

A Technical Framework to Support e-Learning

S Wilson, CETIS, B. Olivier, CETIS, S. Jeyes, CETIS,
A. Powell, UKOLN, T Franklin, Franklin Consulting

Executive Summary

This paper introduces a technical framework designed to support e-learning, and in particular to provide a basis which enables pedagogic diversity.

This is not intended to be prescriptive, nor will it deny practitioners or institutions the capability to deliver distinctive forms of learning. Nor is it intended to restrict the choices of systems that institutions may purchase (whether commercial, freeware or open source). Instead, what we hope to present in this document is a set of patterns that can be used to implement a variety of e-learning strategies.

Although we set out a service framework for e-Learning, we make no assumptions about how many services are deployed in a particular instance, or whether services are delivered by computerised, computer-assisted, or manual processes.

Our starting point has been the “very high level Use Case” of learner-centred education. This is an extreme simplification, and part of the evolution of this framework depends upon teasing out the complexities of much finer-grained processes and looking at learning and teaching from a variety of viewpoints.

This framework is very explicitly centred on the learning and teaching aspects of further and higher education institutions and organisations in the UK. We are very aware that this is only one perspective, and there are other areas, such as logistics, HR and finance, which may also benefit from the approach taken. Although services defined for this framework may be usable for purposes other than learning and teaching, we make no guarantees that the service definitions will be suitable for other domains.

It is believed that the framework will provide benefits to teachers and learners by:

- Supporting pedagogic diversity
- Enabling pedagogy-driven implementations

And benefit institutions by

- Making collaboration between institutions easier
- Providing better returns on technology investment
- Enabling faster deployment of technology
- Providing a modular and flexible technology base.

The paper first discusses the rationale for developing a service oriented framework for managed learning environments (MLE) and e-learning, and then proposes a framework with a brief description of each service.

It is intended as a starting point. The service definitions will need refinement and expansion and many of the details have to be worked out. Additional services will almost certainly be identified and some of the existing ones may be merged or dropped. It is also likely that the standards and specifications listed will not be completely correct, with new ones emerging from time to time. This document is therefore offered to help open discussion and we include just one example of how work might proceed to more fully define the services.

1 The case for a technical framework to support MLEs and e-Learning

If a framework is worthwhile it must lead to benefits for teachers and learners and for institutions. It is our belief that the framework will provide support for this and we describe very briefly the benefits it offers to teachers and learners and to institutions.

1.1 Benefits to teachers and learners

1.1.1 Supporting pedagogic diversity

It becomes possible to support a very diverse set of learning models as it becomes feasible to configure the low-level elements of the learning architecture to fit a variety of pedagogic and institutional business models

1.1.2 Enabling 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.

1.2 Benefits to institutions

1.2.1 Providing 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.

1.2.2 Enabling faster deployment of technology

As components are independent it will often be easier to deploy new components so 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.

1.2.3 Providing a modular and flexible technology base.

The rationale for the framework is specifically to enable the development of modular and flexible systems, where the individual components can be added or replaced more easily than in traditional models

1.2.4 Making collaboration between institutions easier

Through a common framework and thus a common service oriented architecture it becomes easier to define the application interfaces which are needed and thus to share information between institutions (for instance to support student progression). It may also make sharing of applications easier, as it will be simpler to define small applications which are needed in common and can be developed to meet the needs of each institution.

1.3 Why do we need a technical framework

If we look at most “Managed Learning Environments” today, whether implemented or planned, they tend to look something like figure 1:

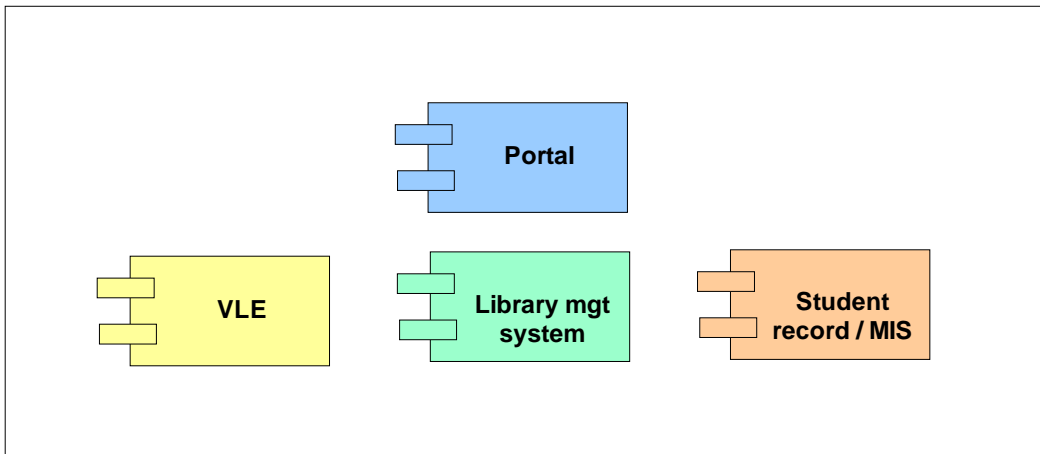


Figure 1: Architecture of a Managed Learning Environment today

So, we have three main systems occupying the main vertical spaces within an institution, and maybe some portal that links the functions together at the end-user level. There is usually some amount of communication between the components, which becomes more obvious when we open up the boxes to see what these components actually contain as shown in figure 2:

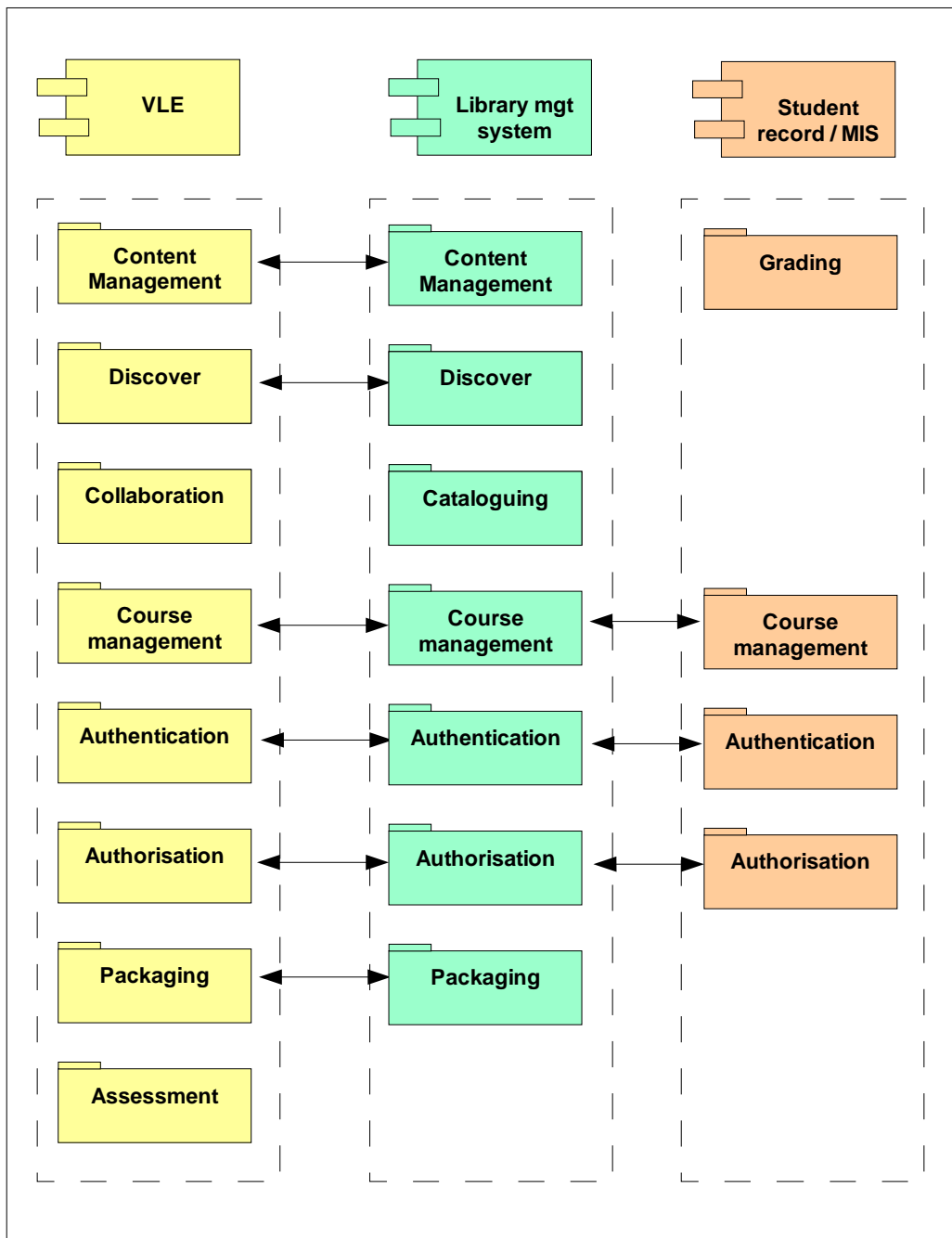


Figure 2: Architecture of a Managed Learning Environment today, expanded to show the components they contain

The communication going on between the components is often because there is considerable overlap of functions and data within the components. Each system tries to manage authentication (making single sign-on more difficult), and each system maintains a complete picture of courses, groups and student enrolments. These overlapping functions mean a lot of data replication is needed to keep the parts in synchronization.

Figure 3 shows the effects of moving the shared functions out of the applications and making these common functions, available to any application that needs them.

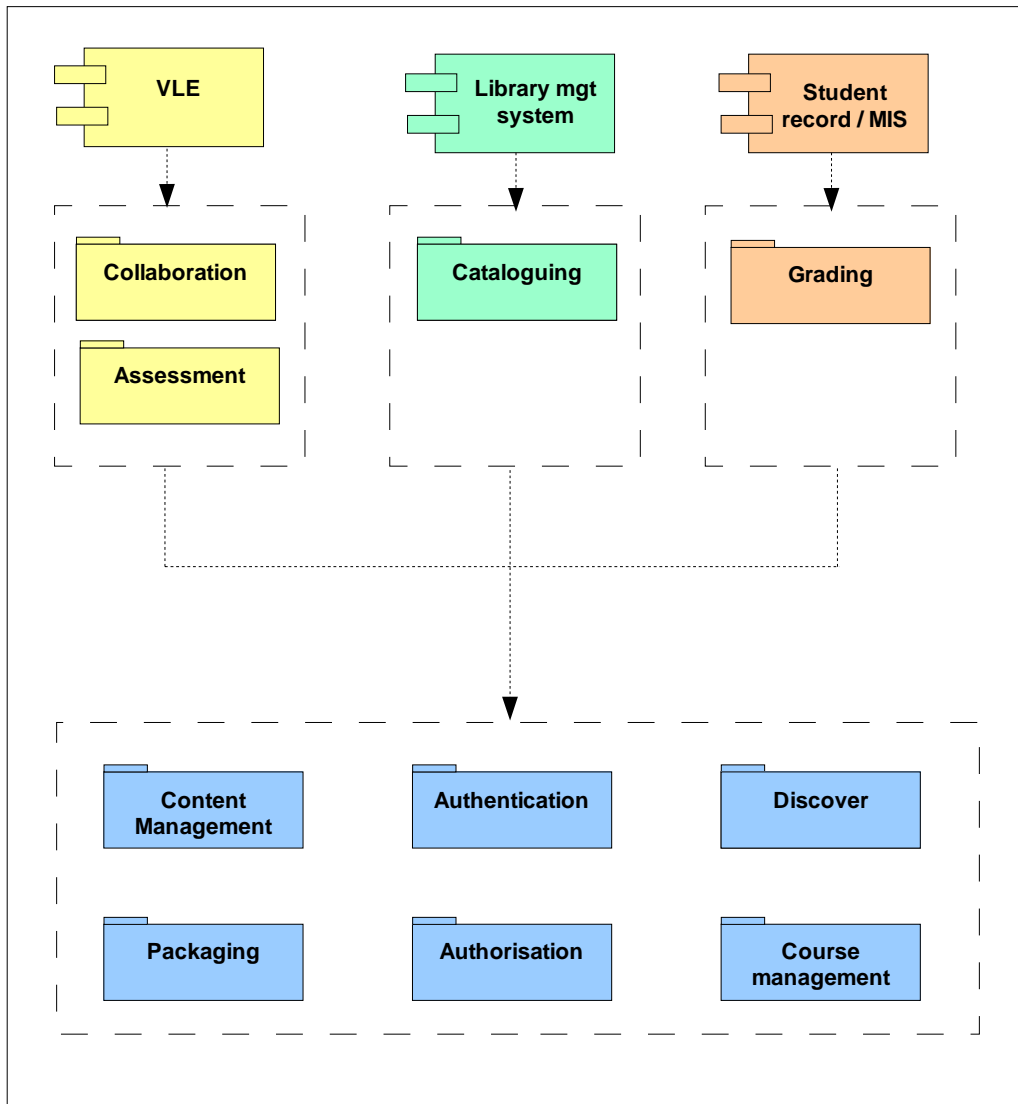


Figure 3: Architecture of a Managed Learning Environment with common services moved out of application

Several things have now changed:

- There is no need to replicate data – all the applications are using the same common data sources
- The individual parts of the MLE are a lot smaller – therefore they contain a lot less actual code, so should be cheaper and easier to write and maintain
- The shared parts of the MLE need to have very well defined interfaces so that our main application components can all make use of them.

Now, it may not be the case that we physically remove all the packages of functionality from the applications – we may simply choose to let one system provide each piece of functionality as a *service* instead. It may also be the case that we can't make some of these common functions into shared services - for various reasons - within a particular institution (for instance, because some of the existing systems cannot interface to external applications in this way due to their design or because their particular requirements are not addressed by the common service).

When we embark on this kind of analysis, identifying the parts of the MLE at a more granular level than monolithic systems, then we eventually end up with a *framework*

of service descriptions. We are no longer interested so much in replicating data between large systems, but instead focus on what kinds of services are needed in the overall architecture to provide certain kinds of behaviour from applications.

What advantages does this approach have over the “giant components” approach we started out with?

First, we reduce the risk of our investments. MLEs, including VLEs, are still very young and rapidly changing. However the costs of changing components is very high as the lack of a common framework and subsequent architecture means that replacing components as needs change is complex and difficult because of the large amount of work that is needed to integrate the replacement into the existing architecture.

Second, we no longer specify the actual architecture of an MLE in terms of its components, but only concern ourselves – at a JISC level - with the shared services that provide its functions. This means that institutions have the flexibility to build an MLE from components of any size and aggregation and yet still take advantage of standardisation efforts.

Third, institutions, vendors and the open-source community can contribute solutions from a much lower cost base. Because components can deliver a small set of functions yet take advantage of a great deal of shared services (e.g. Authentication, Authorization, Course/Student information) the application code is smaller, easier to develop, easier to maintain, and easier to port between institutions which have been using the framework for their service infrastructure. At the moment, every application has to effectively build a replica of the institution infrastructure internally. This also means we enable applications to be developed that support a broader range of user tasks, including those tasks that span multiple areas of responsibility (teaching, information management, administration).

Fourth, Because of the reduced cost of entering the market as developers do not need to build all the services needed for their application there will be a greater diversity of services available to meet a wider diversity of needs.

Fifth, we begin to share a vocabulary with which institutions can discuss the technical aspects of an MLE, enabling sharing of best practice, and facilitating collaborative development.

Finally, by opening up the “black box” of the VLE and other major components, we make possible a far more diverse range of approaches to e-learning within institutions without sacrificing either interoperability or the ability to collaborate and reuse resources and software.

1.4 What does the framework do?

The framework supports the development by institutions of their own architectures, using a flexible *service-oriented* approach.

The framework does not aim to build a generic MLE or VLE, in fact one of the primary goals of the framework is to encourage “coherent diversity”, by providing common toolkits and service definitions which can then be used to meet the diverse goals of the HE and FE community.

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

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

1.5 What is "Service Oriented Architecture"?

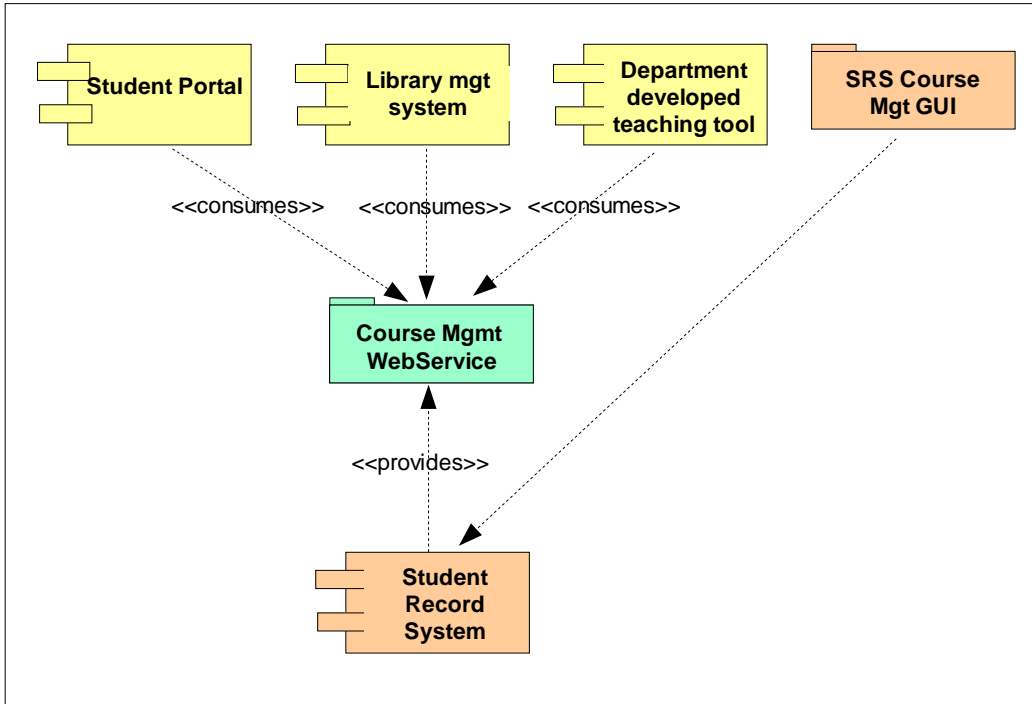
A service-oriented architecture is an approach to joining up systems within enterprises. It is a relatively new approach, but is rapidly gaining popularity because of the lower costs of integration coupled with flexibility and simplification of configuration. Service-oriented architecture builds upon the experience of using Web Services for integration.

In a service-oriented architecture, the application logic contained in the various systems across the organisation – such as student record systems, library management systems, VLEs, 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 for working with enrolment and registration information, which can then be used within a VLE or library system (figure 4).

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 (figure 5.).

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.

However, because integration occurs in this fashion, it becomes a simpler task to replace the systems that provide services within the architecture. Because service consumers are configured to access a service without any knowledge of the system that provides the service, we can replace the underlying system without affecting systems dependent on its capabilities (figure 6.).



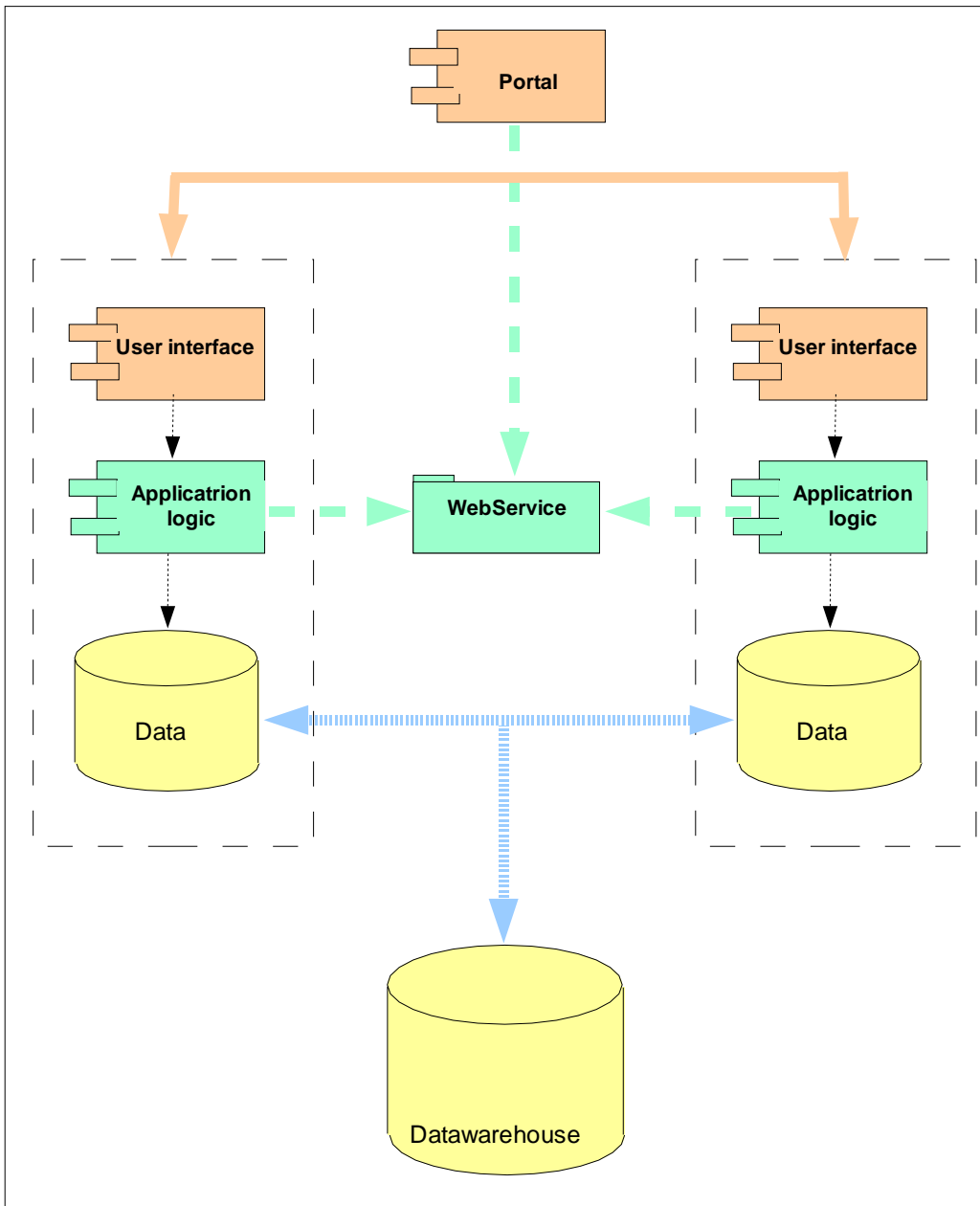
This diagram shows how access to the course management functions of the Student Record System (SRS) can operate in a service-oriented architecture.

The SRS provides a course management web service. This service is "consumed" (ie used) by the student portal to display student data, by the library management system to synchronise data, and in this case a teaching tool has been developed within a department that needs access to enrolment information.

However, this is not the only way to access SRS data, and the SRS Course Management forms can still be used to directly access the SRS by administrative staff.

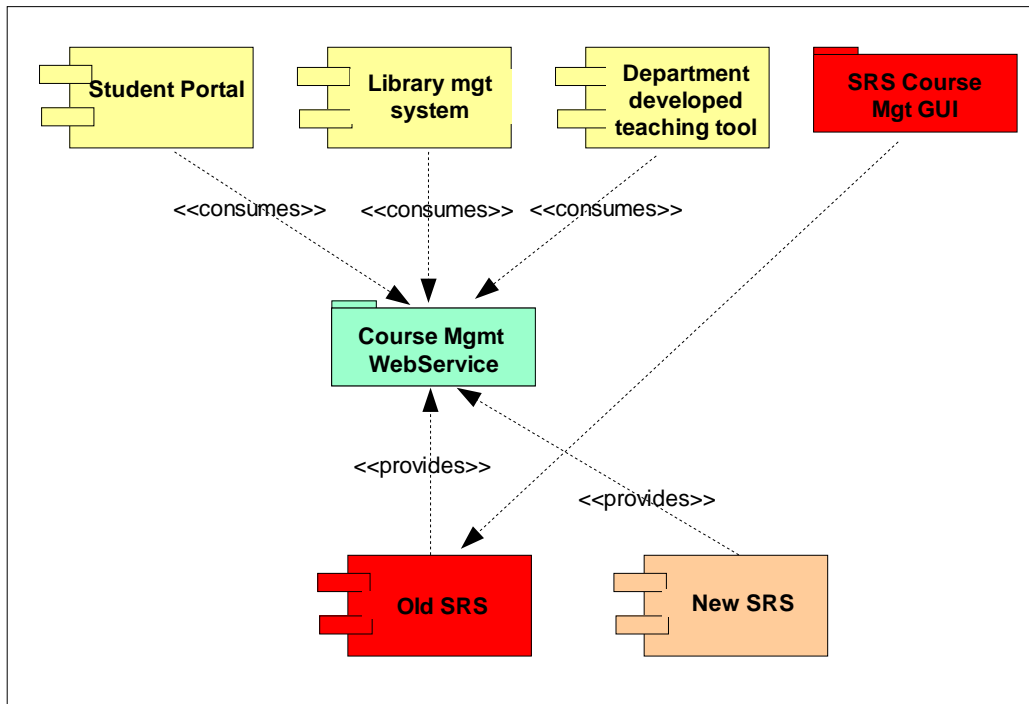
Once the SRS provides the web service, additional applications can take advantage of the capabilities exposed without any changes to the SRS or the service.

Figure 4: Course management within a service-oriented architecture



This diagram shows three different approaches to joining up systems.
The solid "orange route" integrates system capabilities at the end-user level by wrapping their interfaces in a portal.
The dotted blue route" integrates system data by combining it in a single large database.
The dashed "green route" integrates systems by exposing their application logic as a service and sharing them with other applications (and end-users via portals).
Because the application logic is abstracted from both the physical storage and the user interface it is less "fragile" with respect to changes.

Figure 5: Different approaches to integration



This diagram demonstrates replacement of systems in a service-oriented architecture.

Applications which used the Course Management Web Service continue to do so, unaffected by the replacement of the SRS providing the service.

However, the forms provided by the old SRS are now obsolete and cannot be reused.

Figure 6: replacing a student record system in a service oriented architecture

There are other integration approaches that also operate at the application logic level:

- CORBA
- J2EE
- DCOM

However, these technologies are either very costly to implement (CORBA) or restrict platform choice across the organisation (J2EE, DCOM). Web services can take advantage of existing integration using these approaches, however, and many service implementations build upon J2EE integration.

In summary, service-oriented architectures have a number of features that make them attractive for MLEs.

- They are agnostic with regard to platform choices and types of existing systems
- They are less expensive to implement
- Services can be used without knowledge of the internal workings of the system providing the service, allowing systems to be replaced without causing widespread disruption
- Services enable non-replaceable legacy systems to interact with new applications
- By providing access to functionality rather than user interfaces or data it enables institutions and departments to develop applications that relate

better to the tasks they want to perform without duplicating the functionality of existing systems, but instead leveraging existing investment in software. Service-based architectures can be reconfigured to meet changing operational requirements or reflect organisational change.

1.6 How does the framework help?

The framework is intended to support development of flexible, service-oriented architectures in a number of ways:

- Providing a reference set of service definitions
- Providing toolkits to assist developers
- Coordinating related efforts such as standards and shared services

By providing a common set of service definitions, we enable communities to have a shared vocabulary for discussing their MLE and e-Learning activity. Just as the Becta MLE diagram assisted institutions in the earlier phases of MLE development, the framework enables different institutions in different sectors to communicate with one another and the JISC about their technical challenges.

By providing toolkits – not complete solutions – we both enable institutions to build solutions, and also provide assistance for both the commercial sector and the open source community to provide solutions that operate within institutional architectures.

Because we have a shared understanding of the problem space, the areas of concern within the community can be communicated to the JISC so that other efforts – such as the work of CETIS and UKOLN – can be targeted more effectively to solve the pressing needs of the community.

Applications developed using the framework for guidance can, because they have a common specification, be reused far more easily by other institutions, facilitating collaboration between institutions, and inter-institutional integration.

1.7 How does the framework provide value for money?

The core of the framework has a small set of deliverables:

- A list of services identified within the framework, and for each of these
 - The definition of the scope and purpose of the service
 - List of applicable standards and specifications
 - Information on open-source toolkits developed by the programme
 - List of prior work of value, such as projects and case studies, or demonstration projects using the service
 - Information on shared services offered on an ASP basis, for example relevant JISC IE services
- General guidance on creating, exposing, and consuming services

Note that the current version of the framework only contains:

- A list of services identified within the framework, and for each of these
 - The definition of the scope and purpose of the service
 - List of applicable standards and specifications

Also, note that we are interested in producing toolkits for the framework rather than complete solutions. This is because we want to encourage a diverse range of software for institutions to choose from, both open-source and commercial.

Therefore, development activity for the framework will be primarily concentrated on providing libraries that enable developers (again, from commercial as well as open-source communities) to create applications that provide and consume services defined within the framework.

Demonstrator projects will be aimed at testing out the service definitions and toolkits in actual use.

By developing these libraries and toolkits and providing them free-of-charge we substantially reduce the cost of creating software based on services defined in the framework, and reduce the cost of entry to a service-oriented architecture for UK institutions.

We also don't want to build large pieces of software such as content management systems, as these are very complex pieces of software (and there are plenty of commercial and open-source solutions available).

It may be the case that there are some parts of the framework for which there is demand from the JISC community but no available solutions from vendors or the open-source community, in which case it may be necessary to build an open-source reference implementation to fill the gap. This is not an ideal approach, as it tends to lead to problems of sustainability.

Overall, then, the strategy is to create small, tightly scoped pieces of development work that will have maximum impact on the sector. There will be no "white elephants".

1.8 How sustainable is this activity?

It is not envisaged that JISC will manage the definitions of all the services in the framework, but instead JISC will work with other organisations to create and maintain these definitions. Many of the services identified in the work so far overlap with work by the Open Knowledge Initiative at MIT, and the efforts of the IMS Global Learning Consortium, Internet2 and other initiatives. Rather than go it alone, JISC needs to develop the framework in partnership, and consider "exit strategies" for service definitions, including handing over the work to national (e.g. BSI) or global (e.g. ISO, IMS) standards and specifications consortia and organisations.

The toolkits that accompany the framework will be managed through the open-source community. Because these toolkits will be used as integral components within a wide range of products, applications, and services, they are far more likely to gain the necessary support for their continued development from open-source developers than previous educational open source efforts aimed at supplying whole applications such as VLEs. It is also likely that some of these toolkits will be adopted as reference models when accompanying definitions are handed off to BSI, ISO etc.

2 Key Concepts

2.1 Building on prior work

We are not setting out to create a technology framework from a blank page; instead, we are looking to efforts around the world to make sense of learning technologies. We have drawn upon the work of MIT Open Knowledge Initiative, Sun Microsystems' E-Learning Framework, the UK e-University, the Carnegie-Mellon Learning Systems Architecture Laboratory, the IMS Global Learning Consortium and the JISC Managed Learning Environment and Information Environment programmes. We also believe that this framework should be constantly re-evaluated and evolved in response to outside developments that show promise.

2.2 Unified solutions, piecemeal implementation

The idea behind the framework is that where appropriate educational services should be able to make use of common services. This means that there is a need to determine which components are common services that will be used across many services and applications. Common services will include, for instance, authentication and authorisation and content management.

By separating out the different services it also becomes possible to develop systems incrementally. So long as the common services are in place other services can be added as needed.

2.3 Open systems

Open systems provide four advantages:

- Anyone can use them, so that there is a "level playing field" for vendors which will encourage them to use them as.
- Systems can integrate with each other regardless of source which allows institutions to choose the most suitable application and vendors to concentrate on those parts of the systems that they specialise in without having to try to meet all the institution's needs.
- The ability to adapt systems to local needs
- Vendors are not obliged to create everything; instead they can develop just those parts that they have expertise in.

2.4 Building on successful technologies

A key consideration for this framework is the establishment of core technologies for building large-scale modular implementations.

- Java 2 Enterprise Edition (J2EE)
- JINI
- Web Services
- Microsoft .net Framework

This is discussed further under implementation

2.5 The Role of Standards

Standards have an important part to play in the implementation of the framework proposed here. 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.

Standards exist at a variety of levels, and we are not concerned here with the low-level standards (such as IP which belong to the institutional infrastructure layer). However, there are standards at the other levels which are important to ensure that systems will indeed interoperate. Data must be presented in a format which is understood by services that are consuming it in the way that was intended by the services presenting it; this covers the way in which it is transported, the formats that are used and the vocabularies. Without standards for each of these each agreement between producing and consuming services would have to be *ad hoc* and this grows exponentially and rapidly become unsupportable.

2.6 Service-level Abstractions

To make sense of the complexity of business processes, data storage, and application logic, we take the approach of condensing the focus of the framework to a set of services.

Each service encapsulates some form of process meaningful in the context of e-learning, which can be delivered by any number of actual software components.

Service-level abstractions correspond clearly to the notion of Web Services, with each service capable of being defined by using the Web Services Description Language (WSDL). However, this is only one possible interpretation of a service, and services could be realised using a range of technologies.

The important point about services is that we are not concerned, from the point of view of the framework, with either how a service is implemented, or the processes it is used in, but instead only with the set of defined interactions that a service supports. This functional focus allows us to be specific about the range of expected behaviours of an individual service, while remaining agnostic with regard to implementation technologies and overall architecture of particular solutions.

2.7 Managing Identity and Security

One of the drivers for a unified development framework is to consolidate identity and security processes. A common problem within enterprises is the proliferation of access methods, usernames, and passwords, often resulting in a “lowest common denominator” situation with security as strong as the weakest authentication scheme in place.

In this framework, a single point of control is specified for authentication and authorisation, upon which all security-aware end-user applications can depend. This is intended to reduce the number of locations within an architecture where authentication is defined and coded, and support the use of directory services to handle identity management across the organisations.

3 The Layered Services Framework

As with other approaches to e-learning frameworks, the starting point is the abstraction of service layers. We identify four layers of the framework:

User Agents interact with users directly, such as portals, learning delivery systems, authoring tools, administration interfaces and so on. User Agents based on this framework can be either very small and focussed or span many processes to provide a coherent workflow.

Application Services provide functionality required by user agents, such as retrieving learner information, or storing content in a repository. Application Services may be implemented so that they have some sort of user interface, but the key requirement for an application service is that it exposes its functionality for reuse by any number of user agents or other application services, and that it implements a standard interface to support this reuse

Common Services provide lower-level functionality which is not education-specific, such as authentication and authorization services, but upon which application services and user agents depend.

Infrastructure is the underlying network, storage, and processing capability provided for an implementation. This is assumed by the framework, but not defined.

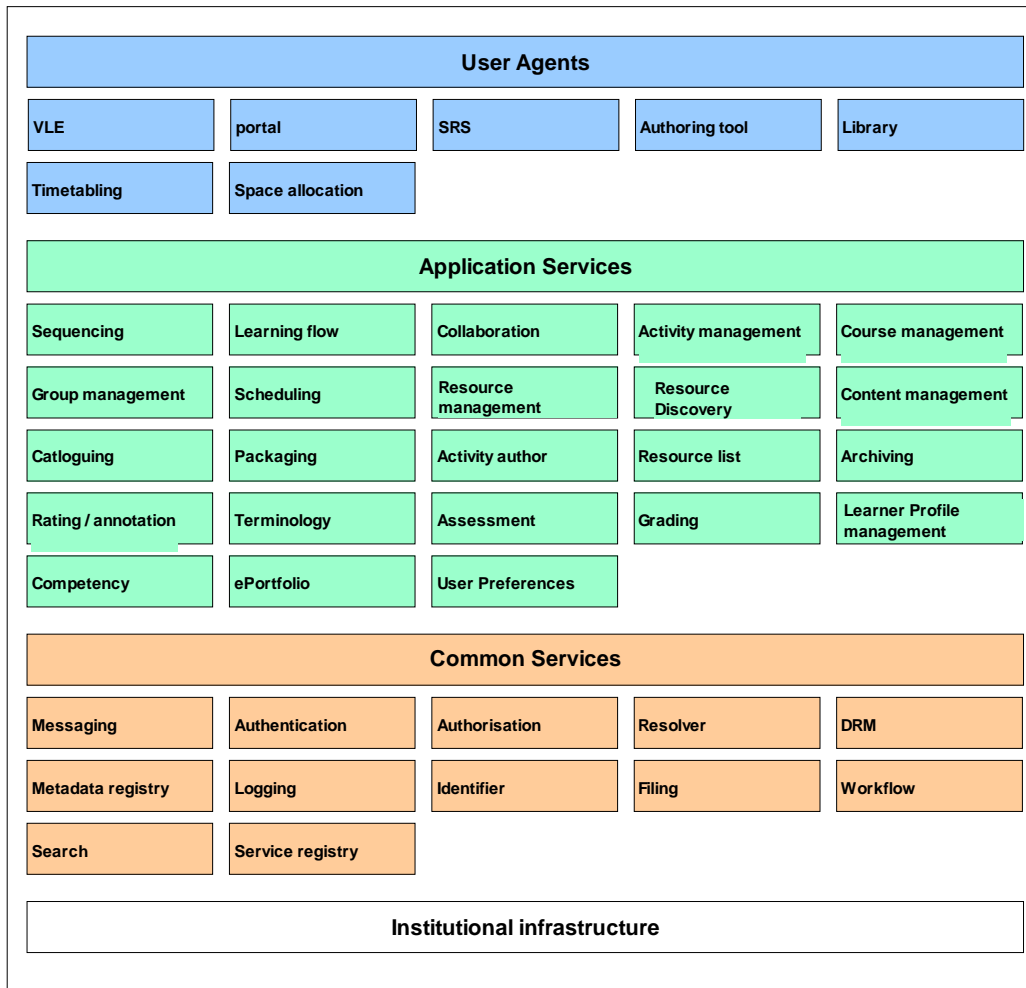


Figure 7: model of services demonstrating common and application services together with selected user agents

3.1 Application Services

We do not consider this to be a definitive list of the services that can be provided, and we hope that additional services are identified, or identified services refined, in the light of future requirements analysis. We have identified the following Application services:

- Sequencing
- Learning Flow
- Collaboration
- Activity Management
- Course management
- Group management
- Scheduling
- Resource management
- Resource Discovery
- Content Management
- Cataloguing
- Packaging
- Activity Authoring

- Resource List
- Archiving
- Rating/Annotation
- Terminology
- Assessment
- Grading
- Competency
- Learner Profile Management
- EPortfolio
- User Preferences

3.2 Common Services

We have identified the following common services:

- Service Registry
- User Messaging
- Authentication
- Digital Rights Management (DRM)
- Logging
- Identifier
- Resolver
- Filing
- Authorization
- Workflow
- Search
- Harvest
- Alert
- Metadata Registry

3.3 Service Specifications

To implement a Service requires a Service Specification – a set of documents that provides a 'blueprint' for building the service. This document set would comprises, at a minimum:

- A narrative description of the component and its role in the framework
- A set of Interface definitions, or references to relevant specification
- A set of Data Type definitions, or references to relevant specifications
- A binding to the implementation technology

Examples of bindings include java interfaces and classes for J2EE, COM interfaces and classes for .net, and WSDL interfaces and XML Schemas for a web services implementation. A service specification may not necessarily cover all of these bindings.

In cases where no service specification currently exists, implementers may choose to develop and publish a service specification themselves. We recommend that JISC provides a suitable structure for publishing and sharing service specifications.

4 Implementation

4.1 Integrating Legacy Applications

In many cases, the services listed may have their functionality embedded within legacy applications. In such cases, one strategy for implementation is to expose the functionality using a service agent. This agent provides the standard service interface, and translates calls upon that interface into messages that are understood by the legacy application. Agent code could be generalized across multiple services.

4.2 Use of standard integration components

It should be possible to use standard industry integration solutions to provide the service infrastructure. For example, Microsoft BizTalk Server, BEA Weblogic, Sun iPlanet and ONE suites, IBM MQSeries, or Sonic MQ/XQ.

4.3 Implementing with Java 2 Enterprise Edition (J2EE)

Sun's Java 2 Enterprise Edition (J2EE) provides excellent support for distributed components and component reuse through the Enterprise JavaBeans technology (EJB). Using this architecture, services can be implemented using a combination of Entity Beans and Session Beans, with the service interfaces exposed through a Session Bean.

To allow a service delivered via J2EE to be accessed by non-J2EE compliant software, each service may also provide a servlet to allow access via Web Services. (This servlet could be written using a standardised agent code base that could be adapted to serve multiple services.)

4.4 Implementing using JINI

Jini is a technology developed by Sun to facilitate the set-up, integration, dynamic configuration and management of distributed service-based applications. Since 1999 it has been refined and successfully applied for demanding and diverse applications such as mission-critical online financial services and tactical battle-field support systems by the U.S. Army. These lean heavily on Jini's 'self-healing' capabilities: systems can be dynamically brought down and up again, added to or replaced and the whole dynamically configures and reconfigures itself accordingly. It is clear that implementing eLearning will require the integration of multiple new and existing systems that will change and evolve over time and these will need to be robust and easy to manage. Thus Jini, or systems with Jini-like properties, will have a crucial role to play in the future evolution of MLEs.

Jini works by dynamically providing service 'proxies' to clients. The client has to talk to the proxy through a pre-agreed Java API. The proxy then talks to its service using a privately agreed protocol. The proxy thus 'hides' the service's protocol from the application, but there does need to be an pre-agreed specification for the API for each service. This is precisely what the MIT OKI project has produced with its Open Service Interface Definitions (OSIDs) as the service specifications 'glue' for the OKI framework for eLearning infrastructures.

4.5 Implementing using Web Services

Web services looks like being the next wave in the development of standards based shared services. Based on four sets of standards: XML for data, W3C SOAP for

message packaging, Universal Description, Discovery, and Integration (UDDI) for registering, publishing and finding appropriate services and Web Services Description Language (WSDL) for describing services. Additional standards cover specific aspects of web services such as security (WS-Security), routing of messages (WS-Routing) and so on. To make sense of the wide range of web services specifications, the Web Services Interoperability group (WS-I) produce a “basic profile” of recommended specifications.

Web Services technology is agnostic with regard to programming languages – services can be developed in Java, C#, Python, or any other language that can handle internet technologies and XML. Toolkits, libraries, and products exist to support development of services on all major platforms, reducing the time and cost of implementation

4.6 Implementing using .net

Like J2EE, Microsoft’s platform provides foundation technologies for distributed and reusable components. The Component Object Model (COM) and Distributed Component Object Model (DCOM) provide CORBA-like interface abstractions that allow applications to share code deployed in dynamic link libraries (DLLs). To share services beyond the firewall, components also need to provide web services interfaces, perhaps through deployment using Microsoft BizTalk Server.

5 Descriptions of Application Services

- Sequencing
- Learning Flow
- Collaboration
- Activity Management
- Course management
- Group management
- Scheduling
- Resource management
- Resource Discovery
- Content Management
- Cataloguing
- Packaging
- Activity Authoring
- Resource List
- Archiving
- Rating/Annotation
- Terminology
- Assessment
- Grading
- Competency
- Learner Profile Management
- ePortfolio
- User Preferences

5.1 Sequencing

A sequencing service provides support for the use of sequenced learning objects, primarily of use within VLEs and similar environments for the delivery of complex learning packages.

The key functions for this service are:

- providing runtime support for sequences, such as getting the next item in a sequence. In future the sequence is likely to be dependent on the learner's activity with the current and previous objects.

Specifications with applicability in this area are:

- IMS Simple sequencing
- IMS Learner Design
- SCORM

Related services are:

- Packaging
- Learning Flow
- Activity Management
- Resource management
- Learner Profile Management

5.2 Learning Flow

A Learning flow service supports the management and use of complex learning scenarios which are more complex than sequencing can support. It is probable that sequencing will be subsumed within learning flow at some point.

The key functions for this service are:

- providing runtime support for learning designs, such as sequencing of activities including for collaborative activities

Specifications with applicability in this area are:

- IMS Simple sequencing
- IMS Learning Design

Related services are:

- Sequencing
- Collaboration
- Activity Management
- Scheduling
- Learner Profile Management
- User Preferences

5.3 Collaboration

A collaboration service supports the use of collaborative tools and provides services for the creation and management of collaborative sessions, including both synchronous and asynchronous communication as appropriate.

The key functions for this service are:

- Managing the set-up of collaborative sessions, including the types of tools and facilities that will be available.
- Supporting synchronous communication

- Supporting asynchronous communication
- supporting collaborative document creation
- supporting collaborative workflow

Specifications with applicability in this area are:

- IMS Learning design
- IMS Learner Information Package
- IMS Enterprise

Related services are:

- Activity Management
- Scheduling
- User Preferences
- Group management

5.4 Activity Management

An activity management service supports the interactions between units of learning (courses, modules, etc.) and the students and staff participating in them. It provides the Learning Design runtime engine. see also "coppercore"

<http://sourceforge.net/projects/coppercore>

The key functions for this service are:

- populating activities by assigning individuals to roles within them
- initialising runtime services required by an activity (such as chat rooms)
- managing the internal state of the set of activities (e.g. who has completed specific activities, which activities are now available). NB. This could be delegated to a sequencing service."

Specifications with applicability in this area are:

- IMS Learning design

Related services are:

- Collaboration
- Course management
- Group management
- User Preferences

5.5 Course management

A Course management service allows applications or services to access and manage courses, modules and other units of learning. This may be merged with Group management.

The key functions for a Course management service are:

- Support creating, reading, updating and deleting units of learning
- Support creating, reading, updating and deleting data regarding membership of units of learning
- Support creating, reading, updating and deleting people information

Specifications with applicability to this area are:

- IMS Enterprise
- IMS Enterprise Services (not yet available)
- OKI Course Management OSID

Related services are:

- Group management

- Activity Management

See also the Detailed Description of the Course and Group Management Services

5.6 Group management

A Group management service is an abstraction of Course management, with applicability to groups other than units of learning. This may be merged with Group management.

The key functions for a Group management service are:

- Support creating, reading, updating and deleting groups
- Support creating, reading, updating and deleting data regarding membership of groups
- Support creating, reading, updating and deleting people information

Specifications with applicability to this area are:

- IMS Enterprise
- IMS Enterprise Services (not yet available)
- OKI Group Management OSID (not yet available)

Related services are:

- Activity Management
- Course management

See also the Detailed Description of the Course and Group Management Services

5.7 Scheduling

A scheduling service supports allocating resources (such as modules, roles, people, or physical resources) against time, supporting basic shared calendaring and resource management.

The key functions for a Scheduling service are:

- Support allocation and deallocation of resources against time slots
- Support requests for schedules by resource and/or timeframe
- Support for optimising schedules and timetables

Specifications with applicability to this area are:

- OKI Scheduling OSID

Related services are:

- Learning Flow
- Resource management

5.8 Resource management

A Resource management service supports the management of finite physical resources, such as equipment and rooms.

The key functions for this service are:

- Support for the allocation and deallocation of physical resources against courses or timetables
- Support for the resolving clashes
- Specifications with applicability in this area are:

Related services are:

- Scheduling

5.9 Resource Discovery

Resource discovery service supports the finding of resources including learning objects, information assets, e-reserves and so on.

- Issuing queries to local and remote search services

Specifications with applicability to this area are:

- IMS Metadata
- Dublin Core
- SCORM
- IEEE LOM
- Other metadata standards
- IMS Learner Information Package
- IMS Digital Repositories Interoperability
- IMS Vocabulary Definition Exchange
- OAI-PMH,
- Z39.50
- SRW

Related services are:

- Metadata Registry
- Search
- Harvest
- Resources
- Service Registry

5.10 Content Management

A Content management service supports the publishing, retrieval, description, and organisations of information resources.

The key functions for a this service are:

- Publishing information to other content management services
- Retrieving and organising information retrieved from other content management services

Specifications with applicability to this area are:

- IMS Digital Repositories Interoperability
- IMS Metadata
- IMS Vocabulary Definition Exchange
- SCORM
- Dublin Core
- IEEE LOM
- other metadata standards

Related services are:

- Resource Discovery
- Cataloguing
- Packaging
- Resource List
- Archiving
- Rating/Annotation
- Resolver

- Metadata Registry
- DRM
- Filing
- Harvest

5.11 Cataloguing

A Cataloguing service supports the management of descriptions for information resources.

The key functions for a this service are:

- Creating, reading, updating and deleting metadata

Specifications with applicability to this area are:

- IMS Metadata
- SCORM
- Dublin Core
- IEEE LOM
- other metadata standards

Related services are:

- Resource Discovery
- Content Management
- Resource List
- Rating/Annotation
- Terminology
- Metadata Registry
- DRM
- Harvest

5.12 Packaging

A Packaging service supports the assembly of packages of information resources by aggregation and disaggregation, and their preparation for transport and delivery.

The key functions for a this service are:

- Support creating, reading, updating and deleting packages
- Support for aggregating and disaggregating objects

Specifications with applicability to this area are:

- IMS Metadata
- IMS Simple sequencing
- SCORM
- IEEE LOM

Related services are:

- Sequencing
- Cataloguing

5.13 Activity Authoring

Supports the process of creating, storing, retrieving, aggregating and otherwise managing learning activities.

The key functions for a this service are:

- Support creating, reading, updating and deleting learning activities

Specifications with applicability to this area are:

- IMS Metadata
- IMS Simple sequencing
- IMS Vocabulary Definition Exchange
- SCORM
- IEEE LOM

Related services are:

- Sequencing
- Activity Management
- Cataloguing
- Packaging

5.14 Resource List

Supports the creation, access and management of reading lists and other lists of resources.

The key functions for a this service are:

- Support for creating, reading, updating and deleting of lists such as reading lists

Specifications with applicability to this area are:

- IMS Metadata
- SCORM
- Dublin Core
- IEEE LOM

Related services are:

- Course management
- Resource management
- Cataloguing
- Rating/Annotation

5.15 Archiving

Supports the long-term preservation of documents and content.

The key functions for a this service are:

- creating, updating and managing archives
- accessing and updating repositories

Specifications with applicability to this area are:

- IMS Digital Repositories Interoperability
- IMS Metadata
- SCORM
- Dublin Core
- IEEE LOM

Related services are:

- Resource Discovery
- Cataloguing
- Metadata Registry
- Search

5.16 Rating/Annotation

Provides support for the use of secondary metadata (user ratings and text annotations) for resources.

The key functions for a this service are:

- creation and management of annotations for objects
- accessing ratings and annotations

Specifications with applicability to this area are:

- IMS Metadata
- SCORM
- Dublin Core
- IEEE LOM
- other metadata standards

Related services are:

- Resource Discovery
- Cataloguing

5.17 Terminology

Provides machine-readable declarations of vocabulary terms, mappings between those terms (either within a particular thesaurus or across multiple thesauri or classification schemes) and automatic classification services. The intention is to allow consumers of the service to map terms, allowing users to use one set of terminology, but automatically mapping their terms to alternative or additional terms in thesauri or classification schemes used by target services. (Note that terminology is not limited to subject but also includes audience level, resource type and certification).

The key functions for a this service are:

- providing decalarions of vocabulary terms
- providing mappings from one term to other terms
- providing automatic classification of resources according to one or more vocularies and/or thesauri

Specifications with applicability to this area are:

- IMS Vocabulary Definition and Exchange (VDEX)
- RDFS
- OWL

Related services are:

- Cataloguing

5.18 Assessment

An Assessment service supports the use of automated assessments.

The key functions for an assessment service are:

- Storing, retrieving, deleting and updating Items (individual questions)
- Storing, retrieving, deleting and updating Sections (groups of questions)
- Storing, retrieving, deleting and updating Assessments (groups of sections or questions)
- Publishing Assessments against a Course, Module, or other unit of learning
- Storing and retrieving Results from Assessments

Specifications with applicability to this area are:

- IMS Question and Test Interoperability (QTI) provides data models for assessments, sections, items, and results reports
- OKI Assessment OSID provides APIs for managing banks of items, sections, and assessments, and for publishing assessments

Related services are:

- Learning Flow
- Scheduling
- Grading
- Authentication

5.19 Grading

A Grading service supports submitting grades against courses, modules, and other units of learning

The key functions for a grading service are:

- Submitting grades against courses, modules, and other units of learning
- Submitting feedback and comments associated with grades

Specifications with applicability to this area are:

- OKI Grading OSID provides an API for submitting grading reports
- IMS Enterprise
- IMS Learner Information Package
- IMS Question and Test Interoperability

Related services are:

- Assessment
- Competency
- Authentication
- Authorization

5.20 Competency

Supports the management of competency frameworks, and the mapping of units of learning, assessments, and activities against specific competencies.

The key functions for a this service are:

- support for the definition and maintenance of competencies and competency frameworks
- support for mapping these competencies an frameworks against courses (and other units of learning)
- support for mapping these competencies against assessments and activities

Specifications with applicability to this area are:

- IMS Learner Information Package
- IMS Learning Design
- IMS Question and Test Interoperability
- IMS Reusable Definition of Competency or Educational Objective

Related services are:

- Assessment
- Learner Profile Management
- ePortfolio

5.21 Learner Profile Management

Supports the management of a learner's personal development plans and personal development records.

The key functions for a this service are:

- support for creating and maintaining personal development plans and personal development records

Specifications with applicability to this area are:

- IIMS Learner Information Package

Related services are:

- Competency

5.22 EPortfolio

Supports the management and assessment of artefacts created by learners, such as essays and projects.

The key functions for a this service are:

- support for the creation and updating and uploading of artefacts
- support for the recording and assessment of such artefacts

Specifications with applicability to this area are:

- IMS Learning Design
- IMS Learning Management

Related services are:

- Collaboration
- Activity Management
- Scheduling
- Grading

5.23 User Preferences

A user preferences service provides machine-readable information about users' personal preferences. The primary intention of this service is to allow user agents, such as portals, to automatically configure themselves for particular end-users and to prevent end-users from having to enter their preferences into multiple user agents.

The key functions for a this service are:

- Support creating, reading, updating and deleting user profiles

Specifications with applicability to this area are:

- IMS Accessibility
- IMS Learner Information Package

Related services are:

6 Descriptions of Common Services

- Service Registry
- User Messaging
- Authentication
- Digital Rights Management (DRM)
- Logging
- Identifier

- Resolver
- Filing
- Authorization
- Workflow
- Search
- Harvest
- Alert
- Metadata Registry

6.1 Service Registry

Provides information about the services that are available on the network, including details of what content is available, what functionality is supported by the service and what protocols and standards are used to access the service (i.e. details of the API).

The key functions for a this service are:

- provide machine-readable descriptions of services
- allow searching for services by 'keyword', type, protocol, etc.
- provide human-readable view of available services

Standards and specifications with applicability to this area are:

- Universal Description, Discovery, and Integration (UDDI)
- Simple Object Access Protocol (SOAP)
- Web Services Description Language (WSDL)

Related services are:

- Resolver

6.2 User Messaging

Allows broadcast of messages to users and groups using appropriate communication technology.

The key functions for a this service are:

- Issuing messages to users and groups of users

Specifications with applicability to this area are:

- OKI User messaging OSID
- Message Service Specification (MSS)

Related services are:

- Alert

6.3 Authentication

Allows the identity of agents to be established.

The key functions for a this service are:

- support for the creation and management and revoking of user IDs
- support for the secure authentication of users

Standards and specifications with applicability to this area are:

- OKI Authentication OSID
- Lightweight Directory Access Protocol (LDAP)
- Public Key Infrastructure (PKI)
- X509

Related services are:

- Logging

- Authorization

6.4 Digital Rights Management (DRM)

Supports the allocation and application of rights policies against resources, consuming data in a digital rights expression language (DREL) to determine access. Works through Authorization services, and is generally intended to be called by Authorization implementations as the result of a request to use a resource.

The key functions for a this service are:

- support for the allocation and updating of rights associated with objects
- support for validation of the use of objects at run time

Specifications with applicability to this area are:

- IEEE Digital Rights Expression Language
- XrML
- ODRL
- CreativeCommons

Related services are:

- Authorization

6.5 Logging

Generic logging service for applications

The key functions for a this service are:

- logging activity to support security, non-repudiation and DRM.

Specifications with applicability to this area are:

- OKI Logging OSID

Related services are:

- Authentication
- DRM
- Authorization

6.6 Identifier

Service that allows the creation, assignment and management of identifiers (DOI, PURL, Handle) to resources (including management of appropriate metadata about each identifier).

Key functions of this service are:

- assignment/creation/registration of identifier
- update/management of identifier mapping to URL and associated metadata

Specifications with applicability to this area are:

- DOI
- PURL
- Handle

Related services are:

- Resource Discovery
- Cataloguing
- Resolver
- Metadata Registry

6.7 Resolver

Provides services for resolving identifiers including OpenURL metadata, DOI, PURL, Handle etc. This includes redirection to document delivery services, redirection to online bookshops, to local library services, and other discovery activities. Resolver services will use institutional and individual preferences to determine resolution preferences.

Key functions of this service are:

- resolving identifier to the URL of the current location of a resource
- resolving metadata to a set of links to appropriate copies of a resource
- providing links to value-added services based on metadata about resource

Standards and specifications with applicability in this area are:

- OpenURL
- DOI
- URI
- Handle
- PURL

Related services are:

- Resource Discovery
- Search

6.8 Filing

Generic storage system support (OKI).

The key functions for a this service are:

- support for handling hierarchical filing systems including creating, updating and organising files and hierarchies.

Specifications with applicability to this area are:

- OKI Filing OSID

Related services are:

6.9 Authorization

Supports the management of access to resources by agents.

The key functions for a this service are:

- support for giving and revoking access rights to services by individuals and other services
- Informing services whether authenticated users or services have access to services when they request access

Specifications with applicability to this area are:

- OKI Authorization OSID

Related services are:

- Authentication
- DRM
- Logging

6.10 Workflow

Supports creation and management of multi-service transactions, usually for administrative purposes.

The key functions for a this service are:

- Supports appropriate usage pattern of a collection of Web Services, in such a way that the resulting composition describes how to achieve a particular business process.
- Supports the interaction pattern of a collection of Web Services; in this case, the result is a description of the overall partner interactions.

Specifications with applicability to this area are:

- OKI Workflow OSID
- Web Services Flow Language (WSFL)

Related services are:

- Learning Flow

6.11 Search

A search service allows application services to support resource discovery by sending a query formatted according to a particular query syntax and returning a result set formatted according to a particular syntax.

Key functions of this service are:

- accept query, return results

Standards and specifications with applicability to this area are:

- SOAP
- XML
- SRW
- Z39.50
- XQuery
- RDF

Related services are:

- Resource Discovery
- Cataloguing
- Rating/Annotation
- Terminology
- Harvest

6.12 Harvest

Allows application services to support resource discovery by harvesting copies of some or all metadata records.

The key functions for a this service are:

- accept harvest request, return set of records

Specifications with applicability to this area are:

- Open Archives Initiative - Metadata (OAI-PMH)
- RSS

Related services are:

- Search
- Alert

6.13 Alert

Allows application services to support resource discovery by alerting them to lists of new resources.

The key functions for a this service are:

- make available an alerting 'channel'

Specifications with applicability to this area are:

- RSS

Related services are:

- User Messaging

6.14 Metadata Registry

A metadata registry provides machine-readable information about the metadata schemas in use by particular metadata-based services. The primary intention of this service is to allow service consumers to automatically determine information about appropriate search terms and the structure of metadata records that will be returned to them. However, metadata registries also provide a useful human-oriented service, allowing people to see what metadata schemas are in use by which services - providing a basis for metadata schema sharing and re-use.

The key functions for a this service are:

- providing machine-readable declarations of metadata schemas
- providing mappings between metadata schemas
- providing human-readable view of metadata schemas

Specifications with applicability to this area are:

- RDFS
- OWL

Related services are:

- Cataloguing
- Terminology

7 Detailed description of Course and Group Management Services

7.1 Overview

A **Course Management** service allows applications or services to access and manage courses, modules, and other units of learning. A Course Management service is a specialization of a **Group Management** service, which provides services to access and manage groups of all kinds, including organisation structures.

Both services offer a similar range of functions.

7.2 Services

A Group Management service consists of three sub-services, which are:

- **Group Management** - Supports creating, reading, updating and deleting groups
- **Membership Management** - Supports creating, reading, updating and deleting data regarding the membership of units of learning

- **Member Management** - Support creating, reading, updating and deleting information about members

The functions for a Course Management service are identical, with the difference that the kinds of Groups concerned are units of learning.

7.3 Deployment and use

Within an MLE architecture, Course and Group Management services would tend to be provided by the system of record for managing this information, such as a Management Information System (MIS) or Student Record System (SRS).

The information provided by these services is of use to a wide range of systems, including systems for delivering learning activities (such as Virtual Learning Environments), library and resource management applications, and portals.

7.4 Group Management

7.4.1 Service definition

This service provides a means to manage information about groups. A group management service needs to support the following features:

- Creating new groups
- Updating an existing group
- Deleting an existing group
- Requesting an existing group by its unique identifier
- Requesting a set of groups by query
- Requesting a set of groups by association

7.4.2 Interfaces and data models

The specification with most direct applicability to this service is the **IMS Enterprise Services** specification, v1.0, which contains both interfaces and data models.

There are two service definitions that apply: **GroupManager** and **GroupsManager**. The former is an interface that applies to single instances of groups, the latter to sets of groups.

The MIT Open Knowledge Initiative also provides two relevant Open Service Interface Definitions (OSIDs), called Course Management and Group Management. These operate at a different binding point to the IMS interfaces, and are intended to be implemented within client applications via adapters rather than as a service layer.

There are many basic similarities between the two sets of specifications, and it is possible to combine them within an implementation:

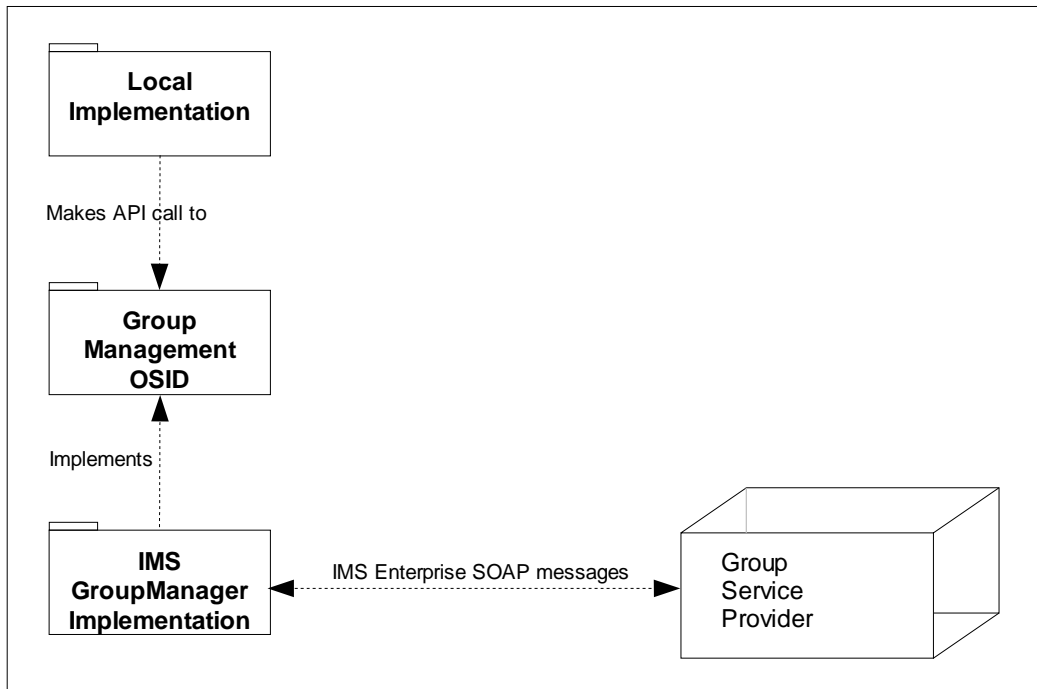


Figure 8. Shows the relationship between the various elements needed to implement the service

Here, the consumer implementation makes calls to the service using the API provided by OKI. The API is implemented to provide web services, defined using the IMS GroupManager specification.

Not all of the OKI API can be implemented using IMS Enterprise at present, as there are some differences between the data model implied by the OKI definition and that provided by IMS.

7.4.3 Bindings

IMS Group Manager and IMS GroupsManager have bindings for Web Services using the Web Services Definition Language (WSDL) and XML Schema (XSD).

The OKI OSIDs have Java API bindings.

7.4.4 Implementations

There is a library under development by CETIS that implements the IMS services for the J2EE platform using the Apache AXIS web services framework.

7.5 Membership Management

7.5.1 Service definition

This service provides a means to manage information about membership of groups. A membership management service needs to support the following features:

- Creating new memberships of a group by a person
- Updating an existing membership
- Deleting an existing membership
- Requesting an existing membership by a unique identifier
- Requesting a set of memberships by query

- Requesting a set of memberships associated with a group
- Requesting a set of memberships associated with a person

7.5.2 Interfaces and data models

The specification with most direct applicability to this service is the **IMS Enterprise Services** specification, v1.0, which contains both interfaces and data models.

There are two service definitions that apply: **MembershipManager** and **MembershipsManager**. The former is an interface that applies to single instances of a membership, the latter to sets of memberships.

The MIT Open Knowledge Initiative also provides two relevant Open Service Interface Definitions (OSIDs), called Course Management and Group Management. These operate at a different binding point to the IMS interfaces, and are intended to be implemented within client applications via adapters rather than as a service layer.

There are many basic similarities between the two sets of specifications, and it is possible to combine them within an implementation (see above, in the Group Management section).

Not all of the OKI API can be implemented using IMS Enterprise at present, as there are some differences between the data model implied by the OKI definition and that provided by IMS.

7.5.3 Bindings

IMS MembershipManager and IMS MembershipsManager have bindings for Web Services using the Web Services Definition Language (WSDL) and XML Schema (XSD).

The OKI OSIDs have Java API bindings.

7.5.4 Implementations

There is a library under development by CETIS that implements the IMS services for the J2EE platform using the Apache AXIS web services framework.

7.6 Member Management

7.6.1 Service definition

This service provides a means to manage information about people belonging to groups. A member management service needs to support the following features:

- Creating new members
- Updating an existing member
- Deleting an existing member
- Requesting an existing member by its unique identifier
- Requesting a set of members by query
- Requesting a set of members for a group

7.6.2 Interfaces and data models

The specification with most direct applicability to this service is the **IMS Enterprise Services** specification, v1.0, which contains both interfaces and data models.

There are two service definitions that apply: **PersonManager** and **PersonsManager**. The former is an interface that applies to single instances of members, the latter to sets of members.

The MIT Open Knowledge Initiative also provides two relevant Open Service Interface Definitions (OSIDs), called Course Management and Group Management. These operate at a different binding point to the IMS interfaces, and are intended to be implemented within client applications via adapters rather than as a service layer.

There are many basic similarities between the two sets of specifications, and it is possible to combine them within an implementation (see above).

Not all of the OKI API can be implemented using IMS Enterprise at present, as there are some differences between the data model implied by the OKI definition and that provided by IMS.

There is also a specification used in conjunction with LDAP called EduPerson that provides a relevant data model. There is a high degree of correspondence between the IMS Person and EduPerson data models.

7.6.3 Bindings

IMS Group Manager and IMS GroupsManager have bindings for Web Services using the Web Services Definition Language (WSDL) and XML Schema (XSD).

The OKI OSIDs have Java API bindings.

The EduPerson schema is provided as a flat-file binding for LDAP.

7.6.4 Implementations

There is a library under development by CETIS that implements the IMS services for the J2EE platform using the Apache AXIS web services framework.

8 Resources

The JISC Information Architecture <http://www.ukoln.ac.uk/distributed-systems/jisc-ie/arch/standards/>

IMS <http://www.imsglobal.org/>

OKI <http://web.mit.edu/oki/>

The Benefits of a Service-Oriented Architecture, Michael Stevens, Developer.com, <http://www.developer.com/services/article.php/1041191>

Service-Oriented Architecture Introduction (2 parts), Michael Stevens, Developer.com, <http://www.developer.com/services/article.php/1010451>

Web Services and Service-Oriented Architectures <http://www.service-architecture.com/>

Service-Oriented Architecture Explained, Sayed Hashimi, O'Reilly http://www.ondotnet.com/pub/a/dotnet/2003/08/18/soa_explained.html

Succeeding at Service Oriented Architecture, Bill Ruh, ZDNet <http://www.zdnet.com.au/builder/architect/work/story/0,2000034884,20276810,00.htm>