MoalimCo : intégration de la dimension coopérative dans un système auteur de Tuteurs Intelligents

Said TALHI* — Mahieddine DJOUDI** — Abdelmadjid ZIDANI*
** Institut d'Informatique, Université de Batna, 05000 Batna, Algérie*
*** Université de Poitiers - UFR Sciences, France*
e-mails : talhi@ univ-batna.dz djoudi@sic.sp2mi.univ-poitiers.fr zidani@univ-batna.dz

Résumé : *Nous proposons une intégration de la dimension coopérative au sein d'un système auteur de STI initialement mono-usager (Moalim). Ceci permettra à plusieurs auteurs géographiquement dispersés de collaborer pour produire un STI. Nous envisageons alors, de leur créer un espace de travail commun et nous définissons des mécanismes permettant de gérer la rétroaction de groupe au sein de cet espace ainsi qu'un support pour assurer l'intégrité des données dispersées à travers les différents sites impliqués.*

Mots clés : Systèmes Tuteurs Intelligents (STI), systèmes auteurs coopératifs, collecticiel, coopération synchrone, modèle client/serveur.

Abstract : *We propose an integration of the cooperative dimension in an ITS authoring system initialy single-user(Moalim). This allow several remote authors to collaborate for a collective ITS production. So, we intend to create a common workspace for these authors and we define mecanisms that allow the awarness group management in this space and a support to guarantee sparse data integrity through the different involved sites.*

Keyswords : *Intelligent Tutoring Systems, cooperative authoring systems, groupware, synchronous cooperation, client/server model.*

1. Introduction

Dans le domaine de l'EIAO (Enseignement Intelligemment Assisté par Ordinateur), nul n'ignore que la réalisation d'un STI est une tâche difficile et pluridisciplinaire où pédagogues, psychologues et informaticiens doivent coopérer pour achever un tel projet. Si ces auteurs sont regroupés dans un même lieu géographique et ont la volonté de coopérer, ceci ne pose pas de problèmes puisqu'ils peuvent se rencontrer et discuter le temps qu'il faut jusqu'à l'opératio-nalisation de leur STI. Par contre, s'ils sont géographiquement dispersés, la collaboration deviendrait une tâche délicate pour ne pas dire impossible. Par conséquent, il est nécessaire de mettre à la disposition des intervenants des supports leurs permettant de communiquer et coordonner leurs activités.

Aujourd'hui, grâce à l'intégration des nouvelles technologies de l'information et de la communication, notamment les réseaux télématiques et les systèmes coopératifs, des rencontres virtuelles à large échelle sont rendues possibles [Coc93] [Rey98]. Le collecticiel permet en effet à des utilisateurs de participer, où qu'ils soient, à la réalisation de tâches communes en exploitant des architectures à base de réseaux et en utilisant des métaphores adaptées aux concepts de partage et de téléprésence [Bux92].

Au cours des deux dernières décennies de nombreux systèmes auteurs ont été développés pour la production de STI [Mur99] [Nka96a] [Nka96b] [Nka96c]. Cependant, ces systèmes ont tous été conçus pour fonctionner en mono-usager. La difficulté de construction d'un STI par un seul auteur comme évoquée précédemment et la répartition géographique des coauteurs dans la plupart du temps nous conduit à penser qu'il serait souhaitable que ces systèmes (spécialement leur *mode auteur*) soient revus pour intégrer l'aspect coopératif à travers un réseau informatique.

C'est dans cette optique que nous présentons le système MoalimCo. Notre objectif est de reprendre un système auteur initialement mono-usager — Moalim — [Tal96a] [Tal96b] afin de

reconsidérer son *mode auteur* dans un contexte coopératif. Nous proposons alors une démarche d'intégration des mécanismes de coopération (espace de travail, rétroaction de groupe, communication, partage des données, etc.) au sein de ce système.

Dans la section suivante nous discutons d'abord des approches de développement de systèmes auteurs coopératifs et présentons le système MoalimCo sous son aspect organisationnel. Nous présentons ensuite la structure de Moalim en insistant évidemment beaucoup plus sur le *mode auteur*. Avant de conclure, nous décrivons enfin le système coopératif MoalimCo en présentant son architecture logicielle ainsi que les différents niveaux qu'elle recouvre.

2. Démarche de conception

Si potentiellement il existe de nombreuses approches possibles pour élaborer un système auteur coopératif, nous pouvons les classer en deux grandes catégories. Une première voie d'investigation consiste à définir une architecture logicielle qui prenne en compte, dès sa conception, les possibilités et besoins des outils de coopération. Cette voie idéale, permet certainement d'édifier une structure souple qui puisse appliquer rigoureusement les concepts de la métaphore de coopération. Mais cette voie est coûteuse. La réalisation de systèmes auteurs mono-usagers pour l'EIAO est déjà suffisamment complexe [Mur99] pour ne pas lui rajouter d'autres préoccupations inhérentes à la coopération.

La deuxième approche, plus pragmatique, et à notre avis plus économique en effort de conception, consiste à enrichir un système auteur préexistant d'un certain nombre de modules visant à lui donner la dimension coopérative. Ce pragmatisme s'appuie sur trois idées raison-nables. La première est l'état de faits: la disponibilité d'un nombre appréciable de systèmes opérationnels. En fait, un effort considérable a été effectué dans ces deux dernières décennies et a donné naissance à de nombreux systèmes auteurs. Murray [Mur99] en a cité plus d'une vingtaine dans sa synthèse la plus récente. La deuxième part du fait que, compte tenu qu'un système auteur est formé d'un mode auteur et d'un mode apprenant, seul le mode auteur peut subir des transformations. L'objectif étant évidemment de faire coopérer les auteurs et non pas les apprenants. La troisième et dernière stipule que de nombreux travaux concernant le travail coopératif dans d'autres domaines telle que l'édition coopérative de documents [Dec94], [Zid96] ont abouti à des résultats satisfaisants. Alors, pourquoi ne pas tirer profit de leurs expériences comme l'a suggéré Grudin [Gru90], [Gru94].

C'est cette deuxième approche que nous avons retenu et dont nous allons en décrire l'ossature, dans les sections suivantes, à travers un modèle de système auteur coopératif que nous avons nommé MoalimCo. Ce dernier est construit par enrichissement du *mode auteur* d'un système auteur préexistant (Moalim) dont nous avons développé un prototype en PROLOG dans nos travaux antérieurs [Tal96a], [Tal96b].

Mais nous devons signaler que la partie logicielle ne constitue pas tout dans les collecticiels. La prise en compte des facteurs humains impliqués par les activités de groupe comme l'a constaté Greenberg [Gre92], constituent également une condition capitale quant au succès des systèmes coopératifs. Ainsi, pour éviter les conflits inhérents à la nature humaine, nous proposons une organisation qui permet de mener la conduite du projet de construction collective d'un STI de manière rationnelle et optimale. Cette organisation facilite également la manipulation des différents constituants du STI pendant toutes les phases du projet et évite ainsi un certain nombre de problèmes tels que les conflits d'accès, la perte de la trace des versions intermédiaires, la non identification des rôles, etc. Nous définissons alors trois rôles à travers lesquels les auteurs peuvent intervenir au cours du processus de construction du STI :

- *Auteur principal :* Un auteur ayant ce rôle est le chef de projet, il coordonne le travail et définit la structure logique du STI à produire en le décomposant en plusieurs éléments (partie, chapitre, paragraphe, figure, image, etc.) puis il distribue les rôles aux différents coauteurs.

Aussi il doit spécifier au système tous les paramètres d'ordre organisationnel qui sont nécessaires à la conduite du projet.

- *Coauteur constructeur :* Un auteur ayant ce rôle n'aura la permission que de modifier la ou les parties auxquelles il sera affecté, sur le reste des parties du STI il n'aura que le rôle de lecteur/commentateur.
- *Coauteur lecteur/commentateur :* Ce rôle ne permet à l'auteur associé que de lire et/ou commenter les parties auxquelles il sera affecté.

3. Le système auteur Moalim

Moalim est un système auteur dédié aux enseignants universitaires pour la création de STI basés sur la *pédagogie par objectifs*. Moalim utilise une hiérarchie à trois niveaux d'objectifs [Ham90],[Tag91]: les *objectifs généraux,* les *objectifs spécifiques* et les *objectifs opérationnel*s. Cette hiérarchie a permis de considérer trois niveaux dans la structuration de la matière à enseigner : les *Parties* (satisfaisant aux *objectifs généraux*), les *Chapitres* (satisfaisant aux *obje-ctifs spécifiques*) et les *Paragraphes* (satisfaisant aux *objectifs opérationnels*). Ces derniers sont évidemment évaluables grâce aux questions théoriques et/ou des exercices qui leur sont attachés. Moalim comporte deux sous-systèmes : *l'éditeur de connaissances* (lié au *mode auteur*) et le *tuteur pédagogique* (lié au *mode apprenant*).

3.1. Le mode apprenant

3.1.1. Modèle du STI créé par Moalim

Moalim s'insère dans le courant des systèmes qui organisent le processus d'enseignement autour d'Unités Pédagogiques (UPs) [Ten94], [Mur99]. Les UPs sont censées recevoir, par instanciation, toutes sortes de connaissances du domaine. La gestion des UPs dans le canevas de STI, est assurée par un système multi-expert basé sur un ensemble de règles de production (dites *règles mères).* Ces règles, complètement paramétrables et formant les différents plans de tutorat, constituent une base de connaissances générique qu'il convient d'instancier pour chaque STI créé par Moalim. L'opération d'instanciation, produisant des *règles filles*, est effectuée automa-tiquement par le système en se basant sur les *paramètres du STI*. Ces *paramètres*, représentés sous forme de prédicats, décrivent l'aspect quantitatif de la matière à enseigner (nombre de parties, nombre de chapitres, nombre de paragraphes, etc.). Finalement, deux niveaux de connaissances sont donc utilisés dans la définition du curriculum :

- Un niveau supérieur correspondant aux *plans de tutorat :* ces plans consistent en cinq paquets de règles qui invoquent les UPs du niveau inférieur, chaque paquet possédant une fonction précise : négociation du point d'entrée dans le cours, détermination des acquis à l'issue d'une négociation, planification des enchaînements, recherche et affichage des UPs, et évaluation de l'apprenant.
- Un niveau inférieur correspondant au *réseau de prérequis :* ce réseau est formé d'une hiérarchie de six sous-niveaux d'UPs. Les quatre premiers sous-niveaux correspondent à des UPs de cours (objectifs pédagogiques, UPs de présentation, UPs d'appropriation et UPs de remédiation) et les deux derniers sous-niveaux correspondent à des UPs d'évaluation (questions théoriques et exercices de résolution de problèmes). Un arc liant deux UPs père-fils représente une relation de prérequis et une transition possible de l'apprenant.

3.1.2. Architecture logicielle

L'architecture logicielle du *mode apprenant* est similaire à celle d'un STI traditionnel [Wen87] ; elle comporte : un *module d'exploration libre*, un *module expert du domaine*, un *module pédagogue*, u*n module de diagnostic de l'apprenant* et une *interface apprenant.*

Comme nous l'avons signalé auparavant, cette architecture reste inchangée dans le système MoalimCo. Tous les auteurs peuvent disposer d'une instance du *tuteur pédagogique* localement, afin qu'ils puissent l'exécuter partiellement et ainsi pouvoir effectuer aisément la mise au point des objets saisis jusqu'à un moment donné.

3.2. Le mode auteur

En *mode auteur*, Moalim est constitué des composants suivants (Figure 1) :

- Un *éditeur d'unités pédagogiques* qui permet la saisie de la matière à enseigner.
- Un *module d'acquisition des paramètres* qui permet la saisie des paramètres du STI.
- Un *générateur de règles filles* qui permet d'instancier les règles mères.
- Un *outil de mise au point* du STI.
- et, une *interface auteur* permettant à l'auteur de dialoguer avec le système.

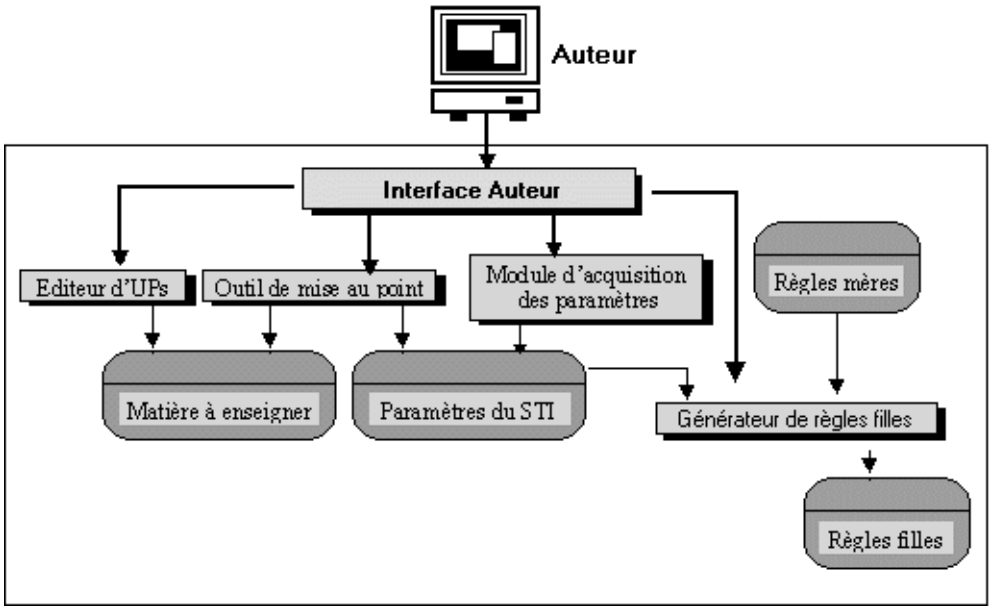

Figure 1. *Architecture du mode auteur*

Pour conclure cette présentation rapide de Moalim nous pouvons dire que du point de vue de l'auteur, construire un STI avec Moalim consiste donc à introduire, via *l'éditeur de con-naissances*, un ensemble d'objets qui seront manipulés par le canevas de STI (*le tuteur pédagogique*). Ces objets sont constitués (Cf. figure 1.) de la matière à enseigner sous formes d'UPs, des paramètres du STI sous forme de prédicats, et de la base de règles instanciée sous forme de règles de production. Quoiqu'il semble que la construction d'un tel STI est maîtrisable par un seul auteur, nous pensons le contraire. En effet, les sciences d'aujourd'hui sont tellement complexes et interdépendantes que cela nécessite souvent la coopération de plusieurs spéci-alistes pour produire un STI dans un domaine donné. Un STI qui enseigne l'intelligence arti-ficielle par exemple doit aborder plusieurs thèmes non vraisemblablement maîtrisés par une seule personne. Il doit traiter de la psychologie cognitive, de la représentation de connaissances, du raisonnement sous toutes ses formes, de l'application de l'IA à différents domaines, etc. Le choix de la matière

à inclure est aussi parfois un casse tête si on tient compte du niveau du public ciblé (expert, novice, etc.). Un système coopératif s'avère donc utile, voire nécessaire dans certaines situations où la matière à enseigner est très complexe.

Dans la section suivante, nous montrons les modalités d'intégration de la dimension coopérative au *mode auteur* de Moalim.

4. Le mode auteur coopératif

Etant donné que notre objectif est de minimiser les changements au niveau du code initial, nous adoptons une démarche [Ata94] qui consiste à s'interposer entre le *noyau* de l'éditeur de connaissances et son *interface auteur* de telle sorte qu'une action réalisée par un auteur soit diffusée vers tous les autres. L'introduction de *contrôleurs de dialogue* [Cou91] entre les différentes couches comme nous le verrons (figure 2.) permettra certainement d'atteindre cet objectif. Les deux composants qui subissent le plus de changements sont *l'éditeur d'UPs* et *l'interface auteur*. Le premier sera découpé en deux morceaux, une partie qui gère le stockage/ récupération et les mises à jour de la copie centrale du STI résidera sur le serveur, l'autre qui regroupe les fonctions d'édition (copier, couper, coller, ...) et de stockage/récupération local résidera sur les sites auteurs. L'interface quant à lui devra fournir, en plus des fonctions actuelles, des mécanismes permettant à l'auteur de contrôler et d'être avisé de l'évolution des contributions apportées au STI par les autres coauteurs. Les trois autres composants ne subiront pratiquement aucun changement et résideront tous sur les sites auteurs.

4.1. Architecture logicielle

L'architecture de l'*éditeur coopératif de connaissances* est une architecture client/serveur centralisée [Orf97] utilisant le mode d'interaction synchrone. Par conséquent, toutes les communications transitent automatiquement par le site central (ou serveur). Sur la figure 2, nous pouvons voir qu'à chaque site client on associe deux processus : un processus de présentation (PRP) chargé de gérer l'interaction d'un auteur avec l'interface et un processus local (PRL) qui accomplit toutes les tâches traitables localement (les tâches d'édition par exemple). Quant au niveau central, nous définissons un processus central (PRC) qui gère toutes les communications entre les différents PRLs et tient à jour le contenu de la copie centrale et la structure logique du STI. Le PRC détient à son niveau les versions les plus récentes des objets du STI ainsi que sa structure logique, tandis que les stations des différents auteurs peuvent contenir des versions qui ne sont pas forcément les plus récentes. Par conséquent, il appartiendra à ces auteurs de les récu-pérer du site central.

L'intérêt de cette approche en terme d'architecture est d'éviter d'alourdir le fonctionnement du serveur central en lui incombant les tâches liées à l'édition de texte et de graphiques qui peuvent être traitées localement. Ceci a pour conséquence d'éviter aux auteurs des attentes inu-tiles. Ainsi, un PRL peut initier une communication avec le PRC, tout en continuant à exécuter des tâches locales et l'auteur associé mènera par conséquent une attente active.

L'idée de la conception d'un système collecticiel en couches comme le montre la figure 2 s'est avéré intéressante à plus d'un titre. En effet, puisque le principe de base de toute concep-tion est la réutilisation logicielle qui repose sur le concept de modularité [Cou91], ceci suggère dans notre cas que chacune des trois couches de la figure 2 ne doit avoir aucune connaissance des objets des deux autres. Mais l'absence de connaissances ne signifie pas absence de communication. La nécessité d'assurer les échanges d'informations au *premier* et *deuxième niveau*, se traduit par la présence de *" contrôleurs de dialogue "*. Nous interposons donc au *niveau-1* de la figure 2, entre chaque PRP et chaque PRL un contrôleur de dialogue (CD), et au *niveau-2* entre le serveur et les différents PRLs le contrôleur principal de dialogue (CPD).

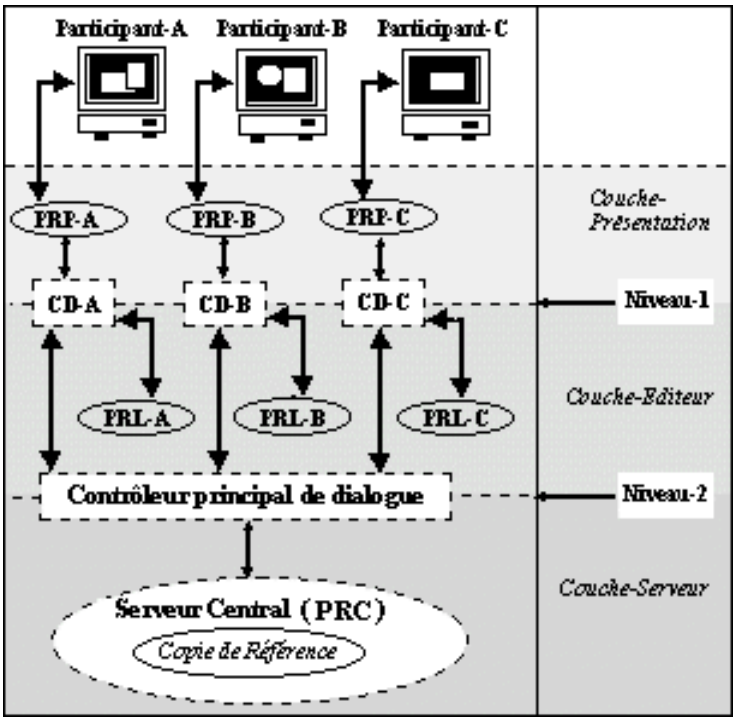

Figure 2. *Architecture en couches de l'éditeur coopératif de connaissances.*

4.2. Description des couches

Couche Serveur **:** Le processus central PRC regroupe plusieurs types de traitements, parmi lesquels nous distinguons ceux qui sont liés à la gestion de la structure logique du STI, ainsi que les contenus des composants du STI. Il permet ainsi aux auteurs de stocker et de récupérer les objets du STI dont la structure logique est déclarée aussi bien au niveau central qu'au niveau local. Le PRC est aussi responsable des opérations de contrôle des droits d'accès, du traitement des événements [Wii91] et de la notification de leurs conséquences aux auteurs. Dans le cas de notification des événements, par exemple, le module qui en est chargé gère un ensemble de files d'attentes telles que la file des engagements, la file de blocage, etc. Chaque fois qu'il est invo-qué suite à un événement, ce processus identifie les auteurs destinataires et procède à la structuration des notifications sous forme de messages transmissibles. Ces messages seront alors mis à la disposition d'un autre module émetteur qui se chargera de l'émission.

Couche Editeur **:** Le processus local PRL regroupe à son tour plusieurs types de traitements permettant à chaque auteur de manipuler les objets composant le STI. Ces traitements incluent aussi bien le support des actions individuelles, que l'aspect de partage et de gestion de la transparence. Par exemple, l'accès à un fichier dans un système mono-usager délivre directement son contenu. Par contre dans notre cas, ce processus déclenchera une suite de traitements tels que la vérification des droits d'accès de l'auteur, de l'état de blocage du composant et enfin l'avertissement des auteurs travaillant sur ce même composant. Au niveau de chaque site, le PRL associé permet à l'auteur d'enregistrer localement les contenus des composants qui lui sont accessibles. Il sollicitera régulièrement le serveur pour maintenir à jour les versions de ces composants.

Couche Présentation **:** Le processus de présentation PRP regroupe un ensemble organisé d'objets interactifs définissant la partie perceptible du système (boutons, curseur, barre de défilement,

thermomètres de progression de tâches, etc.). Ainsi à tout objet modélisant une partie du domaine de notre application, nous associons une technique de présentation matérialisée par un objet interactif qui réagit aux actions de l'auteur. Le PRP supporte un ensemble de traitements parmi lesquelles, nous distinguons essentiellement les traitements d'analyse et les traitements d'affichage qui sont chargés de traquer les gestes de l'auteur et réagir en conséquence, ainsi que d'afficher la représentation approprié à la situation courante. L'objectif de la couche présentation est de rendre le système facile à manipuler en présentant à l'auteur une vue explicite qui ne comporte pas d'ambiguïtés [Mar93].

Les contrôleurs de dialogue CD et CPD : Ces contrôleurs situés respectivement aux *niveau-1* et *niveau-2* (figure 2) se composent chacun de trois modules indépendants réalisant respectivement les tâches de *contrôle*, d'*émission* et de *réception* de messages. Le module de *contrôle* regroupe toutes les fonctions de contrôle qui permettent de coordonner et synchroniser l'exécution des différents modules au sein des trois couches, conformément aux actions des différents auteurs. Il dispose à tout instant de toutes les informations nécessaires pour déterminer exactement quels sont les objets à invoquer au sein des couches dont il est responsable. A chaque fois qu'un évé-nement se manifeste, le message matérialisant cet événement est délivré au module de *contrôle* par le *récepteur* associé. Le module de *contrôle* réagit alors en suivant trois étapes : analyser l'événement, dresser un plan d'actions puis exécuter le plan établi.

5. Conclusion

Le travail présenté dans ce papier porte sur l'intégration de la dimension coopérative dans un système auteur de STI initialement mono-usager. Cela nous a mené à reconcevoir le *mode auteur* afin qu'il puisse permettre à des auteurs géographiquement distants de coopérer pour produire un STI de la même norme que le système initial. Le nouveau système obtenu, *MoalimCo*, permet à plusieurs auteurs de se connecter à une session de travail caractérisée par un espace de coopération et une politique de contrôle. L'espace de coopération, utilisant un mode d'interaction synchrone, est représenté par un ensemble de composants structurés (UPs, paramètres, règles) et des outils d'édition. La politique de contrôle gère la participation des utilisateurs à la session de travail et la négociation du droit d'intervention sur un composant du STI. Les informations échangées entre les sites sont soit des opérations de contrôle soit des flots de données correspondant aux composants du STI.

Ce système repose sur une architecture logicielle centralisée basée sur le concept *client-serveur*. Pour mieux appréhender le fonctionnement de notre système, nous comptons expérimenter le prototype actuel auprès des utilisateurs potentiels. Ceci permettra sans doute de recenser les insuffisances afin d'y remédier. Ainsi, nous pourrons également mettre à l'épreuve l'efficacité du serveur à répondre aux différentes requêtes des auteurs (récupération et stockage des informations, gestion de la structure logique et du contenu du STI, ...) et à un niveau plus haut les performances des contrôleurs de dialogue.

Un autre aspect tout aussi important de notre travail consiste à accroître la tolérance aux pannes transitoires du système et assurer la sécurité. La solution que nous avons proposé consiste à remettre le contrôle aux *contrôleurs de dialogue* en cas de panne du serveur. Ils se chargeront alors de maintenir l'application dans un état cohérent au niveau local, jusqu'à la reprise en service du serveur. Cette solution bien qu'utile demeure insuffisante ; par conséquent, nous projetons à l'avenir, d'étendre cette architecture vers une architecture répliquée.

Bibliographie.

[Ata94]. S.B.Atallah, R.Kanawaki., " architecture de systèmes pour les collecticiels synchrones ", actes d'IHM'94.

[Bux92]. W. Buxton., " Telepresence : Integrating Shared Task and Person Spaces " , *Proceedings of the Graphics Interfaces Conference*, pp. 123-129, 1992.

[Coc93]. A.Cockburn, S.Greenberg., " Making contact Getting the group communicating with groupware ", *Proceedings of the ACM Conference on Organizational Computing Systems*, 1993.

[Cou91]. J.Coutaz., " Interfaces homme-machines, un regard critique ", TSI, Vol10 n°1, 1991.

[Dec94]. D. Decouchant., " Rétroaction de groupe et édition coopérative de documents structurés. Actes d'IHM'94.

[Gre92]. S.Greenberg, K.Roseman, D.Webster., " Human and technical factors of distributed group drawing tools, interacting with computers ", Vol 4, n° 3, pp. 364-392, December 1992.

[Gru90]. J.Grudin., " Groupware and cooperative work **:** problems and prospects ", Communication. ACM, Vol 37, n°1, pp. 93-105, Jannuary 1990**.**

[Gru94]. J. Grudin., " Groupware and social dynamics : eight challenges for developers ", *Communication of ACM*, Vol. 37, n° 1, pp. 92-105, January 1994.

[Ham90]. D.Hameline., " Les objectifs pédagogiques en formation initiale et en formation continue ", ESF Editeur, 8ième édition, Paris, 1990.

[Mar93]. A.Marcus., **"** Human communications issues in advanced UIs ", Communications ACM,Vol 36, n° 4, pp. 101-109, April 1993.

[Mur99]. T.Murray., " Authoring Intelligent Tutoring Systems : an analysis of the state of the art ", Intenational Journal of Artificial Intelligence in Education, Vol. 10, pp. 98-129, 1999.

[Nka96a]. R.Nkambou, G.Gauthier, C.Frasson., " An authoring environment for curriculum and courses building in an ITS ", 3th international conference CAI in Sciences and engineering, Springer Verlag, Berlin, 1996.

[Nka96b]. R.Nkambou, B.Lefebvre, G.Gauthier., " A curriculum based student model for ITS ", 5th international conference on user modeling, Kailua-Kona, 1996.

[Nka96c]. R.Nkambou, M.C.Frasson , C.Frasson., " Generating courses in an ITS ", Procedings ofIEA-AIE'96, 1996.

[Orf96]. R.Orfali, D.Harkey, J.Edwards., " Essential Client/Server Survival Guide ", 2nd edition , John Willeys &Sons, Inc, New York, 1996.

[Rey98]. G.Reynard, S.Benford, C.Greenhalgh, C.Heath., " Awareness driven video quality of service in collaborative virtual environments ", *Proceedings of the ACM conference CHI'98*, 1998.

[Tag91]. C.Tagliante., " L'évaluation " , Edition Clé international , France, 1991.

[Tal96a]. S. Talhi., " MOALIM: un système auteur de l'EIAO ", Proceedings du 18ième Symposium DECUS FRANCE, Paris, 2-4 avril, France, 1996.

[Tal96b]. S.Talhi., M.Boufaida., Z.Boufaida., " MOALIM: vers un système auteur pour l'EIAO ", Proceedings (SNITO'96), pp.29-51, Université de Tizi Ouzou, 11-13 novembre, Algerie, 1996.

[Ten94]. A. Tenachi., "Modélisation de connaissances pédagogiques" , Thèse de Doctorat de l'Université Henri Poincarré de Nancy, Nancy, 1994.

[Wen87]. E. Wenger., "*Artificial Intelligence and Tutoring Systems"*, Addison-Wesley Eds, USA, 1987.

[Wii91]. U.K. Wiil., " Using Events as Support for Data Sharing in Collaborative Work " , *Proceedings of the International Workshop on CSCW*, Berlin 1991.

[Zid96]. A. Zidani., " Conception d'un collecticiel pour l'édition partagée de documents ", Thèse de Magister, Université de Batna, Institut d'informatique, Algérie, Novembre 1996.