Exploiting Web Services (EWES) Project Report

Christ Church Canterbury University
University of Lincoln
University of Nottingham
Roehampton University
Unit4 Business Software Ltd

Author	Manjul Bhasin
Project	EWES Project
Sponsor	JISC
Document Ref	
Document Version	2.0

Version History

Version	Summary of Changes	Changed By	Date Changed
1.0	First Release	Manjul Bhasin	04th October 2010
2.0	Second Release	Manjul Bhasin	28th December 2010

1. Executive Summary

The JISC Flexible Service Delivery Programme and Shared Services Roadmap of Roehampton, Lincoln, Christ Church Canterbury, and Nottingham universities lays out the path and stages which the Universities will move through to enable the establishment of true shared services. This document is a result of the initial analysis of web services which are part of the ACMS product developed by Unit4. This will form the basis for universities to gain an understanding of the web services available from the solution suppliers. It will help in developing a detailed technical analysis and finally integration of their internal applications with ACMS and similar web service based functionality moving towards service oriented architecture (SOA).

The findings of the project help in increased knowledge of the implementation of a services layer on an application module, standards and requirements needed to build and implement good web services. The project outcome also provides indicative understanding of the risks and costs of Web Services and SOA developments in an HE context. The report also highlights some of the potential issues and problems which may be faced while implementing projects.

The report incorporates some of the potential project scenarios in which web service may be implemented and associated key factors for consideration. They are described as an aid to provide clarity about integration with the ACMS web services.

The project could not document all aspects of working of ACMS Web Service due to logistical constraints. The project has considered the key components of the ACMS web service but more analysis may be required while considering any integration project to cover all aspects and overcome any potential issues.

This project report sums up the achievement of core objectives set for the success of the project and should act as a significant step in implementation of SOA across participating universities under JISC Flexible Service Delivery Programme.

The lessons learned in Project EWES would be carried forward and addressed in Project Cumulus.

Contents

2. Introduction

The JISC Flexible Service Delivery Programme and Shared Services Roadmap of Roehampton, Lincoln, Canterbury Christ Church, and Nottingham universities lays out the path and stages which the Universities will move through to enable the establishment of true shared services. The institutions have similar ambitions focusing around Service Oriented Architecture / Service Layers and each of these Universities has taken a similar incremental approach with a view to being able to exploit shared services initiatives and other opportunities like SaaS.

The first stage was part of the JISC funded Project Cairo which has explored the use of Enterprise Architecture and Web Services in the HE context. This project, Exploiting Web services (EWES) is next step for strategic development along the Roadmap with supplier partners. Student System Vendor (Unit4, formerly Agresso) was approached for the project and agreement was reached to study the “Services Layer” made available with their Curriculum Management Module (ACMS) application.

The project explored how the ACMS Services Layer is defined and built and looked at access to the data contained in the Module to enable users/other applications to safely and accurately consume the data. Thus the project has strived to increase the understanding of how the Services Layer provided by ACMS can be exploited and sheds light on the problems to be overcome to achieve this. The project also establishes the key best practices parameters for web services and how ACMS web service fairs against them. It also provides guidelines for using ACMS web service in various common application types.

The project seeks to begin to bridge the gap between vendors and users, opening up the wider exploitation of data contained in best of breed systems in a way which is standardised, re-usable and is not dependant on specific databases or application structures. The project increases knowledge of the implementation of a services layer on an application module and increases understanding of the standards and requirements needed to build and implement web services.

3. Background

Christ Church Canterbury University, the Universities of Lincoln and Nottingham, Roehampton University have similar ambitions focusing around Service Oriented Architecture/Service Layers. Although the concept of delivering “services” is now well understood there are few examples of the implementation of an SOA in the HE Sector and what there is tends to be expensive and a result of work carried out with one of the major database vendors. Each of these Universities has taken a similar incremental approach with a view to being able to exploit shared services initiatives and other opportunities like SaaS which closely aligned with the JISC Flexible Services Delivery (FSD) Programme.

The first stage of the programme was part funded through the JISC Institutional Exemplars Programme: Project Cairo, which has explored the use of Enterprise Architecture and Web Services towards achieving a Service Oriented Architecture. Through this project, the university has successfully replaced key modules with Web Services. For example, its in-house Identity Management System functions for Barcode Creation, Access/ID Card System and LDAP integration have been r-engineered as Web Services illustrating the key characteristics of re-usability and loose coupling.

The next steps for strategic development along this path are based on Shared Services with supplier partners. The JISC FSD Programme worked to bring vendors and suppliers into this discussion, inviting them to make their applications more “open” through web services based data accessibility outside their own product portfolios. Student System Vendor (Unit4, formerly Agresso) were approached for the project and responded positively. This Project focuses on a workplan to explore the detailed activities required to put in place a web services approach to system integration. As such, the project will be a significant move forward in the JISC aims for the Flexible Services Delivery.

Following this project, the next steps for strategic development along the Roadmap are based on Shared Services with supplier partners followed by consolidation of services into applications. At the end of the roadmap is the move to a full shared services scenario based on application and module independence.

4. Concept behind the project

The project has focused on web services exploitation of a vendor provided “Services Layer” which Unit4 have made available with their Curriculum Management Module (ACMS). This Services Layer is already a component of the ACMS Module. The project seeks to begin to bridge the gap between vendor and user, opening up the wider exploitation of data contained in best of breed systems in a way which is standardised, re-usable and is not dependant on specific databases or application structures, in other words is truly interoperable.

The project outcome is expected to:

- Increased knowledge of the implementation of a services layer on an application module
- Increased and accurate documentation regarding the offering and consuming of data in this manner
- Increased understanding of the standards and requirements needed to build and implement web services.
- Increased understanding of the risks and costs of Web Services and SOA developments in an HE context.
- Contribution to the illumination of the issues, problems and costs of web services development and the development of common schemas

These developments are key facets of the JISC FSD programme as they open up the way to interoperability and services re-use in a loosely coupled software environment.

5. Execution and deliverables

The project was structured over 6 months with six weekly board meeting. Project involved few visits to Unit4 Offices as most activities were carried out offsite with the help of remote server access and communication through emails and phone.

The deliverables of the project are:

- Documentation on how the “services layer” is defined and built and the validation that it is a truly “open” access to the data contained in the Module
- If this is not the case, recommendations as to how this can be achieved
- Developed software and documentation to ensure that the layer is truly and comprehensively accessible to the users to enable them safely and accurately to consume the data. Unit4 have agreed to make systems and staff available for the Project officer at their site.
- Descriptions of the business, application and infrastructure architectures which are crucial to the use of web services in this context
- Developed web services which can consume the data presented by the vendor services layer and subsequently offer this data for “consumption” by Users e-Learning systems, timetabling systems and on-line module selection. The project will be operated with a consortium of Universities offering different approaches to curriculum data and different databases. All 4 users have agreed to allow the Project Officer to work on their sites. The use of Blackboard/Webct Vista, Moodle and some Web 2.0 systems such as Wordpress are common to all four universities.
- Documented issues involved in the development of web services which will be re-usable across different user sites.
- Full documentation of any outstanding problems or issues which are fundamental to the concept of web services development in this context

6. Project Timelines

The key phases of the project were:

January 2010	JISC approval for the project
February – March 2010	Detailed project planning
February 2010	Invitation to tender for project contractor
30th March 2010	Project initiation
April – September 2010	Project execution
October 2010	Project report

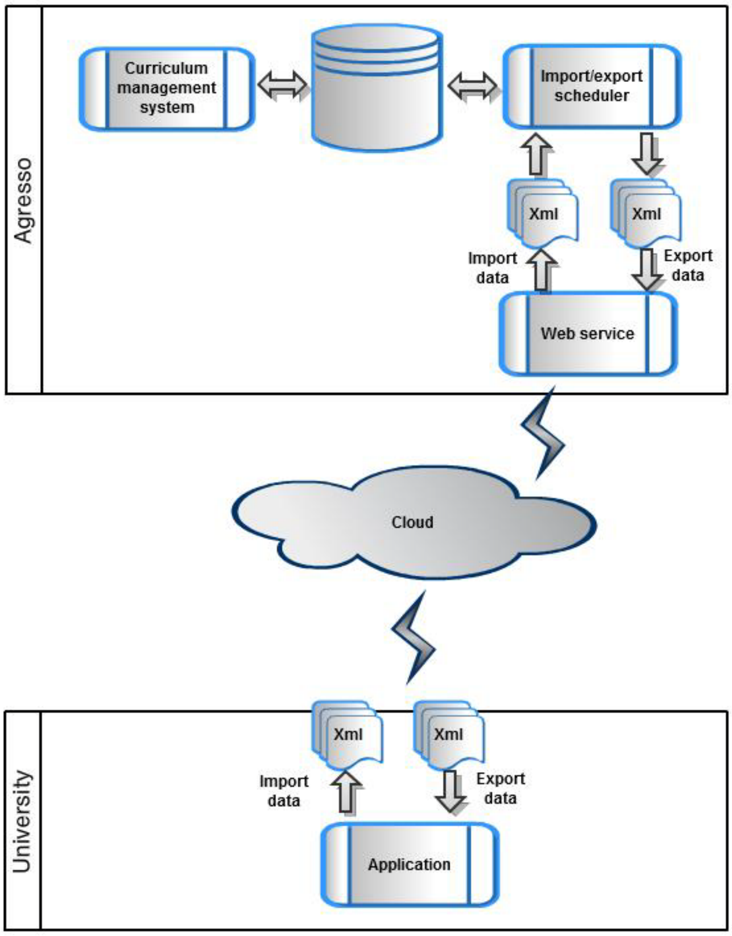
7. Methodology

The project methodology involved:

- The project scope was defined in the project initiation document which was reviewed and agreed upon in the first board meeting.
- Significant amount of work in understanding the Unit4 ACMS and Web Service was carried out through on site review meetings, documentation, email exchange, and remote access to ACMS on Unit4 servers.
- Universities defined curriculum management profiles were uploaded onto the ACMS for the study of web service.
- The project review and testing of ACMS has been done through remote access to Unit4 servers.
- This report is towards conclusion of the project.

8. Functionality Overview of ACMS Web Service

The Unit4 ACMS is composed of three components. Name them. All 3 components work together to enable the data import and export functionality through web services. The components are described below:



8.1 ACMS User interface

The curriculum management system is a Windows based user interface application for managing the various entities of the education domain. Full analysis of CMS is beyond the scope of this document.

The ACMS UI application helps in performing the first step of enabling import and export of data using web services. A user can create different profiles in ACMS UI application. A profile represents a subset of the data which can be used for import and export processes. The user can pick and choose different kinds of entities that are relevant for the application which require the import/export functionality. The client applications making the web service calls refer to the profiles in which they are interested.

It is important that the profile specifies only the essential entities, because including too many entities can result in very large data set for import/export processes.

The client applications working on the data received from the CMS web services will presumably work with only a subset of the entire CMS data. Hence, a profile will be configured for each such subset.

8.2 Import/Export manager

The scheduler application gets the data in and out of the ACMS application. Based on the user profiles configured in ACMS, this process runs periodically (minimum of 1 minute interval) and exports data from ACMS into XML files, or imports XML data sent by client applications back into ACMS. The frequency of its execution is configurable.

Since the import/export manager runs every few minutes, the data available through it is never real time data.

8.3 Web Services

The web service is the interface for other applications to work with ACMS data. There are two web methods defined in the web service:

1. **ExportData**: When called, this web method returns the XML file created by the Import/export manager. This web service is called with profile name as one of the parameters. This profile name should match the profile created using the ACMS application. Hence the data expected will comprise of the entities selected for that profile.
2. **ImportData**: When called, this web method queues up the XML data for import/export manager. This web method expects the XML data to be imported as one of the parameters. At scheduled intervals, the import/export process picks up the given XML and processes it for import. It does not give any indication whether the import processes succeeded or failed, partially or fully.

Both import and export method calls work on a subset of data representing the profile. A profile is created by selecting tables and fields from the entire dataset to be part of the profile, naming it and then scheduling it to produce the XML file for the web service at specified intervals. The scheduler service creates the profile.

The specifics of the web methods are given below:

8.3.1 Credentials

The credentials are used for authorising the web method calls. The credentials used must be registered in ACMS, and must be associated with the profile of the data that is imported or exported. The three components of credentials are:

- **UserName:** Represents the login user name for ACMS application.
- **Password:** Password for authentication of the user name.
- **Client:** Name of the client to which this username and password belongs. This is the same client name used for logging into ACMS.

8.3.2 ExportData Web Method

This web method takes 3 parameters:

- **ExportProfile:** The profile representing the subset of the data to be exported from ACMS. This should match exactly with the name of the profile created in the ACMS application.
- **DocumentTitle:** This represents an informative title identifying the document. This title can contain a timestamp to uniquely identify the document to help in troubleshooting.
- **Credentials:** The credentials for authenticating the user, described above.

8.3.3 ImportData Web Method

Import data web method is called with these parameters:

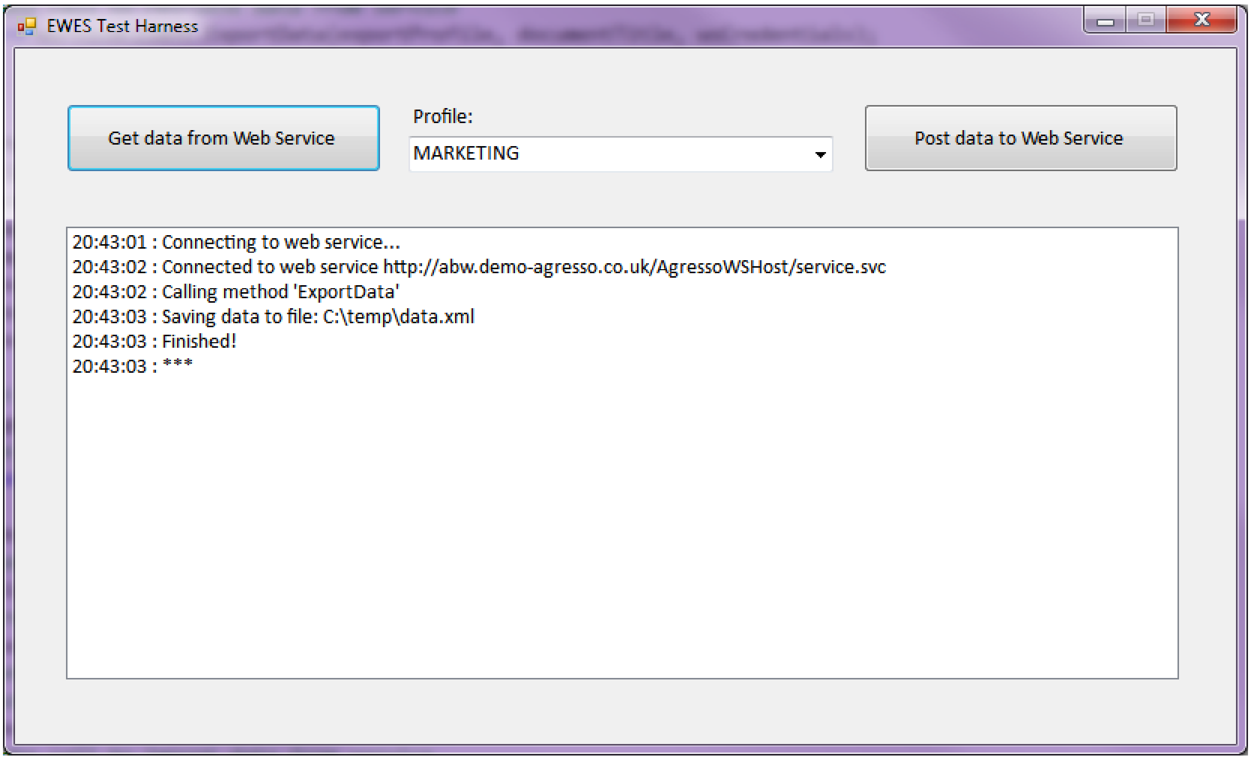
- **Client**: The identity of the client sending the data to the web service. This is the same client ID that you use to login to ACMS application.
- **ImportProfile**: The profile representing the subset of the data to be imported into ACMS. This should match exactly with the name of the profile created in the ACMS application.
- **Data**: The XML containing the data to be imported. This should match the profile that is used while calling this web method.
- **OverwriteExisting**: If this is set to false, the import process will not import the data if a record already exists in the CMS database. If this flag is set to true, then it will overwrite the records in the CMS database.
- **DocumentTitle**: This title can contain a timestamp to uniquely identify the document to help in troubleshooting.
- **DocumentDescription**: Additional informative description of the document.
- **DocumentComments**: Another field for adding informative description for the web service calls.
- **Credentials**: The credentials for authenticating the user, described above.

8.4 Test Harness

The test harness is a .NET based application for demonstrating the import and export web method calls.

It has a very simple user interface allowing the users to select a profile and then either import or export the data using the two buttons. As the application goes ahead with calling the web service, it logs the progress in the large text box.

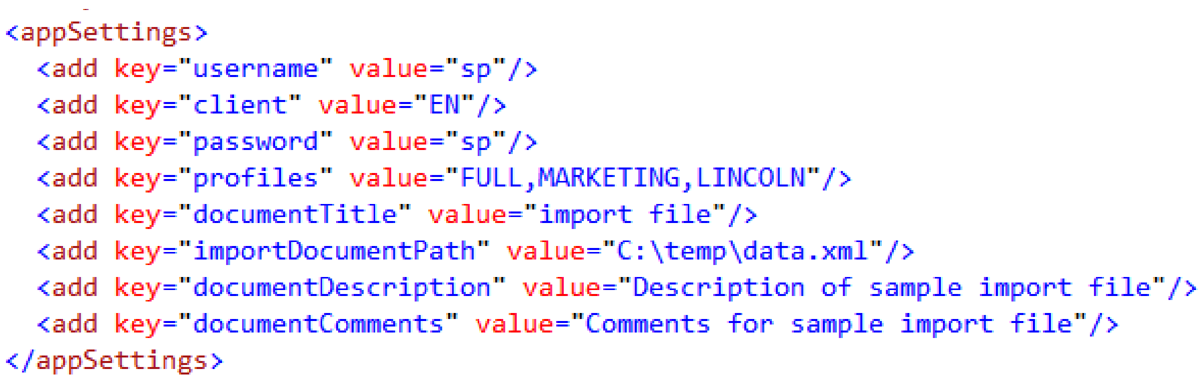
The various parameters used for the web methods such as username, password etc are stored in the web.config file of the test harness.



Profile: MARKETING

```
20:43:01 : Connecting to web service...
20:43:02 : Connected to web service http://abw.demo-agresso.co.uk/AgressoWSHost/service.svc
20:43:02 : Calling method 'ExportData'
20:43:03 : Saving data to file: C:\temp\data.xml
20:43:03 : Finished!
20:43:03 : ***
```

Settings in the web.config file.



```
<appSettings>
  <add key="username" value="sp"/>
  <add key="client" value="EN"/>
  <add key="password" value="sp"/>
  <add key="profiles" value="FULL,MARKETING,LINCOLN"/>
  <add key="documentTitle" value="import file"/>
  <add key="importDocumentPath" value="C:\temp\data.xml"/>
  <add key="documentDescription" value="Description of sample import file"/>
  <add key="documentComments" value="Comments for sample import file"/>
</appSettings>
```

9. ACMS Web Service Analysis

A thorough complete analysis of the web service can be done only after the requirements of the client application requirements are known. However, a few generic conclusions can be made with regards to the requirements of SOA web service best practices.

9.1 Type / standards of web service

Web services can be created using SOAP/REST

9.2 Technology used:

Since the interaction with the web service is based on open standards (SOAP), it does not bring the client application into using a specific platform for development. The client applications can be made using any technology of choice, as long as the client technology can call a web services and work with XML data.

The demo application is however created using Microsoft .NET 3.5.

10. How does service oriented architecture benefit the universities?

Below are key criteria which define key factors in a web service and how ACMS Web Service stands against them.

10.1 Service Contract

The web service exposes web methods which can be called by client applications, as long as the address of the service, name of the web methods and input and output parameters remain unchanged. This is known as a service contract.

The client applications interoperate with the web service based on service contracts. The only coupling between the service and consumer applications is the service contract. All implementation is abstracted from each other. The service contracts are never expected to change without agreement from all parties. However more service calls can be added as and when required to support new functionality. It is also possible to add newer versions of service calls, but the existing versions should always be supported so that all existing client applications continue working well.

It might be an idea to analyse the given web methods with respect to requirements of a few sample client applications. This will ensure that the given service contract is suitable for most common purposes. After such a suitability study the service contract can be frozen.

10.2 Service Granularity

The quantity of functionality encapsulated by a web service call determines the service granularity. This granularity affects the design, performance and various other aspects of the consuming applications.

If the service exposes very fine grained calls, then the consumer applications become very chatty i.e. they depend heavily on service availability and responsiveness. If however, the service exposes only a few coarse grained calls then the consuming application is responsible for implementing much of the functionality but with lesser network traffic. This also means that there will be a large amount of business logic and validation logic associated with each service call.

In ACMS web services, the service calls are very coarse grained. It simply exposes 2 service calls – ImportData and ExportData, each of which can return a very large chunk of XML data and can potentially take several seconds to complete the processing.

In this scenario the consuming applications are likely to be designed to have their own data stores. These applications will also contain most of the business logic related to the dataset that they add or update. These data stores can be periodically synchronises with ACMS using the import/export web service calls.

10.3 Security

Web service can be implemented with two levels of security - Transport level security and Message level security. Both kinds of security operate independent of each other.

The web services use HTTP protocol which is inherently insecure. HTTP is an inherently insecure protocol because all information is sent in clear text between the client and the server. To secure HTTP, SSL security can be applied.

The current web service does not implement SSL. However, the currently deployed version is only a demonstration version. It is expected that the production version of the web service will incorporate SSL.

The message level security can be provided using WS-Security protocol or using custom security mechanism. The ACMS web service expects security credentials with every call to a web method. It authenticates the security credentials before performing the requested action.

10.4 Transactions

Web services play an important role in integrating applications. A complex business process that accesses multiple web services needs to ensure that no part of the business data gets out of sync with the other parts. Transactions help in maintaining the ACID properties of the various data sources exposed by web services, but locking the data records till the transaction is not successfully committed.

The web methods exposed by ACMS web service can take a long time to complete. Transactions should be as short as possible. Keeping the transactions open for long durations is not a viable option. Hence the current web methods are not suitable for supporting transactions.

10.5 Administration / Troubleshooting

While importing the data sent from client applications to ACMS, it is possible that import fails due to one of several reasons like business rules validation failure, incorrect format of XML file etc. In such cases it is helpful to find out the result of the import process.

At the moment there is no web method to allow finding out the status of the last operation or any error that results from that. It would be advisable to identify and agree in troubleshooting mechanisms before developing applications that updated data using ACMS web service.

Since the web service is very coarse grained, a lot of the business logic will have to be duplicated in the client applications to avoid failures due to them. The solution could be – to provide additional fine grained web methods which encapsulate business logic and help in developing client applications that can display those validation errors to users in real time.

10.6 Logging messages for auditing

Logging helps in keeping a complete, persistent record of web service requests, responses, and faults. During development it is often very convenient to be able to inspect the contents of message sent to diagnose problems.

Auditing also enables you to design a strategy that records and analyzes security-related events. A history of audit records can produce an audit trail enabling the reconstruction and examination of a sequence of events. The audit trail can be used to detect attacks, confirm compliance with policy, deter abuse, and so on.

Since there is a possibility that ImportData can fail due to validation logic or other reasons, it would help if the exact XML sent is logged. This can help in conflict resolution and/or troubleshooting.

10.7 Availability/Outages

These should be covered by SLA (service level agreements). The consuming applications are likely to be designed with their own data stores; hence the outages are less significant as compared to error handling and error communication.

10.8 Concurrency

There is a possibility to write many applications importing and exporting data to ACMS, with overlapping profiles. The updates from one application can overwrite updates from another application. The data can contain last modification information to minimise this, but in that case conflict resolution becomes more important and tricky.

The design process of the client applications should take this into account and implement workarounds for checking concurrency or failures in importing data sue to concurrency.

10.9 Reliability

Reliability refers to the ability of a system or component to perform its required functions under stated conditions. If we call a web service but we cannot be certain that it has updated the data, then we have created a point of failure which will need troubleshooting.

Reliability of the application can be improved by:

Testing: A very thorough testing process can enable the application to anticipate and handle the unexpected situations like loss of network connection, etc. This will enable identifying the various kinds of error handling situations and notifications to the user.

Monitoring: Regular monitoring the service calls on regular basis can help identify the problems early. Early identification of problems makes resolution comparatively easier.

10.10 Performance

The calls to the import and export web methods potentially involve transferring large amounts of data. Due to this, the web service calls are not likely to be very high level of performance.

However, if the applications work with their own data stores then the performance will depend on their individual design and build.

10.11 Data Size

The size of the data transferred between client application and web service depends on the entities selected while creating the profile. It is possible that the size of the data transferred is in several megabytes. Transferring such a large dataset can take several seconds or even minutes. The client applications should be designed to handle this kind of delay.

To help reduce the size of the data transferred, one of the compression mechanisms such as HTTP compression should be employed.

10.12 Standards

Web service can be created using 2 different standards – SOAP or REST. Each has its own strengths.

SOAP currently has the advantage of better tools where they will generate a lot of the boilerplate code for both the service layer as well as generating clients from any given WSDL.

If the client applications are more likely to be RIAs or Ajax clients, REST is a simpler, can be easier to maintain as a result, lies at the heart of Web architecture, allows for better protocol visibility, and has been proven to scale at the size of the web itself. . If you keep your requests simple, you can make service calls directly from your JavaScript, and that comes in very handy.

The ACMS web service is based on SOAP protocol. Hence, it should be possible to use the tools provided by development environment to easily call the import and export web methods.

11. Using ACMS Web Service as per various scenarios

This section discussed some common type of the applications which can consume the ACMS web service. The content below is only to provide clarity, these are not design recommendations.

11.1 Administration applications

The administration type applications would generally have low number of users but probably high number of transactions. The user of the administration applications is usually a staff member who can be trained and expectations can be set about responsiveness of the application. However, for efficiency reasons the reliability of the data exchanges is paramount.

If the admin users update any data, it would be good to easily ensure that the changes are reliably reflected in the ACMS application. The current web services do not cater for real time response to the import process. Probably some further changes in the web service can be expected from Unit4 to cater for this.

For the data required for read only purposes, no such problems are expected. However, it should be understood that the data received from the web service is not real time.

11.2 Public facing applications

This type of applications would generally have a high number of users but probably a high number of transactions/ internet traffic and their user expects would be set as per normal internet based applications which are responsive and reliable.

The kind of responsiveness desired can be achieved if:

1. The ACMS data profile represents a small amount of data, which can be transmitted from web service in real time for read purposes
2. The application has its own data store.

If the application encounters any problems while updating the data, it should be handled transparently.

11.3 Multiple applications with overlapping data

Applications can be created to work with different profiles/ web services of data from ACMS. However, two or more profiles can refer to the same entities. For example, marketing application and student administration applications are likely to work on profiles containing student data. Due to the non real time nature of the web service, the applications are likely to have their own data stores. It is also possible that data is modified using ACMS application and the same data is modified using other client applications.

If the two applications are created such that they work on different data stores and they modify the same student data, we can end up with concurrency issues.

Hence, it is advisable that if the applications work on common entities they do not end up modifying the same entities. Even if the applications need data for read only purposes, they should expect to work on data which may not be real time data.

11.4 Offline applications

In offline applications the internet availability is restricted hence an application is not capable of maintaining regular contact with the web service. Thus offline applications do not expect to be working with real time data and would have their own data stores which would be updated through the export from web service. Once updated, the users can access the information even when they are offline.

They can synchronise data with ACMS web service when internet connection is available. The read only data can be refreshed using ExportData web method. For updating the data with ACMS, it can batch up all the data as an XML and send to the web service for import.

The concurrency issues can be amplified in case of data updates as there could be a lot of time delay between updating the data locally and updating it on the ACMS application.

11.5 Third party vendor applications

Usually the third party vendor applications provide the events where custom code can be run. This custom code can help in integrating the application with other applications or web services.

This custom code will have to work within the constraints of the ACMS web service.

If the application does not provide events for plugging in custom code, then a scheduled application can be made which reads the data from the 3rd party database and creates the XML file which can be sent to the web service for importing the data.

11.6 Real time working applications

The data available from the ACMS web service is not real time. Hence any real time application has to ensure that nothing else is updating the data in ACMS platform while it is working with it.

In addition, the real time applications can be expected to have their own transactional data store to provide sufficient performance. All other constraints regarding updating the data and dealing with the problems related to importing data apply to real time applications as well.

12. Key Lessons learned from project

Two general conclusions can be made about the ACMS web service which are likely to have the biggest impact on the design and functionality of client applications.

12.1 Coarse grained

Although it can be argued that the granularity of the data being imported and exported using the web services depends on the granularity of the profiles created in the ACMS application, the web service does not enable the client applications to reuse any ACMS business rules validations. Each client application is likely to duplicate the business validation rules, so that the data sent to the web service has maximum chance of being updated successfully.

The ACMS application simply acts as a central data store, which allows reading and updating any slice of data desired by creation of profiles.

12.2 Scheduled / not-real time

Import and export processes work after certain periodic intervals, and so the data available to read or the process of writing the data back into ACMS is not real time. The client application design and functionality will have to take this into account.

This will not have significant impact if the client applications only read data which changes infrequently. It will have quite a bit of impact on all other types of applications and also on the troubleshooting/administration aspects.

13. Further work - Action on lessons learned

The lessons learnt from the EWES Project will be carried forward and addressed in Project Cumulus which will look more closely into the technical aspects of vendor web services with the objective and imperative of producing a production module. This is in accordance with original vision of moving towards implementation of cloud hosting and SOA across participating universities under the JISC Flexible Service Delivery Programme.

The Key issues to be addressed from this project include:

- Move from synchronous wrapper around an existing batch process to the real-time data.

- The data returned by the web service needs to be much more granular.
- Improvement in security of web service by implementation of technology from the Windows Identity and Communication Foundations.
- Transactions within service to support high levels of consistency and independence.
- Concurrent transactions should be supported by the use of threading.

Project Cumulus will enable the involved universities to work with the vendor to break the link between ACMS and the major systems software of which it is a part. This will allow the ACMS module to exist as a service within the cloud, carrying out a particular business function within a Service Oriented Architecture context. The issues raised in the EWES Project have been vital to indicating necessary changes in the ACMS Application Architecture. These issues have been taken on board by Unit4 and are being monitored to conclusion by the Project Cumulus Project Board.

14. Glossary of terminology used

This section identifies and defines some of the terms used in this document.

- **SOA:** Software oriented architecture is an underlying software architecture that supports the connection between various applications and the sharing of data.
- **ACMS:** Curriculum Management System is the application developed by Agresso which is being used by the universities involved in this project.
- **SOAP:** Simple Object Access Protocol is an XML based message protocol used to communicate with web services.
- **WSDL:** Web service discovery language is an XML based language which describes the details of the web service, its operations and parameters. It can be easily consumed by development tools to automatically generate classes which can work with the web service.
- **SSL**: Secured sockets layer (SSL) is cryptographic protocol which provides security for communications over the internet.