WEB BASED COLLABORATIVE DOCUMENT CREATION AND REVIEW SYSTEM

Marius Ioan Podean, Raluca Arba
Faculty of Economics and Business Administration
Babes Bolyai University of Cluj Napoca, Teodor Mihali 58-60, Cluj Napoca, Romania

Loredana Muresan
Faculty of Economics and Business Administration
Babes Bolyai University of Cluj Napoca, Teodor Mihali 58-60, Cluj Napoca, Romania

Keywords: Collaborative document editing, document management, workflow, virtual teams, XML.

Abstract: An important aspect in distributed teams and organization is their ability to manage documents. A collaborative document editing system that integrates functionalities from document management systems, workflow, collaborative editing with support for virtual teams can increase team efficiency and allow users to concentrate their efforts on content development. This paper reports a case study implementing this approach in collaboratively creating scientific papers. The use of XML when treating documents proves to be the appropriate solution to develop user and document centered systems.

1 INTRODUCTION

Creating scientific papers is most of the time a very complex and elaborate task. This type of documents usually is written by more then just one person, and often the authors are located in different parts of the world. This collaborative process is sometimes slowed down by the fact that dealing with the technology used to support it is very much time consuming. Taking in consideration the authors need for location and time independence, the usage of different operating systems and applications, at this moment technology offers solutions that satisfy just partially the aforementioned requirements.

Document management systems (DMS) focus on tracking and storing documents created and exchanged by there authors (Aversano et al. 2001). They provide components for defining metadata for the documents (i.e. date of creation, authors, version etc.), indexing (usually based on metadata), storage and retrieval (based on the unique document identifier).

Workflow management systems (WMS) allow the automation of processes within an organization, enabling greater coordination and control among geographically distributed teams (Nallaparaju et al. 2005). Using WMS, the organization can integrate different software technologies, leading to the improvement of the collaborative activities (Aversano et al. 2001). This technology leaves authors with the consumption of a great deal of time.

Collaborative editing allows multiple persons to edit simultaneously the same document, see who is working on the document and watch in real time the changes that they have made (Raikundalia and Zhang 2004). In order to have sufficient knowledge about the changes that others perform upon the document, group awareness mechanisms have been created, such as: telepointers (multiple cursors of users appear within the document), radar views, multi-user scrollbars and, as shown by Raikundalia and Zhang (2004), structure-based multi-page view, point jumping mechanism and user info list. The main downside of this solution is that the implementations available are platform-specific and can generate conflicts between members when someone changes often content created by others. The collaboratively edited file is stored on the document owner's computer, leading towards versioning problems when participating members make local copies of the document.

Ioan Podean M., Arba R. and Muresan L. (2008).
WEB BASED COLLABORATIVE DOCUMENT CREATION AND REVIEW SYSTEM.
In *Proceedings of the International Conference on e-Business*, pages 437-442
DOI: 10.5220/0001907404370442

Wikis represent “a piece of software that allows users to add, modify, and/or delete information from a knowledge base via the web” according to Spek (2008). As the author of this definition emphasis as a main characteristic, wikis are anarchistic systems, many implementations allowing anonymous users to modify the content. On wiki systems “conflicts can quickly result in ‘edit wars’ when multiple users keep on reverting each others changes because they don’t agree” as shown by Spek (2008).

Creating research papers requires a system that is simple to use and allows users to focus their efforts on the content rather that on the technology used to create it. The system has to support the space independence of the authors and integrate their collaborative effort in a common workplace in order to obtain greater team efficiency (Guerrero et al. 2004). Collaborative work involves information exchange in order to support negotiation and communication between group members and different mechanisms through which the team can regulate and manage itself in order to be goal directed (Millward and Kyriakidou 2004). A more supportive system would have document management facilities and support for task automation. To achieve greater efficiency the system will have to be user centered and non–restrictive regarding de operating system.

In this paper, we start by presenting current technologies used for document management and editing highlighting their main characteristics and downsides. We will continue in Section 2 with the theoretical approach of our system and then discuses in Section 3 the implementation and the technical details concerning it. Based on the model we’ve proposed, in Section 4 we shall present some conclusions and further work.

2 THEORETICAL APPROACHES

In this paper we shall present our implementation of such a system that aims to cover the aforementioned requirements. **Dante** is a web based system designed to be a good support for virtual teams in elaborating scientific papers. It offers document management facilities and process automation for repetitive tasks. Since all the data the system uses is stored in XML files, **Dante** can be document and user centered, allowing authors to easily edit, review and export in different formats their work. Teams are building around the document allowing them to be goal directed; all members of the team having the same rights. Each author is responsible of editing different chapters of the document, chapters on which the others could only place comments, content changing not being allowed, avoiding conflicts and stepping on each others and allowing individuals to reconcile with the teams goals. The application facilitates communication through synchronous and asynchronous channels. Documents being stored in XML files and using a web based user interface makes the system work on different platform and allows users to export documents in different open formats, representing a combination of best practices specific for the previous discussed systems. At this moment the application offers no version control capabilities, and therefore authors could not revert documents to older versions.

As mentioned earlier, **Dante** is both a user and document centric system, supporting collaboration in virtual teams and efficient document management. As described by Millward and Kyriakidou (2004), it is important for virtual teams to be a “singular concrete entity” with the following characteristics: stability, regular interaction, symbiosis and member proximity. Following this requirements, in **Dante** teams are organized around the document that they are creating. Member proximity results from the fact that each member can view the most recent version of the chapters that the others have created and that everybody knows who is responsible for a particular chapter. Each author can review others work and make suggestions related to each piece of text using the commenting tools. The symbiosis of the team is supported by the fact that each person’s responsibilities are clearly drawn and all members have the same rights, all depending upon others in improving their work. Interaction between members is supported by both synchronous and asynchronous mechanisms.

In collaborative real time editing systems several users can edit a file using different computers. An important aspect for this type of systems is group awareness (GA). GA provides users information about the status of a document and changes made by others. As shown in Raikundalia and Zhang (2004), several GA techniques have been identified: telepointers (multiple cursors are shown within the document), radar views, multi-user scrollbars and structured-based multi-page view, point jumping mechanism and user info list. In **Dante**, the GA problem is solved by using a structured-based multi-page view panel for displaying a project. One of the main downsides of all collaborative editing systems (CES) is that the document is saved on the document owner’s computer, all others participants being allowed to save a copy of the document leading in time to lost edits and versioning problems. **Dante** being a web based system it has the advantage of delivering file storage facilities. All changes made by the authors are saved on the server which will offer only the most recent version of the document. Authors edit different parts of the document and send changes to the server via AJAX.

As shown in Leone, Hodel and Gall (2005), combining CES and DMS can result in greater performance improvements. **Dante** takes a similar approach providing solutions and process automation for creating, storing retrieving, editing and exporting documents collaborative, anytime on the most used platforms. All documents have metadata to easily manage them and provide extra search capabilities. Another important aspect is the presence on the internet and the collaboration with persons from outside the team using e-prints. Lawrence (2001) shows that articles freely available online are more highly cited and recommends, in order to achieve greater impact and faster scientific progress, that authors should aim to make research easy to access. Content can easily be transformed into HTML and allow others to post comments on the article if the team wishes so.

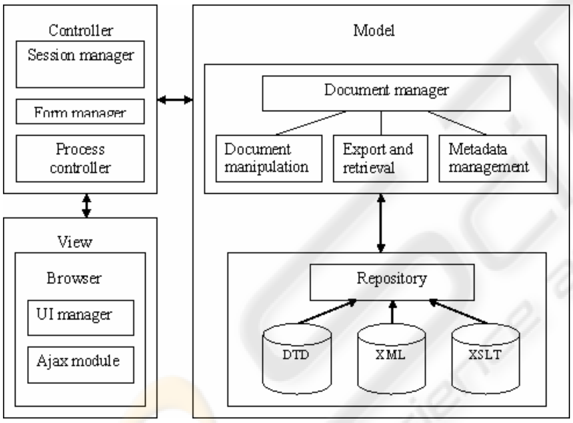

Figure 1: Dante architecture.

To summarize, the main characteristics that we consider essential for an efficient collaborative document creation and review system are:

support for virtual team efficiency

user and document centered

support users location and time independency

platform independent

efficient group interaction

allow users to concentrate on the content and waste as little as possible time with the technology that supports the process

document management facilities

automation for different parts of the process (i.e. formatting the document according to different templates)

As shown in Figure 1, **Dante** is a web based system design upon de model-view-controller design pattern which stores all data in XML files. This approach allows a great flexibility in handling document and user interaction. The presentation layer sets the appropriate tools according to user level and rights and manages the client-side communication with the controller layer using AJAX.

The presentation layer defines de following main tools: *Editor*, *Partners* and *Chat*. The *Editor* allows users to manage old projects, start new ones, collaboratively edit current projects and export documents in different open formats and format the content using predefined templates. When starting a new project, the project owner must define a set of properties, like name (works like an identifier for the project), type (according to the type that has been selected, a certain template will be used to format the content when users export the document) and a list of members that will participate to the project (after defining the project, all participants will have the same rights) (Leone, Hodel and Gall 2005). During development extra sets of metadata will be added to the project allowing users to consult the state of the project, last modified date etc. Each author can edit one or more chapters from the document and place comments on those chapters edited by the others. Chapters are presented as elements in a tree menu and can bee accessed in different windows. When accessing a particular chapter, the system checks the metadata associated to it and determines if the current user is the author of the chapter and displays a new menu that allows him to edit the content if so, otherwise allowing him only to add new comments to the component elements or edit comments define earlier. The editor is not made up from elements that will allow authors to format the content (i.e. defining font types, paragraph alignment etc.), but from elements that represent structural components of document (i.e. paragraph, note, quote, table etc.). The formatting of document will be done automatically by the application according to the template that the document owner had defined. Each structural component of the document can be commented by the rest of the team, the editor defining special zones at the end of each element where these comments can be consulted. When accessing chapters for whom the current users is not the owner, a different editor is loaded allowing the user to define and edit comments and reply to those that others have defined. Before finalizing the project, the chapters can be exported in different open formats (i.e. PDF) or to a link in order to be accessed by those that are not participating to the project.

The *Chat* sections allows user to communicate using synchronous channels. The section defines two main channels: *Groups* and *Personal*. The *Groups* channel lists all active projects for the current user and allows him to communicate with the members of each project in different panels. When a particular group is selected, the user can communicate with the members of that project that are online, the messages being available for all co-workers. All messages are stored in XML files and associated to the project, allowing user to consult the discussion archive or the project anytime. On the other hand, the *Personal* channel allows user to communicate with each friend defined in the *Agenda* using private channels. These messages are not stored in the message archive, only if the users decide so.

The *Partners* section has two main subcomponents: *Agenda* and *Invitations*. The *Agenda* allows users to manage their partners' contacts and export them to micro-formats like HCard. The user's personal data can be exported in VCard format. The *Invitations* subcomponent manages the user's invitations to participate at different projects (only after accepting an invitation a user can actively participate to a project). All personal data is stored in XML files, allowing the system to easily integrate them into the projects content when exporting a project in a final state according to different templates.

In the next section we shall discuss the details regarding the implementation of our web based collaborative document creation and review system.

3 IMPLEMENTATION

As previously mentioned, **Dante** is a web based system design upon de model-view-controller design-pattern which stores all data in XML files. The *View* layer personalizes the user interface according to user's rights and permissions and displays the appropriate editor after reading the documents metadata. It also implements a communication module which transfers data to the *Controller* layer using AJAX. The *Controller* layer handles the events triggered from the UI and calls the appropriate handler from the *Model* layer. We will concentrate our attention on the *Model* layer witch manages all XML documents.

The *Model* layer consists mainly from two subcomponents: the *Document manager* and the *Repository*. The *Repository* is a collection of DTD, XML and XSLT files used by the *Document manager* module.

The *Model* layer consists mainly from two subcomponents: the *Document manager* and the *Repository*. The *Repository* is a collection of DTD, XML and XSLT files used by the *Document manager* module.

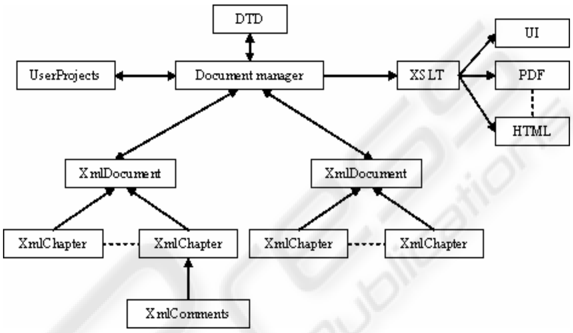

Figure 2: Document workflow.

The document type definition files describe the structure of the documents and all metadata that can be added to them. In this case, to describe the structure of the scientific papers we've created a document type called xmlDocument. The structure of this type is in a way similar to the more mature *DocBook* xml vocabulary (Walsh 2005), being more simplified and particularized for scientific papers. The xmlDocument DTD defines the structure of the whole document viewed as a project. The two main elements of an xmlDocument are *docInfo*, which describes the metadata associate with the document (i.e. document name, authors, document type, last update, version etc.) and *docBody*. The *docBody* defines the content of the document (i.e. abstract, references, appendixes etc.) and for to each chapter the metadata associated (id, title and author) and a link to the XML file that defines its content. This separation of each chapter in different XML files allows a more flexible management of content and metadata. The vocabulary that describes the chapter's elements is defined by the *xmlChapter* document type definition, implementing means of storing and identifying elements. As mentioned earlier, each author can define comments for the chapters that he is not editing. These comments are stored in separate XML files, and the vocabulary describing them defines elements to uniquely identify de comment, describe its characteristics (i.e. author, date etc.) and link it to the element to which it referees.

In order for the *Document manager* to easy set the corresponding roles and rights, a document type definition is implemented to store the projects. This DTD allows defining for each user the projects for which he is a collaborator or the owner. The *Document manager* module consist in several classes based on the DOM implementation which automates the processes concerning the creation, storage, retrieval and the export of the documents. All server side scripts have been implemented in PHP which includes a good support for XML, complying with the commonly used standards (SAX, DOM, SimpleXML, XMLReader, XMLWriter, and the XSLT). Some of the classes defined in the *document manager* come as a wrapper for the DTD residing in the repository: the *xmlChapter* class, for example, handles the creation and the update of different document objects, prepares the content for printing accessing the appropriate class when exporting (i.e. the *pdfPrinter* class when exporting to PDF). This module generates the appropriate user interface according to the user's role and rights, loading the corresponding XSL and parsing the requested XML file. For the same DTD the *Repository* defines different XSL transformation files for each role and user-level, restricting the access to different actions that the user can perform upon the document. Most of the UI used by **Dante** is generated using XSL transformations.

The *Chat* module works in a similar way, defining a DTD for storage purposes and different XSL transformation schemes for display. As mentioned previously, the user can communicate using private channels or rooms dedicated to different projects. The messages exchanged between users in these project related rooms are bound to the project using metadata (although physically residing in different locations) allowing project based message archiving. This approach allows members to catch-up with the team when not being able to join the group. The *Agenda* module stores member's personal data (this being done also using XML files) and allows users to export this information in HCard and VCard microformats. All templates require a minimum of identification data for the authors, the project importing all required data when formatting from this module. We have tried to structure the implementation as much as possible according to functionality of the whole system, defining a modular structure.

The user interaction is managed using JavaScript. Because the communication from the server to the client is mostly done using XML chunks (or an entire file), the classes defined on the client-side are in some manner a transposing of those residing on the server being also a wrapper upon the document. The *Session Manager* module from the *Controller* layer determines which script to be loaded according to conjuncture determining the operations that a user can execute on a particular document. When editing a particular chapter, all new elements and all changes are stored in different queues and only when the users decides to save the document the content of these queues is send using AJAX to the server. These classes are also based on the DOM model, the content being sent to the server representing a DOM node to be inserted in the document residing on the server. The queues are gradually discharged when a component element is successfully saved on the server. This approach allows defining specific UI elements and behavior to each document type being treated. The user screen is blocked while saving the content of the queues, ousting the chances to make unsafe changes to the content. When appropriate, an XSLT processor is used on the client-side to reduce the charge on the server.

We have chosen this modular approach based on document types and user interfaces particularized on roles/rights and document types in order to facilitate further development of the system in such a manner that it could handle a lot more document types.

4 CONCLUSIONS

The work reported in this paper has addressed the problem of integrating document management functionalities and workflow capabilities into single system with support for virtual teams in order to achieve a very efficient solution for collaborative document editing. Allowing users to edit collaboratively documents and supporting their needs for time and location independence can result in more team efficiency. Users have to be able to concentrate their efforts on content and reduce as much as possible the time used to handle and integrate technologies. An approach that integrates functionalities from document management systems, workflow management and collaborative editing proves to be a much more user centered and supportive collaborative solution. As such an implementation, **Dante** tries to offer sufficient support in order for the teams to be goal directed and efficient. Using a web based solution does not impose restrictions on the users regarding operating systems or software. Taking in consideration the multitude of solutions available, we have to consider using as much as possible open formats in order for the content to be easily integrated. As shown in this paper, using XML as building blocks allows the system to be document and user centered. We have presented a case study of a particular field where such an implementation would be a great support. As further work, we intend to develop our system toward a framework that can handle a wide range of documents, taking in consideration the increased need for collaborative document editing in a multitude of working fields.

ACKNOWLEDGEMENTS

The work presented has been funded by the research grant “Intelligent System for Business Decisions Support”, Director Nitchi Stefan, PhD., Professor, PNII Program, 91-049/18.09.2007 supported by Higher Education Ministry.

REFERENCES

Aversano, L, Canfora, G, Lucia, A & Gallucci, P 2001, ‘Integrating Document and Workflow Management Systems’, *Proceedings of the IEEE Symposia on Human-Centric Computing Languages and Environments*, http://csdl2.computer.org/comp/proceedings/hcc/2001/0474/00/04740328.pdf

Guerrero, L & Collazos, C & Pino, J & Ochoa, S & Aguilera, F 2004’Designing Virtual Environments to Support Collaborative Work in Real Spaces’, in *Journal of Web Engineering*, vol.2, no.4, pp. 282-294

Hampel, T, Keil-Slawik, R 2000, ‘A Collaborative Document Management Environment For Teaching And Learning’, in *Proceedings of the Third International Conference on Collaborative Virtual Environments.* San Francisco (Ca.), USA, pp.197-198.

Lawrence, S 2001, ‘Online or invisible?’, Nature, vol. 411, no. 6837, p.521, http://www.idemployee.id.tue.nl/g.w.m.rauterberg/publications/CITESEER2001online-nature.pdf

Leone, S & Hodel, T & Gall, H 2005, ‘Concept and Architecture of a Pervasive Document Editing and Managing System’, in *Proceedings of the 23rd annual International Conference on Design of Communication (SIGDOC)*, Coventry, United Kingdom, ACM Press, pp. 41-47

Millward, L & Kyriakidou, O 2004, ‘Effective virtual teamwork: a social-cognitive and motivational model’, in *Virtual and Collaborative Teams: Process, Technologies and Practices*, ed. Idea Group Inc., United Kingdom

Nallaparaju, V & Raikundalia, G & Brand, C & Bain, C & Hutchinson, A 2005, ‘An Experimental Study of Workflow and Collaborative Document Authoring in Medical Research, in *Proceedings of Tenth Australasian Document Computing Symposium*, Sydney, Australia, http://eprints.vu.edu.au/archive/00000648/

Raikundalia, G & Zhang, H 2004, ‘Novel Group Awareness Mechanisms for Real-time Collaborative Document Authoring’, *The Ninth Australasian Document Computing Symposium (ADCS 2004)*, University of Melbourne, Australia, http://www.cs.mu.oz.au/~alistair/adcs2004/papers/paper05.pdf

Spek, S 2008, ‘Wikis are good for knowledge management’, submitted 6 February, 2008, arXiv:0802.0745v1

Walsh, N 2005, DocBook 5.0: The Definitive Guide, O'Reilly & Associates