An Integrated Workflow Management Solution for Heritage Information Mashups

{A.Al-Barakati, W.Zhang, m.z.patoli, M.Gkion, N.Beloff, P.Newbury, M.White} @sussex.ac.uk
Department of Informatics, University Sussex, Brighton, United Kingdom, BN1 9QJ

Abstract
This paper outlines the process of developing and deploying an integrated workflow management solution for our system that uniquely integrates heritage data mashups whose digital content is derived from social network repositories and a specific museum digital collections repository and presentation system called ARCO. This workflow solution accommodates a number of integration techniques, based on social networking with user defined content, and using virtual and augmented reality in a Web 2.0 mashup to dynamically present digital heritage content. Other technologies exploited in this scenario include a web service based Grid solution for generating 3D virtual reconstruction animations. The implementation of the workflow solution is based on the Windows Workflow Foundation while the adoption of the multi-tiered human workflow architecture leads to a fully integrated workflow management engine.

1. Introduction
Over the last decade workflow management systems have become an integral part of most enterprise projects for the production of tools concerned with data and control flow management. Additionally, the proliferation of web services and Internet technologies has led to the advent of enterprise information mashups where data is pooled from a number of web sources into a single integrated source for the management, preservation and sharing of information. Devising workflow management solutions for managing systems that deal with a diverse range of data as well as complex processes, data acquisition, visualization, processing and management poses a number of unique challenges. Such challenges range from the need to have effective mechanisms to utilize the available resources to developing workflow management mechanisms to coordinating the involved software components and the data that operate and flow through the system.

The lightweight simple programming model that Web 2.0 adopts provides an ideal implementation framework that encourages resource sharing and communication on the web, which also leads to applications that can meet various data distribution and sharing requirements. Web 2.0 mashups and service orientation are based on a combination of web and networking technologies where APIs and web services are becoming increasingly available for the vendors of information mashups to utilize. The popularity of such applications was further enhanced by the introduction of innovative dynamic Web 2.0 technologies like AJAX that allowed for richer user experiences [Fahringer 2007]. The ability to build flexible, efficient and rich data applications utilizing service-orientation and mashups will be further illustrated below.

Such Rich Internet Applications (RIAs) require efficient workflow management, tracking and monitoring means to supervise their underlying control and data workflows and activities so that coherent applications can be built. One approach that can be adopted here is the implementation of a dynamic workflow management framework that can run on top of an existing mashup to provide the necessary management and coordination functions. This generic framework can be designed to integrate with the data services in place to arrive to a fully integrated system that is entirely manageable through the provided workflow management tools.

2. The Anatomy of a Mashup
The term data mashup is a term used to describe the applications that combine data of different formats from various disparate sources into a single integrated data source that can be easily accessed by the intended end users[Tatemura et al. 2007]. Data mashups offer endless resource sharing possibilities enhanced by the effective and highly customized delivery of data that they can achieve compared to other data integration and extraction techniques [Rahm 2007].

Users can benefit from customized frontends and services especially in an enterprise web environment where contents can be drawn from a variety of external data sources to create new totally integrated data services [Merrill 2006]. Moreover, According to [Rahm 2007] enterprise data mashups represent an effective approach for data integration while avoiding the common problems that are usually associated with the traditional data extraction and delivery approaches, which also suffer from a number of shortcomings. Such shortcomings range from setup times that are too high for data crawlers for example to queries that are not sufficient enough to extract the required data.
The last few years has witnessed the advent of a number of innovative mashup applications that proved to be flexible and effective mediums for data enabling in an enterprise domain. This was mainly fueled by the pioneering combination of dynamic web technologies for instance, XHTML, CSS, Java Script, RSS, REST and the Document Object Model (DOM) [Zangerl 2007]. This certainly made the implementation of mashups technically more viable by exploiting some of the powerful above mentioned tools. For instance, XMLHttpRequest has proved to be a very useful and effective solution for page content reloading opening new possibilities and abilities for the existing web applications.

Among the most notable current online mashup applications comes the Google Mashup Editor which is an AJAX based development framework with a set of tools for creating web applications and services that are integrated with Google services such as Google Maps and Google Base. The anatomy of web data mashups is further illustrated in Figure (1) below.

---

**Figure (1). The Anatomy of a Web Data Mashup**

http://web2.socialcomputingmagazine.com/making_the_most_of_the_web_creating_great_mashups.htm

3. The Workflow Model

Workflow management applications have become an important tool for managing business process and data flows especially in enterprise applications. Workflow management can be defined as:

“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.” [AccTech 2007].

While, workflow management systems play an important role in modern enterprise environments by providing means of managing and controlling the existing business processes that operate within the system, both for workflow and for that matter mashups discussed in the previous section, little work has been published on there application to digital heritage, an area which our current research work is focused on.

The whole notion of a workflow system is to control the data flow and business processes that interact with each other from one side and the system users form the other. Furthermore, [De Smet et al. 2007] points out that those workflow management systems mainly fall under two categories. Firstly, the ‘Human Workflow’ model that as the name suggests involves the interaction between the user and the machine to achieve the functional goals of the system. Secondly, the ‘Machine Workflow’ model which essentially involves the interaction between the internal system agents via specialised service calls to achieve the required overall functionality.

On the other end of the scale, from the perspective of data flow and communication [Fox 2006] describes a workflow model in an enterprise environment as a model that is based on a combination of data flows and procedure calls which is generically called ‘Distributed Parallel Programming’. In such a model the system users invoke actions via service calls that interact with the system. The approach that is adopted in this paper is similar to the Distributed Parallel Programming paradigm while accommodating the unique needs that may arise in the mashup implementation and delivery to the designated integrated data frontend.

The reason for integrating a workflow management framework within the data integration and delivery services provided by the web mashup environment is evident in the need to have effective tools by which the running and management of the mashup are maintained efficiently. This can consequently pave the way for maximum performance and data output.

4. The Mashup Scenario

The proposed workflow management engine will be aiming at managing and monitoring the different operational aspects of the abovementioned enterprise web mashup solution. This will be mainly concerned with implementation of a heritage mashup solution that aims to provide heritage resource sharing for its users. The backbone of this mashup implementation is ARCO the service-oriented solution for the storage and provision of 3D data objects that was created in our lab. This data source was complemented with pooling related contents from a number of other sources including Flickr, YouTube and Google Maps. In this mashup application, the RIA approach is followed to enhance the user experience.

As stated above, the underlying mashup solution uses the web services provided by ARCO for the pooled 3D
objects as well as the standard APIs/Web Services provided by the other vendors to create a rich mashup interface for the end user to be able to search and browse the concerned data objects. The main processes and activities to be managed here are:

1. Ingestion of Enterprise Data: A variety of data objects that are pooled from different web sources are delivered to the users by using the mashup logic, e.g., Flickr and YouTube heritage objects.

2. Search Mechanism: Users are able to locate the required data objects such as video clips, images and sound clips by the provided search facility that returns the relevant results based on the backend logic upon which the mashup was built.

3. Management Services: The user is provided with a web frontend that is used to handle the search queries and present the related search results.

The main aim here is to arrive at an effective tool to manage the underlying enterprise mashup infrastructure as further outlined in section (5.2) below. The investigated scenario is based on the process whereby the mashup interface is used to search for a specific heritage topic/item which will result in invoking the different services that constitute the mashup functionality as will be further discussed below. This scenario is illustrated in Figure (2) below.

The proposed workflow should provide sufficient management for the sequential workflow activities that respond to the user’s actions, as they operate, in the example below of the mashup in Figure 3. Figure 3 represents one of our early mashup prototypes, and as discussed below this example workflow controls the search functions over YouTube, Flicr, and the ARCO database, returning the results to the mashup.

A BPEL workflow definition that satisfies the workflow requirements of the underlying mashup can be created, for which the .NET WF workflow design diagram in Figure 4 is a representation.

This approach allows the business rules (i.e. in this example, the rules that govern the search functionality over the digital heritage collections) will be separated from the mashup application code making the mashup easier to manage and monitor. Consequently, the proposed Human Workflow Model will be incorporated facilitating the ability to integrate different activities and users with a specific workflow instances at different stages of the execution lifecycle.

5.2. The Mashup Application

For testing purposes, we implemented a dummy mashup implied in the workflow of Figure 4, as the mashup in Figure 3 wasn’t quite ready at the time, nevertheless the principle is the same. The dummy mashup application that was integrated with the workflow engine was based on a Web 2.0 mashup solution that uniquely integrates enterprise data mashups with ARCO (our solution for the digitization, management and presentation of virtual exhibitions). The ARCO project provides a very effective tool enabling museums to build virtual exhibitions and museums for online resource sharing and distribution [Walczak et al. 2006].

In addition to the data acquired from ARCO extra data items are retrieved from YouTube, Flickr and Google by utilizing their public APIs that conform to the Web 2.0 services so that existing social networking techniques can be used to capture community knowledge.

The dummy mashup implementation itself is based on service oriented architecture (SOA) where applications can be built from different collaborative applications. Based on that approach, the concerned APIs were called by mainly using JavaScript in conjunction with the C# coding that achieved the dynamic server-side functionality of the mashup. Further, this Web 2.0 mashup implementation makes use of REST based Web Services to access the required social networking repositories such as Flickr and YouTube as mentioned above. The retrieved data that is highly structured based on XML is then combined and manipulated to be represented to the end user, as can be seen in Figure (3), the latest version of the mashup.
Some special arrangements were made to deal with Flickr as it does not have standard JavaScript APIs. This was overcome by using some alternative techniques being mainly passing standard REST GET-requests to the service domain with some custom request parameters. As far as ARCO was concerned, the mashup interfaced with it by communicating with its web services that provided the necessary data manipulation and retrieval services. This enterprise mashup solution was then integrated with the created workflow engine as will be explained in Section (6) below.

5.3. Workflow Engine Design

As discussed above, the backbone of the workflow management framework implementation will be the Human Workflow Model which is particularly suitable for the illustrated scenario as it involves the coordination of the business processes that involve the system users. In this model the program logic will be totally separated from the workflow management by means of building a workflow interface that holds all the workflow rules and definitions.

Additionally, the workflow interface will also hold all the decision making workflow logic such as the workflow runtime services, the initialization and termination of the workflow and workflow custom activities among other workflow management and control services. On the other hand, the mashup application itself will act as a separate client application that uses the underlying workflow management framework to manage its intersecting activities by passing parameters to the workflow runtime followed by the execution of the corresponding actions as imposed by the workflow engine.

In this model special emphasis will be placed on data management due to the data-centric nature of enterprise mashups. Integrating the workflow management framework in such a manner will add depth and dimension to the whole mashup implementation. As stated above, the workflow implementation adopted here will make use of the distributed parallel programming model [Fox and Gannon 2005] where there is an explicit component designated to control sequential workflow activities that occur within the system.

5.4. Implementation of the Workflow Engine

This workflow engine that was built by utilizing the WF’s capabilities is based on a sequential workflow implementation that accommodates the adopted human workflow model. In this human workflow model the starting point of the workflow is an external event which ultimately in this scenario is initiated by the mashup users. The WF’s persistence services provided the necessary means by which the workflow instance is preserved so that the workflow activities are handled efficiently. The workflow services handle the different aspects of the mashup via the defined business rules achieved by the utilization of a combination of workflow implementation tools provided by the WF programming model as follows:

1. **Ifelse Activities**: Ifelse activities were employed to provide coherent set of rules resulting in precise decision making procedures and responses to the requests and data passed to the mashup. A combination of Declarative Rules and Code Conditions were used here to manage the integrated mashup. Such Ifelse activities can be used for example to determine the type of services requested whether it be YouTube, ARCO or Flickr among the other available mashup services. Consequently responses with the appropriate actions are delivered to the integrated mashup backbone as demonstrated in the code snippet below.

2. **Executable Code Activities**: Executable code activities were exploited here to define some of the workflow steps that contribute to the overall workflow functionality of the system. They were also used where there was a need to define some custom activities. This can serve a number of purposes including service orchestration via specialized code handlers.
4. Workflows are initiated by calling the workflow initiation routine which is in this scenario is WorkflowInstance.Start().

The practical implementation and running of this workflow engine in conjunction with the aforementioned mashup application proved that the discussed implementation model was a viable one as it offered a number of advantages as illustrated in Section (7) below.

7. Results Analysis

It was evident that integrating the workflow engine with the mashup application offered a number of advantages that are mainly related to better management abilities and higher efficiency. Compared to the traditional approach, integrating the workflow engine within the underling enterprise environment that the information mashup operates on increased the overall usability of that model. Additionally, this integrated architecture provided a better framework for the management of the system workflows and the associated data flows and service calls.

Running the developed prototype proved that combining a traditional information mashup application with a workflow engine that runs in the background resulted in adding significant performance and management advantages without undermining the performance of the mashup component itself. This included better execution of the system workflows, easier integration of the system components and a more flexible approach towards data flow and message passing. Table (1) below illustrates the gained advantages as opposed to the traditional information mashup approach.

<table>
<thead>
<tr>
<th>Traditional Mashup Approach</th>
<th>Workflow-based Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invisible execution of the system workflows.</td>
<td>Workflow implementation that allowed for human inspection during the execution of the system workflows by utilizing the workflow tracking services.</td>
</tr>
<tr>
<td>Usability issues especially when introducing new workflows/business rules.</td>
<td>Increased usability by providing means of workflow editing and customization.</td>
</tr>
</tbody>
</table>

6. Putting it together

An interesting aspect of the system implementation was the way that the workflow engine and the associated web mashup client application were all integrated together to provide the required functionality. In this respect, it was decided to adopt a straightforward approach where the mashup application acts as a hosting application. In this scenario the mashup application communicates with the workflow engine that has the required logic to manage the workflow activities of the system invoked by the messages passed to it. The mashup client application acts as follows:

1. It creates a WorkflowRuntime instance based on the workflow engine DLL that was referenced in the application.
2. Workflows are then executed within the defined boundaries and rules of the referenced workflow functions.
3. An instance of the WorkflowInstance object is created to run the required workflows.
Introducing new components may result in alterations across the system that may destabilize it.

Allowed for an easier integration of new heterogeneous components.

Rigid data flow and message passing approach that is built in the mashup logic.

More flexible approach to data flow and message passing enhanced by the adaptive workflow model.

<table>
<thead>
<tr>
<th>Table 1. Results Analysis Comparison</th>
</tr>
</thead>
<tbody>
<tr>
<td>Venturing upon the integration of a workflow engine within the existing information mashup application succeeded in making it more flexible and easier to manage and expand. It can be argued that information mashups can benefit from flexible and adaptive workflow management frameworks to manage their operation and maximize their performance and outcome to the end user. Furthermore, the integration of a workflow management framework within the existing enterprise data mashup contributed to more robust and functional mashup services while maintaining a good level of management and the ability to customize and control the used activities while separating the mashup logic from the workflow logic to suit the particular needs of the user. Consequently, the implemented model proved to be capable of supporting different mashup operations and services and this resulted in the successful delivery of the requested data items according to the scenario outlined in section (6) above.</td>
</tr>
</tbody>
</table>

8. Conclusions

The practical incorporation of a workflow management engine within an existing enterprise information mashup application succeeded in a limited test environment, i.e. the dummy mashup. Our next step is to rebuild this with the final mashup application, and apply the concepts to other workflows within our heritage system.

The utilization of some traditional workflow components such as conditional code statements and the associated rules and branches in a separate workflow engine implemented on top of WF helped to have a seamless integration with our mashup application test paving the way for more powerful workflow management, tracking and mentoring capabilities.

The advantages gained from the customized workflow engine ranged from better management abilities to optimized dataflow and system performance which could profoundly boost the performance and output of existing mashup applications.

9. References


