

# Workflow and web services

Scott Wilson, CETIS ([s.wilson@bangor.ac.uk](mailto:s.wilson@bangor.ac.uk))

## 1. Introduction

This paper examines the topic of coordinating workflows of web services within an e-learning context, and identifies the requirements, challenges, and technology choices involved.

Various approaches to web services are considered, from aggregating simple RSS feeds, to orchestrating multiple parallel and branching processes using languages such as BPEL4WS. The aim is to assess the requirements for workflows in education, and identify the state of the art in workflow standards and techniques.

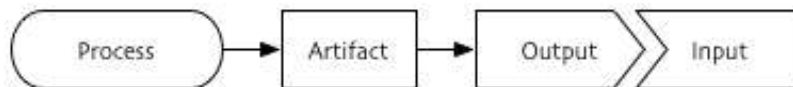
It will be argued that, in the short term, the most feasible approach involves the deployment of simple chains of public, stateless, ReST style services. Such an approach can be achieved in a production environment today using current technologies.

More complex workflows, though important for supporting collaborative learning activities, have considerable implications for security and privacy. Also, due to the relative immaturity of current technologies, such workflows can only be deployed within very controlled environments, such as within a single system, or within a single security domain.

The paper will conclude with a description of some of the gaps in currently available workflow specifications that would need to be filled for collaborative learning activities to be truly service enabled.

### 1.1. About the diagram notation

The notation used for all the diagrams in this paper (apart from those from external sources) is based on a slightly modified UML 2.0 notation (Figure 1).



**Figure 1 - Diagram notation**

A “Process” is any form of activity, manual or automated; an “Artifact” is any data, documents, executables, sources, or files managed or created by the “Process”, an “Output” is any method for exposing the “Artifact” as a service, and “Input” is any sort of client that can access the “output”. The connection between “Output” and “Input” is assumed to be loosely coupled.

The terms “output” and “input” are derived from UML 2.0 convention, and in this case I’m mapping them to the web service concepts of “provide” and “consume”.

## 2. Setting the context

### 2.1. Web Services have become web services

One of the most significant developments in service-oriented architecture (SOA) has been a transition from a very concrete concept of Web Services as SOAP and the WS-\*

stack of specifications, to a “lowercase” (Hanson, 2005) web services concept with a range of implementation patterns that also includes Representation State Transfer (ReST) and XML-RPC.

This transition marks a stage of greater maturity in SOA, with implementers seeking an optimum trade-off between functionality, performance, and simplicity. While the major consultancies and vendors have been extending the functional capabilities of web services through initiatives such as WS-ReliableMessaging/WS-Reliability, WS-Federation, WS-SecurityPolicy and the like, developments at the other end of the scale include the widespread take-up of simple web service APIs, such as RSS, Atom, and iCalendar. These simple services are based on the use of http GET and POST requests, combined with a machine-readable format (this is not even XML in the case of iCalendar).

While much has been made of the “SOAP vs REST” debate (e.g., Elkarra, 2002), a broad spectrum of capabilities and degrees of complexity in web service technology is generally a good thing. For those interested in using SOA, a range of increasingly complex technologies enables solutions to be used that fit an equally wide range of requirements and capabilities<sup>1</sup>.

Currently, while we have yet to see any widespread adoption of SOAP-based web services in the UK education sector, we can already see simpler web service implementations. This may change as the use of SOA develops, or we may discover that the optimum level of sophistication and complexity of web services for use in education is relatively low, and there is little demand for the more esoteric WS-\* technologies.

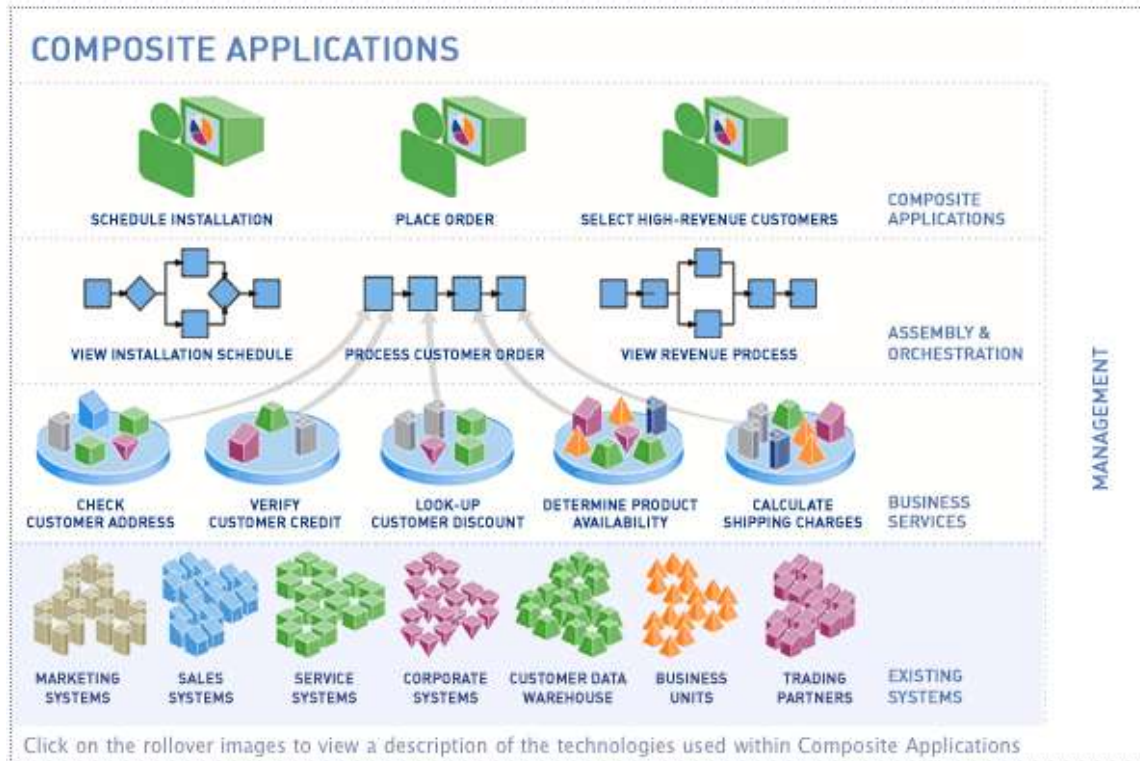
The next challenge for SOA is to enable the coordination of different services and different styles of services in meaningful ways; this is the topic of the rest of this paper.

## ***2.2. Composite Applications and Services***

A Composite Application (Figure 2) is an application that provides a rich user experience by leveraging a collection of services. Typically, the Composite Application itself provides relatively little functionality, but instead orchestrates calls to services in the environment to provide features.

---

<sup>1</sup> This does raise the issue of compatibility and interoperability – is it appropriate to specify for the lowest common denominator, or to the highest barrier?



**Figure 2 - Composite Applications (source: <http://www.seebeyond.com/resource/index2.asp>)**

A Composite Application can also be viewed as the logical evolutionary step that enables a traditional enterprise application to migrate to a service-oriented environment; for example, it isn't difficult to picture the learning platform as the 'ultimate consumer' that pulls together the services offered by the enterprise and presents them in a unified user experience.

In other words, the Composite Application is a useful model of the relations between services, applications, and workflow.

However, the Composite Application model typically assumes an asymmetric relationship between applications and services; that is, models of composite applications typically show how an application consumes a set of services, but rarely how applications can also provide services. For example, an application may consume one type of service, and provide another, or it may consume and provide a service of the same type.

The latter is a common pattern in web services generally, and in content feed services in particular. Content or news feeds are commonly aggregated, analysed, annotated, and then exposed as new feeds. For example, the Edu\_RSS service<sup>2</sup> provides an RSS feed<sup>3</sup> of the latest updates it has aggregated from its source feeds. Another example is a service such as FeedBurner<sup>4</sup>, which consumes and re-publishes a feed with added capabilities, such as statistics, advertising, or creative commons license metadata.

<sup>2</sup> [http://www.downes.ca/cgi-bin/xml/edu\\_rss.cgi](http://www.downes.ca/cgi-bin/xml/edu_rss.cgi)

<sup>3</sup> [http://www.downes.ca/edu\\_rss.rss](http://www.downes.ca/edu_rss.rss)

<sup>4</sup> <http://www.feedburner.com/fb/a/about>

Such patterns highlight an interesting property of simple web services: by providing a machine-readable alternative to web content and functionality, the possibility emerges to apply processes such as mixing, adding, annotating, ranking, and modification to input, before exposing output as a new service. Furthermore, the services concerned do not need to be implemented as complex SOAP services to achieve such subtle and powerful functionality.

### 3. Simple Workflows

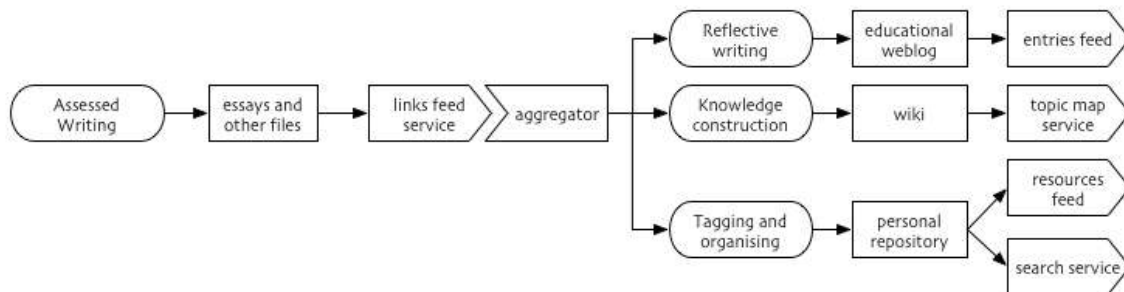
The simplest types of workflows involve the assembly and orchestration of a set of services in a fairly linear fashion.

#### 3.1.Chains

A chain is a set of services that are connected linearly, where the output of one or more services acts as the input for a process that provides one or more further services.

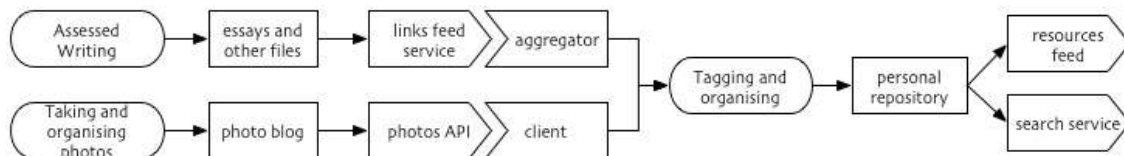
For example, an application may aggregate a links feed service from an assessed writing process, and apply several processes to it (Figure 3);

- § Reflective writing on the artefacts, and the creation of an educational weblog that provides an entries feed
- § Knowledge construction on the artefacts, perhaps resulting in the construction of a wiki, and the provision of a topic map
- § Tagging and organising of the artefacts, and the creation of a personal repository that provides a resources feed and a search service



**Figure 3 - Processes in a service chain**

Likewise, a chain may have multiple inputs. In Figure 4, the input is drawn from multiple services, both inside and outside the education domain.



**Figure 4 - Chain with multiple input sources**

In this second example, the processes of writing for assessment and of taking and

organising photographs result in services that are consumed by the process of tagging and organising items for a personal repository, which then provides services for working with the broader collection of artefacts now available.

### 3.2.Loops

In a loop, two processes are connected by both consumer and provider relationships.

For example Figure 5a student may keep a blog, which is used to develop pieces of assessed writing such as essays (see Figure 5). The feed of essays is aggregated by the blog, and so completed essays can be the subjects of new blog entries (“Hey, I just published this!”).

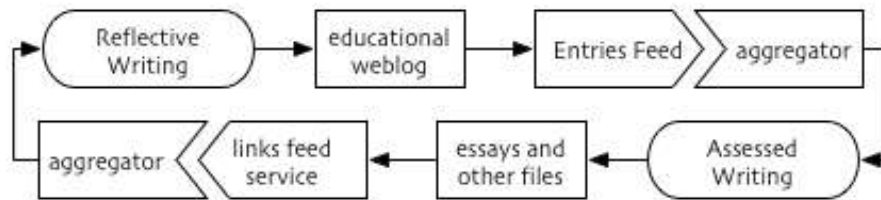


Figure 5 - Loop between two services

Generally, system architects are wary of loops in processes, as there is always a potential for ‘hunting’ behaviour occurring in automated systems, as the system oscillates between a pair of states. However, loops like the one described above avoid such conditions since they rely on human state change in two different places. More generally, such loops are perfectly normal examples of good pedagogy, and the technology needs to be developed to support them.

Another example is the typical back-and-forth seen amongst blogs, where a post in “Blog A” is collected by the aggregator of the author of “Blog B”, who generates a new post in response. This post is then collected by the author of “Blog A” who then posts a response to the response, and so on.

### 3.3.Isolated versus transactional workflows

Most existing examples of simple chains and loops in ReST-type services (for example, the various integration pairings between Flickr<sup>5</sup>, LiveJournal<sup>6</sup>, 43Things<sup>7</sup>, weblogs, Google<sup>8</sup>, Google Maps<sup>9</sup>, Amazon<sup>10</sup> and so on) are also examples of *isolated* rather than *transactional* workflows.

By this, we mean that the components of the workflow are managed without a dedicated coordinating process. For isolated workflows, each process is responsible for the next ‘link in the chain’, but there is no overall transaction that encapsulates and manages the complete workflow from end-to-end. This is a very loosely coupled type of workflow that is fairly simple to manage.

<sup>5</sup> <http://www.flickr.com>

<sup>6</sup> <http://www.livejournal.com>

<sup>7</sup> <http://www.43things.com>

<sup>8</sup> <http://www.google.com>

<sup>9</sup> <http://maps.google.com>

<sup>10</sup> <http://www.amazon.com>

Some of the more sophisticated blogging tools provide a kind of transactional workflow that manages a complete chain. Such tools draw a collection of RSS or Atom feeds into a link repository, and allow users to annotate or comment on these links or create weblog entries out of them. The result is then published as an RSS or Atom feed. This transactional workflow is generally part of the program logic rather than being a scriptable workflow, however (see section 5).

Still, even this type of “transactional” workflow is essentially stateless across the set of services in the chain. One of the possible reasons for the proliferation of these simple types of workflow is that they lack the requirement for maintaining state across a set of distributed transactions, which is something that is found in more complex workflow situations, such as learning activity management workflows.

Stateless web services, especially those providing idempotent<sup>11</sup> requests, are highly robust when compared with stateful transactions, and more suited to providing services in “mass production”, as each request can be treated in a generic fashion (He, 2003).

## 4. Learning Workflows

Educational processes can also involve the modelling of workflows. These are typically more complex than chains and loops, as they often involve the choreography of multiple agents acting in parallel rather than just a single agent acting in isolation. There are also cases where such parallel sets of workflows with separate agents must provide additional support for some form of cross-communication and choreography; we’ll consider some of these more complex cases later in this section.

Compared to most business processes, the agents involved in learning processes are more likely to be humans than static data sources or automated services, even if a learning process can sometimes be expressed at a technical level as the choreography of a set of web service interfaces.

For these reasons, learning activity management workflows are best envisaged as transactional rather than isolated. That is, the workflow is managed from a master process of some sort which has been programmed with a plan or design, and which is then responsible for orchestrating the required services. This implies that there is usually a set of states that must be maintained for the collection of parallel workflow instances, at the very least within the master process, but potentially across the complete set of distributed transactions in each workflow instance.

When workflows choreograph a set of processes that are user-driven, additional consideration needs to be given to how processes can be monitored, and how “checkpoints” can be managed.

Checkpoints within a simple automated process typically include things such as branching after validation checks. For example, not processing a reservation if the application data is incomplete or invalid.

---

<sup>11</sup> An *idempotent* request is one that is guaranteed not to alter the state of the target; constraining a service to be idempotent means that consumers and providers need not be concerned with issues such as rollback and partial transaction recovery – they can just issue the request again if it fails the first time. An example of idempotent service design would be the http GET request, although not all web developers have respected its original design parameters.

However, checkpoints within a workflow of human agents often involve more sophisticated interventions, such as providing feedback and coaching, performing qualitative evaluation and approval, sorting users into groups based on various factors, and so on.

#### 4.1. Subordinate processes

A simple example of a subordinate process is the primary use-case of the IMS Tools Interoperability activity. In this use case, a system needs to invoke a tool, which then returns processing to the launching application (Figure 6).



Figure 6 - Simple Launch Workflow

Task 1 involves dynamically establishing a session context for a user, which is under the control of the parent process. To accomplish this we need to augment the web services involved with the ability to initialise and close the process. In the Tools Interoperability guidelines, the Launch Protocol, and the Results Protocol provide this ability.

This does, however, raise one potential issue around workflow for web services: the need for service specifications to be written with workflow in mind.

For example, Pasley (2005) notes the need to construct a service definition in WSDL with BPEL extensions, before a service can be deployed in a BPEL workflow environment.

This means that anyone thinking of deploying a web service that may be used within an automated workflow really needs to consider, in advance, the needs of the workflow system when designing the service interface. Because workflow, like security, is heavily dependent on the deployment choices made in an implementation environment, generalization and standardization of “workflow-aware” services may prove problematic.

#### 4.2. Parallel processes

Another example of a common learning activity management workflow involves a group of people being given an individual task to complete before engaging in the next activity.

The workflow includes a checkpoint between the two types of tasks, which requires that all the individual tasks have been completed (Figure 7).

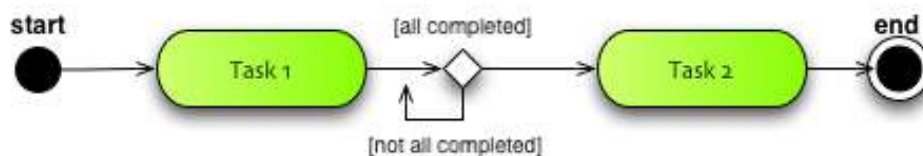


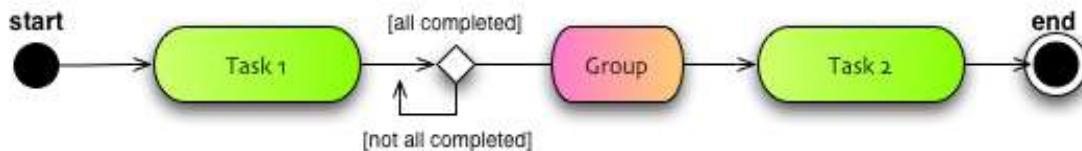
Figure 7 - Concurrent chain workflow

This is considerably different from the previous model, because now we have the concept of “all concurrent processes” not just “this process”. This requires that the workflow engine is keeping track of the state of all its “sessions” and can judge when all of the Task 1 instances reach the ‘completed’ state.

### ***4.3. Multiple parallel processes with grouping***

As well as parallel individual tasks, we can also envisage group tasks. For example a group of 20 students may be asked to individually read a set of excerpts. Then, when everyone has finished, the tutor might assign each student to one of four groups that are to discuss the excerpts and answer some questions.

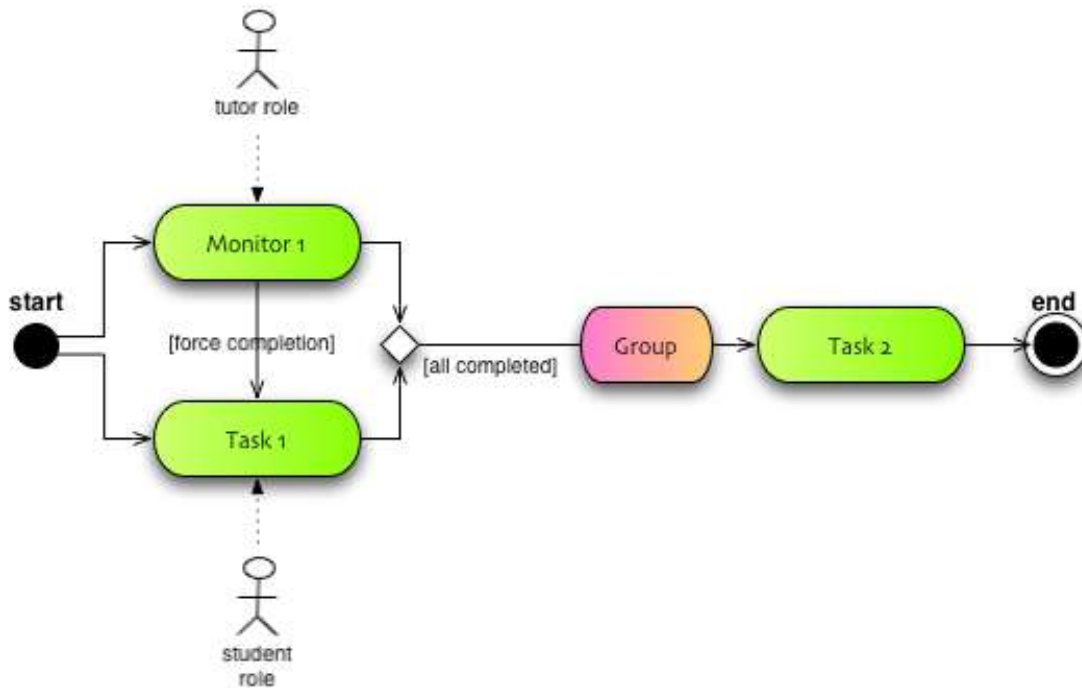
In this scenario, Task 2 is defined as a group activity, so we need to add a grouping process to the model to show that we’re doing something different with the processes that are being instantiated (Figure 8). Task 1 here would have 20 parallel instances, while Task 2 would have four parallel instances. Depending on the scenario, the grouping may be done manually by the tutor or by student opt-in, or automatically, either by random selection, or to provide a particular mix of students within each grouping.



**Figure 8 - Concurrent flow with grouping**

### ***4.4. Multiple parallel processes with monitoring and intervention***

In practice, rather than just let the workflow wait until everyone has completed Task 1, we need to provide an intervention capability, so that the tutor can force everyone to the completion state. We may be in a classroom situation and only have an hour to work within, or someone may have given up on the task and wandered away from the computer. In any case, it’s a desirable behaviour for an educational workflow, so we need to consider it in our model (Figure 9).



**Figure 9 - Monitoring and intervention**

This sort of workflow should look fairly familiar to anyone who has used a learning design tool such as LAMS<sup>12</sup>. In the lab example discussed above, this enables the tutor to move the group along at an appropriate pace, speeding up the transitions between tasks so that the fastest students don't get bored, while enabling the tutors to identify struggling students so that they can provide additional coaching.

Communication between processes within a workflow has a lot of uses in an educational context; for example, to hint, guide, or coach a student through an activity, or to facilitate a group.

This does, however, require us to think outside a single-process model of workflow and to consider workflows with inter-process communication, with all the issues that brings with it.

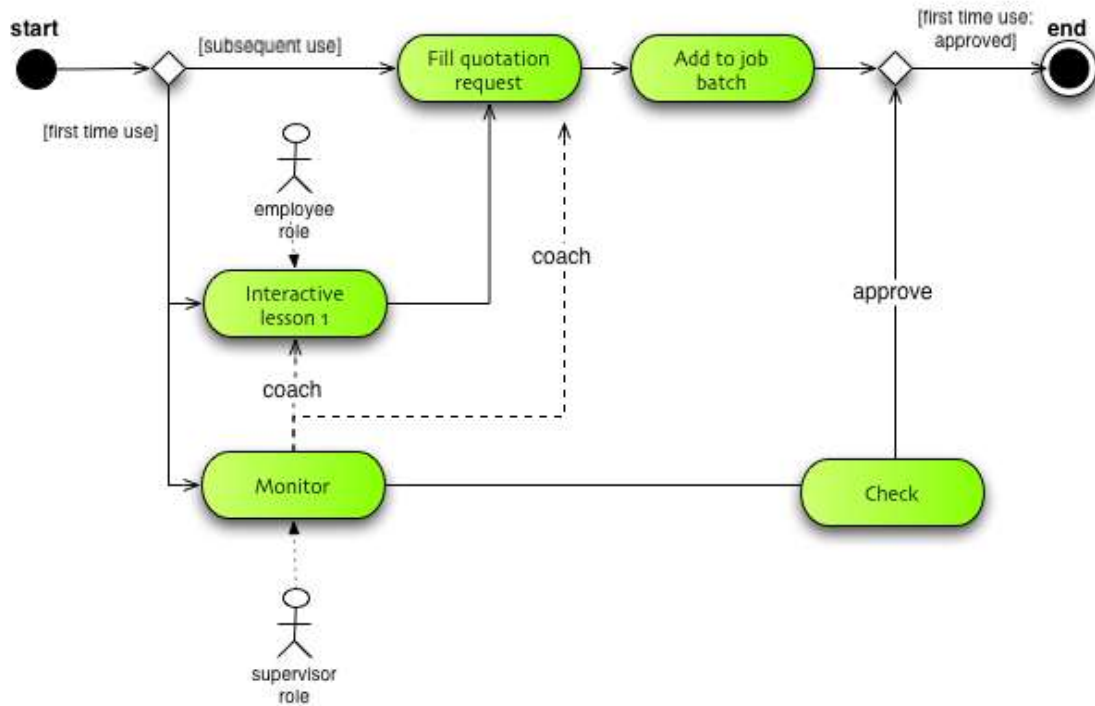
For example, we may have cases where a user in one process may intervene in the workflow state of another process (such as the tutor forcing the completion state, in the example above). However, we may also have cases where a user in one process needs to directly intervene within another process; for example, a tutor entering into a group discussion and adding a provocative question, or entering an individual assessment task and triggering hints and suggestions for a student who is struggling.

#### **4.5. Combined business and learning workflows**

Another perspective on workflow considers the role of e-learning processes within broader business workflows. This leads to multiple parallel processes of a different kind. For example, the process of learning about customer quotation preparation could be incorporated into the workflow that manages customer quotations itself. Such a flow can

<sup>12</sup> <http://www.lamsinternational.org>

be interrupted for just-in-time e-learning to be initiated, or for first-time preparation and coaching (Figure 10).



**Figure 10 - Workflow integrating business and e-learning processes**

Ultimately, however, the goal of workflow learning is to make the learning and the work process a single integrated whole (Cross, 2005), so that learning and business decision-making are indistinguishable (Figure 11).



**Figure 11 - Workflow Learning (image © Cross, 2005)**

While not currently a major concern within higher education, there is considerable interest in the potential of “Workflow Learning” within the commercial sector. Unlike learning flows as envisaged within a typical educational context, workflow learning is concerned with the modelling and orchestration of the whole organisation rather than discrete learning “events”; indeed, one of the criticisms of workflow learning is that discrete training activities create problems of transitioning experiences successfully from learning to actual tasks (the “forgetting problem”).

Workflow learning is therefore more likely to involve the integration of learning within

the design of teams, environments, performance and decision support services, and so on, than the design of any sort of learning activities that stand alone.

Creating a SOA with workflows that seamlessly merge business processes and e-learning is going to be one of the more challenging problems of e-learning, and will require a range of techniques for modelling and executing workflows involving both automated and human processes, as described in the previous sections.

## **5. Describing and specifying workflows**

Workflows are an important topic in system architecture, and we have seen the emergence of a number of standard XML languages for expressing workflow concepts, such as BPEL4WS, ebXML, and XPDL. The e-Learning community has also produced the IMS Learning Design specification for learning activity management workflows.

### **5.1. Encapsulation in program logic**

The simplest method for managing a workflow is to encapsulate the flow within program logic. For simple chains this is probably the most appropriate mechanism. The disadvantage is that the workflow is difficult to modify or reuse, but this need not be a concern in many situations (see section 3.3 for an example).

### **5.2. BPEL4WS: Business Process Execution Language For Web Services**

Originally developed by IBM, BPEL4WS (OASIS, 2005) is a language for modelling business processes. BPEL4WS has since been taken over by OASIS for further development and ratification as an open specification.

BPEL4WS provides a sophisticated language for expressing business processes that involve the interaction of many different services offered by the various partners involved. BPEL4WS can express decision logic and error handling, and has a number of sophisticated constructions for expressing a workflow. BPEL4WS is essentially a scripting language for web services orchestration, supporting conditional and parallel interactions amongst web services (Ma, 2005).

BPEL4WS requires that the workflow processor component will manage state information on behalf of the various orchestrated services, so that properties can be set across multiple processes and used for branching decisions.

However, the specification is primarily aimed at automating processes involving machine agents, and does not really address collaboration or involvement of people in the business process. Most of the learning processes described in Section 4 cannot be represented at all in BPEL4WS.

Furthermore, there is some scepticism about the ability of BPEL4WS to manage workflows that cross organisational or domain boundaries: according to Ma (2005), “for now, organizations can use BPEL4WS in controlled, internal transactions—those that don’t traverse the public Internet or share data with anonymous entities”. This is partly because of problems of trust and identity management between services, but also because of differences in data management, quality processes, and reliability.

### **5.3.ebXML: e-Business XML**

e-business XML (ebXML, 2005) is a UN/CEFACT standard for electronic business, superseding legacy technologies such as EDI (Electronic Data Interchange).

ebXML contains a whole range of specifications, including BPSS – Business Process Specification Schema, which is used to represent the business process realised using an ebXML exchange of messages. This works with CPPA – Collaboration Protocol Profile and Agreement, which is used to represent the contract between the parties.

ebXML is a mature and proven technology for managing business workflows, although new implementations seem to be favouring BPEL4WS over ebXML.

### **5.4.XPDL: XML Process Definition Language**

Another workflow language, XPDL (XPDL, 2005) is somewhat “lower level” than BPEL4WS. It is primarily aimed at interchange of workflows between workflow management tools, particularly in an EAI context.

XPDL is quite mature, but does not have the sort of mindshare that BPEL4WS or ebXML currently enjoys.

### **5.5.IMS Learning Design**

IMS Learning Design (IMS, 2003a) is a specification that attempts to model learning scenarios involving multiple users, and includes some workflow-like constructs. IMS Simple Sequencing (IMS, 2003b) and IMS QTI (IMS, 2003c) also include some workflow modelling aspects, though limited to a single-user process.

Like BPEL4WS, IMS Learning Design provides mechanisms that can be used for branching, conditions, and roles, and has a mechanism for managing state within the workflow. Unlike BPEL4WS, IMS Learning Design is also concerned with flows within web content.

Another difference between the two specifications is that IMS Learning Design does not provide any mechanisms for service activation and initialization, and in that respect could be seen as a more “abstract” workflow language than BPEL4WS. Dalziel (2005), for example, characterizes Learning Design at three levels – as theory, standard, and software – and considers the possibility of using other workflow standards to realize the theoretical model of IMS Learning Design.

In the current IMS Learning Design specification, the role of resolving, initialising, and calling services is delegated to the workflow management system (also referred to as the “runtime”).

Because IMS Learning Design only considers services in the abstract (as in “a conference service”) rather than concretely as specific services with locators and initialization parameters, the workflow management service is tasked with handling failover situations when suitable services cannot be located, or cannot be initialized. For example, while a flow can be designed using IMS Learning Design that states that a discussion forum should be launched at a particular point within the flow, it cannot be used to determine when the forum should be terminated, or what the initial threads or topics are that should be presented, or whether the results of discussion should be exported elsewhere.

Learning Design’s state model (the user “folio”) also has no provision for security or

privacy of user attributes, and assumes that the workflow management system provides such capabilities.

## 6. Workflows and security

In some cases the services utilized within a workflow will be deployed within different security domains. The critical issues with cross-domain workflows concern how and when state and context information is passed from one process to another, and how cross-process interaction can be handled safely.

To take the first point, its quite common for a process to need to know some attributes about the user for whom the process is being instantiated. When the process is part of a workflow within a single domain, its quite normal to pass this information along with the instantiation context, or to provide a fairly simple API by which the delegated process can acquire this information. This is how the plug-in architecture of Blackboard's Building Blocks<sup>13</sup> system and the LAMS tool architecture operate.

When transferring the user context between domains, this is a more problematic issue, as we now need to ensure that (a) the party disclosing the information is entitled to do so, and (b) the party receiving the information can verify that the disclosing party is a valid source for the information.

For example, the Shibboleth<sup>14</sup> and Liberty Alliance<sup>15</sup> reference models of the SAML<sup>16</sup> specification provide mechanisms that can be used to safely disclose attributes across domains while respecting the privacy needs of users. New initiatives such as LID<sup>17</sup> and Microsoft Infocard (Ernst, 2005) also provide a model based on the selective disclosure of attributes by users.

Not all workflows and services will need such user context initialization; many services, in fact, are both public and stateless.

Managing cross-process interaction in a safe manner is a complex problem, as in addition to initializing each process, we now have the situation where one process can take control of its peers given sufficient "privileges"; for example, a process in the "tutor" role being able to force "student" processes to terminate on command.

This distributed state problem is a complex one, and very few of today's systems exhibit this kind of behaviour, which is perhaps why we've seen instead a proliferation of stateless workflow applications instead (see Section 3.3).

## 7. Conclusions

In this paper we have considered a variety of workflows, from very simple stateless, single-agent chains, through to complex learning activity workflows incorporating multiple agents, branching, grouping, and cross-process interaction. We have also looked at some of the available technologies for implementing these workflow types. So what is possible and feasible in this area within the short to medium term?

Simple workflows such as chains and loops are already in evidence using very simple

---

<sup>13</sup> <http://www.blackboard.com>

<sup>14</sup> <http://shibboleth.internet2.edu/>

<sup>15</sup> <http://www.projectliberty.org/>

<sup>16</sup> [http://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=security](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security)

<sup>17</sup> <http://netmesh.info>

web service specification such as Atom, and this may represent the “low hanging fruit” in this area, and where the most practical applications of service workflows will be developed over the next year. The technologies involved are sufficiently mature, within their limited scope, to enable the deployment of production-capable workflows.

Stateless, preferably idempotent, workflow chains are reliable, easy to deploy and manage, and can be readily combined. They match well with the “simple, sloppy, standards, scalable” technology success criteria defined by Google’s Adam Bosworth (2005).

Security within simple stateless workflows is also achievable using existing mechanisms for securing Internet communications, such as https.

For the learning workflows, especially those involving multiple parallel processes, the solutions are more complex, and the available standards and specifications lack both the scope and the maturity to enable the deployment of production services with any confidence outside the boundaries of self-contained applications or highly controlled environments.

Taking each current workflow technology in turn:

- § The Tools Interoperability work, while a useful starting point, is not sufficient, in its current scope, for handling parallel processes with cross-process collaboration, such as monitoring and intervention. It is also very lightweight even in respect of single-agent workflows, and can be characterized presently as being a launch and termination protocol for a single service, rather than constituting a full workflow model. The activity also does not address cross-domain security and privacy issues, but instead assumes trust exists between systems, and that there is implicit consent to share personal information.
- § Learning Design is not sufficient in and of itself for managing the lifecycle of processes that are subordinate to the learning activity management process itself. The specification needs to be combined with a sophisticated interoperability framework for learning design tools.
- § Generic web services workflow languages such as BPEL4WS are not intended for managing processes involving orchestration of user processes (especially multi-user processes) and cross-process interactions. Therefore, they may not be suitable yet for cross-domain processes on their own.

Security is also going to be a recurring issue for workflows that cross domains, and in particular the privacy issues that emerge from sharing user’s attributes may require a substantial effort in providing distributed selective attribute disclosure technologies.

Until the issues of cross-domain security, cross-process interaction, and selective disclosure are addressed, the only place in the short term for complex learning activity workflows at a production level is within highly controlled environments, such as within a single system or collection of components within a single security domain.

## **8. Acknowledgements**

The author wishes to thank James Dalziel and Wilbert Kraan for their comments on early drafts of this paper.

## 9. References

- Bosworth, A., 2005, "A Web of Data", Powerpoint presentation, [http://www.sdforum.org/images/events/presentations/Google\\_AdamBosworth.ppt](http://www.sdforum.org/images/events/presentations/Google_AdamBosworth.ppt)
- Cross, J., 2005, "On demand, in the soup, and on the way to glory", Workflow Institute Weblog, <http://workflowinstitute.blogspot.com/2005/03/on-demand-in-soup-and-on-path-to-glory.html>
- Dalziel, J., 2005, "From re-usable e-learning content to re-usable learning designs: Lessons from LAMS", EduCause 2005, <http://www.lamsfoundation.org/CD/html/resources/whitepapers/Dalziel.LAMS.doc>
- ebXML, 2005, "e-Business XML", <http://www.ebxml.org/>
- Elkarra, N., 2002. "SOAP vs. REST", <http://www.bawsug.org/archives/000003.html>
- Ernst, J., 2005, "What is Microsoft InfoCard?", <http://netmesh.info/jernst/2005/05/13#what-is-msft-infocard>
- Hanson, D., 2005 "Web services are going lowercase", 37 Signals, 1<sup>st</sup> June 2005, [http://www.37signals.com/svn/archives2/web\\_services\\_are\\_going\\_lowercase.php](http://www.37signals.com/svn/archives2/web_services_are_going_lowercase.php)
- He, H., 2003, "What is Service-Oriented Architecture?", XML.com, 30<sup>th</sup> September 2003, <http://webservices.xml.com/pub/a/ws/2003/09/30/soa.html.webloc>
- IMS Global Learning Consortium, 2003a, "IMS Learning Design Specification, v1.0", <http://www.imsglobal.org/learningdesign/index.html>
- IMS Global Learning Consortium, 2003b, "IMS Simple Sequencing Specification, v1.0", <http://www.imsglobal.org/simplesequencing/index.html>
- IMS Global Learning Consortium, 2003c, "IMS Question and Test Interoperability Specification, v1.2.1", <http://www.imsglobal.org/question/index.html>
- Ma, K., 2005, "Web Services: What's Real and What's Not?", IEEE Internet Computing, May-June 2005.
- OASIS, 2005, "Business Process Execution Language", [http://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=wsbpel](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel)
- Pasley, J., 2005, "How BPEL and SOA Are Changing Web Services Development", IEEE Internet Computing, May-June 2005.
- XPDL, 2005, "XML Process Description Language", <http://www.wfmc.org/standards/XPDL.htm>