIWG'99, 5th CYTED-RITOS Intl. Workshop on Groupware, IEEE Computer Society, Cancun (Mexico), pp: 286-295.
18. Orfali, R., D. Harkey and J. Edwards , 1997. Essential Client/Server Survival Guide, John Willeys and Sons Inc., 2nd Edn., New York.
19. Muhammad, A., A.M. Martínez. and D. Decouchant, 2005. Awareness and Coordination for Web Cooperative Authoring. In Proc. of AWIC'2005, The 3rd Intl. Atlantic Web Intelligence Conference, Lecture Notes in Artificial Intelligence, No. 3528, Springer Verlag, Lodz, Poland., pp: 327-333.