

Service-Oriented Frameworks:

Modelling the infrastructure for the next generation of e-Learning Systems

A Paper prepared on behalf of DEST (Australia), JISC-CETIS (UK), and Industry Canada

Principal Contributors:

Scott Wilson, Centre For Educational Technology Interoperability Standards
 Kerry Blinco, Department of Education, Science and Training (Australia)
 Daniel Rehak, Carnegie Mellon Learning Systems Architecture Lab, July 2004

1 Introduction

Technology is becoming increasingly embedded in the systems and processes of our educational organisations. Research, learning and teaching and information management now rely on technology to support processes. Although technology has the potential to extend and improve educational and training activities, this potential can only be fully realised if the activities are built upon a stable and coherent technical infrastructure.

The active involvement of the United Kingdom's Joint Information Systems Committee (JISC), the Australian Department of Education Science and Training (DEST) and Industry Canada (IC) in research into e-learning systems and the development of international learning technology standards and specifications, and the US Advanced Distributed Learning Initiatives (ADL) activities has highlighted the potential of technical frameworks to provide a common basis for designing e-learning infrastructures.

Organisations such as JISC, DEST, IC and ADL along with many others have already made a considerable contribution to the development of e-learning in their own communities and in supporting interoperability standards. Each of these organisations has identified the need to produce a coherent vision of how to integrate systems to support organisational and cross-organisational processes for enabling effective e-learning.

Much of work on developing integrated systems has relied upon frameworks based on a service-oriented approach which provides both direct and indirect benefits to teachers, learners, administrators, organisations and those who supply and develop content and software to the respective communities.

A framework consists of a set of services, each of which may be defined at different levels of detail. The core task of creating a framework is to define a broad set of services required to support the business of a community. A service offers functions and content through agreed behaviours and interfaces.

This paper explains the potential benefits to the e-learning community of adopting a service-oriented frameworks approach to infrastructure development, and the additional activities required to realise these benefits¹.

1.1 Who should read this paper?

This paper was developed for:

- Decision and policy makers for e-learning system

development, including the activities of defining or selecting standards and specifications.

- Decision and policy makers for IT/ICT systems that interoperate with e-learning systems.
- e-Learning system technical architects.

1.2 What is the purpose of this paper

This paper aims to

- Provide a common set of concepts and terms that can be used in conversations about e-learning infrastructure
- Present a case for developing and maintaining service-oriented e-learning technical frameworks
- Enable the audience to understand the nature of framework and its components
- Describe the supporting activities that are required to realise the benefits of a framework
- Examine some possible next steps in developing an international e-learning technical framework as a collaborative activity

2 Understanding the value proposition

If an e-learning framework is worthwhile it must lead to benefits for teachers, instructors and learners, for organisations, that deploy e-learning and all those who access and supply e-learning services and content. It is anticipated that a service-oriented approach will provide benefits that can be grouped as follows:

2.1 Benefits to Policy Makers

An e-Learning Framework:

- Provides a platform for policy makers by offering a coherent vision of how to integrate systems to support organisational and cross-organisational processes to enable effective e-learning supporting planning activities.
- Facilitates communication of technical policy through a single cohesive overview of technical policy.
- Allows diversity with cohesion through enabling implementations to leverage widespread practice without

¹ In the remainder of this document where ever the term "framework" is used with out qualification, it represents the concept of a service-oriented framework

compromising specific community or organisational requirements.

- Supports planning for technical and interoperability specifications and standards development. A framework can be used to document the state of the specifications and standards required to support e-learning in the community of practice. Activities and gaps can be identified, providing a focus for decisions about priorities and resource allocation.

2.2 Benefits to organisations that deliver and manage learning/training

An e-Learning Framework:

- Driven by business processes, services are aligned with business processes and support the business model. Analysis of the business processes determine what services are needed rather than the services in the framework defining what processes are implemented. The resulting services enable business processes to be realised
- Supports adaptability of the organisation. Organisation and business process changes can be reconfigured to meet changing operational requirements or to reflect organisational change.
- Provides a modular and flexible technology base. The specific rationale for a framework is to enable the development of composable modular and flexible systems, where the individual components can be added or replaced more easily than in traditional models and where entire new systems or subsystems can be composed from the collection of available services.
- Makes collaboration between organisations easier through defining the behaviours and processes which are needed to share information between organisations. It may also make sharing of applications easier, as it will be simpler to define small applications which can be deployed to meet the common needs of each organisation.
- Provides better returns on technology investment. Applications can be developed or acquired as needed, which means that only those parts of the system that really need to be changed are replaced retaining the rest of the systems so reducing both purchasing and implementation costs, particularly in terms of staff development and training. Leveraging standardised interfaces and component behaviours, systems can integrate with each other regardless of source thus allowing organisations to choose the most suitable application for their purposes.
- Enables faster deployment of technology. As components are independent entities it will often be easier to deploy new components as long as the needs of the new components are compatible with the existing interfaces. Even where this is not the case it may still be simpler to alter or replace other components to supply the requirements of new systems.
- Supports “buildability”. Well defined behaviours and interfaces allow software components to be developed independently, and ideally these components are reusable granular building blocks.
- Strategy independence. This enables organisations to make choices from a wide variety of development strategies and to

combine different strategies, as appropriate, for each component. It allows institutions to experiment with new methodologies and assists new developers/vendors by reducing the risk at a single decision point.

- Supports durability. Services are defined via their behaviours and interfaces and thus can be used without knowledge of the internal workings of the component providing the service, allowing components to be replaced without causing widespread disruption.
- Supports transportability. Because service interfaces have agreed interfaces, applications can be more easily implemented at another organisation or utilised by other applications in the organisation.
- Supports staged implementation. A framework can be used to provide a “Road Map” that allows organisations to implement their solution in stages. Organisations do not need to implement infrastructure all at once if modelled using a service-oriented framework. They can choose to implement within a heterogeneous environment the standards based interactions that are appropriate for their infrastructure.
- Protects legacy investment. Legacy systems can be integrated into a service-oriented infrastructure by providing a services layer over the top of the existing applications.

2.3 Benefits to Communities of Practice

An e-Learning Framework:

- Supports pedagogic diversity. It becomes possible to support a very diverse set of learning models through having the capacity to configure the low-level elements of the learning architecture to fit a variety of pedagogic models.
- Enables pedagogy-driven implementations. By exposing modular processes as separate services, which can be configured in multiple ways, the construction of technology solutions can become driven by pedagogical imperatives, rather than the reverse.
- Faster response to community needs. New applications can be created by combining existing services in new ways in response to community needs. Often these new applications can be developed relatively quickly and within existing budgets allowing shorter times between identification of a requirement and implementation.

2.4 Benefits to Suppliers and developers of e-learning services and content

An e-Learning Framework:

- Increases Return on Investment (ROI). By providing access to functionality rather than user interfaces, data applications can be developed that relate better to the tasks to be performed without duplicating the functionality of existing systems.
- Lowers the cost of entry. The granular size of the components, clear behavioural and interface definitions, the availability of toolkits, and lightweight development environments should all facilitate the implementation of open standards and allow easier connectivity to enterprise systems.
- Supports market differentiation. Describing systems in terms of

well-defined service behaviours means there is a more “level playing field” for vendors, which should encourage the entry of small innovative players into the market. Vendors can concentrate on those parts of the infrastructure that they specialise in without having to try to meet all of an organisation’s needs.

- Provides a common understanding between suppliers and purchasers / consumers.

3 Service-oriented approaches

In recent years there has been a shift from monolithic application silos towards service-oriented approaches where flexible granular functional components expose service behaviours accessible to other applications via loosely coupled standards-based interfaces. In the strict sense, this approach is known as Service Oriented Architectures (SOA’s) which includes an assumption of delivery by Web Services. However a less formal service oriented approach (small “soa”) has usefully adapted the general approach of loosely coupled composable services whilst allowing for greater flexibility in the technology of implementation².

Service-oriented approaches have been around for sometime, but are rapidly gaining popularity with the wide adoption of Web Services, and because of the lower costs of integration coupled with flexibility and simplification of configuration. Service-oriented architecture builds upon the experience of using Web Services for integration.

In a service-oriented approach, the application logic (behaviours) contained in the various systems across the organisation – such as student record systems, library management systems, Learning Management Systems (LMS) / Virtual Learning Environments (VLE) directories and so on – are exposed as services, which can then be utilised (consumed) by other applications. For example, a student record system may expose services defining enrolment and registration processes and related information, which can then be used by a LMS/VLE or library system.

This approach is somewhat different to two other common ways of integrating systems, which are to integrate at the user interface level using portals, or at the data level by creating large combined datasets or data warehouses. A service-oriented approach does not preclude also using portals or data warehouses, and is in fact agnostic about how the rest of the enterprise is configured, which is why it makes a good approach for a framework.

4 Key concepts

In developing this paper, the term “Architecture” has been deliberately avoided to prevent any confusion as the term is used by different communities to describe many different views of an e-learning system or infrastructure, in both general and technical contexts (though not with a consistent technical interpretation).

Factoring of the key concepts of Framework, Reference Model, Design and Artifact, has been made with reference to the work of the ‘software patterns’ community (derived from the writings of architect Christopher Alexander). However the actual terms adopted to represent the concepts as factored have been chosen because their meanings will be more intuitive to the e-learning community than the equivalent tokens in the pattern community.

² A service-oriented approach would in fact allow a service to be delivered by a “human” service provider through a manual

interaction process.

4.1 Framework

A framework creates a broad vocabulary that is used to model recurring concepts and integration environments and is equivalent to the concept of a pattern in the software community:

“The goal of patterns [Frameworks] within the software community is to create a body of literature to help software developers resolve recurring problems encountered throughout all of software development. Patterns [Frameworks] help create a shared language for communicating insight and experience about these problems and their solutions. Formally codifying these solutions and their relationships lets us successfully capture the body of knowledge which defines our understanding of good architectures that meet the needs of their users. The primary focus is not so much on technology as it is on creating a culture to document and support sound ...design.”³

A framework supports the development by organisations of their own implementation infrastructures, using a flexible service-oriented approach.

The ultimate aim of a Framework is, for each identified Service, to be able to reference one or more open specifications or standards that can be used in the implementation the Service.

A framework can support a large number of unique organisational infrastructures that are still coherent and consistent with respect to one another.

A framework does not aim to build a generic LMS/VLE, or any other learning technology systems, in fact one of the primary goals of a Framework is to encourage “coherent diversity”, by providing common Service definitions which can then be used to meet the diverse goals of the organisation.

As an analogy, the framework provides a vocabulary and grammar and it is left up to individual organisations to write the stories.

Although a service-oriented framework provides support for organisations developing service-oriented architectures, it does not presume that all organisations will want to do so, or that those who do adopt this approach will want to do so across the whole of the organisation.

4.2 Reference Model

A Reference Model is a selection of Services defined in one or more Frameworks together with rules or constraints on how those Services should be combined to realize a particular functional, or organisational goal. A Reference Model constrains the number of unique organisational infrastructures.

For example, within a Framework for digital repositories, a Reference Model for supporting distributed resource discovery may specify how Services such as 3 Appleton, Brad, 2000, Patterns and Software: Essential Concepts and Terminology

www.bradapp.net/

“Harvest”, “Search”, “DRM”, “User Preferences” and “Metadata Management” are combined to support a specific discovery strategy that can then be implemented in multiple Designs.

²A Service-oriented approach would in fact allow a service to be delivered by a human service provider through a manual interaction process

³Appleton, Brad 2000, Patterns and Software: Essential Concepts and Terminology www.bradapp.net/

4.3 Design

A Design specifies an Artifact, such as a piece of software, and is a collection of specific technologies applied to either a Reference Model or the Framework. The choice of technology may apply additional constraints.

A Design may incorporate elements derived from Reference Models and Frameworks. For example, the Search Service specification within a Design may use the definition from a Framework, or it may be part of a workflow in the Design that follows a Reference Model.

4.4 Artifact

An Artifact is a realization of a Design. An Artifact may be a piece of software, a workflow or process, a combination of software components, a piece of content, or anything else that can realize a Design. Within an organisation the artifacts implement an infrastructure (process and technology) to meet the communities agreed upon e-learning goals and operations.

5 Relationships between Frameworks, Reference Models, Designs, and Artefacts

Frameworks, Reference Models, and Designs build upon each other in a top down fashion in potentially complex ways.

A simple example is where a Framework is used to derive a Reference Model, which is used in a particular Design which then results in an Artefact, such as a piece of delivered software. This is shown graphically in Figure 1.

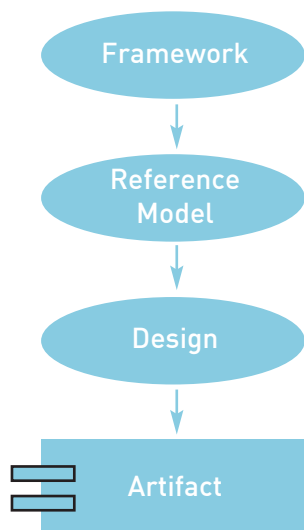


Figure 1: Simple relationship model

An example of such a scenario would be where a Framework for digital repositories provides a vocabulary of Services, from which a Reference Model is derived describing content management and retrieval based on the Services defined in a Framework. An analyst then creates a Design that implements the Reference Model, for example a repository management application, which is then realized in code by the programming team as the final Artifact.

This simple model is however only one of several possible scenarios. In practice, it is quite likely that in practice multiple Reference Models will be drawn from any Framework in order to provide common solutions to typical business problems, such as content management, collaborative learning, or managing the

lifecycle of courses. More complex Designs may then draw upon multiple Reference Models. This scenario is shown graphically in Figure 2.

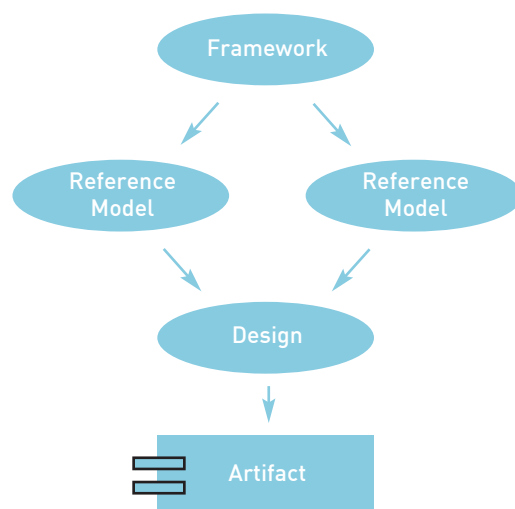


Figure 2: Multiple Reference Models derived from a single Framework

An example of this type of scenario would be where two Reference Models – one which defines the content management lifecycle and one which defines the course creation and accreditation lifecycle – are both used to create a Design for an application that supports a complete course development workflow.

An overall Framework is expected to grow as different behaviours are integrated into it.

The inverse of this sort of model is one where a single Reference Model is derived from more than one Framework. An example of where this would occur would be where the subject of the model crosses over between domains. For example, a Reference Model which describes how to integrate from remote sensor feeds into collaborative learning may need to draw upon Services defined both within a Framework for e-learning, and a Framework to support e-science.

This scenario is shown graphically in Figure 3.

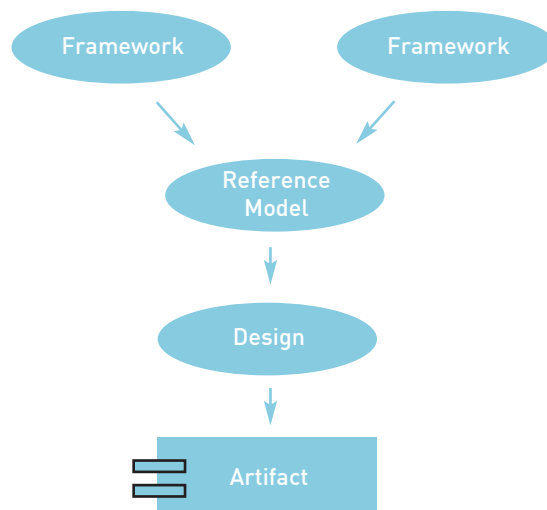


Figure 3 A Reference Model derived from multiple Frameworks

All the possible relationships between Frameworks, Reference Models, Designs, and Artifacts can be enumerated to derive a few principles of relationship:

- Designs can be derived directly from one or more Frameworks without the need for a Reference Model
- Designs can be derived from any number of Reference Models
- Reference Models can be derived from any number of Frameworks
- An intermediate step in combining two Frameworks in a reference model may be to develop a new combined internally consistent framework
- A Reference Model may be consistent with more than one framework
- A Reference Model need not cover the full extent of the framework from which it derived
- Multiple Reference Models derived from the same framework will be consistent, but their Artifacts may not be interoperable
- Any number of Artifacts may be derived from a Design (for example, different instances, or different versions of an instance)

Even in a complex scenario with multiple reference models, it is worth bearing in mind that all the resulting Artifacts should, as a result of being ultimately based upon a common Framework, display a strong degree of coherence in their behaviours, which would be apparent to anyone familiar with the Framework, even though their Designs and implementations may be very different.

Being able to accommodate this type of complexity is essential in realising the goal

Service Oriented Frameworks of delivering processes and systems that support the range of activities that organisations engage in that may need to integrate with e-learning, such as e-science, administration, finance, human resources and marketing.

6 Constructing Frameworks

A Framework consists of a set of Services, each of which may be defined at different levels of detail. The core task of creating a Framework is to define the broad set of Services that need to be defined. This process is called the factoring services.

In general, the more fine-grained the Services, the more flexibility there is for Reference Models and Designs to group Services for a particular purpose.

Factoring is likely to be an ongoing process for organisations developing a Framework, rather than a single unit of work, as experience informs the choice of Services, identifies gaps, and indicates that Services require splitting or joining. There is no current "best practice" for factoring Services within a Framework, but one rule of thumb is to define Services that are fine-grained in overall scope, yet whose operations are still capable of encapsulation for efficient remote access.

There is no current "best practice" for factoring Services within a Framework, but one rule of thumb is to define Services that are fine grained in overall scope and define a single behaviour or step within a composable business process.

7 Defining Services

In the context of a Framework the term Service refers to a pattern that can be used to solve a particular type of problem. A Service

may be realized in a number of ways, including as a Web Service, but also as an API, or as a manual business process. The key characteristic of a Service is that it is a common way of fulfilling a particular task, or to provide a specific behaviour to which different technology-specific specifications can be applied.

From the point of view of a Framework, an important characteristic of this service-oriented approach is that there is no concern with how a Service is implemented or the processes it is used in. Instead, the concern is only with the set of defined interactions that a Service supports. This functional focus provides the capacity to be specific about the range of expected behaviours of an individual Service, while remaining agnostic with regard to implementation technologies, choice of user interface type and overall design of particular solutions. These decisions will be taken during Design stages and depend on functional and deployment requirements.

Services can be characterized within a Framework at several levels:

- As a definition of function and scope
- As an abstract model of data and behaviour
- As a data representation specification, using XML for example
- As an Application Programming Interface (API) specification
- As a Web Service specification, using WSDL for example

Taken together, this set of information provides sufficient detail to create fully interoperable Web Service implementations of the Service. However, not all Services will have all of these levels of detail available.

The relationships between these types of Service specification are shown graphically in Figure 6. Ultimately, all aspects of a Service are derived from its functional definitions, which will tend to be expressed in natural language rather than in a programming construct.

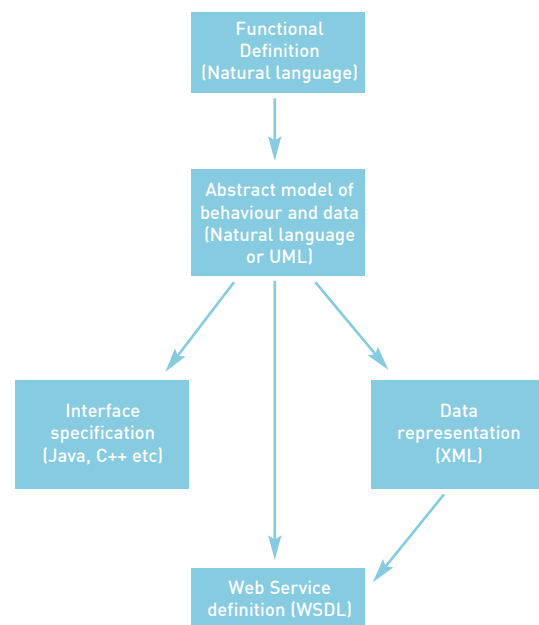


Figure 4 Deriving the parts of a Service definition

From this functional definition, an abstract model may be developed that sets out in a more formal manner the expected behaviour of a realization of the Service, and the structure of the information it deals with. This can be expressed using natural

language, however the Unified Modelling Language (UML) is an extremely useful tool for developing this type of model.

8 Service-level Abstractions

From the abstract model of the Service, it is then possible to derive XML schemas that define data to be exchanged, and WSDL definitions to enable the Service to be implemented as a Web Service and APIs for particular programming languages. (Note that WSDL definitions also rely upon XML representations of data, hence the relationship between the two).

By mapping these parts to the structures needed to enable integration between two parties using a Service, the result is the model in Figure 5.

Both parties need a shared understanding of a common business process for the Service, with a shared formal model of the behaviour and data. This model is realized in an application with an Interface to access a Web Service that has commonly agreed operation definitions (WSDL) and data structures (XML schemas). This Web Service can be used to integrate with the other partner across a shared Messaging infrastructure (for example, using SOAP⁴, or REST⁵).

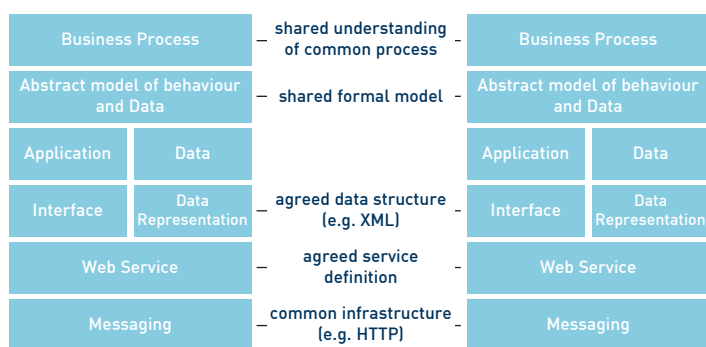


Figure 5 Integration of business partners

Note that in order to integrate the two systems in this model it is necessary to agree to a range of specifications at different degrees of abstraction. However, some aspects of the systems may remain proprietary to each partner, including the data, the application code, and the interface (API) used to integrate the application code with the Web Service.

In practice, common APIs can have a beneficial role in sharing development efforts and producing reusable code, especially where Web Services is only one of several options for integration between parties.

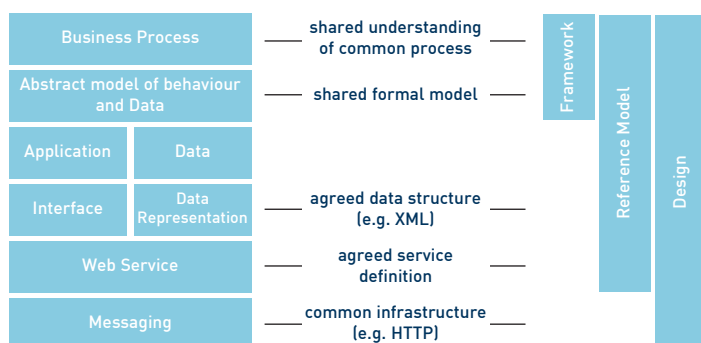


Figure 6 The roles of Frameworks and Reference Models in specifying agreements

Although all these agreements need to be in place to support integration, it is not necessary for a Framework to define all these agreements. It is reasonable to assume that for some Services, agreement in a Framework may be limited to a fairly abstract level, and the details of data representation and Web Services definitions to be specified by a Reference Model (see Figure 8).

9 The Role of Standards in Service Definitions

Technical standards have an important part to play in Service definitions and agreements. Given that the idea is to enable people implementing systems to make use of components that can be selected from a variety of sources, standards are critical to ensuring that components will work together. Without standards for each of these, each agreement between producing and consuming Services would have to be ad hoc which rapidly become unsupportable.

Because the specification development process in e-learning has to date been focused on data interchange specifications that do not include behaviours, almost all the domain Services for e-learning need to be defined.

Early adopters are often implementing ahead of standards and the evolution of best practices. To this extent a well defined and highly granular Framework with high level Service interface definitions would be of great assistance to these early adopters.

10 Frameworks are not enough

Frameworks are intended to lead to the development of compliant artifacts. Early experience with implementing Services indicates that take up is encouraged by making it easy for applications to expose and consume defined Services and that it is important to feed back implementation experience into the Framework development and specifications processes. Strategies that encourage adoption of Frameworks and Reference Models include:

- Prototype new specifications. The development of new specifications required to complete the definition of a Framework should be informed by the development of prototype implementations so that obvious implementation issues can be identified and resolved before release of the specification.
- Provide toolkits or code libraries that can be integrated into developments or overlay existing implementations. Toolkits enable organisations to build solutions, and provide assistance to both the commercial sector and the open source community in providing applications that operate within organisational infrastructures.
- Maintain registries of tools and resources. Resources that are of assistance to implementers include: information on open-source toolkits developed, lists of prior work of value, such as projects and case studies, or demonstration projects using the Service, information about Services offered on an ASP basis and, general guidance on creating, exposing, and consuming Services.

⁴SOAP is a specification for packaging XML data representations for transport across a network using a protocol such as HTTP. This is usually combined with the Web Services Description Language (WSDL), which can be used to define the types of messages that the Web Service supports, such as "getStudentRequest" or "findLearningObjectResponse".

⁵REST is an alternative approach to SOAP for deploying web services, using simple HTTP messages with defined semantics for existing commands such as GET, POST and DELETE.

11 Examples of Frameworks.

In February 2004 the JISC released a technical framework to support e-learning (ELF), which was developed as a way of making sense of its funded development activities within the learning and teaching space, and to focus future efforts. In May 2004, JISC, DEST and IC agreed to collaborate on testing the potential of an expanded JISC framework with a view to becoming an international e-Learning Framework.

Based on the original Framework developed by JISC and extended by the experience of the JISC, DEST, IC and the LSAL, the resulting expanded and re-factored ELF is an evolving factoring of Services designed to support e-learning applications. The upper set of Services is considered to be largely within the domain of e-Learning, whereas the lower set of Services defines access to functions which cross several domains, such as Authentication.

The ELF is described in the accompanying document "An e-Learning Framework - A Summary".

The current state of this Framework can be viewed by visiting www.cetis.ac.uk:8080/frameworks

For comparison, Figure 7/8 shows a draft e-Research framework being developed by the JISC Committee For Supporting Research (JCSR). The upper sets of blocks represent Services considered within the domain of e-Research, and the lower sets of blocks represent Services that are common across domains.

Service Oriented Frameworks Although there is some overlap, the areas of concern of the two groups are different, and this is reflected in the factoring chosen. For Reference Models or Designs that incorporate aspects of both e-Learning and e-Science, it would then be reasonable to draw upon Services from both Frameworks.

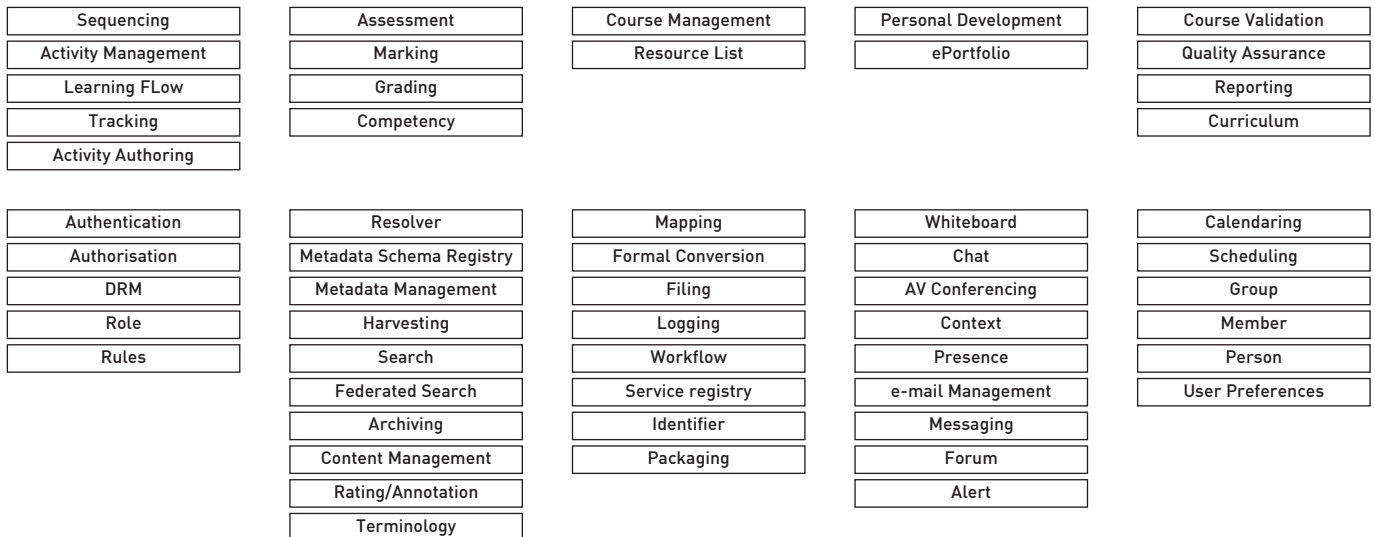


Figure 7 The e-Learning Framework

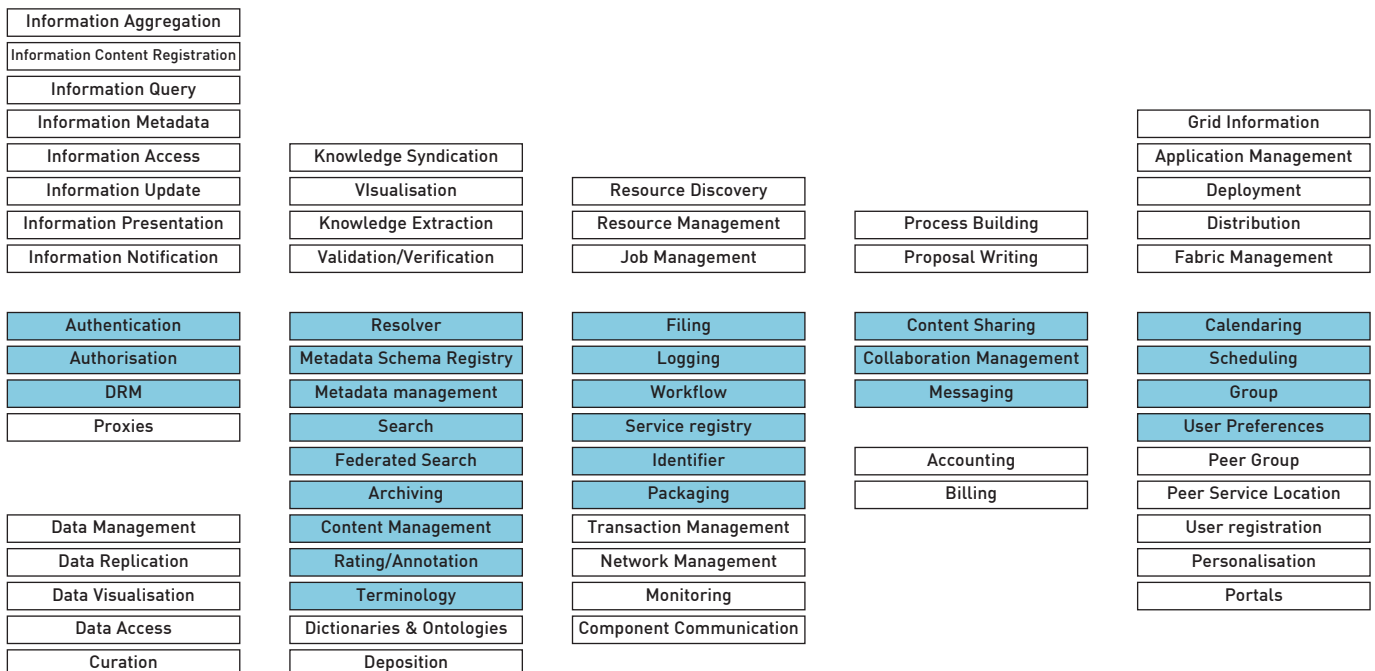


Figure 8 The e-Science Framework (JISC JCSR)*

*Shading indicates Services that are present in both the e-Learning and e-Science Frameworks

12 What next?

The ELF has not been developed from a blank page. The first iteration of the Framework has drawn upon the work of MIT Open Knowledge Initiative, Sun Microsystems' e-Learning Framework, the Carnegie-Mellon Learning Systems Architecture Laboratory, the IMS Global Learning Consortium, and the JISC Managed Learning Environment and Information Environment programmes, and DEST Case studies. The organisations intend the ELF to be an ongoing activity which will be constantly re-evaluated and evolved in response to outside developments and experience in developing artefacts from the ELF.

It is not envisaged that the organisations will independently create the definitions of the Services in the ELF, but instead will work with other organisations to create and maintain these definitions. Rather than go it alone, the organisations need to develop the ELF in collaboration with others, and consider life-cycles for Service definitions, including handing over the work to national (e.g. BSI, NISO) or global (e.g. ISO, IMS) standards and specifications consortia and organisations.

The organisations will also collaborate on supporting activities to encourage adoption of the ELF in their constituent communities and the development of a diverse range of software applications that provide and consume Services defined within the ELF.

The next steps, to be completed by the end of 2004 are:

- Complete a Gap analysis on the first iteration of the Framework
- Prioritise actions resulting from gap analysis
- Establish policies for sustaining and evolving the Framework across the lifecycle of development
- Develop an understanding of organisational processes and organisational responsibilities and activities, within and without the collaboration
- Develop and agree processes to achieve our shared objectives

12.1 Extending the collaboration to other e-Learning communities

Should the service-oriented approach to developing an e-learning framework as outlined in this paper prove to be a viable working model, the aim will be to seek wider participation and input from other bodies representing e-learning communities of practice.

12.2 Extending the collaboration to other domains

The idea behind Frameworks is that, where appropriate, educational applications and Services should be able to make use of Services that are common to multiple domains, which will include, for instance, authentication and authorisation and Service Oriented Frameworks content management. Frameworks also enable e-learning Services to be made available to applications developed in other domains, for example to provide training in context.

A model illustrating the relationship between domains, applications, domain Services and common Services is shown in Figure 9. When several domains adopt a service oriented approach it is possible to develop applications which address functions from across two or more domains and consume Services from multiple domains.

Seeking engagement of communities of practice across domains is a more challenging but worthwhile goal. It remains to be seen as to how relevant national and international bodies foster such engagement. An essential prerequisite however is to have in place a coherent framework for the e-learning space that can be used as a point of reference in establishing cross domain exchanges.

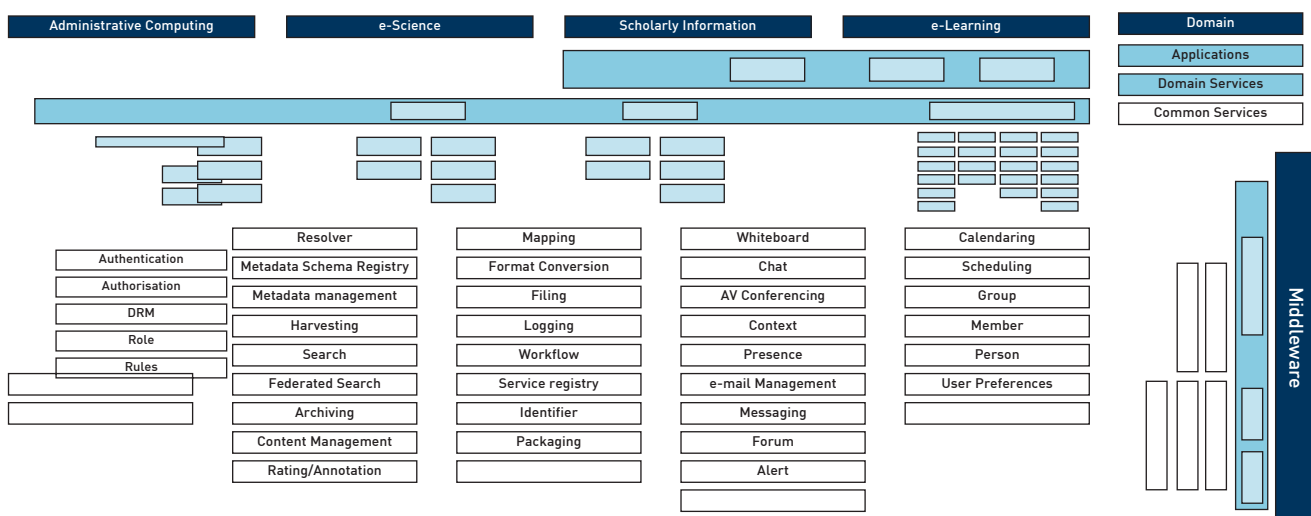


Figure 9 - Cross Domain Modelling and Common Services