

Name of lead institution/organisation BROCKENHURST COLLEGE
Project Name: SWEET.Net (Enterprise Web Services with Timetable Extensions for Microsoft .Net)
Full contact details for primary contact Name: ROBIN GADD Position: MLE MANAGER Email: RGADD@BROCK.AC.UK Address: BROCKENHURST COLLEGE LYNDHURST ROAD BROCKENHURST SO42 7ZE Tel: 01590 625575 Fax: 01590 625573
Total cost to the JISC: £30,000
Outline proposal description The project will implement the IMS Enterprise Web Services specification on the Microsoft .net platform, extended to allow the storage and retrieval of timetable data. The project deliverables will include C# source code, compiled assemblies, a fully documented timetable API, an exemplar database for Microsoft Sql Server 2000/MSDE, extensible "bridge" classes to facilitate developer integration with other systems, and install scripts to enable the toolkit to function "out of the box". A Java based desktop client application with full documentation and source code will also be produced (to demonstrate web service interoperability across platforms), and a full suite of unit tests.
Names and contact details of any additional contacts JON ROWETT MLE DEVELOPER JROWETT@BROCK.AC.UK Tel: 01590 625555 x.422

Overview of Project

1. Background

Brockenhurst College have developed their MLE Strategy around a home grown application called Emily. Emily is used for attendance tracking, web registers, read-only student records access, staff and student timetables, messaging, progress reviews & action planning, and Individual Learning Plans (ILPs). The system is integrated with other systems (MIS and VLE) via the exchange of IMS Enterprise v1.1 XML documents. During the development of the project, the College has gained considerable experience using the Enterprise specification, web services, and timetable data. The college developed custom timetable extensions to the Enterprise specification, and the MIS department have been using their own custom software for building timetables for many years.

The project is designed to address some common scenarios which appear time and time again during institutional MLE development. Some of them are:

- Assigning students and staff appropriate access rights to VLE content and services;
- Smaller applications which need to be aware of up-to-date class lists, locations of people and groups, enrolments and related data, without duplicating existing data;
- Providing staff & student access to timetables from a range of applications (institutional portals, mobile phones, messaging systems, Personal Information Managers, college calendars).

Many institutions (and some software vendors) have adopted IMS Enterprise v1.x to address some of these needs, partly because the specification is relatively easy to understand and implement. While the new IMS Enterprise Services specification finally defines a much-needed standard transport mechanism for Enterprise data (namely SOAP), it also adds significant complexity to its implementation. As such, there is a clear need for an easy to use toolkit which will reduce the development time needed to “Enterprise enable” an application. Microsoft’s .net technology (and Visual Basic / Active Server Pages before it) has traditionally been popular with smaller institutions such as FE colleges because of its popular Rapid Application Development tools, low cost, good documentation and good support. In most FE colleges, Microsoft languages are widely understood by academic IT staff (much more so than C++ or Java). IMS do not currently publish a web service binding for the storage and retrieval of timetable data; it is our intention to build on our work in this area, and fill this gap with an easy to use timetable web service API, written in Microsoft C#, which works well alongside IMS Enterprise Services (which we will also implement).

2. Aims and Objectives

- To provide an easy to install, easy to use, highly adaptable implementation of Enterprise Web Services (EWS) for the Microsoft platform;
- To extend EWS (using the extensibility provisions in the spec) to allow the transport of timetable data;
- To evaluate the iCalendar / xCal specifications as possible formats for timetable exchange.
- To demonstrate and evaluate interoperability between .net and Java web service technology;
- To present MLE developers with an easy way to move away from duplication of data and functionality, and towards service orientation;
- To evaluate a “test driven” approach to software development.

3. Overall Approach

The work will be structured as a series of coding projects to be undertaken by three developers, using a test-driven software development methodology. Source code and documentation will be regularly released on the development bay, and the team will participate fully in the ELF Project community development process. The .net and Java components will be developed in parallel, as part of a process of continuous testing and review.

Critical success factors include the stability and quality of the EWS specification; the specifics of the .net interpretation of the WSDL files; the ability of Java and .net to interoperate.

We aim to provide a solid and extensible implementation of EWS and of a timetable API. In most scenarios, web services of this type will be subject to authentication, encryption and access control. While we can and will make recommendations for further investigation in this area (such as out-of-band SOAP header authentication, WS-Security), it is outside of the scope of this project to either provide an implementation or dictate the best way to go about this.

4. Project Outputs

- A fully functional, configurable and conformant open-source implementation of Enterprise Web Services (EWS) for the .net platform.
- An extensible framework for incorporating EWS into new applications, or retrofitting existing ones.
- A web service API for the storage and retrieval of timetables.
- Extensions to EWS to enable the location and retrieval of timetables.
- An evaluation of the iCalendar and xCal specifications.
- An evaluation of web service interoperability between the .net and Java platforms.
- An evaluation of test-driven development (TDD).
- A guide to implementing EWS using the toolkit.

5. Project Outcomes

A more robust, loosely coupled way for institutions to solve common integration challenges.

Developers will be able to avoid writing (and re-writing) applications which maintain copies of Enterprise-style data and timetables. Through the use of our framework, it will be easier to Enterprise-enable existing applications, and settle on EWS as a standardised means of data exchange, regardless of the underlying applications, and without developing fragile custom solutions whose data can rapidly go stale.

An easy way to provide electronic access to timetables regardless of the client device or platform.

Many institutions provide web-based access to timetables, usually using the web pages which come with their chosen MIS software. Use of this toolkit will allow them to go far beyond this and present consistent timetables to users on a range of application and hardware platforms, and to develop innovative solutions such as mobile services or applications which are aware of a user's current location by querying a timetable service. For instance, newly enrolled students could opt to have their daily timetable delivered via SMS each morning for the first two weeks of term, and then access the same timetable via the web if after that they still can't remember it.

Promote and demonstrate platform diversity

Java (particularly J2EE) is still predominantly favoured as an enterprise platform by the e-learning community. While this is not a bad thing in itself, we hope that this project will raise awareness of development activity on other platforms, and (through the Java/.net interoperability elements of the project) demonstrate that common schema and not underlying implementation details are the key to successful service orientation, and that homogenous MLEs are both possible and even desirable.

Raise awareness of MLE activity within Further Education

The most high-profile MLE projects tend to be the work of Higher Education institutions, with FE perhaps not gaining full recognition for the large amount of work they put into their own MLEs, e-learning initiatives and business processes. We plan to change that!

6. Stakeholder Analysis

List key stakeholder groups and individuals that will be interested in your project outcomes, will be affected by them, or whose support/approval is essential, both within your institution and in the community, and assess their importance (low/medium/high).

Stakeholder	Interest / stake	Importance
Technical staff in institutions	Need a usable, reliable, well documented product	High
JISC	Value for money	High
CETIS	Feedback on the IMS EWS spec	High
Brockenhurst College	Innovation, collaboration, networking with other project teams	High
Commercial vendors (MIS, VLE)	Scope for interoperability of existing applications	Low

7. Risk Analysis

Risk	Probability (1-5)	Severity (1-5)	Score (P x S)	Action to Prevent/Manage Risk
Staffing	2	5	10	Project staff are all directly line-managed by the project manager and are part of an existing motivated and effective team
Organisational	1	2	2	Project progress will be monitored by the College's MLE Strategy Steering Group to ensure effective embedding into other MLE development work
Technical	1	4	8	Continued active involvement in CETIS SIGs community. Regular attendance at relevant conferences and technical briefings to ensure up-to-date knowledge of developments

8. Standards

This will be a conformant implementation of the IMS Enterprise Web Services specification. We will investigate the use of the iCalendar specification for exchange of

timetable data where appropriate. All server-side code will be targeted for Microsoft's Common Language Infrastructure (ECMA-335).

9. Technical Development

The Web Service Layer, Exemplar Implementation and server-side admin tools will be written in C# using the Visual Studio.net 2003 IDE. The services will be hosted at the College on Windows Server 2003 but will also be compatible with Windows 2000 and Windows XP Professional.

The desktop client will be written using Sun Microsystems Java 2, and should work on any platform with the appropriate Java Virtual Machine.

We will adopt a test-driven approach to the project in order to deliver a robust and reliable solution in a timely fashion.

10. Intellectual Property Rights

The toolkit will be released under an open source license. Any third-party libraries or APIs used (most likely in the Java components) will also be strictly open source. Although the .net platform is a Microsoft development, the Common Language Infrastructure itself, and the C# language, are international standards and as such we are free to develop on them.

Project Resources

11. Project Partners

N/A

12. Project Management

Robin Gadd (Project Manager).

Robin will be responsible for contractual stuff and ensuring work packages are delivered to schedule. He has managed the College's MLE strategy from its inception, from vision through to implementation to deployment and end-use staff training.

Jon Rowett (.NET / Database Developer)

Jon will undertake the majority of the application design and programming work. He has been involved in Managed Learning Environment development since 2000 and has experience working with Microsoft development technologies. Through the college's MLE projects, he has gained a thorough knowledge of IMS Enterprise and its uses, timetabling and MIS systems, and is an active member of the CETIS Enterprise SIG.

Ed Merritt (XSLT / Web Developer)

Ed has 5 year's web development experience and has a thorough knowledge of XHTML, XSLT, CSS and accessibility standards.

A.N.Other - TBC (Java Developer)

We plan to bring in a second developer to handle the Java components of the project, and to assist with other secondary activities such as testing, gathering community feedback etc.

13. Programme Support

We have identified a need for developer training in the use of the Unified Modelling Language (UML).

14. Budget

See appendix A

Detailed Project Planning

15. Workpackages & Schedule

See appendix B

16. Evaluation Plan

Indicate how you will evaluate the quality of the project outputs and the success of the project. List the factors you plan to evaluate, questions the evaluation will answer, methods you will use, and how success will be measured. Expand as appropriate on how you will conduct the evaluation.

Timing	Factor to Evaluate	Questions to Address	Method(s)	Measure of Success
31/10/2004	Appropriateness and functionality of toolkit	Ease of use Installation Adaptability Conformance to spec Reliability	Test cases User feedback Xml validation	Successful testing Positive user response
31/10/2004	Java/.net interop	Do they?	Part of the development process	Java application works with toolkit and any other EWS implementations
31/10/2004	Effectiveness of Test Driven Development	Productivity gains Quality gains	Development team feedback Success in meeting project deadlines	Meeting deadlines Positive developer feedback

17. Quality Assurance Plan

Explain the quality assurance procedures you will put in place to ensure that project outputs comply with JISC technical standards and best practice, and what will constitute evidence of compliance.

Timing	Compliance With	QA Method(s)	Evidence of Compliance
	Fitness for purpose	User feedback	Successful use of toolkit; working code; interoperability
	Best practice for processes	Test driven development	Successful running of test suites
	Adherence to specifications	Validation of SOAP messages against IMS schema Interoperability with other implementations	Successful validation by software Successful interoperability

	Adherence to standards		
	Accessibility legislation	Validation of web client tools against W3C standards	Successful validation by software

18. Dissemination Plan

Explain how the project will share outcomes and learning with stakeholders and the community. List important dissemination activities planned throughout the project, indicating purpose, target audience, timing, and key message.

Timing	Dissemination Activity	Audience	Purpose	Key Message
Oct 2004	JISC Dissemination event	All interested parties	Demonstrate working product in context	This is what it does! It works! Try it!
Ongoing	Public development via development bay	Developer community	Peer review of code and practices; sharing of expertise	This is what we're doing!
18/06/2004	CETIS Enterprise SIG Meeting	SIG members	Promote the project and encourage discussion around our approach	This is what we're doing!
Ongoing	Project website	Allcomers	Promotion	""

19. Exit/Sustainability Plan

Project Outputs	Action for Take-up & Embedding	Action for Exit
Toolkit CDROM	Distribute at JISC event	
Web service suite	Embed in college systems (Emily, MIS, VLE and portal)	

Project Outputs	Why Sustainable	Scenarios for Taking Forward	Issues to Address
The toolkit	Open source, conforms to spec, extensible, well understood technology	Public availability; encourage further development through ongoing engagement with community	

Appendixes

Appendix A. Project Budget

Item	Cost (£)
Staffing	24,500
Capital expenditure	2,000
Sub-contracting and consultancies	2,000
Travel & subsistence	1,500
	30,000

Appendix B. Workpackages

1. Debugging of IMS WSDL Files, generation of server stubs, conformance testing of .Net-generated SOAP messages.
 - It is anticipated that the WSDL files provided by IMS will (due to their pre-release nature) need a small amount on debugging before the .net WSDL & codegen tools can be used on them. Once code stubs have been generated, we will need to verify that the SOAP messages being produced by .net conform to the IMS schema.
2. Information model, Back end database and test data.
 - For simplicity, the database schema for the Exemplar Implementation will be closely based on the IMS Enterprise information model, extended to allow for timetabling. The back end database will be produced, including stored procedures for data access, then filled with convincing Student Records System data.
3. Unit tests for Application Layer objects.
 - Methods of the Application Layer objects will closely match the functionality exposed by the service layer defined by IMS. Before any functional code is written, the unit tests for these objects and methods will be written and allowed to fail.
4. Application Layer objects.
 - The objects will be continuously developed until all of the unit tests from work package 4 run correctly.
 - The objects for the Exemplar Implementation will then be loosely coupled to the service layer via a configuration file.
5. Web based admin tools.
 - ASP pages will be built for the twin purposes of providing useful admin functionality, and testing the web services themselves. These pages will not be as "rich" as the Java client.
6. Server Installation Scripts
 - Windows Installer Packages will be created so that the web services can be deployed on a server or development machine in a user-friendly fashion and with a minimum of user intervention.
7. Server Documentation
 - Machine-generated API documentation and sample source code, generated using the XML commenting features of the C# language.
 - Installation notes, all UML for the project, typical use cases
8. Unit tests for Java client.
 - JUnit test fixtures, as with Work Package 3
9. Java client.
 - Functional application which makes use of the entire range of web services
 - Testing against the C# web service implementation
 - Testing against any other IMS Enterprise Services implementations available
10. Client Documentation
 - JavaDoc documentation and UML
 - Report on interoperability between Java and .net web services
 - Installation notes

Schedule

The server (C#) and client(Java) elements of the toolkit will be developed in parallel.

WP	May		June		July		Aug		Sept		Oct	
1	*	*										
2			*									
3				*	*							
4				*	*	*	*					
5							*	*				
6									*	*		
7				*	*	*	*	*	*	*	*	*
8			*									
9				*	*	*	*	*				
10				*	*	*	*	*	*	*		

JISC Project Management Framework
22 December 2003