An Experiment in Light Workflow

Stewart Baillie, Anthony Bennett, Anne-Marie Vercoustre, Ross Wilkinson

Division of Mathematical and Information Science
CSIRO
723 Swanston St, Carlton 3053, Australia

[Stewart.Baillie, Anthony.Bennett, Anne-Marie.Vercoustre, Ross Wilkinson]@cmis.csiro.au

Abstract

Workflow tools have been successfully applied to automate work in many situations where the work is well regulated, there is a stable pattern of work, and there is a sufficiently high volume or sufficiently high importance to justify the cost of automating the activities. In many other circumstances there is a very mixed story of success and failure of workflow implementation. The Web also has changed work practices and increased the role of electronic documents, in particular, forms, as a support for many distributed tasks. In this paper we explore using a workflow approach based on fully self descriptive documents, that embed the information and instructions necessary to support processing the document, within the document. The traditional workflow engine or server that is typical of current workflow tools is discarded, but the document still allows a full work process to be applied, without necessarily enforcing the process. Ideally one would need a Web browser, and an email client, and no workflow system at all. This paper shows how this is not quite possible, but one can build a very small supporting application to achieve light workflow.

Keywords Workflow, Document Flow. Web-based Workflow, XML, XSLT, Co-operative work.

1 Approaches to Workflow

There are various ways companies implement their internal procedures to effectively accomplish their routine tasks. These range from manual approaches to fully automated, and can involve one person or several.

In a manual approach, a task is modelled or at least described in a procedural manual. Workers take responsibility to cany out the steps described in the manual, and use standard office tools - word processor's, email and web browsers - to carry out the work. Often this works well, but it relies on human validation and execution, which is prone to error.

The aim of a workflow system is to automate in some part critical business processes within an organization. According to the Workflow Management Coalition, workflow is:

'The automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules'[1].

There are numerous workflow products on the market, some of them now quite mature [4], [8], [9], [10]. The approaches taken by these products can be placed in one of three classes [11]:

- E-mail or Message based. These systems only require a pre-existing e-mail system to operate. Generally, tasks are mailed to the next person in the process, along with any relevant documents for them to complete the task. Once their stage is complete, the information is mailed back to the server.
- Web Based. These systems make use of a web server, and a browser to perform the workflow This gives the advantage that users may access the workflow system from essentially anywhere. The main problem with the system is that client's must log in to receive the list of tasks that need to be completed.
- Monolithic and proprietary based (production based). Applications in this category are generally quite large. Users access the system through customised client software, which attaches through custom communication techniques (ie not e-mail or http protocol) to a server. These systems attempt to cover every scope of current workflow technology.

[1] www.aiim.org/wfmc/standards/docs/glossy3.pdf

Common to all products is a server for controlling the workflow and a commercial database system on the server side for the storing of information. They also all require substantial process modelling, then a workflow description to support that model.

In order to exploit the potential of the Web. many companies now provide a Web interface to their Workflow Management System and a notification through e-mails, e.g. [8], [9], which results in systems that may cross over the boundaries of the classes defined above.

In addition to workflow approaches it is also worth mentioning the cooperative work approach, where a group work together to cany out an information task, using tools to make relevant information available, as well as tools that help the group cooperate. This work might still require workers to use a manual, but may better support work in an ad hoc environment when the process is not clear, even though the outcome may be. The cooperative approach tends also to move towards a Web-centred approach [1], [3]. The cooperative approach may benefit the introduction of light workflow in order to improve the coordination of work carried in these environments.

In this paper we will concentrate on support for those information tasks that can be earned out by the production and manipulation of documents. Thus we concentrate on document flow.

2 The Need for Light Workflow

Commercial workflow tools are based around a complex, and often sizeable server. This server acts as the hub for executing processes, including the storing of process data, view generation and the enforcing of rules, by data validation, and forwarding.

This approach works very well for workflow types that are frequently executed and static in approach. However, as workflow types become less frequently executed, an increased requirement for ad-hoc processing arises, or a need to cross company boundaries appears, systems taking this approach become less effective.

Effectiveness can be compromised for a number of reasons, including cost, human behaviour and incompatibilities with other workflow products.

In order to support more distributed workflows, and those less well catered for by current tools, a system needs to be smaller (and thus more cost-effective), cross platform compatible, and be simple enough to use so that ad-hoc processing is actually effective.

A typical example is the Conference paper review process which is often earned out via a central server where reviewers enter their review through a form. The drawback is that reviewers are not always connected to Internet when making their review and may also wish to keep a copy of their review for future use or in case the review is lost. They would rather connect to the server to get the fonn on their computer, complete the form using their local browser, save it locally, and send it when they are later connected, by opening it again and clicking on a submit button.

For this we propose a system of light workflow, where much of the server functionality is removed, such as the enforcing of flow rules, and is implemented in a self descriptive 'transaction document'. This document moves from client to client, providing everything that is needed to process the document on the client side.

In order to develop the idea of light workflow, we will first present an example application appropriate for this approach, and our system that implements this example.

3 An Example Application - The Paper Submission Process

In order to test the concept of light workflow, a test application was needed. Chosen was the CSIRO process for getting a paper approved for submission to publication. This becomes a 'workflow type' (often refened to as a process or business process). Broadly speaking, the paper submission process (if executed in correct order) consists of the following steps:

- Gain approval to write the paper from project leader (providing paper title, abstract, etc).
- Write paper
- Obtain reviews of the paper.
- Make changes according to the reviews, and submit paper for acceptance.
- Determine what to do if paper not accepted.
- Obtain copyright agreement and modify according to CSIRO guidelines.
- Make changes to the final paper and submit.
- Send copy of final paper to publications officer, and record the publication on the publications database.

Why do this electronically? Often, many steps of the process are left out, especially those that are of no direct consequence to the author(s). This leaves scattered records of papers that have been accepted for publication. By doing this process electronically, we may make automatic records of accepted publications, and provide a record of the transactions that took place to get the paper published, should this ever be needed.

The other reason is in tenns of light workflow, this process consists of some challenging points to consider:

- Some steps are simply document uploads. These need to be supported in an attractive manner, so that users will still want to upload these documents (to form a permanent record). Users need to see the value of recording such documents.

- The process requires flexibility. Firstly, for some steps, users complete them in multiple ways. This is typical of reviews. The system must make it easy as possible for users to complete such steps in the manner they are used to. Secondly, ad-hoc processing is required - not all steps will necessarily be completed, and may also be completed out of order. This needs to be supported.

In order to use this as a test platform, we created a detailed step-by-step breakdown of an appropriate process. This involved discussions with researchers (users) to obtain a model of how they approach the process of writing a paper, and what would be important to them in a system. This resulted in a detailed step-by-step breakdown of the process.

Using this, work could begin on the actual light workflow system.

4 A Light Workflow System

The light workflow approach provides a system that is document-centric, as opposed to the data driven approach of most commercial tools. This technique aims to provide a portable, workflow enabled document. The design of this approach keeps in mind the following points:

- The system should be document-centric (self-descriptive).
- Clean abstraction of document data, views and flow.
- Web standards should be used, to allow the workflow to function on any client with a modern browser. (XML, Javascript, Mail).
- Ad-hoc/flexible processing is required.
- Processing is moved from the server to the client, to remove the need for a server.
- User of standard data model and public API for data processing (XML/DOM, XSLT).

Instead of a server acting as the hub of activities, the 'transaction document' becomes the central component. The transaction document stores both the data and the instructions that describe the processing of each step, such as data validation and storing, and the relationship between steps.

When a step is completed, the entire transaction document is forwarded to the next person in the flow.

As is the characteristic of workflow applications, the transaction document produces a differing view of itself for each different step. Additionally, processing instructions may or may not check which step was executed previously (and thus check if it is their turn or not) thereby allowing for ad-hoc processing, by being able to complete any step at any time. To utilise this ability, some form of 'step selector" is available, allowing quick navigation and examination of each of the steps available (as seen in the left frame of figure 4).

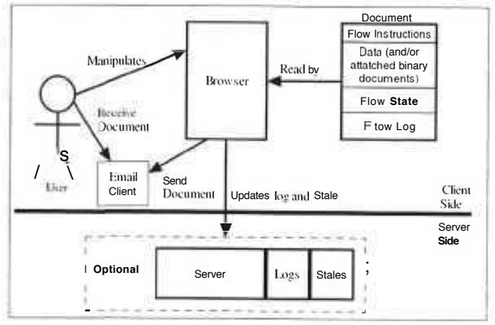

Figure 1 - A simple Architecture for a light workflow system

Figure 1 illustrates an architecture for light workflow. A browser is used to parse the transaction document. The transaction document is rendered according to the step, and the user interacts with this. Data is input to the system, and the instructions for that step process this data, modifying the transaction document as appropriate. When processing is completed, the updated document is sent to the next person in the process.

A server is still presented in this architecture, but is optional. However, often is important for a user to be able to find the current state of a process, or a need

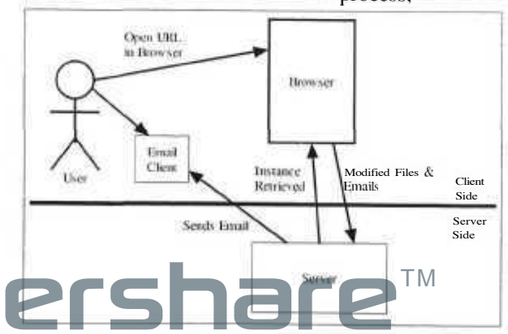

Figure 2a - Compromise made due to issues with browser security

to analyse logs of workflow usage will arise. In this case, the server acts as a 'message board' for this data.

If this functionality is not needed, then it can be left out.

However, when attempting to implement this architecture, we needed to make some concessions. Figure 2a illustrates these concessions, whilst Figure 2b illustrates the implementation architecture.

Technically, the architecture presented in 2a & 2b differs greatly to the architecture presented in figure 1. but conceptually, they are similar.

The primary reason for differences in architecture is the default security level in browsers. It is not possible to do any sort of file manipulation on the client (other than loading XML documents), nor is it possible to automatically send an e-mail. It is possible to set non-default security to circumvent these problems, but this creates additional tasks to allow functionality, and users may not be happy with this scenario, given the recent high profile of internet scripting viruses. Thus the server is now used for two additional tasks - storing the transaction documents (and related files), and sending e-mails.

The server, shown in Figure 2a, runs on top of an Apache web server, which is equipped with the Jserv module to allow it to run Servlets. Three separate items are served from this:

1. Workflow Engine - This is not technically an engine, but a collection of small utilities. Users connect to this to create a new workflow instance, to specify which instance they want opened, (and to what step) and to view the status of other instances, and log files.

2. Servlets - These allow the transaction documents to access the facilities of the server to create new instances, send e-mails and save modified instances back to their original location.

3. Instances Folder - This folder stores the templates and the running workflow instances. The template is a transaction document that represents a particular workflow type. Copies of these templates arc made each time a new workflow is created. These arc referred to as workflow instances.

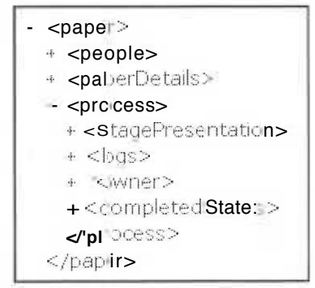

Figure 3 - Structure of the Transaction Document, from the sample workflow type.

The transaction document is constructed from two files, an XML document and an XSL document. The XML document consists of the workflow data, and the instructions for each step. The instructions currently are specified in JavaScript, for interpretation by a web browser. Figure 3 shows the structure of such a transaction document, in this case for the paper submission process.

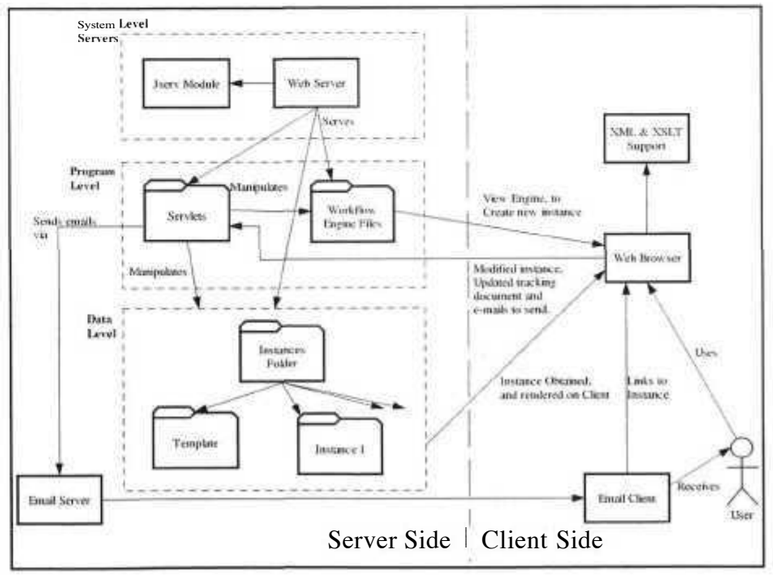

Figure 2b– The architecture of a implementation of a light workflow system.

This document is broken into two parts - the workflow data, and the transaction document data. The former is specific to the workflow type. For the paper submission process, there is an element (people) for storing data about the people involved in the process, and another for storing data about the paper itself (paper details).

The second part to the document structure is the process data. Each workflow type has this structure, completed according to the particular flow. The stagePresentation tag (under the process tag) represents all data that is needed for displaying a step, except for the forms, and the actual appearance (which is handled by the XSL document). Therefore for each step, it contains the processing instructions, text to be displayed to the user, the contents of e-mails and the explanation to be displayed to the user once processing is completed. The Log tag records each step that takes place, when and what data changed. Owner represents the creator of the process, and completedStates records the states that have completed fully.

The XSL document consists of display information. By interpreting a parameter provided to it at display time, the XSL creates a display appropriate to the step to be viewed. The appropriate text and code is extracted and formatted into the display, and then the appropriate form inserted.

Figure 4 shows an example screen of the paper submission process running. The left hand 'frame' shows the stage selector structure. Clicking on any of the stages automatically opens it on the right hand side. The right hand side has been totally rendered from the transaction document, and is displaying the contents that specify each field.

The user can also be prompted by e-mail to open an URL. When the URL is clicked on. the appropriate transaction document is opened, rendered and displayed to the client according to the step specified.

It is important to realise that despite the reliance on a server, conceptually, the two different architectures are almost the same. If you were to remove browser security, then only minor adjustments would be required for the system to work purely client side. The transaction document carries everything with it to describe the workflow.

5 Discussion

We have argued that workflow systems are valuable for some activities in a workplace, but that there are times in which a fully prescribed workflow is not possible, or a commercial workflow tool is not effective. We thus designed and created a light workflow system to address some of these needs. In order to evaluate this system, we can analyse the approach against the following criteria:

- Does the paper submission process work?

Yes. The system has been used successfully to gain pennission to write a paper, right through to publication.

- Is the effort needed to create a new application too high?

At this stage, yes. For a typical process of 8 - 10 relatively different steps, it look us several months to fully complete a new process. This will no doubt improve as expertise is gained, and generic libraries of functions are obtained. However, the development time is still likely to be too high. Most current workflow applications include a graphical tool for specifying workflows, and we should assume such a graphical tool to be built for our system. Prior to this, we need to develop a language, preferably in XML for specifying the processing details for each step. This language should include XML fonn definition and manipulation[5], variables, and calls to DOM functions

- Can a workflow be easily changed?

Freedom to change and alter the system depends largely on the level of change required, but generally it is not an easy task. For example, adding a new data field may require updating the DTD, altering several forms, and including the new data field into processing code. Identifying where changes need to be made, and then testing them, is likely to take several hours. This is again a call for a process creation tool - without it the cost of defining and maintaining a flow becomes too high.

- We haven't been able to get away with just using a browser, email clients and "smart" documents.

Is the system still actually light?

Needing a server to complete basic tasks is a compromise on the original design. However, we believe this is still a light workflow system. Firstly, to its cunent stage, it has not taken very long to implement. None of the complexity that you might find in a commercial workflow tool is present. Secondly, very few changes would be needed to allow the system to run purely on the client side, and we could even do this now on a machine running Outlook. (We could easily reduce security on browsers, and if necessary just use a public e-mail server, which does not really comprise the design).

- What are the limitations of this approach?

There are a number of limitations of this approach that need to be considered.

- Without a server, this approach does not support central functionality such as stage and audit tracking for workflows. However this additional functionality could be provided by external services to which the workflows would send appropriate infonnation.
- Since we do not enforce a flow, we rely on the users to use the system conectly. We have less of a guarantee that the process will be completed, or completed conectly. A commercial workflow app enforces the rules more rigorously, and has the capacity to send out reminders, set deadlines etc. However, if we don't display a stage selector, the flow is then enforced, so enforcing of rules may become an option that can be selected when a new instance is created.
- Since the XML is processed on the client side, we can't easily prevent users tampering with data that they should not. A 'trust' system is worked on. where it is assumed that users will have no interest in completing a stage that is not theirs, or changing data that they should not. In an environment where this is not the case, we may need to extend the system to allow for user authentication, placing an increasing need on a central server, or support the use of digital signatures on sections of the document.
- Data is inherently less secure, since it may be located on multiple clients, thus making it easier for unauthorised individuals to obtain data.

6 Conclusions

We have demonstrated the feasibility of a light workflow based on a document that carries all the data and instructions necessary to its processing. Should the ability to save locally, and send e-mails without user intervention arrive, then we can have a totally client side workflow system. This is what makes the major difference of our approach with cunent Web-based Workflow systems such as Lotus Notes and others. Another advantage is the use of standard Web languages and processors, such as XML, XSLT. and Javascript that allows the transaction document to be reused and further processed by new or external applications, which is not possible with a proprietary format.

We believe that this system can be effective for workflow types that are distributed, infrequently executed, require flexibility (ad-hoc) or are driven by documents. This system has the advantage that any client with a modem browser will be able to participate in the flow. This allows company boundaries to be easily crossed, e.g. the purchase order to an occasional supplier, or where an ad-hoc group has formed, e.g. a conference review process.

However, it is unlikely that we can ever move totally away from requiring a server, unless we change the base technologies and go away from using only standard technology. If we used our own custom technique, we would need our own client side parser/viewer.

There are times where light workflow is less appropriate, and a centralised approach is best. As noted, these are times where the flow is frequent, and unchanging, or the process is data driven. Central systems also have improved security, and allow the

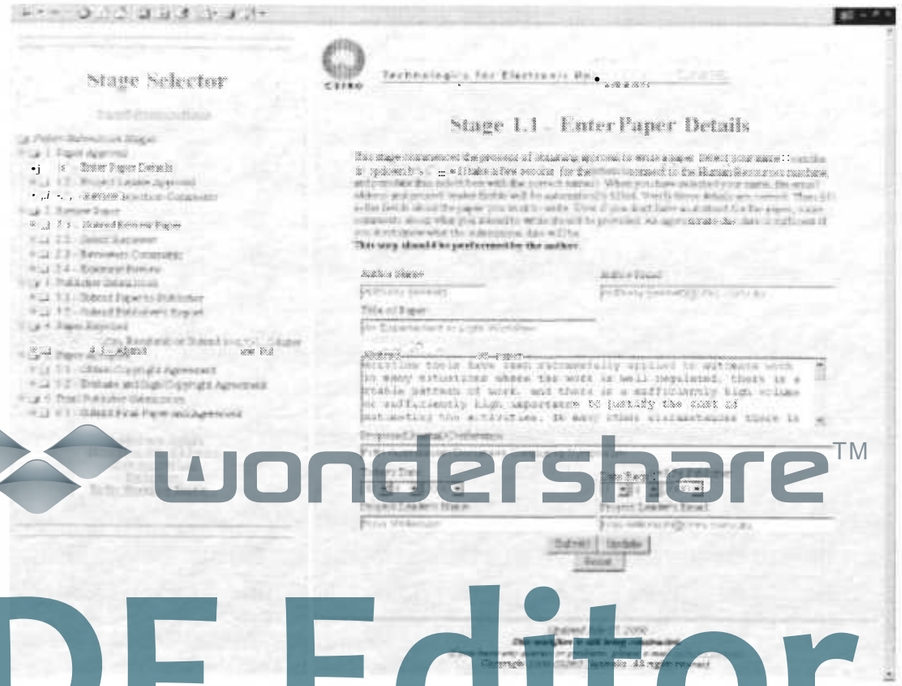

Figure 4 - Screen shot from running paper submission system.

generation of complex statistics on usage. This may be important for improving efficiency of processes, through identification of sections of the process that result in a delay.

Finally, there are also times where no form of workflow automation may be best. Such situations might be in collaborative work, where group-ware type systems are more appropriate, and situations where for legal or security reasons, only hardcopy is acceptable.

Further testing of the paper submission process, as well as other types of workflows will be conducted to more fully assess the capabilities of this approach.

References

[1] Natalie Glance, Antonietta Grasso, Uwe M. Borghoff, Dave Snowdon and Jutta Willamowski, Supporting Collaborative Infonnation Activities in Networked Communities, In: Bullinger, H.-J., Ziegler, J. (eds.): Proc. 8th Int'l Conf. on Human-Computer Interaction (HCI'99), Munich, Germany, August 22-27, 1999.

[2] Koch. T.: XML in practice: the groupware case, in Proceedings of IEEE WET ICE Workshop 1999, Stanford University. http://bscw.gmd.de/Papers/wetice99/wetice99.pdf

[3] Bentley, R.. Horstmann. T., Trevor. J.. The World Wide Web as enabling technology for CSCW: The case of BSCW. in *Computer-Supported Cooperative Work: Special issue on CSCW and the Web, Vol. 6(1997), ©Kluwer Academic Press.*

[4] Microsoft Exchange 2000 Worflow Engine, http://msdn.microsoft.com/msdnmag/issues/0700/exchange/exchange.asp

[5] Anders Kristensen, Formsheet and the XML Forms Language XML Forms reference, in Proceedings of the WWW8 Conference, Toronto, 1999.

[6] Charles K. Ames, Scott C. Burleigh, and Stephen J. Mitchell,WWWorkflow: World Wide Web Workflow, in *Proceedings of the 30th Hawaii International Conference on System Sciences (HICSS-30)*

[7] John S. Davies, IBM MQSeries Workflow: Staff Assignment Techniques, White paper. http://wvvw-4.ibm.com/software/ts/mqseries/workflow/

[8] Lotus Notes, http://www.lotus.com/

[9] Oracle, SQLFlow Workflow Management System, http://170systems.com/SQ/flow.htm

[10] GFI - Email Flow - www.emailflow.com

[11] GFI, Workflow Technology - an introduction (www.workfl.owsoftware.com/workflowwp.pdf).