KNOWLEDGE MANAGEMENT AND INFORMATION SYSTEMS BASED ON WORKFLOW TECHNOLOGY

Iván Martínez Toro, Daniel Gallego Vico and Joaquín Salvachúa Rodríguez
Departamento de Ingeniería de Sistemas Telemáticos, Universidad Politécnica de Madrid
Avda. Complutense 30, Ciudad Universitaria, 28040 Madrid, Spain
{imartinez, dgallego, jsalvachua }@dit.upm.es

ABSTRACT

Knowledge management is critical for the success of virtual communities, especially in the case of distributed working groups. A representative example of this scenario is the distributed software development, where it is necessary an optimal coordination to avoid common problems such as duplicated work. In this paper the feasibility of using the workflow technology as a knowledge management system is discussed, and a practical use case is presented. This use case is an information system that has been deployed within a banking environment. It combines common workflow technology with a new conception of the interaction among participants through the extension of existing definition languages.

KEYWORDS

Knowledge management, awareness, groupware, workflow technology, distributed software development.

1. INTRODUCTION

Within the information management area, one of the most common examples of virtual community is the one generated to support Distributed Software Development (DSD). DSD is nowadays a common practice adopted by companies to increase their competitiveness and minimize costs. During a software development project, most of the time different stakeholders (e.g. clients, users, developers, project managers, experts, and so on) are not physically co-located. In this new DSD scenario, the usual development team which used to meet periodically to discuss, take decisions and solve problems, is substituted by a virtual team composed of smaller sub-teams located in different places that employ communication technologies to interact with each other. Consequently, the use of complex information systems allowing telework is critical.

There are some important benefits of using DSD. The main one is related to costs savings achieved by off shoring and outsourcing, just like at any other industry context. Another advantage is that time difference can lend to intense developing activity as some sub-teams can work while others are resting; employing a sub-team of developers near to the client can be an important benefit too, avoiding periodical journeys. Differences in experience and technical knowledge of distributed stakeholders usually mean and extra value for the project.

However, DSD has some difficulties imposed by distance. It is in need of permanent cooperative work. Problems related to distributed work groups have been deeply investigated by Damian and Zowghi (2002). Due to that survey, there are four main problem categories: inappropriate communication, cultural diversity, knowledge management and time difference.

Today, a growing list of groupware tools aiming to solve those problems can be found. These tools have been classified by their functionality or their synchronization needs by Ellis et al. (1991). By functionality, there are message systems, multiuser editors, group decision support, conference systems, intelligent agents and coordination systems. The other classification divides groupware into synchronous and asynchronous tools. The fact is that DSD is still supported by common non-optimized tools such as mails lists, forums, chats and artifacts repositories.

On the other hand, workflow technology is growing and being adopted by an increasing number of companies and workgroups due to its benefits related to process optimization. Up until now workflow technology has been poorly used as a knowledge management tool because of the static processes it is able to manage, different from the highly dynamic needs of information intelligent systems. Once improved this technology to make it dynamic, easy to adopt and useful as knowledge manager, it can be an important groupware tool to take into account.

In this article, the use of an improved workflow technology system to support knowledge management and awareness is discussed, and a practical use case is presented. This use case implements an information system architecture that can be classified as an asynchronous coordination system following the ideas of computer supported cooperative work (CSCW) tools (Grudin, 1994). It helps maintaining global awareness within the DSD group. This tool, called ItecDesk, has been deployed within a Spanish banking environment and is being utilized for banking distributed software development.

2. KNOWLEDGE MANAGEMENT IN VIRTUAL COMMUNITIES

In every virtual community and, as a consequence, in every distributed working environment, knowledge management and awareness are essential to accomplish different tasks in a satisfactory manner. Each participant must be aware of every action done by the rest of the participants within the group to achieve a coordinated and effective work done.

Europarc et al. (1992) stated that "*awareness is an understanding of the activities of others, which provides a context for your own activity*". A critical part of the awareness maintenance is the knowledge management, which allows this context to be shared among all participants in the community. This context is useful to enable coordination, work distribution, decision making, anticipation and help finding (Gutwin and Greenberg, 2002). In the concrete case of software development, with its inner complexity and interdependency; it is advisable to take actions to assure knowledge maintenance through the use of correct practices and specific tools.

To allow different members of a distributed team to work in an efficient way it is important to minimize communication and knowledge management problems. It is also necessary to employ the appropriate technologies for each participant of the team. When they feel comfortable with the methodology and technology in use, it is more probable that the information will be exchanged fluidly and a better understanding will be achieved.

3. WORKFLOW TECHNOLOGY AND KNOWLEDGE MANAGEMENT

Workflow technology studies operational aspects of a working activity: the way tasks are structured, how they are completed, the order in which they are accomplished, their synchronization, the way related information flows among different participants and the methods used to monitor the process, the performance of each task and its global impact.

When talking about virtual communities, this technology is an essential part of every collaborative tool. Workflows enable the automation of a whole work process or part of it. During this process, documents, information and tasks are passed from one participant to another when concrete actions are taken, following a set of previously defined rules. The process performance, represented as a workflow, is defined in a 'process definition' which identifies the different activities, rules and control data.

Nowadays this technology is becoming more and more interesting because of the economic benefits it provides directly related to the process automation. Human and material resources as well as process time are optimized. Another reason for this increasing adoption is the existence of tools that integrate the set of applications needed to run one of this workflow based environments.

Due to the importance of this technology, some standards have been established to standardize workflow definition languages and their graphic representation. The most extended definition languages are XPDL (2008) and BPEL (2009), and the standard graphic notation is BPMN (2010).

There is a wide group of applications around workflow technology. Most of them are oriented to workflow creation and edition. The most complex ones are integrated solutions, capable of creating, editing and even executing workflows as a service by including workflow engines. These workflow systems allow collaboration and knowledge sharing among non co-located participants, making it possible to optimize different project processes when they are designed for that purpose.

Until now, workflow technology has been pushed into the background, used rarely to support knowledge management and awareness, due to the fact that one of its main characteristics is its inflexibility. Therefore, it has been useful only to support well-defined processes that suffer no changes as time goes by. As a consequence, workflow technology has been a minimal component of groupware packages, working together with decision taking support tools, group coordination techniques, time management, planning methods, etc.

With the tool presented in this paper, a new way of taking advantage of workflow technology is introduced, employing workflows in a new flexible way that enables the dynamic workflows definition, edition and customization to cover every different project needs. Business organizational aspects related to information management have been considered during its design. As a result, this technology acquires the main role in a new paradigm of knowledge management.

4. PRACTICAL USE CASE: ITECDESK

ItecDesk is a project oriented to use the power of workflow technology to automate, organize and, over all, optimize different tasks with special stress in distributed software development processes, where a significant number of distributed professionals employ workflow management to coordinate their daily activities. The results obtained can be directly generalized to cover common virtual communities' needs.

In this project the selected workflow engine was OpenWFEru, introduced in the following sections. This workflow engine is executed in the application's server side, using Ruby technology (Ruby, 2010), while the launching of workflows and the interactions made during each step of the workitem are done using a Flash web client programmed using Adobe Flex (2010).

The use of the application starts with a user accessing the web client with a common web browser. Once the Flash client is launched, the user must authenticate against a Single-Sign-On central authentication service (RESTful API CAS, 2010), used to provide security and make easy a future integration of this application with other tools. Once user's credentials have been accepted, a main view is showed containing the possibilities of starting a new software development workflow instance (a new work item) or executing actions over pending work items that have reached a point when human interaction is needed, specifically interaction with the connected user.

The method to interact with work items is using dynamically rendered forms. These are rendered from form definition data contained in the workflow definition. This concept will be explained in subsequent sections. The user completes these forms and, after accepting, data introduced are collected and sent to the execution engine as the result of the current work item step in which the workflow had stopped. The engine evaluates such results and, after processing them, the workflow is resumed. During each step, external functions can be launched automatically to add extra value to the data entered by the user. This is repeated step by step interacting with different participants until the end of the workflow is reached.

During a work item lifetime, from its generation to its final step, some events are thrown such as notifications configured in the workflow definition file, to allow a better and more complete information management.

4.1 Architecture

The developed application follows the client-server paradigm that enables a distributed working environment. The system architecture is shown in Fig. 1. It is composed by two main modules. First, the server containing the core functionality which is the workflow engine in which work items are started and modified as they advance step by step. The second main module is the client, ready to interact with the server via a REST API, and able to dynamically render forms from retrieved definition data.

Apart from the two main modules, there is a set of components with different complementary functions. It uses a lightweight directory access protocol (LDAP) containing different participant roles and a CAS server implementing a Single-Sign-On authentication service. A configuration management data base (CMDB) is used to store different artifacts created during the distributed software development process that is supported by one or more work items. Finally, there is a workflow definition tool formed by two small applications. They allow the user in charge of defining a new workflow to do it graphically instead of programming it in plain text. They also allow this user to design the forms to be shown in a graphical way too.

There is a REST interface enabling interaction between the client and the server part. This communication is done using XML-based Atom (RFC 4287, 2005) as the main data format and simple form definition language (SFDL) (Moreno et al., 2010) to exchange form definitions to render.

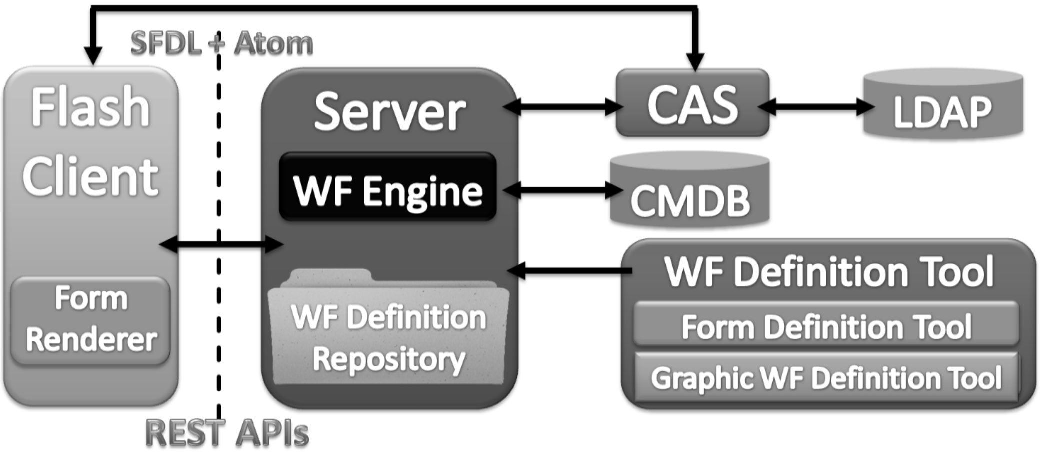


Figure 1. System architecture.

4.1.1 Client

As the architecture of this application is based on a REST wrap of a workflow engine, allowing a standard interaction with it, the client part of the application could have been implemented with almost any modern programming language, due to this interface versatility.

In this case, it was decided to develop a web based client to make it easy to execute in a common web browser. This way the final user does not need to install it as a stand-alone application, making the tool available for multiple platforms. As it has been introduced before, Flash technology was chosen. Flex programming language facilitates the creation of rich, easy to use user interfaces with an attractive look and feel. This technology is based in MXML, a XML-based markup language to define the elements to be shown, and ActionScript as the main scripting language. The outcome is a rich internet application or RIA (Moritz, 2008).

The most important characteristics of the development were the ones responsible for the REST communications between the client and the engine’s interface located at the server part. Therefore, the client has the functions to comprehend resources information, including work item related data in every different step of the workflow and dynamic forms definition information. All this data is exchanged using extensible XML serialization language. With the received data the client is capable of generating in real time rich forms with which the user interacts to act over the work item. The module on charge of this generation is the Form Renderer. After every interaction, information is sent back through the REST interface to the workflow engine, which updates the work item attributes, executes requested external functions and resumes its flow.

4.1.2 Server

In this architecture the server implements a standard REST interface to allow communication among the workflow engine and the rest of the modules. Besides, external functions required by work items to add extra value to the data they carry, are executed at the server part. It also maintains interaction with CMDB data base and CAS and LDAP authentication elements and stores workflow definitions able to be executed.

The technology used to develop the server is Ruby, an agile and dynamic programming language that provides a fast and flexible development environment. The server part works as the workflow engine container, and therefore it supplies the whole set of knowledge management properties related to the workflow technology itself.

The workflow engine in which the whole tool is based is called OpenWFEru or Ruote. OpenWFEru (2010), as it was at first known, means Open Source Ruby Workflow Engine. It is an open source workflow engine that manages and executes workflows, developed using Ruby programming language. It is based on the mature and extended discipline of business process management BPM.

Ruote defines a new extensible workflow definition language. It can be serialized in XML, JSON or Ruby languages. It is not oriented to the graphical representation of the workflows defined, so there were no graphical editors except for a very simple approach made by John Mettraux which is called Ruote-Fluo (part of the Ruote Project). It is hard to use and is neither flexible nor powerful. As an answer to the lack of editors to create and edit workflows for Ruote engine, and understanding it as an essential part of every workflow project, a Flash based graphical and interactive workflow editor have been constructed as part of the project presented in this paper and it is described below.

4.1.3 Workflow definition tools

The workflow defining process is not a hard task for advanced users as it can be done in three different simple formats: XML, JSON and Ruby. This language is described in detail afterwards. To make this tool useful as a knowledge manager for a wider range of users, a big effort has been made to develop an easier way of defining, not only workflows, but also forms that will be used for final user interaction. For that reason, two different small applications, accessible from the main client, are included with the main tool:

- A graphic workflow definition tool, developed as a web application, based in Flex, allows the user to drag and drop workflow parts from a set of graphic elements on a resizable canvas and link them using arrows. At any time the user can obtain the definition code of the workflow that is being edited and save it in the local drive to continue editing it later, or in the server’s workflow definition repository to make it executable. It can work the other way round, loading or typing a workflow definition using the definition language and creating its graphical representation to continue editing.
- A form definition tool, made in Flex, lets the user drag form elements from an elements bar and drop them on a grid. That way, the user can observe the final look of the form while designing it. Every element added can be edited and its properties can be changed, such as height, width, content, and so on. Once the form is designed, the SFDL correspondent source code can be obtained directly and it can be saved in the local drive or in the server’s repository to make it accessible from the workflows.

4.2 Workflow definition language

Ruote project uses its own workflow definition language. This is a concise language that does not carry any information oriented to the direct graphical representation of the workflows defined, avoiding cages and rows positioning data and obtaining a much simpler and cleaner language. This information is present in some other workflow definition languages such as XPDL, making this kind of languages hard to understand and use.

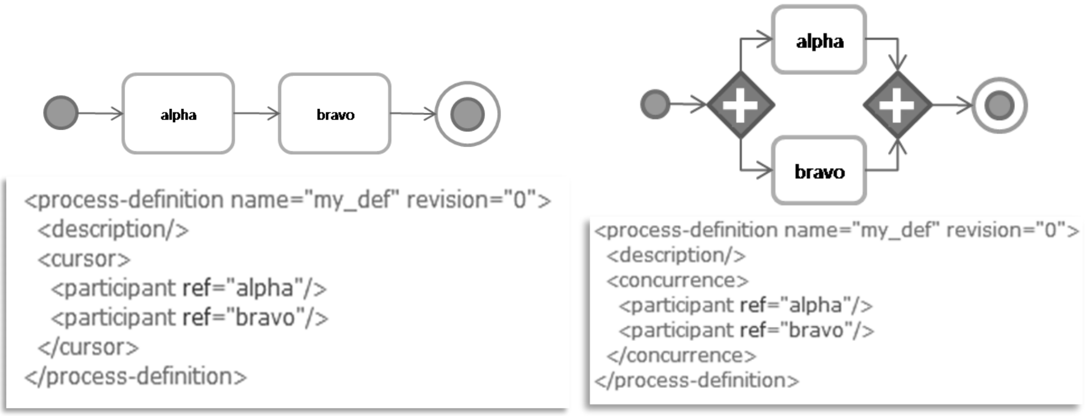
The lack of positioning information, on the other hand, makes it difficult to obtain a graphical representation of workflows defined. This difficulty has been overcome by the development of the previously introduced graphic workflow definition tool.

This language is formed by four main constructors: *process-definition*, used to establish the beginning of a workflow or a subprocess definition; *participant*, used to move the action in a concrete step of the workflow to a specific participant; *sequence*, that establishes that the next steps of the workflow will be executed as a sequence, one after another; and *concurrence*, which establishes that the next steps of the workflow will be executed at the same time, as concurrent threads.

To illustrate the use of these four main expressions, in Fig. 2 there are a simple sequence workflow and a simple concurrence workflow with their definition. In the first example, the expression cursor acts as a simple sequence one (the only difference is that cursor is more potent than the second one as it accepts some special attributes as break, skip or jump to customize the execution). The action passes from the participant alpha to bravo sequentially. In the second one, the action reaches at the same time both participants and the workflow waits until both of them have acted.

Apart from these four main expressions, there is a large set of expressions that allow some functionalities and special performance. Some are *loop, repeat, if, set, wait,* etc.

In the ItecDesk project an extension of Ruotes definition language has been defined to cover two main aspects: the dynamic generation of forms and some basic operations to allow access from workflows to external functions as explained before. The first aspect was covered by the creation of a new XML based language called SFDL, explained in the next section. The second issue was solved by adding some anchors to the original language that launched special functions executed at the server part (mathematical functions, complex decisions, external communications, etc.).



```
<process-definition name="my_def" revision="0">
  <description/>
  <cursor>
    <participant ref="alpha"/>
    <participant ref="bravo"/>
  </cursor>
</process-definition>
```

```
<process-definition name="my_def" revision="0">
  <description/>
  <concurrence>
    <participant ref="alpha"/>
    <participant ref="bravo"/>
  </concurrence>
</process-definition>
```

Figure 2. Simple sequence and concurrence workflow definitions.

4.3 Simple form definition language

One of the extensions carried out to improve Ruote project's workflow definition language is the one that allows the definition of forms that will be rendered at the client part of the tool. The final user will interact with these forms to perform operations over each work item. Depending on the user's role, a different view can be presented, achieving the user-specific reaction needed to attain a complete information system.

To reach this, a new extensible definition language has been developed: the Simple Form Definition Language (SFDL). The main design requisites were to support a wide range of web forms elements (selectors, tables, different inputs, etc.), to support multiple forms for each view showed at a single step, the facility to be defined with different serialization languages such as XML, JSON and YAML, and the inclusion of support for external functions execution at the server part.

SFDL is not the only form definition language. XForms (W3C 2009) was ruled out because it lacked support of nowadays web clients and it is CSS dependent. HTML 5.0 Forms (W3C 2010) was not chosen because external functions cannot be triggered from its forms and it is hard to serialize in JavaScript.

SFDL works with a simple scheme based on tags that indicate position, type, value and extra parameters of each form element. Data model access is achieved using external functions that can be executed when the workflow is being processed or when it is executed.

The form definition tool introduced in earlier sections allows the graphical and dynamic definition of forms using SFDL.

4.4 Knowledge management achieved

ItecDesk tool is oriented to distributed software development supported by the organized interaction among participants using an extension of the workflow technology. In other words, it is a system that enables a complex and powerful knowledge management to improve a concrete distributed working environment.

The type of awareness achieved is task-oriented, as every single task, identified as a software development or a concrete part of it, is modeled as a process and executed in the workflow engine.

Once a new process or workflow is launched, in this case according to a new development task needed, the launcher is registered and he or she will be notified with every event related to the evolution of this development. The workflow engine is the one in charge of, after the launching, transfer the action to a participant that, because of his or her skills, is the most adequate person. This participant will be able to contribute with the actions and information needed to complete the step in which the work item is stopped and let it continue to the next participant in charge. Every involved participant will be notified, with a completely customizable notification level, with the most relevant events occurred during the process. That way, the knowledge is maintained through the whole process for every participant working in the same development course, even when they are physically far from each other.

Furthermore, the work item carries accumulative notes attached by different participants along the workflow in the form of comments that allow explicit textual communication among participants in a much optimum way than the common textual interaction via e-mail, mail lists or chats. This is much more efficient due to the fact that it happens within the development tool itself and, therefore, there is the possibility of accessing software artifacts related with the comments or the implementation as these artifacts are attached to the work item too. This is one of the main keys of the proper information management.

With the capability of executing external functions at the server out of the main workflow engine from each work item, the context is improved giving the capacity to each participant to write in the data base or add developed software artifacts to a compilation machine (that compiles each source reliably), for example.

Proper coordination of the work to be done by distributed professionals is achieved with the use of ItecDesk tool. It also adds new knowledge management mechanisms and attains the work and time optimization of each process.

4.4.1 Evaluation

One of the main tasks after the deployment of the tool was the elaboration of an evaluation plan to carry out the monitoring of the benefits obtained and the possible lacks in this corporative environment related to the use of this tool. The selected approach for this study was based on interviews to the final users after a period of employment, and on observing the way they make use of the tool, just like what has already been done in previous similar scenarios (Gutwin, 2004).

The first monitoring data have been collected and the first iteration of questionnaires has already been processed. The first results obtained from this evaluation method show a process duration optimization directly related to the improvement of knowledge management and the supporting information system. On the other hand, questionnaires reveal that users still utilize suboptimal tools (e-mail) complementing the presented system, what may indicate the need of new features to achieve a better usability.

CONCLUSION

In this paper, the feasibility of using the workflow technology as a knowledge management system is proved. The presented approach, extending existing definition languages to improve workflow technology, helps the maintenance of group awareness and the coordination within a distributed software development process.

An important fact to remark is that these ideas have been the foundation of a real system deployed in a Spanish banking environment. This way, the proposed innovative approach is validated. Also, it makes possible to run a detailed evaluation considering the bank workers as the system final users. The very first results of the evaluation show a remarkable optimization of processes directly related to the improvement of knowledge management and awareness.

Nevertheless, traditional groupware related challenges described by Grudin (1994) have been overcome during the development, deployment and evaluation of the presented tool. Specifically, the adoption process and the difficulty of evaluation have been the hardest challenges.

There are some interesting areas to explore as future work. First of all, the design of workflow templates that could cover well-defined business tasks that appear in every project would speed up the use of this application within a new project, being customizable with the use of the previously introduced graphic workflow definition tools. The same way, some form templates would be useful for workflow designers too. A specially interesting but hard to complete task is the combination of this kind of tools with the main integrated development environments (IDEs) used today, as it would increase the use of this knowledge management tools.

As a conclusion, it is evident that there is a need of a flexible, powerful and easy to adopt tool to optimize the management and coordination of distributed teams. None of the existing alternatives is being widely used as they all entail an excessive complexity when compared to the benefits they produce.

Workflow technology is a mature field poorly used in knowledge management and information systems, only employed to automate rigid and well-defined processes. In this paper a new way of taking advantage of this technology is presented, using workflows in a more flexible way and enabling their customization to cover every different project needs.

ACKNOWLEDGEMENT

This work has been supported by the ITECBAN project, sponsored by the CDTI and the Spanish Ministry of Industry, Tourism and Commerce. The authors would like to express their gratitude to INDRA Sistemas S.A.

REFERENCES

Adobe Flex, 2010. *Open source framework.* [Online]. Available: http://www.adobe.com/products/flex/ [23 December 2010].

BPEL. 2009. *Oasis.* [Online]. Available: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel [23 December 2010].

BPMN, 2010. *Information Home.* [Online]. Available: http://www.bpmn.org/ [23 December 2010].

Damian, D.E. and Zowghi, D., 2002. The Impact of Stakeholders. Geographical Distribution on Managing Requirements in a Multi-Site Organization. *Proceedings of 10th Anniversary IEEE Joint International Conf. on Requirements Engineering*, Washington, DC, pp. 319-330.

Ellis, C.A. et al, 1991. Groupware: some issues and experiences. *In Communications of the ACM*, Vol. 34, No. 1, pp. 39-58.

Europarc, C. et al, 1992. Awareness and Coordination in Shared Work Spaces. *Proceedings of Conference on Computer-Supported Cooperative Work*, Toronto, Canada, pp. 107-114.

Grudin, J., 1994. Computer-supported cooperative work: history and focus. *In Computer*, Vol. 27, No. 5, pp. 19-26.

Grudin, J., 1994. Groupware and social dynamics: eight challenges for developers. *In Communications*, Vol. 37, No. 1, pp. 92-105.

Gutwin, C. and Greenberg, S., 2002. A Descriptive Framework of Workspace Awareness for Real-Time Groupware. *In Computer Supported Cooperative Work*, Vol. 11, No. 3, pp. 411-446.

Gutwin, C. et al, 2004. Group awareness in distributed software development. *Proceedings of ACM Conference on Computer Supported Cooperative Work*, New York, Chicago, Illinois, USA, pp. 72-81.

Moreno, D. et at, 2010. SFDL: MVC applied to workflow design. *Proceedings of IADIS International Conference on Collaborative Technologies*, Freiburg, Germany, pp. 61-68.

Moritz, F., 2008. Rich Internet Applications (RIA): A Convergence of User Interface Paradigms of Web and Desktop - Exemplified by JavaFX. *Diploma Thesis of University of Applied Science Kaiserslautern*, Zweibrücken, Germany.

OpenWFEru, 2010. Route - open source ruby workflow engine [Online]. Available: http://openwferu.rubyforge.org/ [23 December 2010].

RESTful API CAS, 2010. *User Manual*. [Online]. Available: http://www.ja-sig.org/wiki/display/CASUM/RESTful+API [23 December 2010].

RFC 4287, 2005. *The Atom Syndication Format*. [Online], Available: http://www.ietf.org/rfc/rfc4287.txt [23 December 2010].

Ruby, 2010. *Programming language.* [Online]. Available: http://www.ruby-lang.org/ [23 December 2010].

W3C, 2009. *XForms 1.1* [Online]. Available: http://www.w3.org/TR/xforms/ [23 December 2010].

W3C, 2010. *HTML 5: A vocabulary and associated APIs for HTML and XHTML*. [Online]. Available: http://dev.w3.org/html5/spec/ [23 December 2010].

XPDL, 2008. *Workflow Management Coalition*. [Online]. Available: http://www.wfmc.org/xpdl.html [23 December 2010].