IMS Abstract Framework: White Paper

Version 1.0

IPR and Distribution Notices

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the specification set forth in this document, and to provide supporting documentation.

IMS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on IMS's procedures with respect to rights in IMS specifications can be found at the IMS Intellectual Property Rights web page: http://www.imsglobal.org/ipr/imsipr_policyFinal.pdf.

If you wish to copy or distribute this document, you must complete a valid Registered User license registration with IMS and receive an email from IMS granting the license to distribute the specification. To register, follow the instructions on the IMS website: http://www.imsglobal.org/specificationdownload.cfm.

This document may be copied and furnished to others by Registered Users who have registered on the IMS website provided that the above copyright notice and this paragraph are included on all such copies. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to IMS, except as needed for the purpose of developing IMS specifications, under the auspices of a chartered IMS project group.

Use of this specification to develop products or services is governed by the license with IMS found on the IMS website: http://www.imsglobal.org/license.html.

The limited permissions granted above are perpetual and will not be revoked by IMS or its successors or assigns.

THIS SPECIFICATION IS BEING OFFERED WITHOUT ANY WARRANTY WHATSOEVER, AND IN PARTICULAR, ANY WARRANTY OF NONINFRINGEMENT IS EXPRESSLY DISCLAIMED. ANY USE OF THIS SPECIFICATION SHALL BE MADE ENTIRELY AT THE IMPLEMENTER'S OWN RISK, AND NEITHER THE CONSORTIUM, NOR ANY OF ITS MEMBERS OR SUBMITTERS, SHALL HAVE ANY LIABILITY WHATSOEVER TO ANY IMPLEMENTER OR THIRD PARTY FOR ANY DAMAGES OF ANY NATURE WHATSOEVER, DIRECTLY OR INDIRECTLY, ARISING FROM THE USE OF THIS SPECIFICATION.

Document Name: IMS Abstract Framework: White Paper

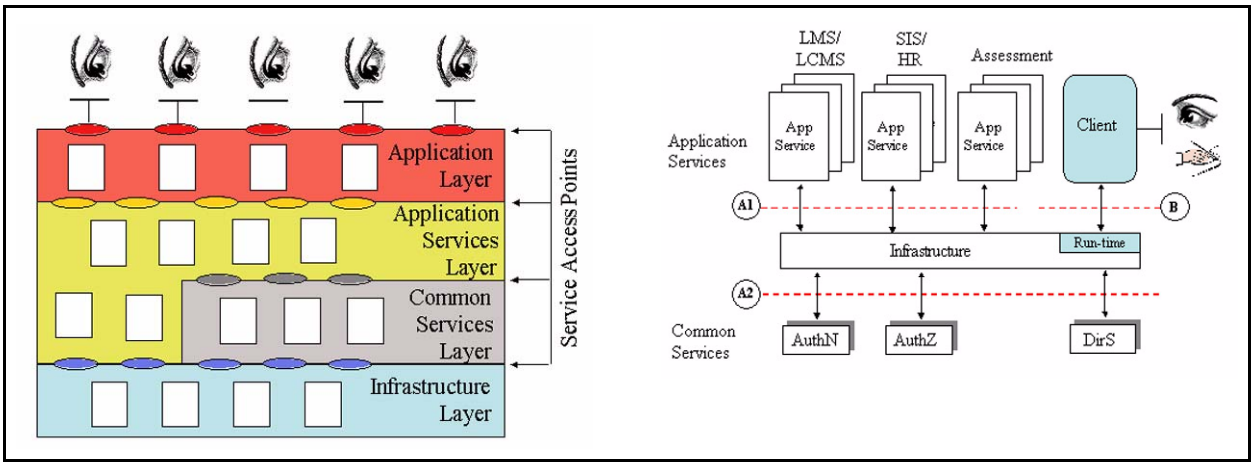
Date: 01 July 2003

Executive Summary

The IMS Abstract Framework is a device to enable the IMS to describe the context within which it develops its eLearning technology specifications. This framework is not an attempt to define the IMS architecture, rather it is a mechanism to define the set of interfaces for which IMS may or may not produce a set of interoperability specifications. The IMS Abstract Framework is so named because:

- It is an abstract representation of the set of services that are used to construct an eLearning system in its broadest sense;
- It is focused on the support of distributed electronic learning systems;
- It is a framework that covers the possible range of eLearning architectures that could be constructed from the set of defined services.

It is the intention of IMS that this Abstract Framework and the associated IMS specifications produced to realize the exchange of information between the identified services will be adopted in a manner suitable for a particular system requirement. The Abstract Framework is represented as a layered model, as shown in the figures below; this approach was derived from an extensive survey of the state-of-the-art for eLearning architectures.



The core features of the framework are:

- Application layer – the set of systems, tools and applications that are constructed from the suite of application and common services to provide a particular set of eLearning functionality;
- Application services layer – the set of entities that provide the eLearning specific services e.g., course management. It is these services that constitute the primary focus for IMS specification development;
- Common services layer – the set of entities that provide the generic services to be used by the application services e.g., authentication;
- Infrastructure layer – the underlying services that enable the exchange of the data structures in terms of physical communications, messaging and corresponding transaction needs;
- Service access points – the access points, or interface, to the corresponding service. Each access point provides access to one service capability;
- Entities – the processes that are used to represent a particular service. The realization of an entity with its service access points is termed a component and its abstract representation is called a Class.

The IMS specifications are used to define the data structures, and when appropriate the permitted behaviors for the exchange of those data structures, between the Components. This interoperability is defined in terms of the exchange of the appropriate XML instances as defined by their XML Schema Definitions. The IMS specification of a service is such that any number of bindings can be created e.g., native XML, web services description language, Java, etc. It is

the corresponding information and behavioral descriptions that enable interoperability between these different bindings. The Abstract Framework also describes the transport mechanisms that can be used to exchange the XML documents.

The adoption of the Abstract Framework will be based upon the creation of suitable Domain Profiles. Domain profiling is the process that is undertaken to define which specifications and the detailed usage of the data objects within each specification are to be adopted to provide a particular solution. The domain profiling process is based upon:

- Identification of the appropriate specifications by matching their functional capabilities to the requirements of the application;
- Refinement of each IMS specification made by increasing the constraints on the information model to more accurately reflect the needs of the domain. This refinement includes definition of profile-specific extensions;
- The new binding for the domain profile should now be produced. In general, a instance document for a particular profile should validate against the new binding control document for the profile and the binding control document for the corresponding IMS specification. The profile-specific binding control document will be capable of identifying errors in the instances due to the increased constraints on the permitted data content;
- The corresponding strong conformance statement and certification test specification should now be produced for the domain profile.

This is a 'living' document i.e., it is not archival in nature. Our ideas for various parts of the Abstract Framework are constantly being developed and so the information contained herein should always be considered in that context. This white paper is one of a set of closely related documents, the others being:

- The IMS Abstract Framework: Glossary – the definitions of the key terms used throughout the ALF and the associated specifications;
- The IMS Abstract Framework: Applications, Services, and Components – the identification of the set of applications and services and their corresponding implementation components which can be used to support eLearning system interoperability (the separation of the detailed descriptions of the applications, services and components allows the details of this white paper to focus on the abstract representation itself);
- The IMS Learning Activity Model (LAM) – the description of the underlying content model and the learner design mechanisms to be adopted for the provision of learning content (this document will not be available until mid 2004);
- The IMS Use Case Portfolio – the collection and collation of the core set of use cases that reflect the interoperability needs within eLearning systems (work on the collection of these use cases is underway);
- The IMS Specification Development Methods and Best Practices – the identification of the methods and best practices that must be used when developing and documenting IMS specifications.

'Domain Conformance' will not be defined with respect to an IMS specification. These specifications contain too many optional features and are subject to regional and sector specific amendments e.g., the inclusion of the appropriate vocabularies. Therefore, conformance will be against a particular Conformance Profile that has been derived from the corresponding Domain Profile. Conformance certification is considerably easier when all of the functionality is mandatory and so it is important for Conformance Profiles to remove as much optional functionality as possible. In turn, there is a generic requirement that for a Domain Profile to be 'Strictly Conforming' to an IMS specification then an XML instance of the domain profile must also validate against the corresponding IMS specification.

Table of Contents

1. Introduction

1.1 Scope and Context

The IMS Abstract Framework (IAF) is a device to enable the IMS to describe the context within which it will continue to develop its eLearning technology interoperability specifications. This framework is not an attempt to define the IMS architecture, rather it is mechanism to define the set of interfaces for which IMS may or may not produce a set of interoperability specifications. In the cases where IMS does not produce a specification then every effort will be made to adopt or recommend a suitable specification from another organization. The IMS Abstract Framework is so named because:

- It is an abstract representation of the services and their interfaces that are used to construct an eLearning system in its broadest sense;
- It is focused on the support of distributed electronic learning systems;
- It is a framework that covers the possible range of eLearning architectures that could be constructed from the set of defined services and interfaces.

It is the intention of IMS that this Abstract Framework and the associated IMS specifications will be used as the starting point for the definition of many different eLearning systems. Each eLearning system will only adopt those parts of the IMS specifications that are of immediate use. This process is called the *Profiling* of the framework to define an architecture or reference model for a particular implementation domain; the process of profiling a particular specification is similar to refining the best practice to produce mandatory facets for an implementation domain. The *Profiles* can then be used to support conformance testing, based upon a particular conformance-profile for that domain, to ensure compliance to a particular requirement.

1.2 Using this Document

This is a 'living' document i.e., it is not archival in nature[1]. Our ideas for various parts of the IAF are constantly being developed and so the information contained herein should always be considered in that context. This white paper is one of a set of closely related documents, the others being:

- The IMS Abstract Framework: Glossary [IMS, 03a] – the definitions of the key terms used throughout the ALF and the associated specifications;
- The IMS Abstract Framework: Applications, Services, and Components [IMS, 03b] – the identification of the set of applications and services and their corresponding implementation components which can be used to support eLearning system interoperability (the separation of the detailed descriptions of the applications, services and components allows the details of this white paper to focus on the abstract representation itself);
- The IMS Learning Activity Model (LAM) – description of the underlying content model and the learner design mechanisms to be adopted for the provision of learning content (this document will not be available until mid 2004);
- The IMS Use Case Portfolio – the collection and collation of the core set of use cases that reflect the interoperability needs within eLearning systems (work on the collection of these use cases is underway);
- The IMS Specification Development Methods and Best Practices [IMS, 03c] – the identification of the methods and best practices that must be used when developing and documenting IMS specifications.

At the current time only the Glossary and the Application, Services, and Components documents are available as support to this white paper. During the next twelve months the other documents will become available and consequently new releases of the IAF may be required. This white paper is constructed in three sections:

- Basic structure of the IAF – definitions of the requirements and the corresponding overall structure of the IAF;

1. One immediate example of this is the IAF Common Services development activity that will be undertaken in late 2003. This activity will be responsible for documenting and describing the best practices to be adopted when using a predefined set of Common Services. These Common Services will be taken from other specification sources e.g., OKI, and so the IMS will be showing how the OKI APIs can be used with IMS Application Services.

- Abstraction to implementation – the process through which the IAF can be used to define an architecture for implementation;
- Detailed background information – a series of appendices that contain the detailed technical information that forms the background to the IAF.

The IAF provides the context for the development of the next series of IMS specifications. The IMS specification development process uses the IAF to:

- Identify some of the eLearning components that are to be specified. Once these components have been specified then any appropriate amendments to the services identified in the IAF will also be made;
- Identify the components that will be adopted from other specification and standards activities. These components may be tailored to enhance their adoption for eLearning;
- Demonstrate the process by which the UML representations of the information model descriptions of the components are converted to their XML binding equivalents;
- Demonstrate the ways in which the 'sea of components' is combined and profiled to support the specification of eLearning architectures.

As with all of the IMS specifications there is a need to develop best practice for each of the activities summarized above. The IAF will be updated on a regular basis, typically once every six months, and as a part of other activities e.g., development of the learning activity model. While these updates will include important new information, the underlying principles of interoperable, component-based service definitions within a layered framework will not be changed.

1.3 Structure of this Document

The structure of this document is:

2. Requirements Definition	The definition of the requirements in terms of the underlying principles, scoping and use cases that define the context within which the abstract framework was developed;
3. The Abstract Framework	The definition of the overall structure of the abstract framework. This shows the relationship between the various sub-structures within the framework;
4. Applications, Services, and Components	The description of the ways in which the requirements and capabilities of applications, services and components can be defined;
5. Infrastructure layer	The detailed description of the Infrastructure Layer within the abstract framework. This includes the definition of the sub-layers and the identification of the core technologies that can be adopted to realize the corresponding functionality;
6. Service Bindings	A description of the different service bindings of the abstract framework and the implications for implementations based on these bindings;
7. Profiling and Conformance	A review of the profiling of the abstract framework and the accompanying IMS and other specifications to support a particular application, to define a specific architecture and to enable conformance;
Bibliography	The set of documents that are referenced within this document or which provide context to the contents of this document;
Appendix A – List of Acronyms	The list of acronyms used throughout this document;
Appendix B – IMS Specification Roadmap	A visualization of the development roadmap of the set of IMS specifications released and under development and their relationship and adoption timeline by other specification and architecture definition initiatives;
Appendix C – eLearning and Related Architectures & Models	Brief descriptions of each of the architectures and models currently in use or under development within the eLearning domain.

2. Requirements Definition

2.1 Historical Context

During the past 30 months IMS has released a unique set of interoperability specifications within the eLearning technology community (that work was founded upon an extensive range of experimental activities undertaken during the prior 24 months). During this period the various eLearning technology activities have begun a slow convergence that has been the result of extensive work behind the scenes by IMS and other organizations. It is now apparent that the current set of IMS activities and processes need to evolve to tackle the next series of technical issues [IMS, 02a]. As a consequence, IMS has produced its Abstract Framework to the context within which:

- The new set of IMS specifications are developed;
- The migration from the current to the new specifications can be defined and managed;
- The relationship between IMS and non-IMS specifications can be explained and clearly demonstrated;

The current set of IMS specifications released and under development is listed in Table 2.1.

Table 2.1 - The set of IMS specifications.

IMS Specification	Description	Version	Release Date
Meta-data	The tagging of any learning content.	1.2.1	December 2001
Enterprise	The exchange of Person and Group information.	1.1	July 2002
Learner Information Package	To exchange a Person's profile or life-long learning log.	1.0	March 2001
Question & Test	Used to support computer-based Assessment.	1.2.1	April 2003
Content Packaging	Exchanging content with its associated learning structures.	1.1.3	May 2003
Simple Sequencing	Adaptive learning routes through a set of learning content.	1.0	March 2003
Reusable Definition for Competency and Educational Objectives	A syntax for the description of competencies.	1.0	August 2002
Learning Design	The unified representation of different learning activities.	1.0	February 2003
Digital Repositories Interoperability	Search and retrieval using meta-data tagged resources distributed across a federated set of databases.	1.0	January 2003

The IMS also has a set of guidelines that have been released to act as support documents to the specifications. These guidelines are:

- IMS Persistent Location-independent Resource Identifier (PLIRI) – the recommended IMS generic user identifier format [IMS, 01a];
- Using IMS Content Packaging to Package Instances of LIP and Other IMS Specifications: An Implementation Handbook – recommendations for the usage of the Content Packaging specification as a generalized packaging mechanism [IMS, 01b];
- Accessibility Guidelines – guidelines for the development of accessible content [IMS, 02d].

2.2 Underlying Principles

2.2.1 Interoperability

The specifications are focussed on the exchange of information between systems. The specifications make no assumptions on how the data is managed within the communicating systems.

2.2.2 Service-Oriented

The exchange between the systems is to be defined in terms of the services being supplied by the collaboration of the systems. This service collaboration could take many forms as well as being based upon peer-to-peer and client-server techniques.

2.2.3 Component-Based

The set of services will be supplied as a 'sea of components' that can be mixed and matched to form a particular service. A single component may provide all or a sub-set of a service but it will not provide more than one service.

2.2.4 Layering

The total set of services required to make an eLearning system will be modelled as a set of layers with each layer providing a clearly defined set of services. A particular layer will make use of the services in the layer below it and will provide services to the entities in the layers above it.

2.2.5 Behaviors and Data Models

A service will be defined in terms of its behaviors and data model. The behaviors will cause changes in the state of the data model and the state of the data model will only be altered as a result of a clearly defined behavior. It may not be appropriate to define every service in terms of its behavior and in these cases only the relevant data models will be developed.

2.2.6 Multiple Bindings

The information model is to be defined and represented using an established syntax and semantics. This will then enable automatic mapping of the information model into a range of different bindings. The bindings of immediate importance are XML and Web Services Description Language (WSDL), however Java bindings will also be supported.

2.2.7 Adoption

New specifications will only be created as required. Whenever possible, appropriate specifications will be adopted from wherever and either used 'as is' or modified to suit a particular set of requirements.

2.3 Key Use Cases

The service range covered by the abstract framework is based upon a core set of use cases that have been collected from:

- Higher education, community colleges and further education;
- Schools – education in the age range 4-16 e.g., K-12;
- Corporate training – this includes activities such as Professional Certification and Continuing Professional Development.

In all cases these use cases have been taken from different parts of the world, with a particular focus on North America, Europe and Asia. This ensures that the internationalization aspects of eLearning services are considered.

Note: The Use Case Portfolio activity currently underway will complete its work in late 2003. That portfolio will be used as the basis for further material in this sub-section. The key use cases will be used to identify the set of applications and services that need to be supported.

2.4 Scoping of the Abstract Framework

The scope of the abstract framework is set by the requirements generating by analyzing:

- The use cases as described within the Use Case Portfolio;
- The current set of IMS specifications;
- Reference models and architectures already available or undergoing development e.g., SCORM, SIF, OKI, etc.

Note: The Use Case Portfolio activity currently underway will complete its work in late 2003. That portfolio will be used as the basis for further material in this sub-section.

3. The Abstract Framework

3.1 eLearning Systems

A review of a wide range of eLearning systems is summarized in Appendix C and a synthesis of these is shown in Figure 3.1. Figure 3.1 is a logical architecture based upon layer abstraction. The equivalent physical architectural model is shown in Figure 3.2.

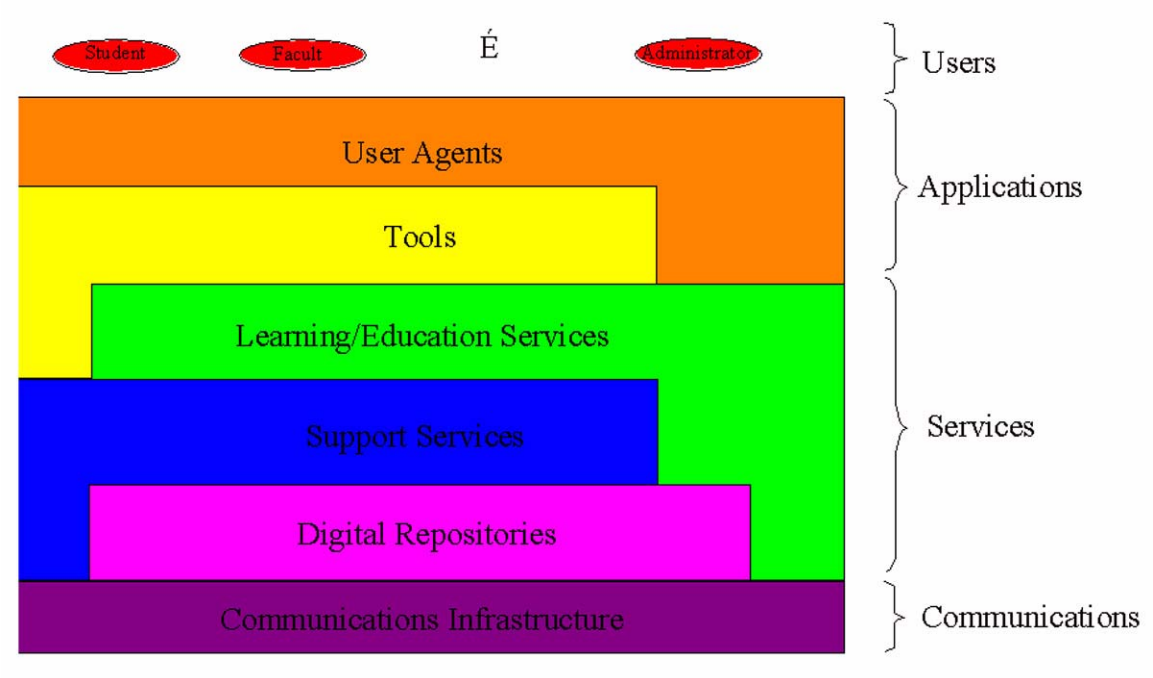


Figure 3.1 - A logical architecture for an eLearning system.

The logical representation consists of the following layers:

- Users – the set of users of the eLearning system e.g., students, administrators, teachers, etc. Users gain access to the system through an appropriate 'user agent';
- User agents (see Appendix C5 for more details) – the agents that deliver the services to the users themselves;
- Tools (see Appendix C5 for more details) – the tools enable the different services to be accessed in a convenient and 'user-friendly' manner. This includes assessment, tutoring, simulation, etc.
- Learning/education services (see Appendices C1, C5 and C7 for more details) – the learning services themselves;
- Support services (see Appendices C1, C5 and C7 for more details) – common services that are also required by non eLearning systems e.g., authentication, resource discovery etc.
- Digital repositories (see Appendix C7 for more details) –
- Communications infrastructure – the basic networking and data transport services that deliver information end-to-end.

The physical representation (Figure 3.2) consists of the following core structures:

- Core network – the primary network that interconnects the core computer systems. This includes what would be termed the 'Internet'. This is a part of the 'Communications Infrastructure' in the logical model;

- Access network – the network that links the delivery devices to the core network. Typical examples are cable networks, wireless networks, PSTNs, etc. This is a part of the 'Communications Infrastructure' in the logical model;
- Federated digital repositories – the series of digital resources that are available in a variety of digital repositories, databases, web servers, etc. This is the manifestation of the 'Digital Repositories' in the logical model;
- Service delivery engines – the systems that are responsible for the provision of the full series of learning services. This corresponds to the 'Learning/educational Services' and 'Support Services' in the logical model;
- Delivery devices – the devices and their client support that deliver the learning material to the user. This corresponds to the 'Tools' and 'user Agents' in the logical model.

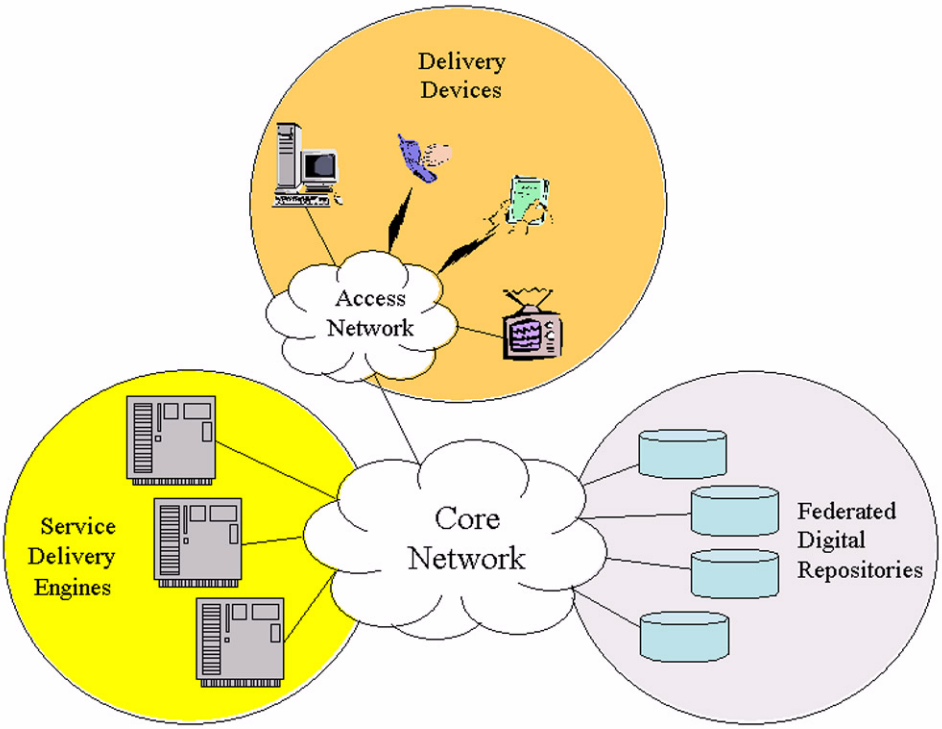


Figure 3.2 - A physical architecture for an eLearning system.

The abstract framework has to be designed such that it can be used to represent a wide range of architectures.

3.2 The Functional Model

There are several possible functional perspectives of an eLearning system. The two perspectives that are introduced herein are (it is these perspectives that dominate the IMS specification activity):

- Content – this describes how content and related information flows and is managed within an eLearning system;
- Individual – this describes how information about an individual flows and is managed within a eLearning system.

These functional perspectives identify where the interoperability points are within an eLearning system i.e., where the arrows are used in Figures 3.3 and 3.4. If these exchange points are exposed as external communications between different systems then an interoperability specification is required.

3.2.1 The Content Perspective[2]

A functional representation of the flow and management of content and related information within an eLearning system is shown in Figure 3.3.

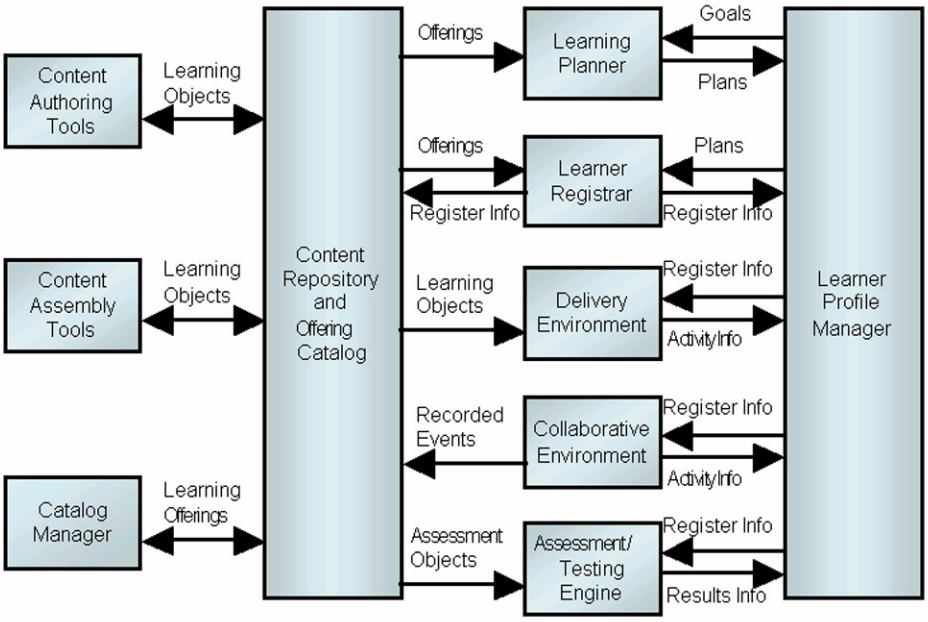


Figure 3.3 - A functional model for using learning material.

There are four key functional themes:

- Content repository and offering catalogue – the storage of the learning materials and the catalogue listing and describing those materials;
- Learner profile manager – management of the learner's profile;
- Content repository and offering catalogue source generators – this includes the set of tools that generate the actual learning content and the catalogue that describes those materials;
- The core activities that generate changes in the learner's profile as a result of undertaking learning. These activities are:
  - Planning of what and when learning needs to be undertaken
  - Registration on the appropriate course offerings
  - Learning with the materials
  - Collaboration with other participants on the course(s)
  - Assessment of what has been learnt.

3.2.2 The Individual Perspective

A functional representation of the flow and management of personal and related information within an eLearning system is shown in Figure 3.4

2. This work is taken, with permission, from a SUN white paper that was produced by G.Collier [SUN, 02].

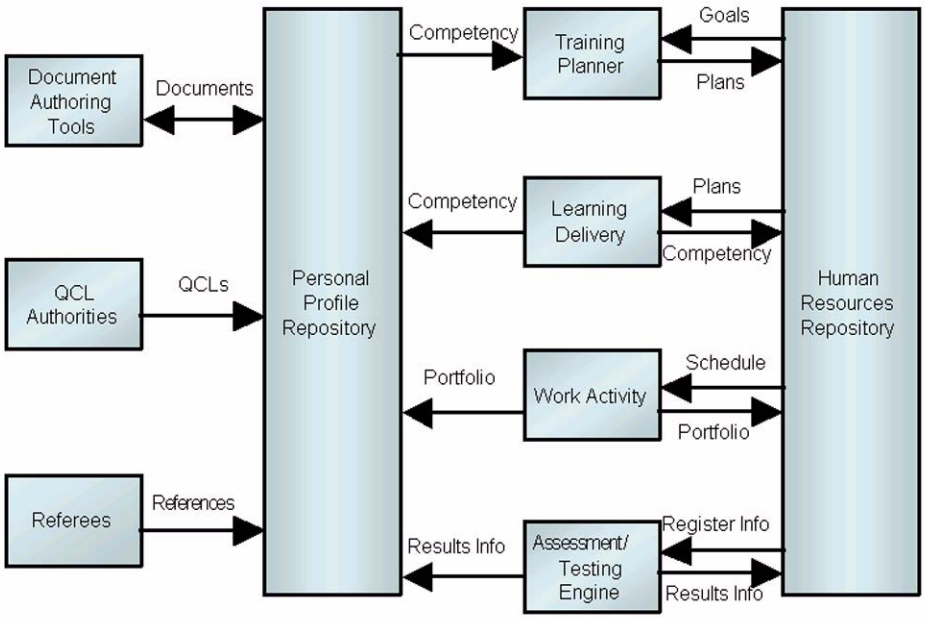


Figure 3.4 - The functional person model.

Again, there are four key functional themes:

- Personal profile repository – the management of the learner's profile in some personally maintained repository;
- Human resources repository – the management of the learner's profile in the institution's human-resources system;
- Personal information and work source generators – this includes the set of tools and sources that generate information about the person;
- The core activities that generate changes in the learner's profile as a result of undertaking learning. These activities are:
  - Planning of what and when training/education needs to be undertaken
  - Delivery of the training (corporate and training focus) and education (qualification focused)
  - Normal work activity that generates a range of materials that can for a portfolio of experience
  - Assessment of what has been learnt.

3.3 The Layered Model

The abstract learning framework can be represented as a layered model, as shown in Figure 3.5, consisting of four layers:

- Application layer – this is a tool, system, agent, etc. that presents the appropriate application services to the user i.e., an application manages the user interface. The application may use one or more application services but whenever possible the system composition should be hidden from the user;
- Application Services layer – a set of services that provide the required eLearning functionality to the applications. An application service may make use of one or more common services. Distributed application services communicate using via the Infrastructure Layer;
- Common Services layer – a set of services that are available to the application services. Common services may use other common services. Therefore, a common service is available to any other service;

- Infrastructure layer – this provides the end-to-end transaction and communications services for the application and common services.

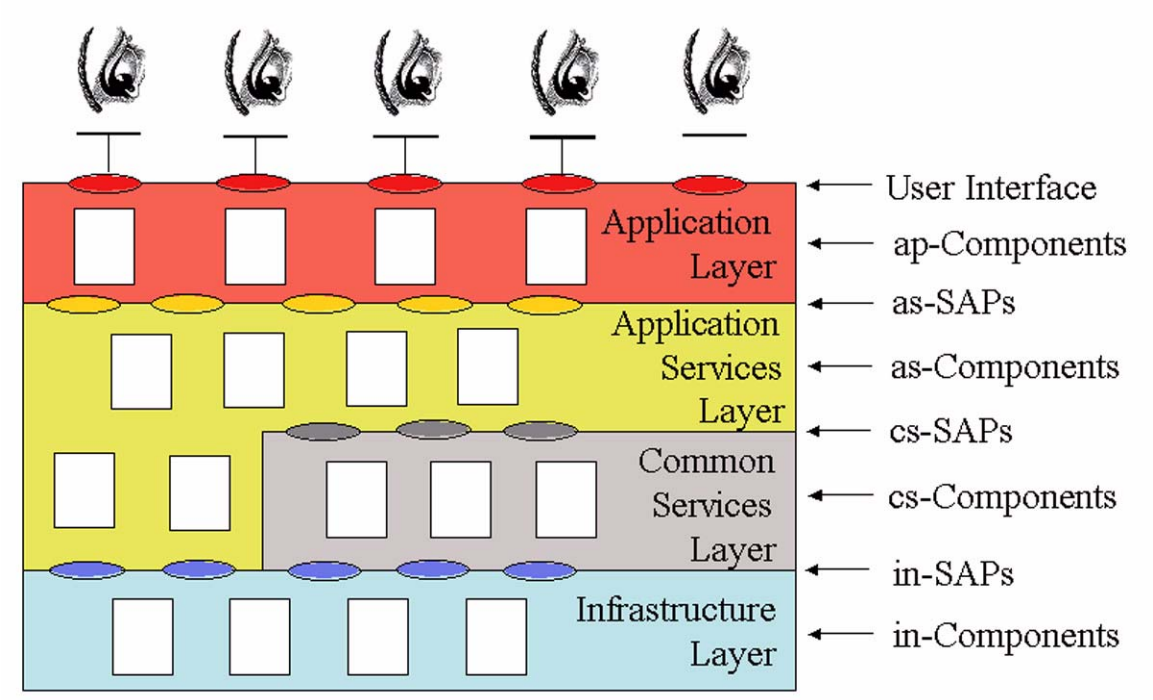


Figure 3.5 - The layered model.

Access to a service is through the appropriate Service Access Point (SAP). Each service has a single SAP. A Component may support one or more SAPs (in an object oriented representation, a SAP could be supported by one or more operators where the class is itself the definition of the service).

In a distributed implementation of this layered framework, as typified in a webs services environment, the interaction between the services would as shown in Figure 3.6. In this interaction framework there are three categories of interface that must be supported by the Infrastructure layer namely:

- The Application Services interface (A1) – this interface is used to provide interoperability between common application services e.g., Enterprise-to-Enterprise systems. One form of the interface is based upon XML-messaging;
- The Common Services interface (A2) – this interface is used to provide interoperability with the set of common services that are made available to the application specific services e.g., authentication and authorization for an Enterprise system;
- The run-time interface (B) – this interface is used to interconnect the client's run-time application with the remote service provider.

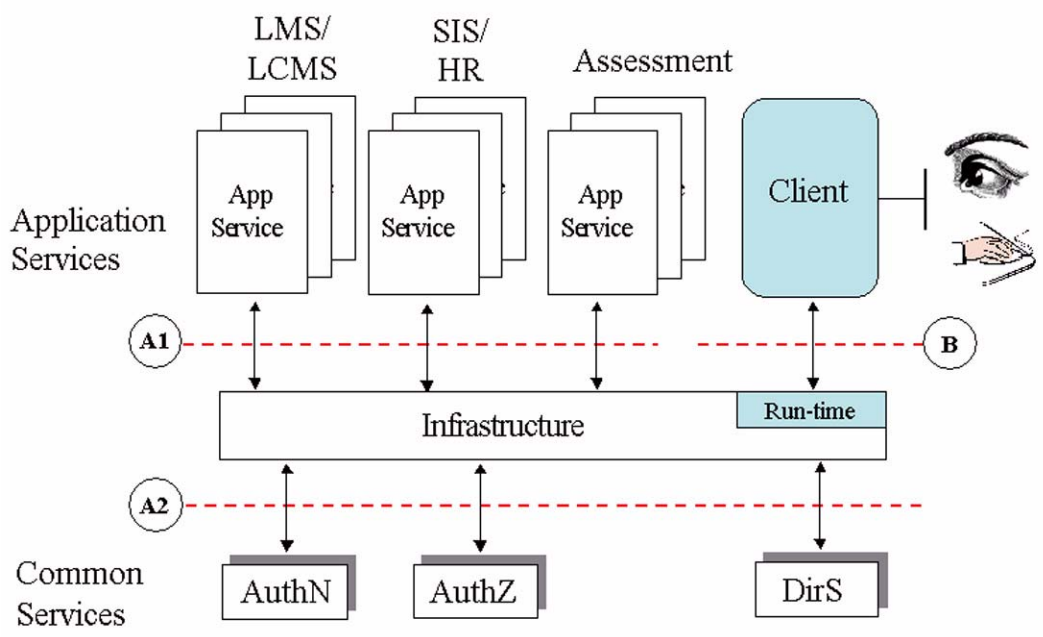


Figure 3.6 - Service interaction through the layers.

Figure 3.6 shows that there are two types of interaction behavior that need to be specified to ensure interoperability, namely:

- Message passing – where information is exchanged between systems using some form or another of message exchange. The content and sequence of the messages define the expected behavior. The communicating systems are expected to process and generate the messages as a response to known and understood events. The systems must be capable of handling unknown error conditions;
- Run-time – where the end systems have to reliably operate on information using some predefined algorithm. The data will determine the outcomes of the algorithms but the behavior is well defined for all possible outcomes.

3.4 The Service Abstraction

One of the design principles for the IAF is the adoption of service abstraction to describe the appropriate eLearning functionality. The service is hidden behind a SAP and can only be accessed by using the SAP (an API could be one way in which the SAP is actually implemented). However, even though the service provision is hidden behind the SAP, an implementation requires that the behavioral capabilities of the service be defined. Once again an object-oriented representation is adopted. In Figure 3.7 we have a schematic representation of a service, namely:

- The service has a clearly defined service access point. Each service has only one SAP. The SAP is now defined in terms of its constituent objects and behaviors;
- The SAP may consist of one or more objects and each object will, in general, will have more than one operator. Each object is defined using a class definition and consists of a group of attributes and operators. The operators describe how the state of the attributes may be changed. The set of behaviors permitted for each class must also be defined. These behavioral definitions ensure that any implementation of the class provides the same predicted behaviors for the same trigger events. Both the classes and their behaviors are defined in an implementation-independent manner;
- The ‘Private Service Implementation’ is beyond the scope of the IAF. The only requirements on this are that it provides all of the appropriate behavioral characteristics of the service and nothing more. This means that an implementation can be changed e.g., for optimization, without requiring a change to any of the systems using that service.

This approach means that every service (application and common) must be defined using this form of abstraction. In many cases the services interact with each other e.g., an application service will use a common service. This interaction is reflected by the service invoking the SAP of the required service.

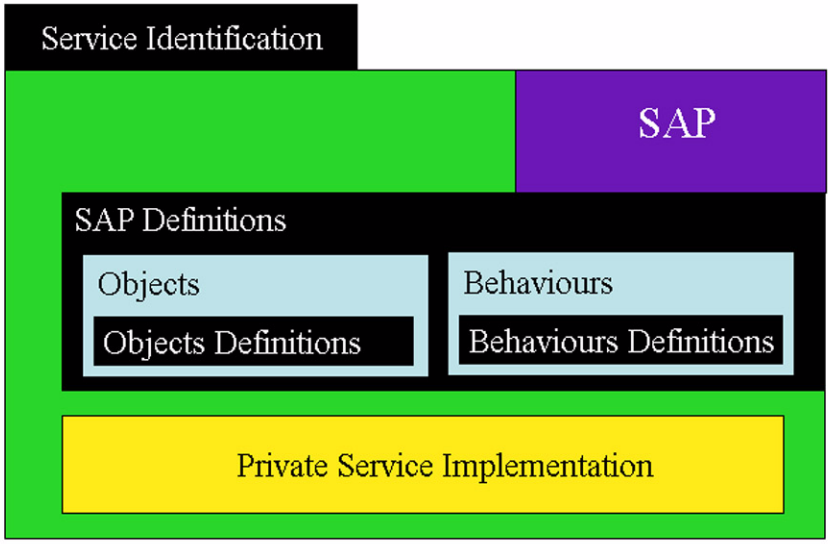


Figure 3.7 - The service abstraction.

The adoption of UML as our representation framework means that each service must be defined by:

- A single class package whose name reflects the name of the service. The functional capabilities of that class package are based upon the aggregation and inheritance of other Classes. The actual SAP is represented by 'Interface Classes'. The relationship between the classes will be shown using Class Diagrams and Package Diagrams;
- Each Interface Class has a set of operators which are the definition of the SAP i.e., the SAP is a single logical definition of one or more operators;
- The set of data structures for the object are defined using the attributes of the classes. All of the data structures must be strongly typed and their domain must be defined as tightly as possible;
- Whenever possible the behaviors will be defined using state diagrams/tables;
- The interactions between the classes will be defined using Sequence Diagrams;
- Implementation details will not be addressed thus Component Diagrams, Deployment Diagrams, etc. will not be produced.

3.5 The Run-time Environment

Few of the current IMS specifications address the issue of content run-time interoperability. This is handled by other specifications e.g., the AICC CMI. However, IMS need to look at several run-time issues such as content-to-content communication, web service based content launch, etc. A schematic representation of the run-time architecture for content is shown in Figure 3.8.

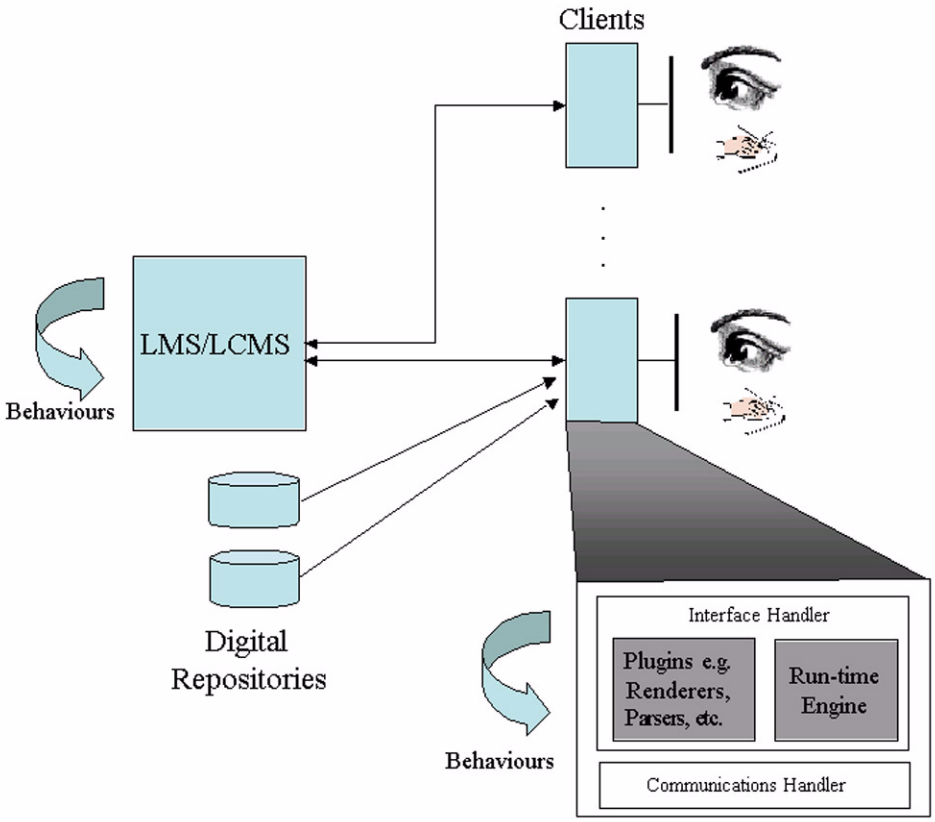


Figure 3.8 - A run-time environment model.

Figure 3.8 shows that the interactions between a content server (LMS, LCMS, etc.) and the delivery clients (typically this includes the usage of a browser); note that in general the server will have a significant number of concurrent users who will be using different learning materials and who may or may not wish to halt and restart their activities at various times. The client should be able to 'pull-in' other content as and when required without requiring communication through the server. The client should also be able to launch multiple concurrent 'learning objects' that co-operate to provide the full learning experience. Therefore the client will need a range of plug-ins that will enable it to parse the content, render the content, etc. before actually playing the material to the user.

This scenario requires clear definition of the internal behaviors of the server and client in response to the user's interactions with the content. The nature of the content being exchanged between the server and client needs to be addressed – at present this is either web pages with or without other material that can be played with the appropriate plug-in or it is executables that can run natively on the client device. The increasing adoption of XML may mean that it is possible to encode content natively in XML – clearly not all content will be XML encoded e.g., video.

3.6 UML Representation

The layered representation shown for the IAF in Figure 3.5 can be redrawn in UML as shown in Figure 3.9. In many regards this is a more appropriate representation as the IAF does not enforce a robust layering in the way that it can be implemented.

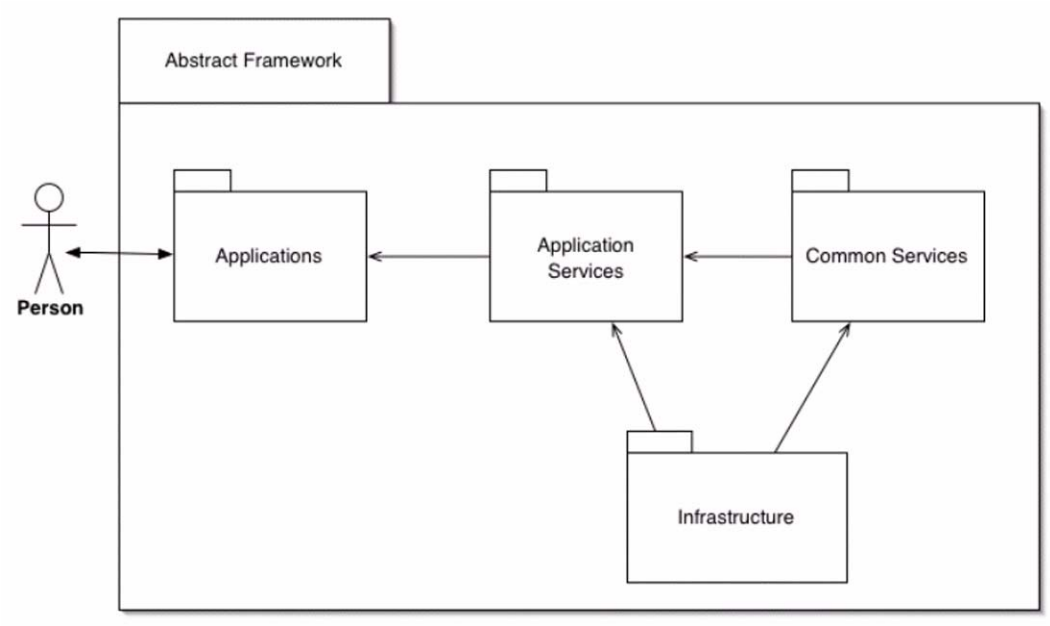


Figure 3.9 - A UML package diagram of the abstract framework.

The object oriented representation in Figure 3.9 shows the dependence between the different ‘layers’ or the services.

4. Applications, Services, and Components

The set of applications, services and components that can be developed using this abstract framework are detailed in [IMS, 03b]. In this Section we are going to describe the characteristics that are used to categorize an application, application service, common service and component.

4.1 Applications

In the context of eLearning, an application is the manifestation of a system or tool that delivers one or more eLearning application services to a user. The key differences between an application and an application service are that an application provides an interface to the user(s) and that it may use more than one application service. A template for the brief description of an application is shown in Table 4.1.

Table 4.1 - A template for the description of an application.

Application	
Application Description:	A description of the application.
User Interface:	Definition of the form and structure of the user interface.
Application Service Dependencies:	Identification of the application services that are used to create the application.

Note: IMS will not be undertaking the specification of applications as a part of its specification development activity.

4.2 Application Services

In the context of eLearning, an application service is responsible for delivering a set of features that support the manipulation of a set of data objects for a common purpose e.g., an Assessment Service for online assessment. An application service may be created by aggregating other application services. A template for the brief description of an application service is shown in Table 4.2.

Table 4.2 - A template for the description of an application service.

Application Service	
Service Description:	A description of the service.
Service Access Point:	Definition of the service access point i.e., the supported operators.
Core Attributes:	The definition of the core data objects that can be manipulated via the operators.
Core Components:	Identification of the component(s) that will be used to realize the application service.
Common Service Dependencies:	Identification of the common services that are invoked by the application service.
Infrastructure Dependencies:	Definition of the infrastructure required for the support of the application service.

Note: The core specification development activity undertaken by IMS will focus on the development of application services.

4.3 Common Services

A common service is responsible for delivering a set of features that are relevant to more than one application service and which, in many cases, are also required by non-eLearning systems e.g., an authentication service. A template for the brief description of a common service is shown in Table 4.3.

Table 4.3 - A template for the description of a common service.

Common Service	
Service Description:	A description of the service.
Service Access Point:	Definition of the service access point i.e., the supported operators.
Core Attributes:	The definition of the core data objects that can be manipulated via the operators.
Core Components:	Identification of the component(s) that will be used to realize the common service.
Infrastructure Dependencies:	Definition of the infrastructure required to support the common service.

Note: IMS will not be undertaking the specification of common services as a part of its specification development activity.

4.4 Components

A component is the realization of an abstract model. IMS produces specifications of services that can be released as components that provide interoperability capabilities for eLearning systems. A template for the brief description of a component is shown in Table 4.4.

Table 4.4 - A template for the description of a component.

Component	
Component Description:	A description of the component.
Source Specification(s):	The data objects for the original specification(s) and standard(s) from which this component is derived.
Interfaces:	Definition of the interfaces that expose the support functionality.
Core Attributes:	The definition of the core data objects that can be manipulated via the interfaces.
Protocol Requirements:	Definition of the protocols required to provide intra and inter component communications.
Component Dependencies:	Identification of the other components that are required by this component.

An IMS specification of a component is the UML-based representation of the abstract application programming interface. This representation describes the set of classes for the component and defines the data and information that is exchanged between the corresponding objects.

5. Infrastructure Layer

5.1 The Composition of the Infrastructure Layer

The IMS specifications are focused on data exchange interoperability. To this end they define a data model of the information to be exchanged and a behavioral model that encapsulates the data model and constrains the way in which the data can be manipulated. An IMS information model is the manifestation of this behavioral and data description and an information model will consist of one or more IMS Components (these will realize either all, or part of, an Application Service). These components can then be realized in a variety of ways; the defined IMS method is as an XML-based binding. As such, this binding describes the way in which the data is exchanged in the form of XML messages/documents, however the actual transfer of these structures requires, at the very least, an appropriate communications system.

The exchange of the data between the XML components within the abstract framework is defined through the Infrastructure Layer. A schematic representation of the system components in this infrastructure description is shown in Figure 5.1. This representation assumes that the system is loosely coupled e.g., as per web services.

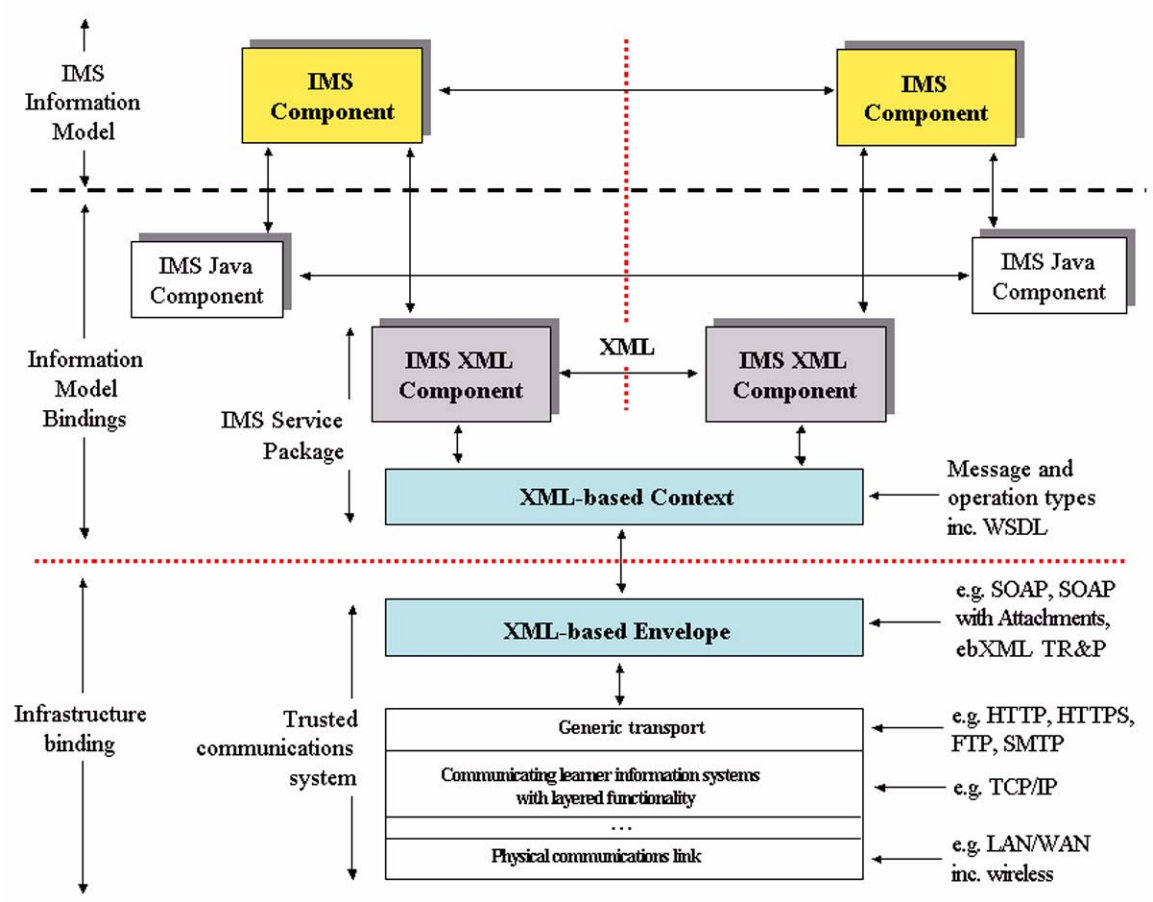


Figure 5.1 - The infrastructure layer.

In Figure 5.1 the key parts of this infrastructure are:

- IMS XML Components – the application and common services components that are combined to create the e-learning system required. It is assumed that these components exchange information in the form of XML documents;

- XML-based Context – the XML documents are transformed to XML messages which are then mapped onto a common XML messaging infrastructure that is designed to support the required end-to-end services e.g., reliable data transfer, datagram, publish and subscribe, etc. The preferred context mapping language is the Web Services Description Language (WSDL). This could be based upon the W3C XML protocol (XP);
- XML-based Envelope – the common XML messaging system can be supported using several types of XML envelope encapsulation e.g., SOAP, SOAP with attachments (used to support file attachments such as images, etc.), ebXML, etc. WSDL supports SOAP, 'http-Get' and 'http-Post' transport mechanisms;
- Generic transport – the envelope is then transported across the network using an appropriate end-to-end file transfer protocol e.g., SMTP, FTP, HTTP, etc.;
- Communications network – this is the actual data network that is used to physically transport the data from one system to another. This will almost certainly be based upon the ubiquitous Internet Transmission Control Protocol/Internet Protocol (TCP/IP) combination providing seamless interworking between wire-line and wireless networks.

5.2 XML-Based Context Sub-Layer

The XML-based Context sub-layer is responsible for mapping the behaviors and associated operators of the XML components onto an appropriate sequence of XML messages. In principle there are a very large number of possible operators for even just a few components. There is however a number of common operators as described in Table 5.1. Table 5.1, or the Beshears Matrix (taken from work produced by the Enterprise working-group) was derived to identify the generic types of operation/methods required to enable the exchange of objects between an initiating system (initiator) and the responding system (respondent).

The corresponding state diagram requires that the completion of each method should be accompanied by the definition of which of the next methods can be invoked successfully on the same object.

The set of XML messages required to support the operators in Table 5.1 must include the following range of functionality:

- Operator action invocation message – the message sent from the initiator to the respondent invoking the functionality of the operator;
- Operator action invocation acknowledgement message – the message sent from the respondent to the initiator confirming receipt of the original operator action invocation message;
- Operator action invocation response message – the message sent from the respondent to the initiator containing the requested information.

Any of the messages may consist of the actual control and core data structures plus any associated resources e.g., jpeg files, etc.

Table 5.1 - The Beshears Matrix.

	Respondent is the owner of the data. Request an action.	Initiator is the owner of the data. Announce data availability for others.	Initiator is the owner of the data. Request synchronization.
Create	Create the object and return the allocated identifier. The object data structure is populated with the supplied data (only the mandatory parts are supplied). The respondent owns the object. **createRequestObj()**	Broadcasting of the fact that an object has been created with the assigned identifier. The initiator owns the object. **createAnnounceObj()**	Create the object using the supplied identifier. The object data structure is initialized as empty. The initiator owns the object. **createRequestObj()**
Delete	Delete the object. **deleteRequestObj()**	Broadcasting of the fact that the object has been deleted by its owner. **deleteAnnounceObj()**	Delete the object. **deleteRequestObj()**
Update	Change the object identified by the supplied ID. This could be a partial update of the component. This is a destructive write but only the fields changed are updated; it is not required to have sent a copy of the old data. **updateRequestObj()**	Announcement that the Object has been changed. This includes the information of which part of the object has been changed. **updateAnnounceObj()**	Change the object identified by the supplied ID. This could be a partial update of the component. This is a destructive write but only the fields changed are updated; it is not required to have sent a copy of the old data. **updateRequestObj()**
Append	This is a functional adding of a new part to the identified parent. The commonality with 'update' equivalent needs thinking thru. This will need an equivalent delete part of object. **appendRequestObj()**	Announcement that the Object has been changed. This includes the information of which part of the object has been changed. This will need an equivalent delete part of object. **appendAnnounceObj()**	This is a functional adding of a new part to the identified parent. The commonality with 'update' equivalent needs thinking thru. This will need an equivalent delete part of object. **synchronizeRequestObj()**
Read	Read the part of the object identified. If read 'All' then only return those parts permitted for view by the requester. Need to consider whether we get flat data or just the reference. **readObj()**		
Query	This provides the capability to ask for all objects that comply with a particular set of search criteria. The details for all of the objects that comply with the search criteria are returned. **queryObj()**		

The equivalent state diagram for the operators shown in Table 5.1 is given in Figure 5.2.

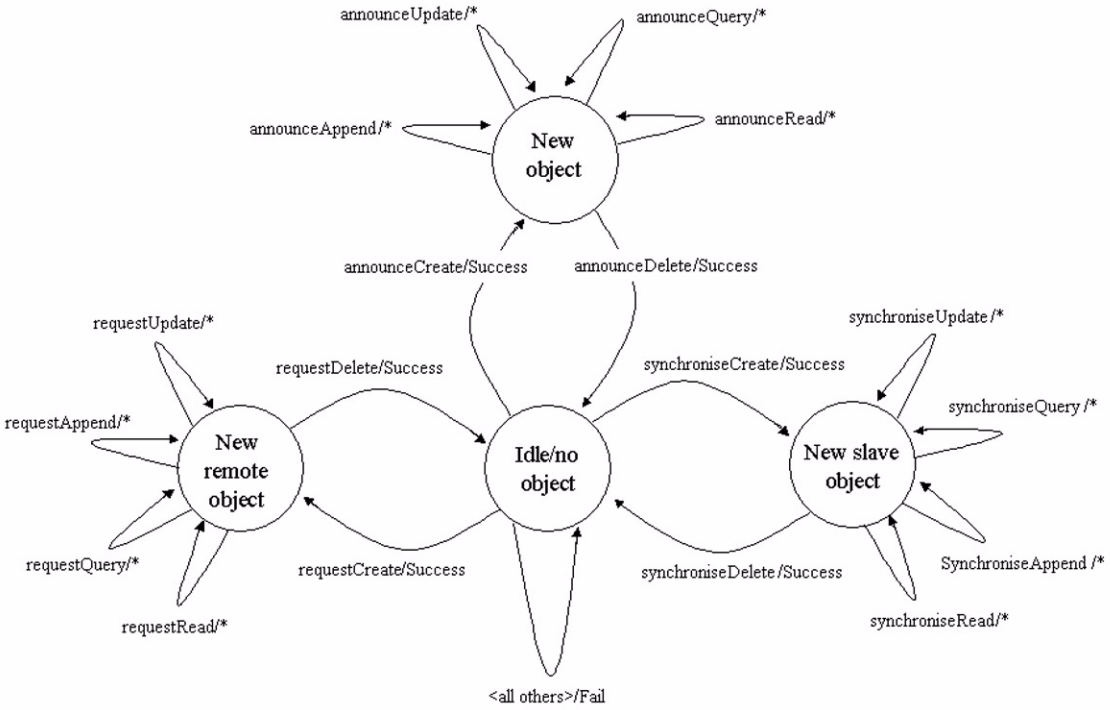


Figure 5.2 - The state diagram for the core operators.

The start state is when the system is idle. An instance of an object can then be created one of three ways:

- New remote object – the owner of the object requests that a copy is made in a remote system. The new object is owned by the remote system;
- New slave object – the owner of the object request that a copy is made in a remote system but this copy is not owned by the remote system;
- New object – the owner of the object announces the availability of the object to other systems.

Once the object has been created all operations are now supported. Once the delete operation has been invoked the object is deleted and the system returns to the 'Idle' state. In Figure 5.2, it is assumed that the state of the operation is reported to the system initiating the operation.

5.2.1 XML Context Messages

In the case of loosely coupled systems then the Beshears matrix is implemented by the exchange of messages. In most cases this is a two-phase message exchange with the initiator of the interaction issuing the 'request' message and the respondent replying with the 'response' message; this is normally termed the 'request-response' service model. It is good practice to ensure that the response message carries back the state of the initial request i.e., was the request successfully process or not and if not what was the cause for the failure. Table 5.2 shows the types of information that should be sent in the request/response messages. The various status codes include:

- Success – successful operation execution;
- NoIdFail – no identifiers available to be allocated;
- UnknownIdFail – the object identifier is unknown to the system;
- InvalidDataFail – the supplied data is invalid in some form or another;
- CommsFail – some form of communications system failure.

Table 5.2 - The logical information contained in the request/response messages[3].

Operation	Request Message Information	Response Message Information
Create	Initialization state Transaction Identifier	Status Code Transaction Identifier
Delete	Object identifier Transaction Identifier	Status Code Transaction Identifier
Read	Object identifier Transaction Identifier	Object data Status Code Transaction Identifier
Update	Object identifier Replacement data Transaction Identifier	Status Code Transaction Identifier
Append	Object identifier New data Transaction Identifier	Status Code Transaction Identifier
Query	Search criteria Transaction Identifier	Set of object data Status Code Transaction Identifier

WSDL is one representation method for the context sub-layer. This enables the operations and their associated logical messages to be expressed in XML but it also provides the binding of those messages to the equivalent transport mechanism (the envelope sub-layer). In the case of SOAP the XML message is carried in the body part of the SOAP message is defined using an XSD. A brief overview of WSDL is given in the IMS specification development methods and best practices guide [SpecDev, 03].

5.3 XML-Based Envelope Sub-Layer

The XML-based Envelope sub-layer is responsible for encapsulating the sequence of XML messages and associated resources produced by the XML-based Context sub-layer in a common XML messaging structure. The three available approaches are:

- Develop an IMS-specific XML-based envelope – this approach is discouraged in preference to adopting an established mechanism;
- Use an established eLearning specific XML-based envelope – the SIF has such a mechanism however this approach could also lead to an idiosyncratic solution;
- The preferred approach, which is to use an established generic XML-based envelope such as:-
  - SOAP
  - SOAP with Attachments
  - http-Get
  - http-Post
  - ebXML Transaction, Routing & Packaging.

3. ‘Transaction Identifiers’ are required to support asynchronous systems. Synchronous system are blocked until the ‘Response’ is received whereas in an asynchronous system other calls can be invoked before the initial ‘Response’ is received.

5.3.1 SOAP

The Simple Object Access Protocol (SOAP) is a messaging transport protocol that can be implemented using XML documents [SOAP, 03a], [SOAP, 03b], [SOAP, 03c]. A brief overview of SOAP is given in the IMS specification development methods and best practices guide [SpecDev, 03].

5.3.2 SOAP with Attachments

SOAP with Attachments is an extension of SOAP that is designed to enable non-XML information to be transported in the SOAP messages [SOAP, 01a]. A brief overview of SOAP is given in the IMS specification development methods and best practices guide [SpecDev, 03].

5.3.3 ebXML Transaction, Routing, and Packaging

The XML-based replacement of the Electronic Data Interchange (EDI) is termed ebXML. ebXML is based upon an extensive range of functionality ranging from the transaction exchange mechanisms to the routing table protocols [ebXML, 01a], [ebXML, 01b], [ebXML, 01c], [ebXML, 01d], [ebXML, 01e], [ebXML, 01f], [ebXML, 01g].

5.4 Generic Transport Sub-Layer

The Generic Transport sub-layer is responsible for transferring the XML messages across the appropriate data network architecture. The specification of these protocols is outside the scope of the abstract framework with the exception of ensuring that the XML-based Envelope sub-layer XML messages can be exchanged using the appropriate generic transport protocol. The detailed sub-structure of the generic transport sub-layer is:

- Generic transport protocol – the application transport protocol responsible for managing the delivery of the XML messages. A variety of protocols are available to provide this capability, including:-
  - File Transfer Protocol (FTP) – the standard Internet file transfer protocol
  - Simple Mail Transfer Protocol (SMTP) – normally used to send emails, with attachments, from the client to the mail server
  - Hypertext Transfer Protocol (HTTP) – the Internet web access protocol. The secure version, S-HTTP, is also available;
- End-to-end communications protocol – this is the protocol that turns the set of linked physical networks into a reliable end-to-end network. In general this protocol will be based upon Transmission Control Protocol/Internet protocol (TCP/IP) with the occasional usage of User Datagram protocol/Internet protocol (UDP/IP) in cases where high performance delivery is required. If UDP/IP is used then a higher level protocol will have to be responsible for ensuring reliable delivery as and when required;
- Physical data network – the actual data network, which may itself consist of several internetworked networks, which physically connects all of the system. The different parts of this network will have different characteristics e.g., terrestrial and mobile links, and so the end-to-end communications protocol is responsible for making this a reliable end-to-end network. The definition of the various networks, devices and networking techniques is outside the scope of the abstract learning framework.

5.5 Building the Infrastructure Stack

A working system requires the profiling of the entire infrastructure layer to support the appropriate application and common services. IMS will be producing a series of recommendations for a set of default infrastructure profiles as a part of its general web services binding project (to be completed by the end of 2003). One of the strengths of this layered approach is that the information model definition remains unchanged for different infrastructure profiles as it is the only the binding to that infrastructure that must be changed. Also, one infrastructure profile is capable of supporting many different information models provided the infrastructure sustains the minimum quality of service capabilities e.g., reliable end-to-end data delivery.

6. Service Bindings

6.1 Binding of the Information Model

The abstract framework is formally expressed as a UML representation. From an IMS perspective, this information model representation is traditionally bound to XML. In the new specifications XML will not be the only supported binding. This representation process is shown in Figures 6.1.

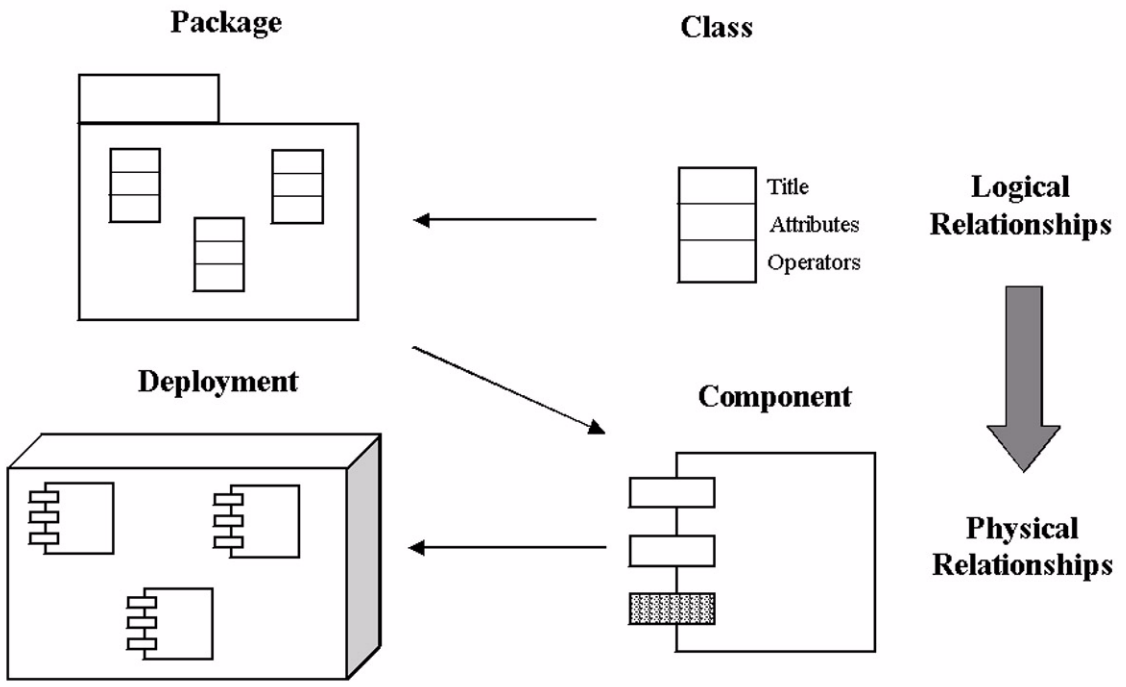


Figure 6.1 - The UML representation relationships.

In Figure 6.1 it is shown that the logical representation of a system is derived from one or more classes that are grouped using packages. The package shows the dependencies between the classes. Each class consists of attributes (the data structures) and operators (the operations permitted on the data structures). The system is then constructed from these components as shown in the deployment diagram. The logical structure is represented using packages and it is representation that must be mapped using an appropriate binding. The advantage of this approach is that both the UML specification (the Information Model) and the corresponding set of bindings (of which XML and WSDL are just two) can be used for implementation. It is then possible to map together different implementations in different ways without loss of capability. The usage of an information model defines the common service capability between the different bindings.

The binding must take into account the attributes and operators of the class. In the original IMS specifications the XML DTDs and XSDs were a representation of the attributes of the class. The behavioral representations are defined through the class operators. In this case the binding is achieved by representing these operators as a series of messages exchanged by the communicating eLearning systems. The behavior is represented by the sequence of messages and the content of the control field in the headers of the messages (a message is assumed to consist of a header, containing the control fields, and the body, containing the data to be transferred).

A simple system showing the exchange of an object between two systems (the initiator and the respondent) is shown in Figure 6.2. In this system the initiator is requesting that the respondent create an object. In the initiator the ‘createObject()’ operator is invoked. The implementation of the object handler in the initiator translates this action into a ‘createObject.request’ XML message. This message is passed to the communications handler object that is

responsible for transmitting this message to the respondent system via the communications network. The respondent communication handler may be required (this depends upon the protocol being implemented) to return an acknowledgment message that is used to inform the initiator that the request has been received. When the respondent actually completes the object creation request, notification of this may be returned to the initiator. This notification results in the transmission of the 'createObject.response' message. Its receipt at the initiator signifies that the request has been completed; this may or may not have resulted in the creation of the object and so some completion status codes should be returned via the 'createObject.response' message.

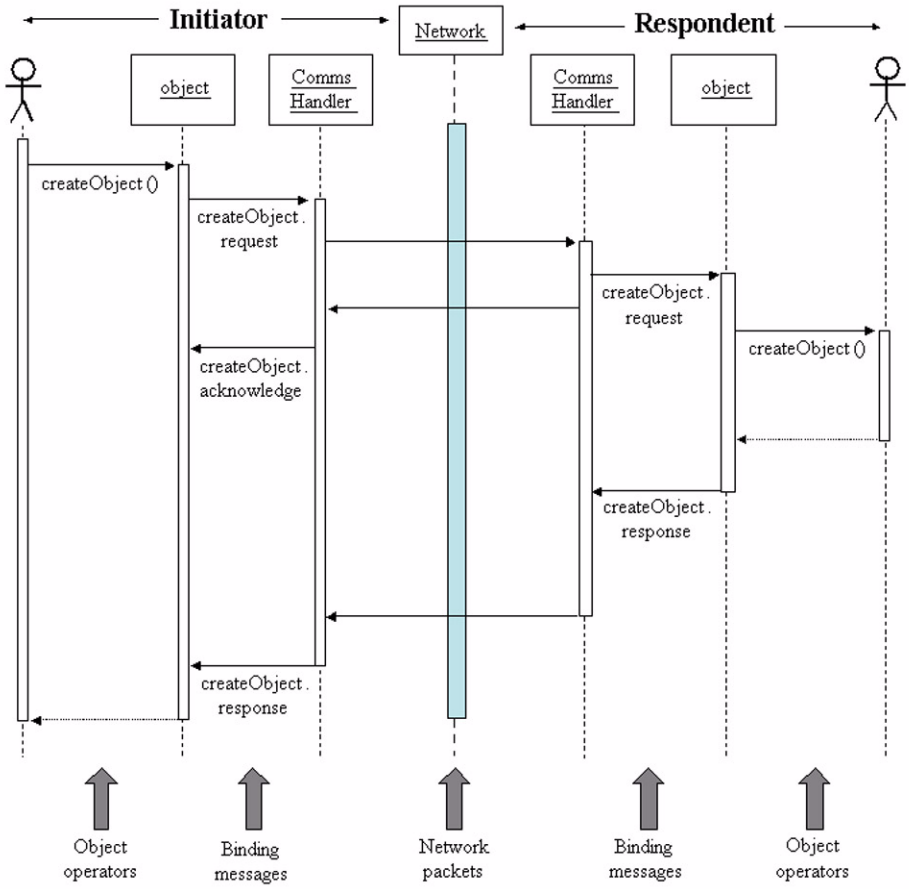


Figure 6.2 - A typical sequence diagram for object information exchange.

This sequence gives rise to three categories of information exchange, namely:

- Object operators – the operations defined in the equivalent class description;
- Binding messages – these are exchanged between the interoperating systems to implement the behaviors of the operator triggering the message exchange;
- Packets – the actual data packets that are sent across the data network (the binding messages are encapsulated in these messages).

From the perspective of the fully layered abstract framework one possible sequence of events is as shown in Figure 6.3. In this scenario two systems (perhaps Enterprise systems) are exchanging information e.g., the initiator wishes to inform the respondent of the creation of a group. The systems use a common application service and one common service e.g., authorization. The infrastructure service underpins all of the other services and provides a reliable end-to-end communication infrastructure. The initiator is using remote services and as such the initiating objects must contain the interface definitions for the service. The respondent objects must contain the functionality to actual handle the service requests.

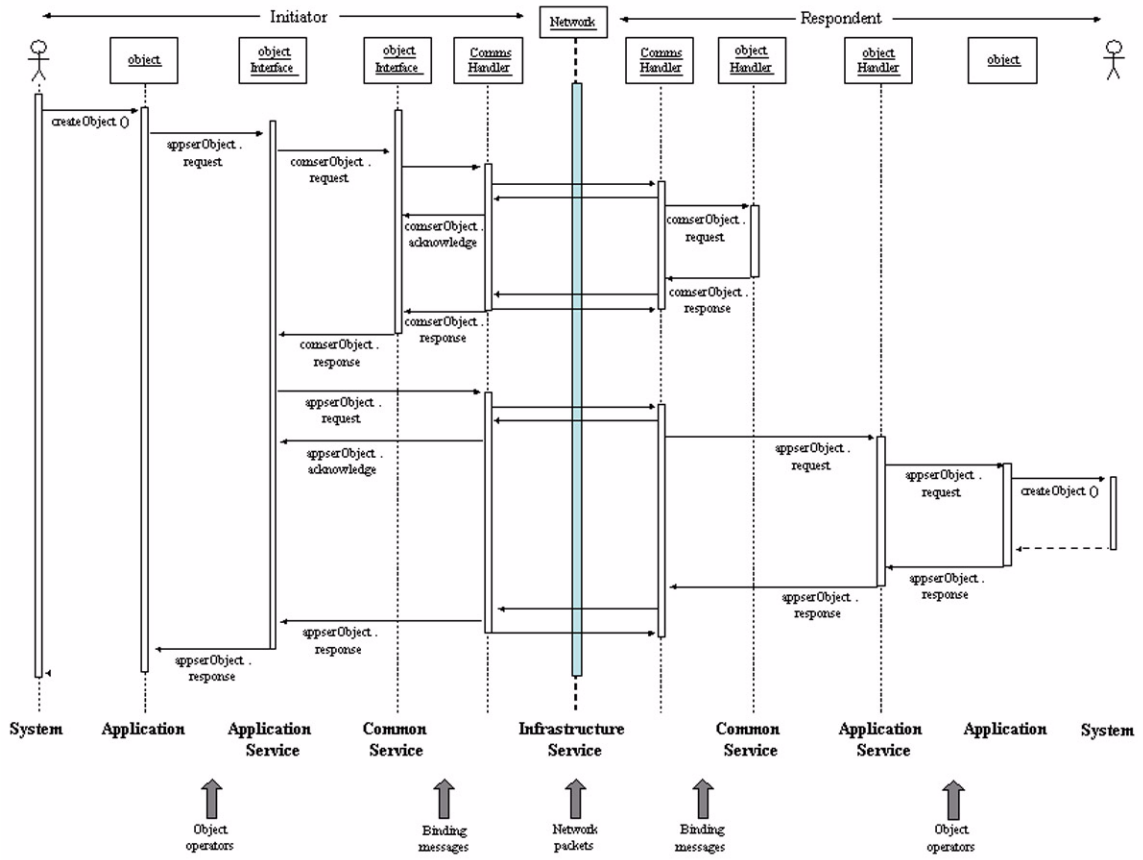


Figure 6.3 - A possible full sequence diagram for object information exchange.

The initiating system issues the 'createObject()' which results in the object invoking the relevant application service. The application service issues the remote object creation instruction. This request requires the application service to obtain authorization though the corresponding common service. Once the authorization has been completed the actual object creation process is started and this request is sent to the respondent system. Once the request has been processed the appropriate response is returned to the initiator.

There are several ways in which this scenario could be developed:

- More than one common service could be invoked by the application service;
- Any common service could itself invoke another common service;
- An application could invoke more than one application service;
- Each service (application, common, infrastructure) could be implemented using several objects. This depends on the functionality of each object and its relationship to the service definitions.

It is important to stress that this scenario has been considered from the perspective of a generic binding of the services. The actual mapping is dependent on the nature of the services and the underlying capabilities in the Infrastructure Layer.

6.2 Possible Bindings

The advantage of using a robust method for representing the behavioral aspects of the information model means that, in principle, different bindings can be automatically generated by the usage of the appropriate tools and the application of suitable transformation rules. Figure 6.4 shows the set of possible bindings that can be created.

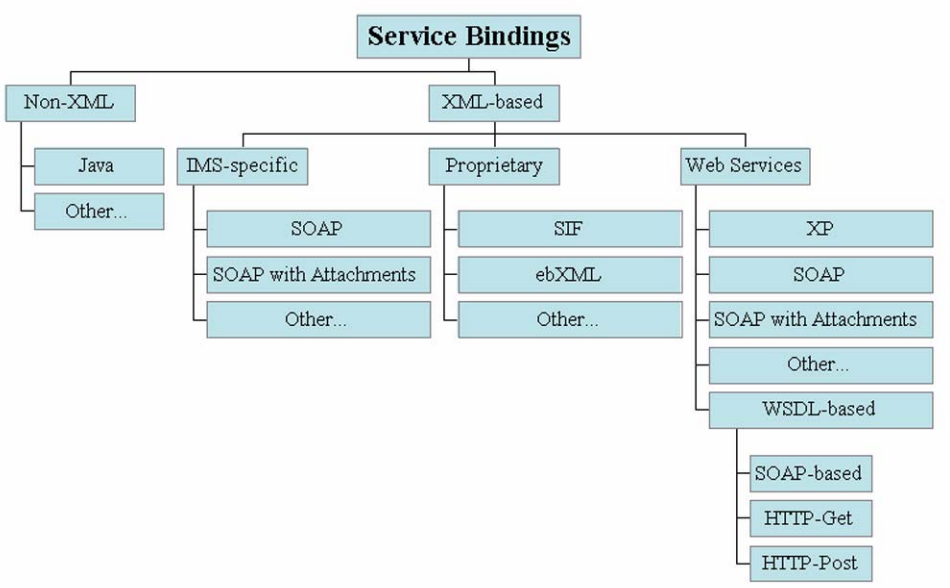


Figure 6.4 - Possible set of bindings of the information model.

The key points from Figure 6.4 are:

- Only Java is shown as an explicit non-XML based binding. Other language specific bindings are also possible. A Java implementation can be created by generating the corresponding Java methods and data structures from the class descriptions. Some UML tools will create a Java implementation as a 'Save As…' option;
- The IMS-specific XML-based binding requires that IMS create their own XML context and XML document representation. These XML documents can then be converted to XML messages using SOAP, SOAP with Attachments or some other technique (including an IMS-specific approach). One approach is that the permitted structure and contents of the XML messages are defined using an XSD. Each type of message is defined using its own XSD and each operator in a class is represented by one or more XSDs. The behavior is represented by the sequence of messages and the content of the control field in the headers of the XML messages (a message is assumed to consist of a header, containing the control fields, and the body, containing the data to be transferred). The disadvantage of this approach is that IMS would not be leveraging the work of organizations such as W3C;
- The proprietary XML-based binding approach makes use of work from other organizations that have developed their own messaging system (this may or may not use some standard underlying technologies). Systems of particular relevance are the SIF messaging mechanism and ebXML. The latter contains a sophisticated messaging system whereas the SIF approach requires a new message object to be created for every new object as opposed to using a single generic message object container;
- The web services binding covers all of the other XML-based techniques. In this category are the standard W3C techniques of XP, SOAP, SOAP with Attachments and WSDL. WSDL is itself a binding representation technique that must have an associated messaging mechanism i.e., SOAP, etc.;
- There will at some time in the future be other web service bindings.

The preferred IMS service binding approach is based upon:

- The usage of XML as the underlying data representation format. UML is the abstract representation of the information model but XML is the underlying binding encoding format;
- The usage of WSDL to act as the intermediate binding representation format. IMS will develop a series of recommendations on how to create the WSDL binding from a UML-based information model. These recommendations will also address the usage of different transport mechanism within the WSDL description;

- The usage of SOAP with Attachments as the underlying common messaging mechanism. The usage of HTTP/HTTPS (as opposed to FTP or SMTP) as the underlying transport mechanism.

It is possible to translate between different IMS bindings provided the binding of the information model has been reliably and consisted. This gives rise to the following forms of communications interoperability issues:

- End systems that use the same IMS binding mechanism can communicate using another binding (either IMS or non-IMS) provided the appropriate binding switch or router encapsulation is used;
- End systems that use the same IMS binding mechanism can communicate using another binding (either IMS or non-IMS) provided the appropriate binding switch or router transformation is used;
- End system that use different IMS binding mechanisms can communicate through a gateway that provides the appropriate binding translation;
- An IMS bindings switch/router fabric could be used to support the exchange of information between non-IMS systems.

7. Profiling and Conformance

7.1 Profiling

Domain profiling is the process that is undertaken to define which specifications and the detailed usage of the data objects within each specification are to be adopted to provide a particular solution. Therefore, the development of SCORM, using this terminology, was domain profiling undertaken on behalf of the US Department of Defense. In general, a resulting 'Domain Profile' or 'Reference Model' will be based upon a variety of specifications and standards i.e., it will not consist, solely, of IMS specifications.

The domain profiling process is based upon:

- Identification of the appropriate specifications by matching their functional capabilities to the requirements of the application;
- Refinement of each IMS specification made by:-
  - Where appropriate, the constraints within the information model and binding are made stronger e.g., optional elements can be made mandatory. The constraints must **not** be relaxed otherwise the profile will not be valid with respect to the IMS specification. All default attributes should be changed to be required or fixed
  - Profile-specific extensions must be defined. These will take the form of new elements and attributes that should, in general, be required. Whenever possible these extensions should only be applied where permitted within the IMS specification. All profile-specific extensions should have their own unique namespace
  - All proprietary extensions locations should now be defined. These must only be permitted wherever the IMS specification allows extensions i.e., the profile may wish to remove some of the IMS defined extension locations
  - All free format data content should be typed according to the required data types e.g., whenever possible a number should be defined as integer, etc. and the permitted range of values should also be defined
  - All of the vocabularies for the element and attribute data entries must be defined. The vocabularies will reflect the market and domain of the application. Whenever possible, internationally and nationally agreed vocabularies should be adopted;
  - The new binding for the domain profile should now be produced. In general, a instance document for a particular profile should validate against the new binding control document for the profile and the binding control document for the corresponding IMS specification. The profile-specific binding control document will be capable of identifying errors in the instances due to the increased constraints on the permitted data content;
- The corresponding strong conformance statement and certification test specification should now be produced for the domain profile.

The issue now becomes one of how to map between different domain profiles i.e., this is inter-domain interoperability (the domain profile supports intra-domain interoperability. In many cases different domain profiles will have a common core with only a small set of important differences. Mapping between domain profiles is simplified due to the usage of XML as the binding format. The mapping process is shown schematically in Figure 7.1 in which there are two, currently theoretical, domain profiles of QTI for SCORM and SIF. Each domain profile has its own XML schema that is the manifestation of the domain profile of the IMS QTI specification. These schemas are then used to validate the QTI-XML documents for the corresponding application profile. In each case the domain profile would be used by an appropriate system e.g., a SIF LMS (it is possible that a single LMS could in fact support both the SCORM and SIF domain profiles).

The advantage of the usage of XSDs is that an XML Stylesheet Transformation (XSLT) can be used to support the transformation between the XSDs; there would be one XSLT for each QTI transformation i.e., SIF-to-SCORM and SCORM-to-SIF.

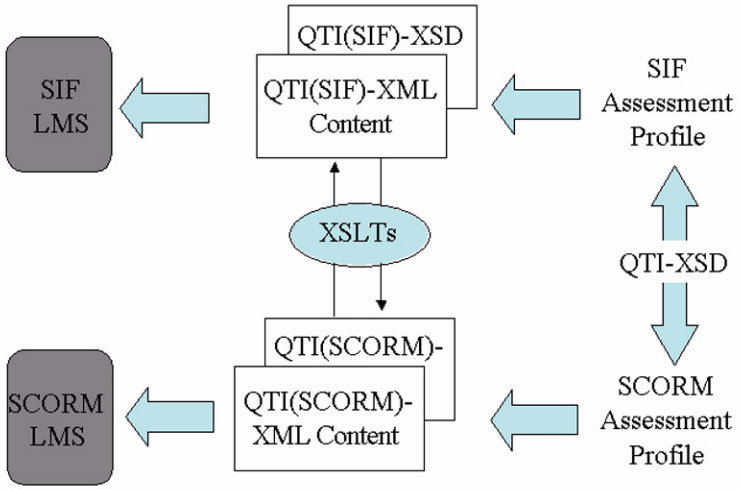


Figure 7.1 - Mapping between different application profiles.

The XSLT can be defined to handle the mapping of each element and attribute in the XSDs, including those defined in any extension – each domain profile must define the information model and XML binding for all extensions. In some cases it will not be possible to map some features from one domain profile to similar features in the other. In these cases there is a loss of functionality but this is a controlled situation in that the situations under which this loss occurs is well understood.

7.2 Conformance

'Domain Conformance' will not be defined with respect to an IMS specification. These specifications contain too many optional features and are subject to regional and sector specific amendments e.g., the inclusion of the appropriate vocabularies. Therefore, conformance will be against a particular Conformance Profile that has been derived from the corresponding Domain Profile. Conformance certification is considerably easier when all of the functionality is mandatory and so it is important for Conformance Profiles to remove as much optional functionality as possible.

7.2.1 First Generation Conformance

The key issue for conformance under the first-generation specifications is that they were not designed from the perspective of being testable i.e., exhaustive testing to show compliance is possible but not cost effective. The reason for this is that the specifications are designed to support practice and not to impose a best practice. This has produce specifications that a largely based upon optional features and so the derived conformance profiles can have many disjoint features. The current specifications are data models and so the associated behaviors are assumed in the information model and not defined in a behavioral model. Once again this leads to important discrepancies in the underlying assumptions used by different application profiles.

Each of the first-generation specifications contains a conformance statement. In most cases these statements are either a simple statement of the obvious i.e., the insistence that the information is followed or consists of a tick list of the features that are supposedly supported. In neither case is it possible to use the statements as a definitive guide to the compliance or otherwise of an implementation to the corresponding IMS specification.

7.2.2 Second Generation Conformance

An underlying objective in the creation of the second-generation specifications is the creation of specifications against which conformance can be clearly shown. This means that the second-generation specifications must:

- Minimize the number of optional elements and attributes. Whenever possible there should be no optional features – this implies that we are moving towards supporting a set of best practice recommendations;

- Whenever possible define the behaviors that are associated with the elements and attributes. These behavioral descriptions must reflect all the perspectives of the service being defined;
- When supporting an XML binding adopt as many as possible of the features of XSD to support extensive validation by the parsers e.g., the use of data typing. Many of the first-generation specification used DTDs and so only structural validation was possible;
- When defining the exchange messages ensure that different functionality is supported using different messages and that each message has no optional components, particularly in the body of the message;
- Be designed to ease the testing activities without compromising functional and performance effectiveness. This implies the usage of many XSDs as opposed to one single monolithic XSD.

Unlike the first-generation specification the second-generation ones will contain an extensive conformance specification that will form the basis for conformance against which the application profiles will build.

8. Bibliography

[ADL, 01a] *SCORM v1.2: The SCORM Overview*, ADL, Version 1.2, October 2001.

[ADL, 01b] *SCORM v1.2: The SCORM Content Aggregation Model*, ADL, Version 1.2, October 2001.

[ADL, 01c] *SCORM v1.2: The SCORM Run-time Environment*, ADL, Version 1.2, October 2001.

[ADL, 02a] *SCORM v1.2: The SCORM Addendums*, ADL, Version 2.0, January 2002.

[ADL, 02b] *SCORM v1.2: Conformance Requirements*, ADL, Version 1.2, February 2002.

[CP, 03a] *IMS Content Packaging Information Model*, T.Anderson, M.McKell, A.Cooper, W.Young, C.Moffatt and C.Smythe Version 1.1.3, IMS, May 2003.

[CP, 03b] *IMS Content Packaging XML Binding*, T.Anderson, M.McKell, A.Cooper, W.Young, C.Moffatt and C.Smythe Version 1.1.3, IMS, May 2003.

[CP, 03c] *IMS Content Packaging Best Practice and Implementation Guide*, T.Anderson, M.McKell, A.Cooper, W.Young, C.Moffatt and C.Smythe Version 1.1.3, IMS, May 2003.

[DRI, 03a] *IMS Digital Repositories Information Model*, T.Barefoot, J.Mason and K.Riley Version 1.0, IMS, January 2003.

[DRI, 03b] *IMS Digital Repositories XML Binding*, T.Barefoot, J.Mason and K.Riley Version 1.0, IMS, January 2003.

[DRI, 03c] *IMS Digital Repositories Best Practice and Implementation Guide*, T.Barefoot, J.Mason and K.Riley Version 1.0, IMS, January 2003.

[ebXML, 01a] ebXML Technical Architecture Specification, ebXML, Version 1.0.4, February 2001.

[ebXML, 01b] *ebXML Business Process Specification Schema*, ebXML, Version 1.01, May 2001.

[ebXML, 01c] *Collaboration-Protocol Profile and Agreement Specification*, ebXML, Version 1.0, May 2001.

[ebXML, 01d] *Message Service Specification ebXML Transport, Routing & Packaging*, ebXML, Version 1.0, May 2001.

[ebXML, 01e] *ebXML Requirements Specification*, ebXML, Version 1.06, May 2001.

[ebXML, 01f] *ebXML Registry Information Model*, ebXML, Version 1.0, May 2001.

[ebXML, 01g] *ebXML Registry Services Specification*, ebXML, Version 1.0, May 2001.

[eGIF, 02] e-Government Schema Guidelines for XML, Version 3.0, Office of the e-Envoy, UK, December 2002.

[eGIF, 03a] *e-Government Interoperability Framework Part 1: Framework*, Version 5.0, Office of the e-Envoy, UK, April 2003.

[eGIF, 03b] *e-Government Interoperability Framework Part 2: Technical Policies and Specifications*, Version 5.0, Office of the e-Envoy, UK, April 2003.

[Ent, 02a] *IMS Enterprise Information Model V1.1 Final Specification*, C.Smythe, G.Collier, C.Etesse and W.Veres, IMS Specification, July 2002.

[Ent, 02b] *IMS Enterprise XML Binding V1.1 Final Specification*, C.Smythe, G.Collier, C.Etesse and W.Veres, IMS Specification, July 2002.

[Ent, 02c] *IMS Enterprise Best Practice & Implementation Guide V1.1 Final Specification*, C.Smythe, G.Collier, C.Etesse and W.Veres, IMS Specification, July 2002.

[IEEE, 02] *IEEE P1484.1/D9 Draft Standard for Learning Technology: Learning Technology Systems Architecture (LTSA)*, IEEE LTSC Publication, Draft 9, 30th November 2002.

[IMS, 01a] *IMS Persistent Location-independent Resource Identifier Implementation Handbook*, V1.0, M.McKell, IMS Specification, May 2001.

[IMS, 01b] *Using the IMS Content Packaging to Package Instances of LIP and Other IMS Specifications Implementation Handbook*, V1.0, B.Olivier and M.McKell, IMS Specification, August 2001.

[IMS, 02a] *Minutes of the IMS System Components & Infrastructure Workshop: Cambridge USA 29th April – 1st May 2002*, Colin Smythe, IMS, May 2002.

[IMS, 02b] *IMS Guidelines for Developing Accessible Learning Applications*, Cathleen Barstow, Mark McKell, Madeleine Rothberg and Chris Schmidt, V1.0, IMS White Paper, June 2002.

[IMS, 03a] *IMS Abstract Framework: Glossary*, Editor C.Smythe, IMS Publication, V1.0, July 2003.

[IMS, 03b] *IMS Abstract Framework: Applications, Services & Components*, Editor C.Smythe, IMS Publication, V1.0, July 2003.

[LD, 03a] *IMS Learning Design Information & Behavioral Model*, R.Koper, B.Olivier and T.Anderson Version 1.0, IMS, February 2003.

[LD, 03b] *IMS Learning Design XML Binding*, R.Koper, B.Olivier and T.Anderson Version 1.0, IMS, February 2003

[LD, 03c] *IMS Learning Design Best Practice and Implementation Guide*, R.Koper, B.Olivier and T.Anderson Version 1.0, IMS, February 2003

[LIP, 01a] *IMS Learner Information Package Information Model*, R.Robson, C.Smythe and F.Tansey, Version 1.0, IMS, March 2001.

[LIP, 01b] *IMS Learner Information Package XML Binding Final Specification*, R.Robson, C.Smythe and F.Tansey, Version 1.0, IMS, March 2001.

[LIP, 01c] *IMS Learner Information Packaging Best Practices & Implementation Guide Final Specification*, R.Robson, C.Smythe and F.Tansey, Version 1.0, IMS, March 2001.

[MD, 01a] *IMS Learning Resource Meta-data Information Model*, T.Anderson and M.McKell, Version 1.2, IMS, May 2001.

[MD, 01b] *IMS Learning Resource Meta-data XML Binding*, T.Anderson and M.McKell, Version 1.2, IMS, May 2001.

[MD, 01c] *IMS Learning Meta-data Best Practice and Implementation Guide*, T.Anderson and M.McKell, Version 1.2, IMS, May 2001.

[QTI, 02a] *IMS Question & Test Interoperability: ASI Information Model*, C.Smythe, E.Shepherd, L.Brewer and S.Lay, Final Specification, Version 1.2, IMS, April 2002.

[QTI, 02b] *IMS Question & Test Interoperability: ASI XML Binding Specification*, C.Smythe, E.Shepherd, L.Brewer and S.Lay, Final Specification, Version 1.2, IMS, April 2002.

[QTI, 02c] *IMS Question & Test Interoperability: ASI Best Practice & Implementation Guide*, C.Smythe, E.Shepherd, L.Brewer and S.Lay, Final Specification, Version 1.2, IMS, April 2002.

[QTI, 02d] *IMS Question & Test Interoperability: ASI Selection & Ordering Specification*, C.Smythe, L.Brewer and S.Lay, Final Specification, Version 1.2, IMS, April 2002.

[QTI, 02e] *IMS Question & Test Interoperability: ASI Outcomes Processing Specification*, C.Smythe, L.Brewer and S.Lay, Final Specification, Version 1.2, IMS, April 2002.

[QTI, 02f] *IMS Question & Test Interoperability: Results Reporting Information Model*, C.Smythe, L.Brewer and S.Lay, Final Specification, Version 1.2, IMS, April 2002.

[QTI, 02g] *IMS Question & Test Interoperability: Results Reporting XML Binding*, C.Smythe, L.Brewer and S.Lay, Final Specification, Version 1.2, IMS, April 2002.

[QTI, 02h] *IMS Question & Test Interoperability: Results Reporting Best Practice & Implementation Guide*, C.Smythe, L.Brewer and S.Lay, Final Specification, Version 1.2, IMS, April 2002.

[QTI, 02i] *IMS Question & Test Interoperability: Overview*, C.Smythe, E.Shepherd, L.Brewer and S.Lay, Final Specification, Version 1.2, IMS, April 2002.

[QTI, 02i] *IMS Question & Test Interoperability: QTILite Specification*, C.Smythe, E.Shepherd, L.Brewer and S.Lay, Final Specification, Version 1.2, IMS, April 2002.

[QTI, 03] *IMS Question & Test Interoperability: Addendum*, C.Smythe, L.Brewer and S.Lay, Final Specification, Version 1.2.1, IMS, April 2003.

[RDCEO, 02a] *IMS Reusable Definition of Competency or Educational Objective – Information Model*, A.Cooper and C.Ostyn, Final Specification, Version 1.0, IMS, September 2002.

[RDCEO, 02b] *IMS Reusable Definition of Competency or Educational Objective – Best Practices and Implementation Guide*, A.Cooper and C.Ostyn, Final Specification, Version 1.0, IMS, September 2002.

[SIF, 00] *Schools Interoperability Framework Implementation Specification*, V1.0, Software & Information Industry Association, June 2000.

[SIF, 01] *Schools Interoperability Framework Draft Data Objects Specification*, Draft, Software & Information Industry Association, December 2001.

[SOAP, 01a] *SOAP Messages with Attachments*, W3C, W3C Note 11, December 2000.

[SOAP, 03a] *SOAP Version 1.2 Part 1: Messaging Framework*, W3C, W3C Final Specification, June 2003.

[SOAP, 03b] *SOAP Version 1.2 Part 2: Adjuncts*, W3C, W3C Final Specification, June 2003.

[SpecDev, 03] *IMS Specification Development Methods & Best Practices*, C.Smythe, Version 1.0, IMS, July 2003.

[SS, 03a] *IMS Simple Sequencing Information & Behavioral Model*, A.Panar & M.Norton Version 1.0, IMS, March 2003.

[SS, 03b] *IMS Simple Sequencing XML Binding*, A.Panar & M.Norton Version 1.0, IMS, March 2003.

[SS, 03c] *IMS Simple Sequencing Best Practice and Implementation Guide*, A.Panar & M.Norton Version 1.0, IMS, March 2003.

[SUN, 02] *E-Learning Content Delivery Functional Model*, G.Collier, SUN Microsystems Technical White Paper, SUN Microsystems, September 2002.

[SUN, 03] *E-Learning Framework*, SUN Microsystems Technical White Paper, SUN Microsystems, February 2003.

[Universal, 02] *System Interface Framework for Building an Educational Brokerage Network*, EU Framework V UNIVERSAL Project Deliverable, UNIVERSAL Project Consortium, Version1.0, September 2002.

[WSDL, 02] *Web Services Description Language*, Version 1.2, W3C, W3C Working Draft 9, July 2002.

[XP, 01a] *XML Protocol (XMLP) Requirements*, W3C, W3C Working Draft 19, March 2001.

[XP, 01b] *XML Protocol Abstract Model*, W3C, W3C Working Draft 9, July 2001.

[XP, 01c] *XML Protocol Usage Scenarios*, W3C, W3C Working Draft 17, December 2001.

Appendix A – List of Acronyms

Acc	IMS Accessibility Guidelines
ADL	Advanced Distributed Learning
AICC	Aviation Industry CBT Committee
ANGEL	Authenticated Network Guided Environment
ANSI	American National Standards Institute
API	Application Programming Interface
ASL	American Sign Language
ASP	Application Service Provider
as-SAP	Application Service Service Access Point
AT	Assistive Technology
CEN/ISSS	Centre for European Normalisation/Information Society Standardisation Service
CBT	Computer Based Training
CMI	Computer Managed Instruction
CMU	Carnegie Mellon University
CP	IMS Content Package Specification
CSS	Cascading Style Sheet
cs-SAP	Common Service Service Access Point
DC	Dublin Core
DCMI	Dublin Core Meta-data Initiative
DR	Digital Repository
DRI	IMS Digital Repository Specification
DTD	Document Type Definition
ebXML	e-Business XML
EDI	Electronic Data Interchange
EJB	Enterprise JavaBeans
Ent	IMS Enterprise Specification
FTP	File Transfer Protocol
HEKATE	Higher Education Knowledge and Technology Exchange
HRMS	Human Resources Management System
HR-XML	Human Resources XML
HTML	Hypertext Mark-up Language
HTTP	Hypertext Transfer Protocol
HTTPS	Secure-Hypertext Transfer Protocol
IAF	IMS Abstract Framework
ICSE	IMS Common Service Infrastructure
IEEE	Institute of Electronic & Electrical Engineers
IETF	Internet Engineering Task Force
in-SAP	Infrastructure Service Access Point
IP	Internet Protocol

J2EE	Java 2 Enterprise Edition
ISO/IEC	International Standardisation Organization/International Electrotechnical Commission
JDBC	Java Database Connection
JNDI	Java Naming and Directory Interface
KM	Knowledge Management
LAM	Learning Activity Model
LAN	Local Area Network
LCMS	Learning Content Management System
LD	IMS Learning Design Specification
LDAP	Lightweight Directory Access Protocol
LIP	Learner Information Package
LMS	Learning Management System
LOM	Learning Object Meta-data
LTS	Learning Technology System
LTSA	Learning Technology Systems Architecture
LTSC	Learning Technology Standardisation Committee
MD	IMS Meta-data Specification
MLE	Managed Learning Environment
MIT	Massachusetts Institute of Technology
OASIS	Organization for the Advancement of Structured Information Standards
OKI	Open Knowledge Initiative
OMG	Object Management Group
OpenUSS	Open University Support System
PAPI	Personal and Privacy Information
PDF	Portable Document Format
PLIRI	Position Location-independent Resource Identifier
PSTN	Public Switched Telephone Network
QCL	Qualifications, Certifications & Licenses
QTI	Question & Test Interoperability
RDCEO	Reusable Definition of Competency or Educational Objective
RFC	Request For Comment
RLO	Reusable Learning Object
RMI	Remote Method Invocation
SAP	Service Access Point
SAS	Student Administration System
SCA	Sharable Content Asset
SCO	Sharable Content Object
SCORM	Sharable Content Object Reference Model
S-HTTP	Secure Hypertext Transfer Protocol
SIF	Schools Interoperability Framework
SIIA	Software & Information Industry Association

SIS	Student Information System
SMIL	Synchronized Multimedia Integration Language
SMTP	Simple Mail Transfer Protocol
SOAP	Simple Object Access Protocol
SOAPwA	SOAP with Attachments
SRE	SCORM Runtime Environment
SS	IMS Simple Sequencing Specification
SVG	Scalable Vector Graphics
TCP	Transmission Control Protocol
TR&P	Transaction, Routing & Packaging
TTY	Text Telephone
UDDI	Universal Data Discovery Interface
UDP	User Datagram Protocol
UML	Unified Modelling Language
UN/CEFACT	United Nations Centre for Trade Facilitation and Electronic Business
USOeC	United States Open e-Learning Consortium
VLE	Virtual Learning Environment
W3C	World Wide Web Consortium
WAN	Wide Area Network
WAI	Web Accessibility Initiative
WSDL	Web Services Description Language
WSFL	Web Services Flow Language
XML	Extensible Mark-up Language
XP	XML Protocol
XSD	XML Schema Definition
XSLT	Extensible Style-sheet Language Transformation
ZIS	Zone Integration Server

Appendix B – IMS Specification Roadmap

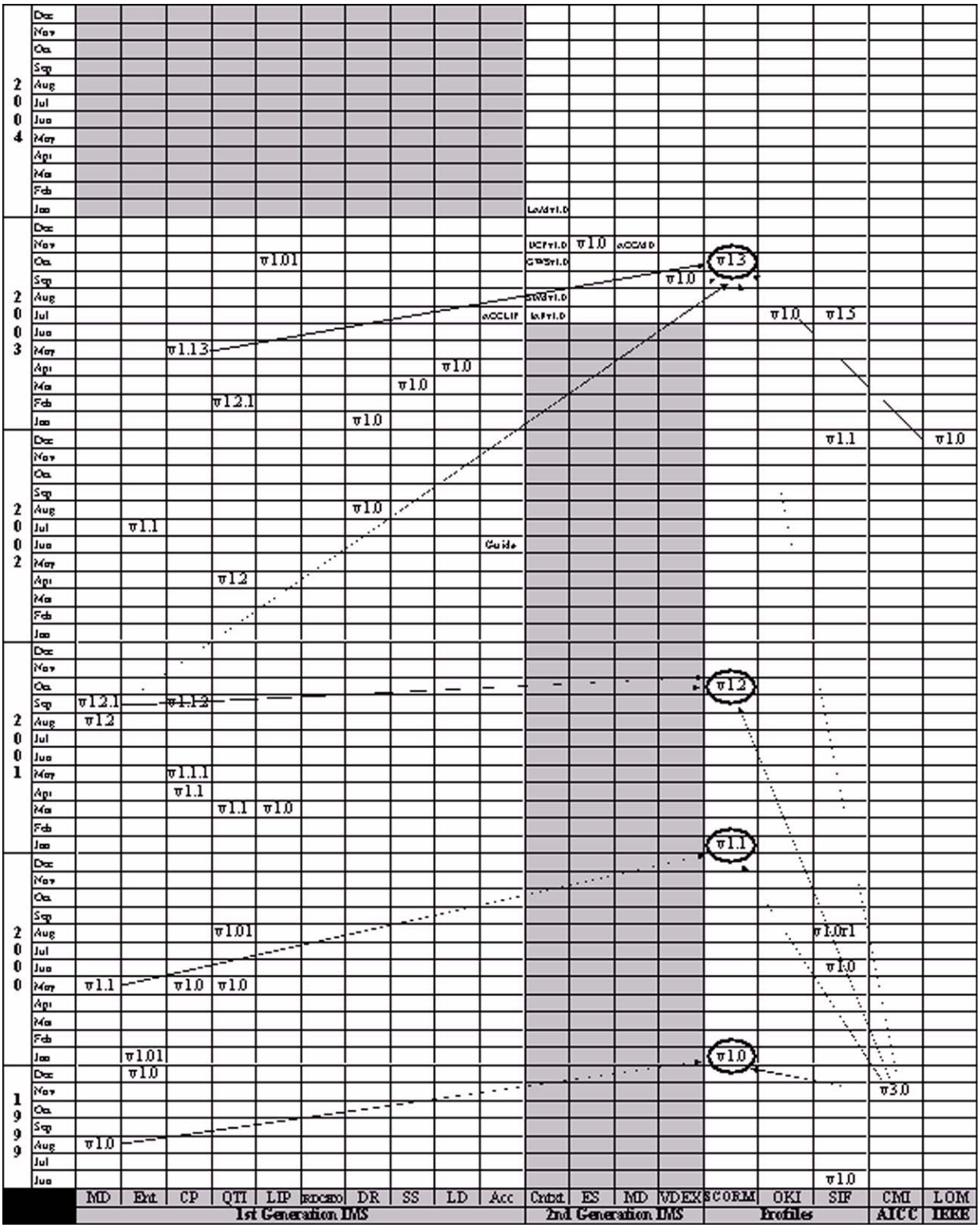


Figure B1 - The eLearning specification release roadmap.

The IMS specification roadmap given in Figure B1 shows the release sequence of the IMS specifications, AICC specifications, the IEEE standards and the key profiles. The arrows are used to denote instances where the IMS specifications have been adopted by a profile. The key for the IMS specifications is:

- MD = Meta-data
- Ent = Enterprise
- CP = Content Packaging
- QTI = Question & Test Interoperability
- LIP = Learner Information Package
- DR = Digital Repositories
- SS = Simple Sequencing
- LD = Learning Design
- RDCEO = Reusable Definition of Competencies and Educational Objectives
- Acc = Accessibility
- Cntxt = Context documents
- ES = Enterprise Services
- VDEX = Vocabularies

Appendix C – Related eLearning Architectures and Models

C1 - Open Knowledge Initiative (OKI)[4]

The core deliverable of the Open Knowledge Initiative (OKI) is an architectural specification in support of learning management and educational application development, the primary feature of which is a set of application programming interface (API) definitions. OKI are specifying the architecture by identifying a set of services, a framework, upon which learning tool developers can base their work. The goal here is twofold:

- To identify and provide these basic services so that developers can concentrate on the pedagogical issues at hand without having to re-invent basic functionality like authentication/authorization, user profile management, content storage and retrieval, etc., or concern themselves with the details of particular underlying implementations;
- To allow easier intra-institution work sharing. Too often, applications built in one place for one purpose are impossible or at the very least very difficult, to port to another location or adapt to a new purpose.

OKI is taking a layered approach to this architectural specification, defining a clean set of boundaries between the various required components that traditionally support learning applications. At the very lowest level of functionality OKI is building on existing work, within its partner institutions and elsewhere, to define an API layer of common services that will help institutions to integrate applications with existing institutional infrastructure. The OKI project is relatively unique in its vision that the issues of LMS architectural design should be approached as issues of institutional infrastructure.

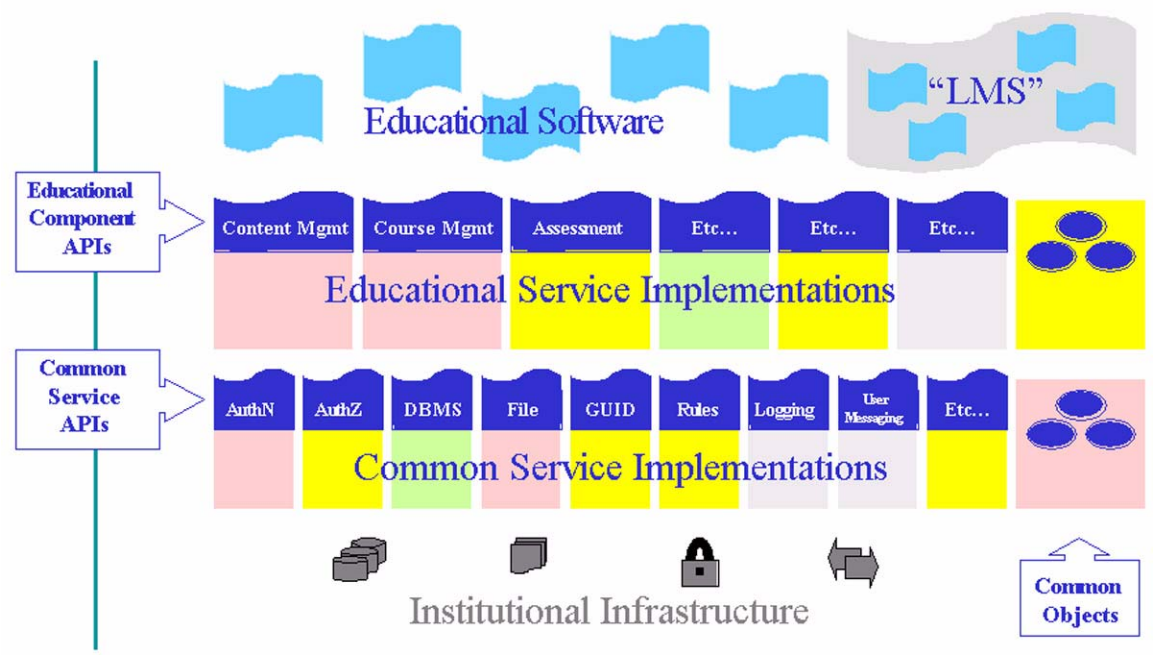


Figure C1 - The OKI layering.

As shown in Figure C1, the four layers from the bottom upwards are:

- Infrastructure – the underlying communications and implementation structure as supported by the local institution;
- Common services – the set of services that convert the institution specific infrastructure into a common services platform that can be used by the educational services;

4. Thanks to Jeff Merriman and Scott Thorne. The contact URL is: http://web.mit.edu/oki.

- Educational Services – the set of services that are available to the institution's applications in the provision of the e-learning framework;
- Applications – the manifestation of the educational services as supplied to the user.

In Figure C1 the actual APIs are denoted as the interfaces to the implementations. The form of the implementations is beyond the scope of the OKI.

The deliverables from the OKI are the two sets of Java APIs for the Common Services and the Educational Services, namely:

- Common services
  - Authentication - the Authentication OSID gathers required credentials from an agent, vouches for their authenticity and introduces the agent to the system
  - Authorization - the Authorization OSID allows an application to establish and query a user's privileges to view, create, or modify application data, or use application functionality
  - Database Control - the DBC OSID allows an application to access and modify the contents of a database by using the java.sql and javax.sql packages
  - Logging - the Logging OSID records and retrieves a variety of application activity history
  - Dictionary - the Dictionary OSID provides a means to support multiple languages, domain-specific nomenclature and culture-specific conventions through interchangeable property files
  - Shared - the Shared OSID contains fundamental objects used in the other OSIDs to provide their functionality
  - Filing - the Filing OSID provides platform-independent means to handle files arranged in simple hierarchical containers
  - Hierarchy - the Hierarchy OSID manages parent-child relationships among elements. In addition to simple tree structures, the OSID supports hierarchies that are recursive and have nodes with multiple parents
  - User Messaging - the User messaging OSID supports communication and notification among users
  - Scheduling - the Scheduling OSID manages events in shared calendars
  - SQL - the SQL OSID provides relational database access functionality at a higher level of abstraction than the DBC OSID. Unlike DBC, it is not dependent on JDBC.
  - Workflow - the Workflow OSID provides a way to manage an interdependent succession of activities each of which has completion constraints;
- Educational Services
  - Class Admin
  - Digital Repository
  - Assessment.

While the OKI Architectural Specification is basically a document, it is the intention of MIT and some of its partners to actually develop implementations of these specifications in order to support OKI applications on their campuses.

MIT will make its implementations available as open source code via the IMS web-site, and other institutions may do the same as they see fit. Modularity at this level should allow sites installing OKI services to begin to pick and choose from among existing implementations where it makes sense. Therefore, it is expected that the OKI partner institutions initially, and eventually other institutions and vendors, will begin to share service level implementations. This will especially make sense for OKI services that currently have no counterpart in a typical institutional infrastructure. A good example of this might be the Rules service. Most sites do not currently have a Rules infrastructure and, therefore, may be more than happy to use someone else's.

However, it is expected that a number of services will require variations in local implementations to hook into existing or planned local infrastructure many different implementations of the Authentication (AuthN) and Authorization (AuthZ) services will be developed to interoperate with the various security infrastructures at our institutions.

The end goal of all of this is to encourage and support a wide range of pedagogical tool and educational application development activity and to allow these tools to be shared more easily among campuses. These pedagogical tools and applications are what the end users of OKI based systems will see and with which they will interact. In other words, users will never need to know about the framework underneath, they will simply reap the benefits of this integration through interoperable applications.

C1.1 - Relationship to IMS Activities

IMS and OKI are working closely together and several OKI members (MIT, University of Wisconsin, University of Michigan and Cambridge University) are IMS Contributing Members. During the development of the new IMS specifications the usage of the OKI Common Services will be described within the corresponding best practices and implementation guides. The IMS specifications of direct relevance to the OKI APIs are Enterprise Services, Digital Repositories Interoperability, Content Packaging, and Question & Test Interoperability.

C2 - IEEE Learning Technology Systems Architecture (LTSA)[5]

The IEEE LTSA is an architecture based on abstract components – as shown in Figure C2 [IEEE, 02]. Higher-level abstractions can be "implemented" in lower levels: either lower level abstractions, or actual implementations. Learning technology systems ("implementations") can be mapped to/from LTSA.

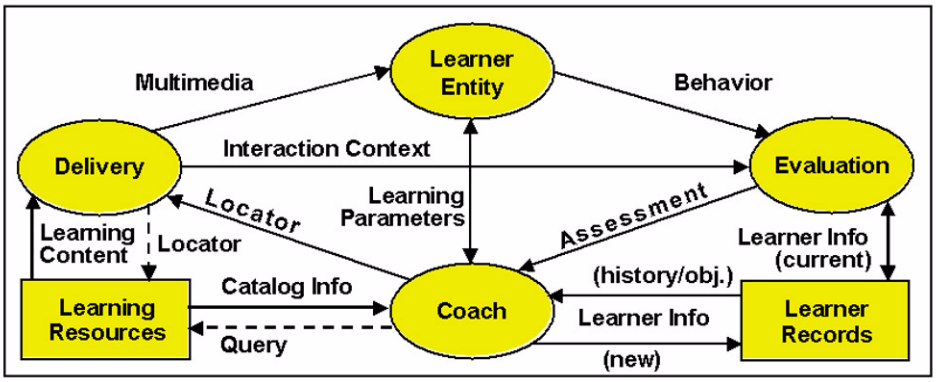


Figure C2 - The IEEE Learning Technology Systems Architecture.

The IEEE LTSA is pedagogically neutral, content-neutral, culturally neutral, platform/technology-neutral. It uses a simple "systems notation" i.e., processes ("bubbles" that transform information), stores (e.g., repositories and databases), flows (information between processes and stores). The abstract components of the IEEE LTSA are:

- **Learner Entity:** represents the "learner" or a group of "learners", e.g., team/collaborative learning;
- **Learning Parameters:** the "negotiable" items of the learning experience(s), e.g., the learner's native language preferences are honoured, except when the learner is learning a foreign language;
- **Behavior:** the learner's observable behavior with respect to the learning experience(s);
- **Evaluation:** a process of combining information from Interaction Context, Behavior, and Learner Records to produce new Learner Information and Assessment;
- **Learner Info:** information about the learner (e.g., performance/preference/relationship information) that may be processed/stored/retrieved by Evaluation, Coach, and Learner Records components;

5. Thanks to Frank Farance and the IEEE LTSC who supplied most of this text on the IEEE LTSA.

- **Learner Records:** stores Learner Information, etc. about the learner's past (e.g., historical data/ records), present (e.g., “bookmarks”, current work), and future (e.g., learning objectives);
- **Assessment Info:** Describes learner's current state vis-à-vis ongoing evaluation (e.g., “where the learner is at”);
- **Coach:** a process that combines information (in an unspecified order) from Learner Entity, Evaluation, Learner Records, and Learning Resources to produce Locators that describe Delivery instructions to support learning experiences. Note: An individual human may play the roles of more than one LTSA component, e.g., playing the roles of both Learner Entity and Coach, such as in self-directed learning;
- **Query:** an inquiry about the availability of Learning Resources. Note: The Query may reflect the learner's capabilities, history, objectives, preferences, and other information;
- **Learning Resources:** a “store” of learning content to support learning experiences. Note: The “store” may be distributed (e.g., not same geographic area), disconnected (e.g., nomadic), or distance (e.g., subject to communication delays);
- **Catalogue Info:** a response to the Query from the Learning Resources. Note: Learning Object Metadata is one kind of Catalogue Info;
- **Locator (Coach to Delivery):** an (abstract) “identifier” that describes the Delivery instructions;
- **Locator (Delivery to Learning Resources):** an “identifier” for retrieving learning content;
- **Learning Content:** information that may be used in the context of learning experiences;
- **Delivery:** a process that receives delivery instructions (Locator from Coach), retrieves Learning Content, and transforms them into (1) Multimedia to support a learning environment, and (2) an Interaction Context to support interactivity (to the level desired);
- **Interaction Context:** the contextual information associated with the Delivery and the Multimedia of the learning experience(s) that may be used within the Evaluation process;
- **Multimedia:** One (mono-media) or more types of electronic or non-electronic media with coordinated delivery.

  Note 1: The boundaries, functions, and decomposition of actual or other abstract learning technology systems might not have the same structure as LTSA, i.e., the mapping *to* LTSA might not be “one-to-one” (mathematical sense).

  Note 2: Not all learning technology systems will have all components of LTSA, i.e., the mapping *to* LTSA might not be “onto” (mathematical sense).

  Note 3: All inputs and outputs are optional to each process and store, e.g., the Evaluation process might or might not use Interaction Context as part of its processing; the Learning Resources might or might not have high quality Catalogue Information.

C2.1 - Relationship to IMS Activities

The IMS is one input channel to activities within the IEEE LTSC. In the future, the IEEE LTSC may or may not decide to adopt all or part of the abstract framework. The IEEE LTSC adopted IMS Meta-data and turned it into a standard.

C3 - ADL Sharable Content Object Reference Model (SCORM)

The Sharable Content Object Reference Model (SCORM) defines web-based learning ‘Content Aggregation Model’ and ‘Run-time Environment’ for learning objects [ADL, 01a], [ADL, 01b], [ADL, 01c], [ADL, 02a], [ADL, 02b]. At its simplest, it is a model that references a set of interrelated technical specifications and guidelines designed to meet DoD’s high-level requirements for Web-based learning content. The work of the ADL initiative to develop the SCORM is also a process to knit together disparate groups and interests. This reference model aims to co-ordinate emerging technologies and commercial/public implementations.

The SCORM applies current technology developments – from groups such as the IMS, the AICC, ARIADNE and the IEEEE LTSC – to a specific content model to produce recommendations for consistent implementations by the vendor community. At the current time SCORM V1.2 is the latest release but V1.3 is slated for release before the end of 2003. SCORM V1.3 will make use of the following specifications:

- IEEE LOM V1.0 / IMS Meta-data V1.2.1;

- IMS Content Packaging V1.1.3;
- IMS Simple Sequencing V1.0;
- AICC CMI V3.1.

Within the SCORM, the generalized model of an LMS is shown in Figure C3. This model shows the potential components or services that are required of an LMS.

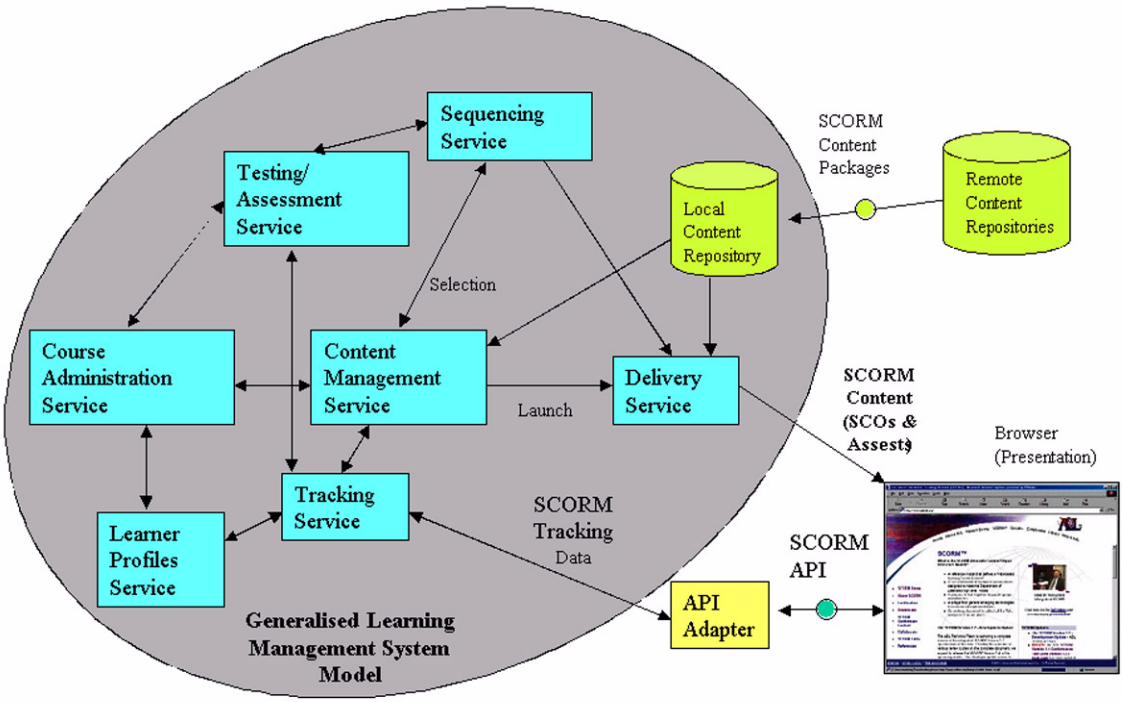


Figure C3 - The SCORM generalized model for an LMS.

Within SCORM, the term LMS implies a server-based environment in which the intelligence resides for controlling the delivery of learning content to students. In other words, in the SCORM, the LMS has the ability to determine what to deliver and when, and track student progress through the learning content.

Even SCORM V1.3 will be missing some key functional eLearning capabilities and so the areas in which other IMS specifications could be used are:

- Assessment model – usage of the IMS QTI specifications;
- Course administration – usage of the IMS Enterprise specification;
- Learner progression – usage of the IMS LIP specification.

The two core parts of SCORM are:

- SCORM Content Aggregation Model (SCAM) – this provides a common means for composing learning content from discoverable, reusable, sharable and interoperable sources. The SCAM further defines how learning content can be identified and described, aggregated into a course or portion of a course and moved between systems that may include LMSs and repositories. The SCAM defines the technical methods for accomplishing these processes. The model includes specifications for aggregating content

and defining meta-data using the IMS Content Packaging and IMS Meta-data specifications respectively. The content is defined as a set of Assets which become Sharable Content Objects (SCOs) when the CMI launch and termination instructions are added. Sharable Content Assets (SCAs) are added as a part of SCORMv1.3;

- SCORM Run-time Environment (SRE) – the purpose of the SRE is to provide a means for interoperability between SCO based learning content and LMSs. A requirement of SCORM is that learning content be interoperable across multiple LMSs regardless of the tools used to create the content. For this to be possible, there must be a common way to start content, a common way for content to communicate with an LMS and predefined data elements that are exchanged between an LMS and content during its execution. The three components of the SRE are defined as 'Launch', API and Data Model.

C3.1 - Relationship to IMS Activities

SCORM is based upon several IMS specifications (Meta-data, Content Packaging, Simple Sequencing). ADL is an IMS Contributing Member and the SCORM project team make regular contributions to the development of the relevant IMS specifications.

C4 - Schools Interoperability Framework (SIF)[6]

One of the most pressing challenges for the K-12 education industry today is software interoperability: how to enable different software applications to share information in order to improve efficiency and reduce cost. To meet this challenge, many software companies in the education industry and educational institutions have spearheaded the Schools Interoperability Framework (SIF) initiative. SIF is an effort to promote interoperability between software applications from different vendors without requiring each vendor to learn and support the intricacies of other vendor's applications.

SIF has two deliverables: the *SIF Implementation Specification v.1.0* and the notice of availability of one or more SIF compliant Zone Integration Servers [SIF, 00], [SIF, 01]. The Implementation Specification defines the software implementation guidelines for SIF. The Implementation Specification does not make any assumption of what hardware and software products need to be used to develop SIF compliant applications. Instead, it only defines the requirements of architecture, communication, software components, and interfaces between them. The goal of SIF is to ensure that all the SIF compliant applications can achieve interoperability, regardless of how they are implemented. SIF is truly an open industrial initiative. SIF Zone Integration Servers will be created by organizations for use on one or more hardware and operating system platforms. They are required to meet the requirements of the Implementation Specification and will be tested as being compliant with that specification but the specifics of the implementation and any additional features will be left to each developing organization to determine.

The target users of this document include software developers who develop SIF compliant products, technology specialists who deploy SIF solutions for K-12 institutions, and executive managers who need to evaluate the feasibility of SIF.

A SIF implementation must enable different applications to exchange data efficiently, reliably, and securely regardless of what platforms are hosting the applications. Nothing should be designed to preclude the specification implementation with any architecture. To this end HTTPS has been chosen as the default transport protocol for sending SIF's XML messages. HTTPS must always be supported in all SIF implementations. When HTTP and XML are used together in this way, a truly platform-neutral wire is created, over which any two SIF compliant applications can intercommunicate, independently of the language they are written in, and the operating system and hardware they are deployed on. The requirement of efficiency implies that the SIF implementation must support both real-time and batch data exchange between different applications. To achieve this goal, the communication mechanism for SIF must be efficient and lightweight. A SIF implementation should also scale to support large numbers of applications.

Reliability implies that when an application sends out a message destined for a receiver, the delivery of the message is guaranteed. This requirement also implies that messages sent by an application will arrive at the receiver in the same order as they were sent and that each message is delivered once and only once.

6. Thanks to Tim Magner who supplied most of this text on SIF.

The SIF security requirement has three aspects, all of which may be optionally invoked by applications, and none of which may preclude compliant application interoperability:

- SIF must enable each message that is sent to SIF to be encrypted in a defined manner so that only the authorized receiver can decrypt and process the message;
- SIF must enable an application to optionally authenticate its partner before messages are exchanged and the data is processed. This is necessary to prevent unauthorized programs from accessing other applications to perform potentially damaging operations;
- SIF must enable access control that can be configured to restrict access to information to designated applications. For example, an application designed for students may not be allowed to request teachers' personal information, nor should it permit students to falsify messages to other students or teachers.

SIF implementation is a distributed networking system that consists of a Zone Integration Server (ZIS) and one or more integration agents organized into a zone. The size of the zone is flexible and could consist of a single building, school, a small group of schools, a district, etc. SIF is a scalable solution to data exchange and supports multiple SIF implementations or zones. A ZIS is a program that provides integration services to all the agents registered with it so that they can provide data, subscribe to events, publish events, request for data, and respond to requests. It is responsible for all access control and routing within the system.

Each application server requires an agent, which typically is provided by the application vendor, to communicate with other application servers via their respective agents. For example, a school may use a student information application, a food service application, and a library automation application. Each of these applications must have an agent for their application that will act as a go-between between the application and the Zone Integration Server.

In SIF, an agent never talks to another agent directly. Instead, an agent communicates with its ZIS that will manage the connection to the other agent. By having the ZIS manage the routing responsibilities, it allows complex, multi-zone communications to occur between agents that have no direct information about the other. The ZIS acts as the trusted intermediary that brokers the data exchange. Figure C4 illustrates a typical single zone SIF implementation that reflects a single school zone.

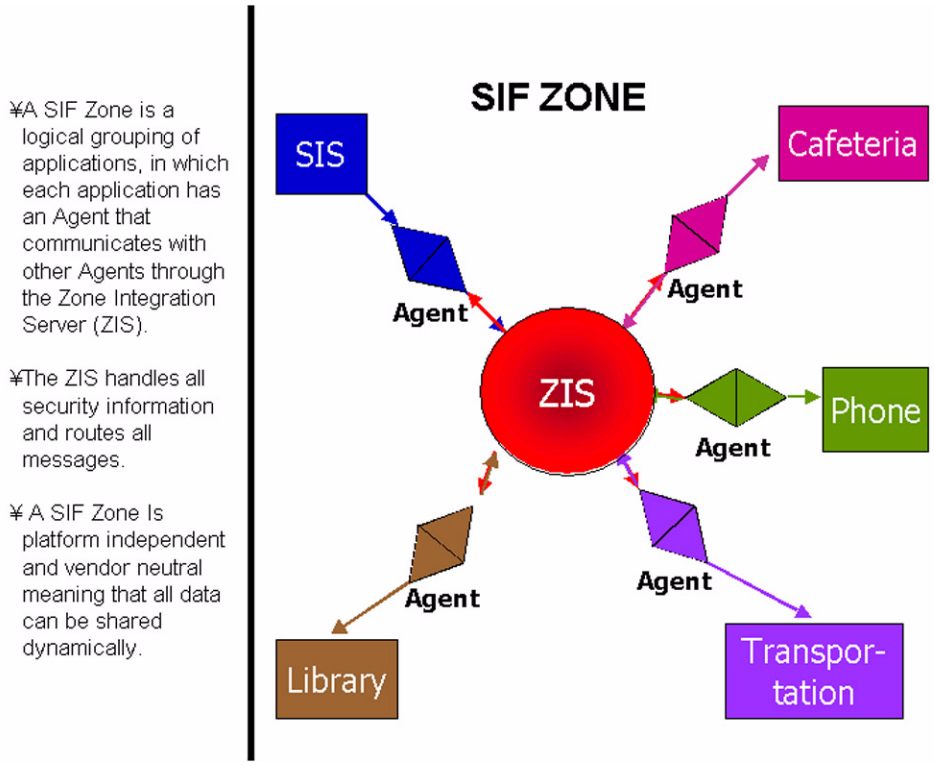


Figure C4 - The Schools Interoperability Framework (SIF) architecture.

In some SIF implementations, it may be necessary for SIF compliant applications to be organized into different zones based on the requirements of ownership, organizational structure, geographical proximity, security, or management. A zone is typically defined according to physical boundaries; for example, a zone can consist of all the applications that are connected over a private network and managed by one organization, such as a school. A ZIS is required in each zone and each application server requires an agent to connect to the ZIS. The ZIS cooperates with the agents to support data exchange between the application servers within the zone. Zone Integration Servers also cooperate with each other to let applications in different zones exchange messages either over a private network or a public network such as the Internet.

C4.1- Relationship to IMS Activities

At the current time IMS and SIF are working together to investigate the ways in which each organization can utilize the others specifications. SIF is looking at the IMS Question & Test Interoperability and Content Packaging to support K12 assessment/quizzes. IMS is looking at the SIF grade-book. Both organizations are looking on the issues of messaging using SOAP technology.

C5 - CMU Learning Technology System Architecture[7]

C5.1 - The CMU eLearning Architecture

The Learning Systems Architecture Lab (LSAL) is a research and development center at Carnegie Mellon University. The LSAL research program focuses on the design and creation of Internet based technologies for education and training. The Learning Systems Architecture Lab has developed the concept of a ***Learning Services Architecture*** and the ***Learning Services Stack,*** as shown in Figure C5, as a framework for developing the next generation of learning technology systems. The Learning Services Stack is based on current approaches used to develop Web Services and service-based systems. It elaborates on the conventional network communications stack. The learning services architecture divides functionality into three major groups or service layers: ***User Agents***, ***Learning Services*** and ***Infrastructure***. Each layer presents a well-defined standard communications interface, and functionality is isolated into the distinct layers or service collections. Services within each layer communicate only with services in the same layer and utilize the services provided by the layer directly below.

User Agents Layer

User Agents are the learning technology systems that provide the human computer interface to access and interact with all learning services and functions; access is only via User Agents. User Agents include systems for content and learning delivery, content and learning management, and authoring and content creation. User Agents are built using the collection of services provided by the Learning Services Layer.

Learning Services Layer

The Learning Services layer includes all services and components that have learning-specific functionality. It is further subdivided into three service sub-layers: ***Tools***, ***Common Applications*** and ***Basic Services***. The ***Tool Layer*** provides the high-level integrated services used to create a public interface to services for use by the user agents. Tools include those for content delivery, authoring, and learning management, e.g., course delivery, tutors, simulators, quiz and assessment, presentation applications, collaboration, grade and record book, registration, course administration and course management. The ***Common Applications Layer*** provides the collection of standard, commonly used services needed by tools and user agents. Common services include: content selection and sequencing, learner profiling, user tracking, learning object management, content management, report generation, and knowledge management. The ***Basic Services Layer*** includes core learning services and learning technology-specific versions of commonly used system services such as storage management, workflow, rights management, authentication, validation, logging, etc.

7. Thanks to Dan Rehak who is Technical Director for the Learning Systems Architecture Lab at Carnegie Mellon University. The contact URL is: http://www.lsal.cmu.edu/.

Infrastructure Layer

The Infrastructure Layer is the base of the Learning Services Architecture and the service stack. It consists of the services that are independent of the learning domain. It contains the core enabling services such as Transport (SOAP), Discovery (UDDI), Description (WSDL), and Workflow (WSFL). It utilizes core networking services such as HTTP, SMTP, FTP, TCP/IP to create Internet and Web-based learning technology systems.

C5.2 - Relationship to IMS Activities

The Learning Systems Architecture Lab at Carnegie Mellon University is a Contributing member of IMS. The CMU Learning Technology System Architecture has been one of the cornerstone pieces of work that have informed the development of the IAF. The CMU Learning Technology System Architecture is considerable more detail within its layered structure and much of this will be used to inform the detailed development of the IMS specifications within the context of the abstract framework.

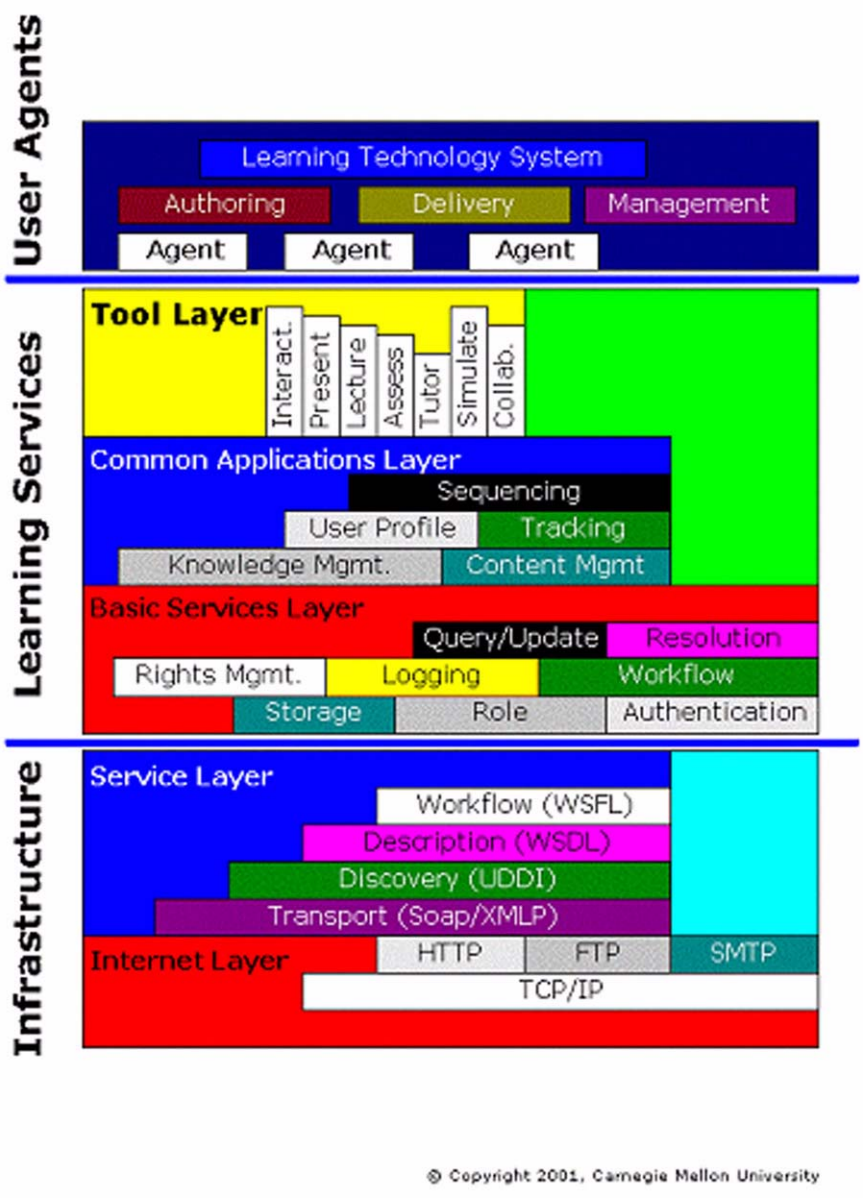


Figure C5 - The CMU Learning Technology System Architecture.

C6 - Open University Support System (OpenUSS)[8]

The OpenUSS project is designed to support all universities, institutes, faculties and students. The project has two kinds of activity:

- Developers – programmers who wish to contribute building architecture, APIs and reference implementation for the OpenUSS. OpenUSS is component oriented using the newest software technologies that are available. It is also important to use available Open Source Software in all parts of this system;
- Users – Users are universities, faculties, institutes and students, which use OpenUSS as their publishing and information centre system (portal). They should have an easy system to support lectures, documents and exercises. The portal should be the communication centre for all of the organization's members. Universities, faculties and institutes that don't have the capability to run the system can use the reference implementation and outsource it to OpenUSS. This is important because many faculties don't have the know-how or opportunity to set up such a publishing system.

The OpenUSS concept is based on the Application Service Provider (ASP) model, which means that one or more organizations (Universities, Schools, Communities, etc.) can be handled within one instance of OpenUSS. OpenUSS gives users the flexibility to use their chosen appliances – the so called multi-channel information delivery – to access the OpenUSS instance e.g., InternetPC, PDAs, and mobile phones.

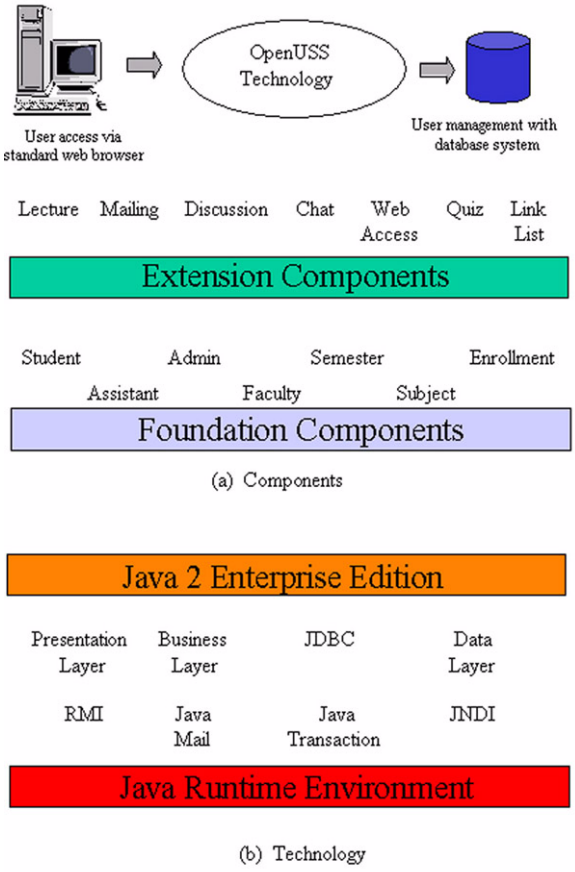


Figure C6 - The OpenUSS architecture.

OpenUSS is based on a component-oriented architecture, which divides the whole components into two parts: Foundation Components and Extension Components. As shown in Figure C6(a), the Foundation Components are the main components within OpenUSS. They represent the eLearning domain-oriented components like Assistant,

8. The contact URL is: http://www.openuss.org.

Student, Enrollment, etc. All the domain neutral functionalities of OpenUSS are implemented as Extension Components. Discussion, Chat, Lecture, etc. represent some of the Extension Components available. The separation of both component types makes it easier to develop more functions separately from the whole system.

From the technology perspective the OpenUSS is built upon a multi-tier architecture and Java2 Enterprise Edition (J2EE), as shown in Figure C6(b). The separation of the Presentation layer, Business Process Layer and Data Layer is the most important part in the OpenUSS technical design. The Presentation layer is implemented with Servlet APIs. The Business Process Layer is designed and implemented with Enterprise JavaBeans (EJB). Any database systems that support Java Database Connection (JDBC) can be used to cover the Data Layer of OpenUSS.

C7 - SUN Microsystems E-Learning Framework

The SUN e-learning framework [SUN, 03] can be broken down into four layers, each with a series of modular components. Beginning with the user, these four tiers are (from top to bottom):

- Presentation Tier;
- Common Service Tier;
- E-Learning Service Tier;
- Resource Tier.

C7.1 - The Architecture

Presentation Tier

The presentation tier allows end-users to interact with the service or application, and comprises both navigation and presentation logic.

- Portal – the portal serves two purposes: it provides a single point of entry for all users to the platform and it aggregates various services through channels and provides both a customized and a personalized view. The portal is customizable and can be tailored to any institution's guidelines and can easily be integrated with an existing portal solution;
- Entitlement – the entitlement service authenticates and authorizes users to enter into the platform, while single sign-on ensures an uniform experience by having the users enter their credentials only once, no matter how many services or systems they interact with while using the system;
- Profiles – this module is where users provide details about themselves and their preferences. These profiles will then dictate the services available to them;
- User Interface – services may have a presentation tier which interfaces with the users. The user interface service provides a uniform approach to integrate the user interface components with their business tier counterparts.

Common Service Tier

This tier represents the services that everyone needs to do eLearning, or e-commerce. Since they are not role specific, and are not dependent on any particular pedagogic function, they should not be part of the E-Learning Service Tier.

- User Management – all services in the framework delegate the task of managing users, groups and roles to a central service. This user management service provides the backbone for the user authentication, authorization and entitlement for all services in the framework;
- User Administration – user administration is a central service that keeps track of all user registration and account information. It manages the lifetime user identifier of every user in the eLearning system;
- Collaboration Services – an advantage of the eLearning framework is the abstraction of common facilities from the typical 'LMS only' model. This is demonstrated in the use of a central collaboration environment being made available to all users of the platform. This is the backbone of an all-pervasive feedback mechanism allowing users

to collaborate on any of the services available in the framework and have their comments directed to the appropriate resource owner. The collaboration service supports multiple forms of interaction and is able to adopt new mechanisms with advances in technology. The main forms of interaction supported include:

- E-mail
- Threaded Discussion
- Web Cast
- Desktop Sharing and Simulations
- White-boards & Chat;

- Schedule Management/Calendar – all services have access to a single schedule management system controlling both synchronous and asynchronous messages between the services and the users. Examples of this functionality might include reminder messages on the progress of a student application for enrollment. User calendar functionality is also supplied by the schedule management service;
- Event Management – all interaction between users and services is captured by the event management service. This data provides invaluable information for both program management and research into the pedagogy of eLearning. This warehouse of events then allows for both canned and dynamic reporting. An example of dynamic reporting might include the ability to ask "How long does it take for a tutor to respond to a student query?".

E-Learning Service Tier

- Learning Content Management Services – the learning content management service (LCMS) provides authoring, sequencing, and aggregation tools that structure content to facilitate the learning process. It uses a workflow driven approach to the production of both online and traditional instructional material to support blended learning. Learning objects are discovered and assembled using a meta-data language allowing flexible course construction. The workflow model supports course specification and author peer review when constructing the learning program. The learning material can be imported from other content systems using IMS interoperability standards and similarly exported to the learning management service for learning delivery;
- Learning Management Services – the learning management service is responsible for the delivery and administration of the course instance or offering, and the management of legacy applications with which it may be associated. It manages all interactions between the learning material and its participants including tracking progress and monitoring the usage levels to detect how the environment is being used. This information is invaluable in assessing the quality of the learning experience and detecting problems early enough to address them. The learning management service is distinguished from a typical 'LMS' in the flexible support for multiple pedagogical models through static and adaptive Reusable Learning Object (RLO) sequences. It also manages a catalog of static sequence learning programs or learning profiles for adaptive sequencing;
- Learning Administration Services – the learning administration system (LAS) manages all reference data in support of the learning services. The collection of reference data is specific to the individual educational institution. LAS consists of components that support the non-learning, delivery-related administration functions. It provides an abstraction between the e-learning platform and the hosted institution's existing billing or back office information system. It supports the institution's admission operations such as application and enrollment, fee payment and student transcript management, tutor record, and tutorial provider management. The assessment system measures student performance against specific learning goals. Both formative and summary assessment is provided through a collection of tools available to the tutor to assess students' progress against learning objectives. The tools support the following assessment types:
  - Automated Assessment—Includes multiple choice questions, multiple right answer, short answer, true/false
  - Complex Assessment—Includes essay assignment or structured document
  - Collaboration Assessment—Includes simulations and group work
  - Using these tools the assessment process become a more interactive model instead of the typical automated solution;

- Digital Resource Services – the repositories contain libraries of digital resources for constructing learning materials. Resources are discovered through an open search interface and made available to both the learning content management service for sequence definition of courses and the learning management service for delivery. RLOs are represented using a meta-data language for markup of digital media including descriptions, specifications and usage guidelines.

Resource Tier

- Learning Content Repository – the learning content repository uses meta-data to store and manage individual learning objects. The data repositories allow multiple developers and subject matter experts to share content and its components over the network;
- Learning Meta Data – a meta-data specification makes the process of finding and using a resource more efficient by providing a structure of defined elements that describe, or catalog, the learning resource, along with requirements about how the elements are to be used and represented. IMS Meta-data Specifications and those from the Dublin Core Meta-data Initiative are the most common sources;
- Learning Assessment Repository – on a platform level, learning assessment repositories contain quality-assured assessment questions as RLOs to ascertain learning gaps or test learning acquisition. However, on an Internet level, they provide access to valid, reliable, and customizable survey instruments that collect program assessment data in a secure, web-based environment. These repositories provide a source of data for researchers interested in comparing programs at different institutions and looking at longitudinal data on program effectiveness;
- Learning Administration Repository – this repository may, in fact, be a series of existing administrative databases, containing non-learning, delivery-related administration data and functions;
- User Repository – all user data, including the user's profile, assessment and transcript information, is contained here.

C7.2 - UK eUniversity Architecture

The UK eUniversity's platform is based upon the Sun Microsystems eLearning Framework (as described above). The framework consists of a collection of service definitions separated from implementation. This allows for vendor implementation to be specialized at the service level for 'best of breed' selection and ensuring the architectural integrity is retained while slotting in additional services.

The model also protects investment by wrapping existing implementations into a service. The service wrappers also allow for future proofing by replacing a service implementation with another while retaining the architectural integrity and without disrupting the surrounding services.

The services are completely interoperable and can be implemented as a complete collection or standalone. In applying the framework to an existing environment, services can be implemented optionally to complement existing implementations.

C7.3 - Relationship to IMS Activities

SUN and UKeUniversity are both IMS Contributing Members. IMS and OKI are working with SUN to evaluate the ways in which the SUN eLearning architecture can make use our respective specifications.

C8 - EU UNIVERSAL Project[9]

UNIVERSAL was a European Union Framework V project that was investigating educational brokerage networks for higher education [Universal, 02]. The idea of building educational brokerage networks is about making learning resources delivered through dispersed delivery systems available via an educational broker. Within the network the broker acts as a central system, providing facilities for managing exchange relationships between learning resources providers and consumers. As an educational mediator the broker combines, both, a collaboration facility for distributed educational activities and a market place for educational materials, in one system.

9. The contact URL is: http://www.ist-universal.org.

Users are able to provide learning resources to the broker and specify offers (i.e., usage terms), which consumers of those learning resources are asked to agree on before accessing the resource. Based on learning resource descriptions and usage conditions, learning resources are advertised through a catalogue and/or mailing lists. Users can choose and access learning resources from various kinds of delivery systems (e.g., video conferencing applications, learning management systems, streaming media servers, etc.), but have to agree on the offer terms before getting access to a resource.

Although the broker provides its service via a central web site, the system architecture is based on a network of specialized educational systems. The broker is a super system interacting with various decentralized delivery systems, which hold educational material or provide educational activities. Within the educational brokerage network the brokerage service is centralized, where as the content provision service is de-centralized. A delivery system can be a web server-based learning resource repository, a streaming media server, a video conferencing system or learning management system. The network can provide, both, static learning content (i.e., educational material in broker terms) and educational events with a specific time-table and a virtual meeting place (i.e., educational activities in broker terms).

The architectural framework consists of three layers as shown in Figure C7. The two top layers (*Application Layer* and *Administration Layer*) contain a number of services that facilitate the broker to support the exchange process and to administer user and system registration. The *Communication Layer* defines the middleware that can be used to remotely invoke services.

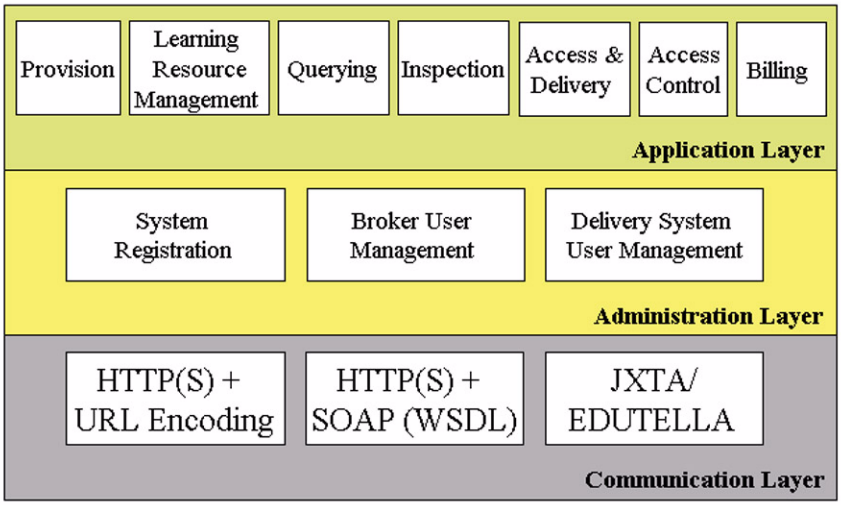


Figure C7 - The UNIVERSAL interface framework.

The application layer provides several application services along the exchange process. It relies on a trusted business environment, where users and systems are authenticated by services provided by the administration layer. There is a *Provision* service, which replicates learning resource descriptions and offer information between the delivery systems and the broker. *Learning Resource Management* provides means for uploading an educational material to a delivery system. The *Inspection* service can be used to check the status (e.g., availability) of learning resources and systems. The *Querying* service provides an interface for realizing multiple, federated brokers. The *Access Control* service grants users access control and the *Access and Delivery* interface supports learning resource access via the broker. The *Billing* service provides an interface to payment systems.

C8.1 - Relationship to IMS Activities

This work is from a completed EU Framework V project. At the current time there is no IMS interaction with this work.

C9 - EU MOBIlearn Project[10]

MOBIlearn is a worldwide European-led research and development project exploring context-sensitive approaches to informal, problem-based and workplace learning by using key advances in mobile technologies. The MOBIlearn project consortium involves 24 partners from Europe, Israel, Switzerland, USA and Australia. Their competencies are integrated and extended by a Special Interest Group that includes 250 of the world's leading IT organizations. The project addresses most of the key objectives of the Multimedia content and tools area of FP5 IST programme and it is strategically positioned to provide relevant research outcomes for the FP6.

For a mobile learning experience, the interoperability of a number of services, even new ones, has to be guaranteed, in the framework of properly defined pedagogical models. One of the major initiatives in this sector is the MOBIlearn project, whose objective is the definition of an abstract model, the development of an application profile and the realization of a Mobile Learning Management prototype. The Mobile Learning Management is a service accessible by eLearning applications (LCMS, LMS) to extend their functionalities to the ubiquitous learning, using:

- *Context Awareness Service.* To understand and support learning across contexts (for example, as a person moves from one exhibit to another in a gallery). This involves technical issues such as tracking the learner's movement, and also pedagogic and human issues (such as determining the learner's needs from the path taken through the gallery and previous requests for assistance);
- *Collaboration*. Service to be adapted for collaborative learning specific for mobile environments (This will allow group members to build the underlying pattern of collaborative learning on a consensus-oriented process through interaction and cooperation).

The main interoperability needs of the Mobile Learning Management service are:

- Interoperability with existing infrastructure including learner profile and institutional security (Authentication/Authorization);
- Identification and delivery of a learning object structured for a specific device (same content/semantic, different syntax/resolution) using a properly defined meta-data or “real-time” rendering process;
- Use of "location based" learning objects (i.e., with coordinates to make it geo-referenced, using extended LOM). In this context some work is being initiated in Dublin Core group;
- Synchronization of tracking;
- Extension of calendarization for mobile learning paradigms;
- Interoperability with QTI specification for location-based answers to special test (the reply is the position/movement of the learner);
- Interoperability with Simple Sequencing for location-adapted learning paths.

C9.1 - Relationship to IMS Activities

This work is from a EU Framework V project. Giunti Labs are the project coordinators and they are an IMS Contributing Member. At the current time the IMS has a special interest group that is looking into eLearning issues with respect to mobility.

C10 - UK Electronic Government Interoperability Framework (eGIF)[11]

Better public services tailored to the needs of the citizen and business, as envisaged in the UK online strategy, require the seamless flow of information across government [eGIF, 03a], [eGIF, 03b]. The UK’s e-Government Interoperability Framework (e-GIF) sets out the UK Government’s technical policies and specifications for achieving interoperability and information systems coherence across the public sector. The e-GIF defines the essential pre-requisites for joined-up and web enabled government. It is a cornerstone policy in the overall e-Government strategy.

10. Thanks to Giorgio da Bormida who is Chief Technical Officer at Giunti Labs. The contact URL is: http://www.mobilearn.org.
11. The contact URL is: http://www.govtalk.gov.uk/schemasstandards/egif.asp.

Adherence to the e-GIF specifications and policies is mandatory. They set the underlying infrastructure, freeing up public sector organizations so that they can concentrate on serving the customer through building value added information and services. It will be for the organizations themselves to consider how their business processes can be changed to be more effective by taking advantage of the opportunities provided by increased interoperability.

The main thrust of the framework is to adopt the Internet and World Wide Web specifications for all government systems. Throughout this section use of the term "system" is taken to include its interfaces. There is a strategic decision to adopt XML and XSL as the core standards for data integration and management of presentational data. This includes the definition and central provision of XML schemas for use throughout the public sector. The e-GIF also adopts specifications that are well supported in the market place. It is a pragmatic strategy that aims to reduce cost and risk for government systems whilst aligning them to the global Internet revolution.

The Framework also sets out policies for establishing and implementing meta-data across the public sector. The e-Government Metadata Standard will help citizens find government information and resources more easily. Stipulating policies and specifications in themselves is not enough. Successful implementation will mean the provision of support, best practice guidance, toolkits and centrally agreed schemas. To provide this, the government has launched the UK GovTalk™ web site. This is a Cabinet Office led, joint government and industry facility for generating and agreeing XML schemas for use throughout the public sector. Schemas can be found at http://www.govtalk.gov.uk/interoperability/xmlschema.asp. UK GovTalk™ is also used for wide consultation on a number of other e-Government frameworks and documents.

At the present time the eGIF recommendations have started to address eLearning (Table 10 in [eGIF, 03b]). The eGIF shows that several of the IMS specifications are under review for adoption within eGIF; see Table C1. Later versions of eGIF will further expand on which eLearning specifications are to be formally adopted. Under e-Government there is also a set of recommendations for the usage of XML Schema [eGIF, 02].

Table C1 - The eGIF eLearning specification/standards recommendations.

Industry Standard and Sponsoring Organization	**Areas covered by the standards developed by the organization**	**eGIF status** A = Adopted; see notes for applicability R = Recommended for consideration U = Under review by an ad-hoc group F = For future consideration	
		Status	e-GIF area of applicability
IMS Content Packaging (V1.1.2) Information Model Sponsor: IMS Global Learning Consortium, Inc. http://www.imsglobal.org/	e-learning	R	Recommended for consideration by OeE/DfES e-learning Working Groups
IMS Content Packaging (V1.1.2) XML Binding Sponsor: IMS Global Learning Consortium, Inc. http://www.imsglobal.org/	e-learning	R	Recommended for consideration by OeE/DfES e-learning Working Groups
SCORM 1.2 Content Aggregation Model application profile Sponsor: ADL http://www.adlnet.org/index.cfm?flashplugin=1&fuseaction=home	e-learning	U	Under review by OeE/DfES e-learning Working Groups
SCORM 1.2 Runtime API application profile Sponsor: ADL http://www.adlnet.org/index.cfm?flashplugin=1&fuseaction=home	e-learning	R	Recommended for consideration by OeE/DfES e-learning Working Groups
IEEE 1484.12.1 - 2002 LOM Sponsor: IEEE http://www.ieee.org/	e-learning	R	Recommended for consideration by OeE/DfES e-learning Working Groups

Industry Standard and Sponsoring Organization	**Areas covered by the standards developed by the organization**	**eGIF status** A = Adopted; see notes for applicability R = Recommended for consideration U = Under review by an ad-hoc group F = For future consideration	
		Status	e-GIF area of applicability
IMS Meta-data (V1.2.1) XML Binding Sponsor: IMS Global Learning Consortium, Inc. http://www.imsglobal.org/	e-learning	R	Recommended for consideration by OeE/DfES e-learning Working Groups
IMS Question and Test Interoperability (V1.2.1) Information Model Sponsor: IMS Global Learning Consortium, Inc. http://www.imsglobal.org/	e-learning	R	Recommended for consideration by OeE/DfES e-learning Working Groups
IMS Question and Test Interoperability (V1.2.1) XML Binding Sponsor: IMS Global Learning Consortium, Inc. http://www.imsglobal.org/	e-learning	R	Recommended for consideration by OeE/DfES e-learning Working Groups
IMS Enterprise (V1.1) Information Model Sponsor: IMS Global Learning Consortium, Inc. http://www.imsglobal.org/	e-learning	U	Under review by OeE/DfES e-learning Working Groups
IMS Enterprise (V1.1) XML Binding Sponsor: IMS Global Learning Consortium, Inc. http://www.imsglobal.org/	e-learning	U	Under review by OeE/DfES e-learning Working Groups
IMS Learner Information Package (V1.0) Information Model Sponsor: IMS Global Learning Consortium, Inc. http://www.imsglobal.org/	e-learning	R	Recommended for consideration by OeE/DfES e-learning Working Groups
IMS Learner Information Package (V1.0) XML Binding Sponsor: IMS Global Learning Consortium, Inc. http://www.imsglobal.org/	e-learning	U	Under review by OeE/DfES e-learning Working Groups
IMS Reusable Definition of Competency or Educational Objective (V1.0) Sponsor: IMS Global Learning Consortium, Inc. http://www.imsglobal.org/	e-learning	U	Under review by OeE/DfES e-learning Working Groups
IMS Digital Repositories (V1.0) Sponsor: IMS Global Learning Consortium, Inc. http://www.imsglobal.org/	e-learning	U	Under review by OeE/DfES e-learning Working Groups
IMS Simple Sequencing (V1.0) Sponsor: IMS Global Learning Consortium, Inc. http://www.imsglobal.org/	e-learning	U	Under review by OeE/DfES e-learning Working Groups
IMS Learning Design (V1.0) Sponsor: IMS Global Learning Consortium, Inc. http://www.imsglobal.org/	e-learning	U	Under review by OeE/DfES e-learning Working Groups
IMS Guidelines for Developing Accessible Learning Applications (V1.0) Sponsor: IMS Global Learning Consortium, Inc. http://www.imsglobal.org/	e-learning	R	Recommended for consideration by OeE/DfES e-learning Working Groups

Industry Standard and Sponsoring Organization	Areas covered by the standards developed by the organization	eGIF status A = Adopted; see notes for applicability R = Recommended for consideration U = Under review by an ad-hoc group F = For future consideration	
		Status	e-GIF area of applicability
BS7988 A code of practice for the use of IT in the delivery of assessments Sponsor: BSI http://www.bsi-global.com/	e-learning	R	Recommended for consideration by OeE/DfES e-learning Working Groups
BS8426 A code of practice for e-support in electronic learning systems Sponsor: BSI http://www.bsi-global.com/	e-learning	F	For future consideration by OeE/DfES e-learning Working Groups
BS8419 Interoperability between Metadata Systems used for Learning, Education and Training Sponsor: BSI http://www.bsi-global.com/	e-learning	F	This is under development and will be considered in the future by OeE/DfES e-learning Working Groups

C10.1 - Relationship to IMS Activities

IMS staff and several of IMS's UK Contributing Members have been advising the UK's eEnvoy Office on the recommendations for the adoption of eLearning specifications within eGIF.

C11 - UK Joint Information Systems Committee (JISC) Architectures[12]

Within the UK, the Joint Information Systems Committee (JISC) is responsible for undertaking applied research and development into information systems for adoption with the Higher Education, and increasingly the Further Education, communities. In the past few years there have been three eLearning architectural focused projects, namely:

- Managed Leaning Environments (MLEs) and Virtual learning Environments (VLEs);
- Digital Electronic Library Integration within Virtual Environments (DELIVER);
- Authenticated Network Guided Environment (ANGEL).

C11.1 - Relevant JISC Projects

Managed Learning Environments

While recognizing that the world at large will continue to use terminology in different and often ambiguous ways, the term **Virtual Learning Environment (VLE)** is used to refer to the "online" interactions of various kinds which take place between learners and tutors. The JISC MLE Steering Group has said that VLE refers to the components in which learners and tutors participate in "online" interactions of various kinds, including online learning. The JISC MLE Steering Group has said that the term **Managed Learning Environment (MLE)** is used to include the whole range of information systems and processes of a college (including its VLE if it has one) that contribute directly, or indirectly, to learning and the management of that learning.

The principle functions that the complete VLE needs to deliver are:

12. Further information on the JISC projects is available from: http://www.jisc.ac.uk.

- Controlled access to curriculum that has been mapped to elements (or “chunks”) that can be separately assessed and recorded;
- Tracking student activity and achievement against these elements using simple processes for course administration and student tracking that make it possible for tutors to define and set up a course with accompanying materials and activities to direct, guide and monitor learner progress;
- Support of online learning, including access to learning resources, assessment and guidance. The learning resources may be self-developed, or professionally authored and purchased materials that can be imported and made available for use by learners;
- Communication between the learner, the tutor and other learning support specialists to provide direct support and feedback for learners, as well as peer-group communications that build a sense of group identity and community of interest;
- Links to other administrative systems, both in-house and externally.

As shown in the Figure C10, the VLE will act as a ‘portal’ to online Curriculum Mapping, Assessment, Communication, Delivery, Tutor support and Tracking facilities. The VLE makes up only one part of the college’s overall systems (both computerized and non-computerized). Interfacing between these systems is possible by 'connecting up' the constituent parts by the use of interoperability standards such as IMS (plus required extensions). Examples of these are between the Student Record Systems and the VLE and between Learning resources (or content) and the VLE.

JISC recently completed a series of pilot projects that investigated interoperability between various eLearning system components as used with the UK Further Education college system. The framework shown in Figure C9 is the result of investigating the ways in which some interoperability specifications can be used to support intra- and inter- College data exchange.

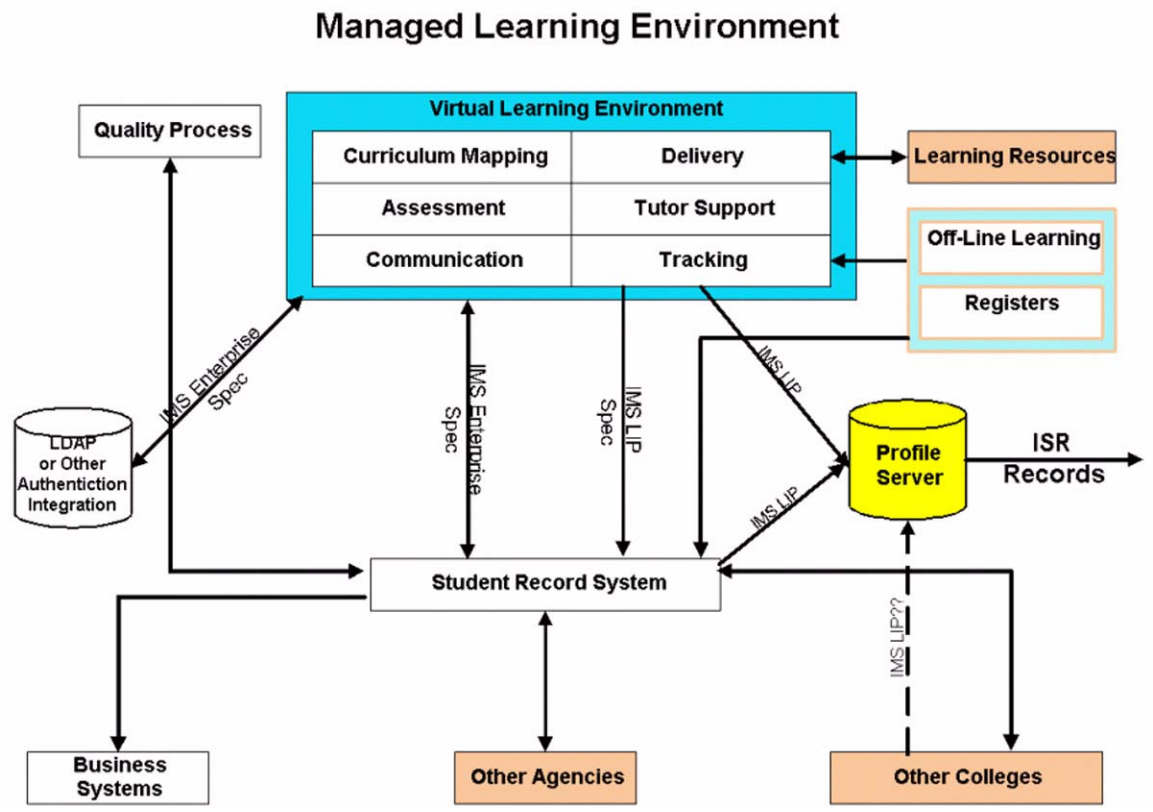


Figure C8 - The JISC MLE framework.

Digital Electronic Library Integration within Virtual Environments

The DELIVER project (this is scheduled for completion in July 2003) will design and implement practical software tools for end-users and administrators of institutional VLEs and Library Management Systems to facilitate the consistent creation and easier use of course-based resource lists.

The overall aim of the project is to create a series of library tools that will allow access to a range of library systems and services on a course-specific basis from within a VLE. This software tools will encompass generic library tools, subject-specific tools and course-specific, directed learning tools.

The specific objectives are to:

- Produce software tools that advance the use of middleware within UK HE and FE;
- Work to establish a highly visible presence for library resources within VLE courses;
- Improve resource management work-flow across departments, addressing duplication of effort;
- Improve communication between departments;
- Work to re-conceptualize the way resource lists are considered, created, used and supported with institutions.

Authenticated Network Guided Environment

With an increasing number of courses available online and the proliferation of networked information resources there is a pressing need to provide integrated access for students and to integrate support and management from academics and librarians. The JISC has funded the development of an Authenticated Networked Guided Environment for Learning (ANGEL) to provide powerful access management tools. ANGEL will be able to offer students:

- Integrated access to: digital library resources, for example from the local library, the JISC's Distributed National Electronic Resource (DNER) and the wider Web; online learning as delivered, for example, by WebCT and similar course content management tools; and locally produced resources such as reading lists, examination papers, lecture notes, electronic course packs, model answers, discussion lists and tutorial advice;
- Customized and personalized views of available resources based on course needs, current progress and personal preferences;
- Seamless access to all the services;
- Support across the whole range of material.

Educators and librarians will be able to:

- Design courses that effectively combine all kinds of material;
- Track student progress through all resources;
- Offer support and guidance across all sorts of networked sources.

ANGEL built on research in integrating access in the hybrid library, such as the Headline PIE, and in VLEs, such as the De Montefort University's Learning Domain. The Networked Authentication part of the effort developed an access management system. ANGEL also worked closely with those responsible within the JISC for authentication and security of information. As well as portable, reusable, open software components for use by Further and Higher Education, the project delivered updates on personalization, authentication and authorization developments and public reports on meta-data, licensing and other interoperability issues.

C11.2 - Relationship to IMS Activities

JISC is an IMS Contributing Member with the Centre for ELearning Technology and Information Systems (CETIS) acting on their behalf[13]. The IMS receives extensive input from CETIS on JISC activities and the CETIS technical team play a significant role in the development of the IMS specifications. IMS has received many core use cases from the UK Higher Education and Further Education domain, particularly regarding the IMS Enterprise, Content Packaging and LIP specifications. CETIS runs many special interest groups aligned with the IMS specification activities.

13. Further information on CETIS is available from: http://www.cetis.ac.uk.

C12 - Electronic Business XML

Electronic Business Extensible Markup Language (ebXML) is an international initiative established by the United Nations Centre for Trade Facilitation and Electronic Business (UN/CEFACT) and the Organization for the Advancement of Structured Information Standards (OASIS) with a mandate to undertake a 15-18 month program of work. As identified in the ebXML Terms of Reference, the purpose of the ebXML initiative is to research and identify the technical basis upon which the global implementation of XML can be standardized. The goal is to provide an XML-based open technical framework to enable XML to be utilized in a consistent and uniform manner for the exchange of electronic business (eb) data in application to application, application to human, and human to application environments—thus *creating a single global electronic market.*™ [14]

ebXML is based on international standards and is itself intended to become an international standard [ebXML. 01a], [ebXML, 01b], [ebXML, 01c], [ebXML, 01d], [ebXML, 01e], [ebXML, 01f], [ebXML, 01g]. A key aspect for the success of the ebXML initiative is adherence to the use of the W3C suite of XML and related Web technical specifications to the maximum extent practical. Although these specifications may not provide the optimal technical solution, acceptance of ebXML by the business community and technical community is tied to XML. However, certain key elements of the ebXML technical framework may require adopting alternative technologies and technical specifications—such as those of the Internet Engineering Task Force (IETF), International Organization for Standardization (ISO), Institute of Electrical and Electronics Engineers (IEEE), International Electrotechnical Commission (IEC), UN/CEFACT, OASIS, and the Object Management Group (OMG).

The major ebXML technical specifications consist of the:

- Technical Architecture Specification - contains an overview of the technical infrastructure that comprises ebXML and itemize the design rules and guidelines;
- Repository and Registry Specification - includes functional specification and technical design, interfaces, services;
- Transport, Routing and Packaging Specification - addresses transport of ebXML messages, the means of security employed, and the physical construction of the messaging used within the scope of the ebXML system. Specific deliverables include -
  - message structure specification
  - message header specification
  - a textual API example
  - choreographic of messages
  - security specification;
- Business Process Modeling Specification - the business process metamodel and the recommended methodology for using it;
- Core Components Specification - the set of ebXML core components and the prescribed methodology for deriving them;
- Trading Partner Specification - A collaboration profile template that supports manual and electronic discovery and agreement.

C12.1 - Relationship to IMS Activities

At the current time there is no formal relationship between IMS and ebXML. The IMS specification development activity is evaluating the ways in which ebXML could be used to support eLearning.

14. "*creating a single global electronic market" is a trademark of the ebXML Working Group*

About This Document

Title	The IMS Abstract Framework: White Paper
Authors	Colin Smythe (IMS)
Version	1.0
Version Date	01 July 2003
Status	Final
Summary	During the past 30 months the IMS has released a unique set of interoperability specifications within the eLearning technology community. The IMS Abstract Framework (IAF) is a device to enable the IMS to describe the context within which it will continue to develop its eLearning technology specifications. This framework is not an attempt to define the IMS architecture, rather it is mechanism to define the set of interfaces for which IMS may or may not produce a set of interoperability specifications. In the cases where IMS does not produce a specification then every effort will be made to adopt or recommend a suitable specification from another organization. It is the intention of IMS that this Abstract Framework and the associated IMS specifications produced to realize the exchange of information between the identified services will be adopted in a manner suitable for a particular system requirement.
Revision Information	01 July 2003
Purpose	The first formal distribution of the IMS Abstract Framework. This document has been distributed to the IMS Technical Board, the IMS Contributing Members, the IMS Developers Network and organizations that participated in its development.
Document Location	http://www.imsglobal.org/af/index.cfm

List of Contributors

The individuals who contributed to the development of this white paper are:

Thor Anderson	Collegis, USA
Bill Blackmon	Carnegie Mellon University, USA
Kerry Blinko	DEST, Australia
Geoff Collier	EduWorks, Inc., USA
Giorgio Da Bormida	Giunti Interactive Labs, Italy
Chris Etesse	Blackboard, USA
Steve Griffin	IMS, USA
Michael J. Halm	CIC/PSU USA
Dirk Herr-Hoyman	University of Wisconsin, USA
Ralph LeVan	OCLC, USA
Tim Magner	SIIA, USA

Jeff Merriman	MIT, USA
Mark Norton	IMS, USA
Dan Rehak	Carnegie Mellon University, USA
Kevin Riley	IMS, USA
Stuart Sim	SUN, UK
Colin Smythe	IMS, UK
Scott Thorne	MIT, USA
Michael Vax	WebCT, USA
Chris Vento	WebCT, USA
Ed Walker	IMS, USA

Revision History

Version No.	Release Date	Comments
Final 1.0	01 July 2003	The first formal distribution of the IMS Abstract Framework. This document has been distributed to the IMS Technical Board, the IMS Contributing Members, the IMS Developer network and organizations that participated in its development. Feedback on the material presented in this document is welcomed. Please send all comments, questions to: csmythe@imsglobal.org.

Index

V

W

X

Z