

JISC DEVELOPMENT PROGRAMMES

ISIS Software Quality Evaluation

Project

Project Acronym	MakingTracks	Project ID	
Project Title	Making Tracks		
Start Date	April 2005	End Date	Sept 2005
Lead Institution	University of Hull		
Project Director			
Project Manager & contact details	Robert Sherratt Email: r.sherratt@hull.ac.uk Address: Brynmor Jones Library, University of Hull, Cottingham Rd, HU6 7RX Tel: +44 1482 466834 Fax: +44 1482 465531		
Partner Institutions	Icodeon Ltd., NRICH, Newark & Sherwood		
Project Web URL	http://www.hull.ac.uk/esig/makingtracks.html		
Programme Name (and number)	ELF		
Programme Manager	Tish Roberts		

Document

Document Title	ISIS Software Quality Evaluation		
Reporting Period	<i>April 2005 – Sept 2005</i>		
Author(s) & project role	Mike Pearson. Demonstrator Construction and Evaluator		
Date	Oct 2005	Filename	SQE_MT.doc
URL			
Access	<input type="checkbox"/> Project and JISC internal		<input type="checkbox"/> General dissemination

Introduction

The ISIS project deliverables have been refactored, and improved during the course of the Making Tracks project so this evaluation includes the refactored code in its scope. We are not here explicitly evaluating the adapter components – the CMIRun web service and the SCORM 2004 player - that were constructed in order to make the demonstrator possible. We do consider however that these adapter components are worthy of further development as they are crucial to building sequenced reusable content.

The new libraries are available through the Icodeon web site at <http://www.icodeon.com>. Where it seems appropriate, we refer back to the original project deliverables posted at http://www.scormtech.com/icodeon_site_html/projects-framework-output.html.

The main difficulty I face in writing this report is that as a content developer using the ISIS toolkit, I have not needed to write code that directly uses the library API or the web service. This is good – it indicates that it is possible to understand and use sequencing at a higher level than the API. It's not so useful when you need to evaluate the API itself.

To overcome this potential problem, we have therefore split the evaluation into two parts:

Part 1 – Quality issues arising in content development

A discussion of the quality issues that arose during the development of the demonstrator. Some of these are related more to the usability of the Simple Sequencing standard itself . We discuss these in the usability report. Here, we shall be more concerned with documentation and installation.

Part 2 – Quality of the implementation

An assessment of the quality of the refactored ISIS implementation of the standard. For this the final arbiter must be conformance tests cast in the form of unit tests.

Part 1 – Quality issues arising in content development

We primarily discuss documentation and installation issues here.

Documentation

There is plenty of scope for confusion to the new user approaching Simple Sequencing. One of the first problems that arises is where to start. ISIS published its project deliverables at http://www.scormtech.com/icodeon_site_html/projects-framework-output.html but much of this information is now superseded by more up to date material on the Icodeon site at <http://www.icodeon.com>, so this should perhaps be advertised more prominently as the first port of call now.

Some links are still more easily found on the ISIS page however. In particular, the IMS Simple Sequencing and QTI Integration Guide http://www.scormtech.com/icodeon_site_html/pdf/ss_qti_integration_v1_0.pdf and links to support from Carnegie Mellon University (Learning System Architecture Lab) including the very important link to their SS-capable branch of the [RELOAD](#) editor.

The Icodeon site has certainly thought through the problem of introducing new users very effectively, so I'll summarise the material that is available there.

Essentially we have:

- A couple of executive introductions – one on SCORM 2004, and one on Simple Sequencing.
- A demonstration area which now includes the Making Tracks demonstrator
- A download area, where evaluation packs can be requested. These evaluation packs contain the core Sequencing Library, the Web Service, the CMI service, and the SCORM 2004 player.
- A 2 minute integration guide, which gives an example of using the Sequencing Library.
- API documentation on the sequencing engine.
- A FAQ
- A support forum

I found these materials very clearly presented. The FAQ and the support forum are partly constructed from work done in this evaluation.

FAQ:

As part of this evaluation we prepared a questionnaire for Icodeon. An edited version of this, with responses, is published in the form of a FAQ at <http://www.icodeon.com/faq.do>.

Support Forum:

The support forum was set up while Making Tracks was underway and was used to assist with installation issues. The ISIS toolkit is supported there at <http://forums.icodeon.com>. If you register your email address with the forum, you are automatically notified of responses to your posts by email.

Installation

The libraries and web services are shipped in zips and there is a war for the SCORM player: icodeon-examples-pack-v1p0p5.zip

- the sequencing library, and an installation guide.

icodeon-web-services-pack-v1p0p2.zip

- the SSRun web service and its installation guide

icodeon-cmi-service-pack-v1p0p0.zip

- the CMIRun web service and its installation guide



Finally, there is a web application to drop into TomCat's webapp directory:

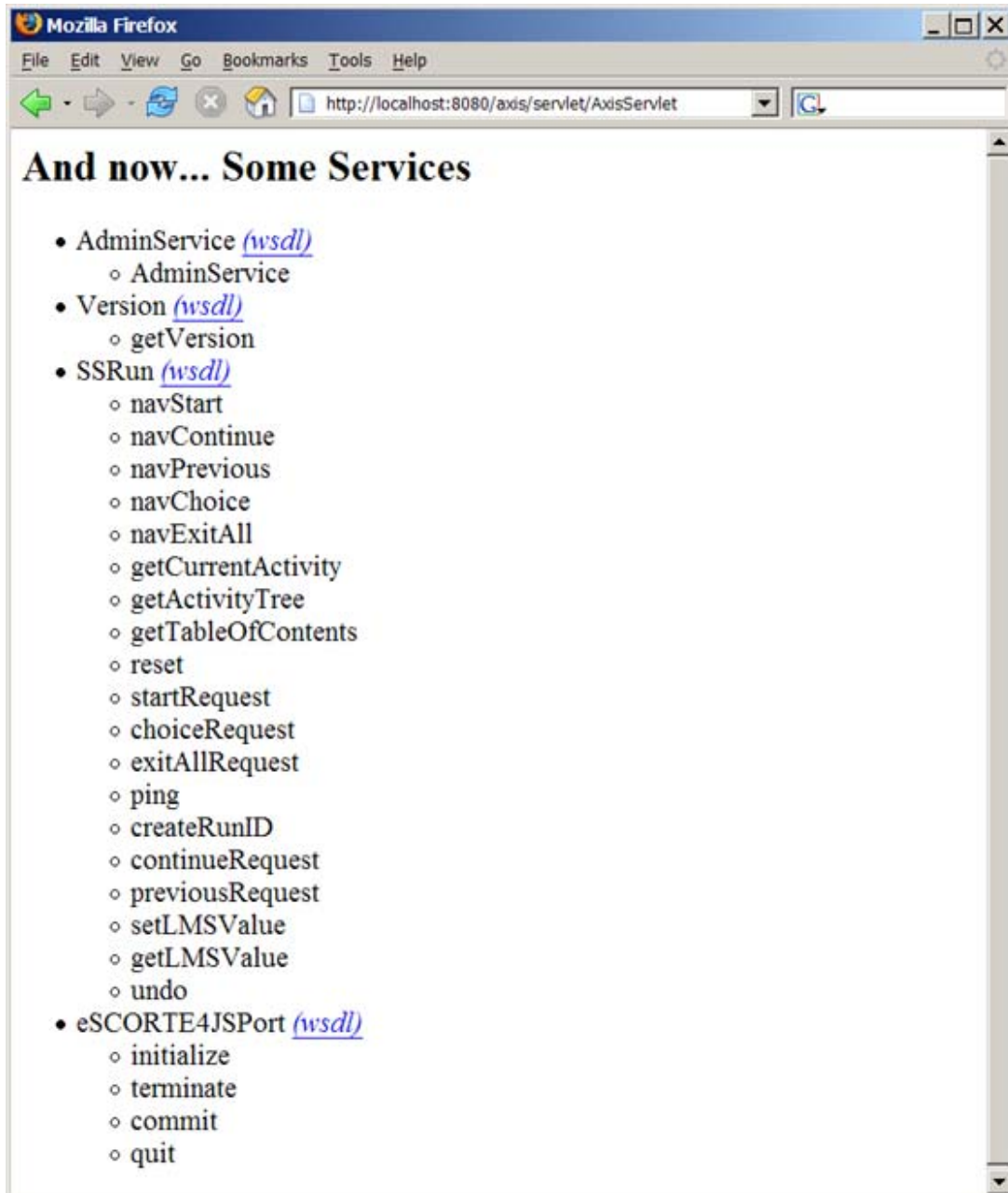
- icodeon.war

Installation of these products is reasonably easy, though not yet fully automated with an installation application. This is however good enough for the systems administrators and programmers who are likely to be tasked with the installation. There were some problems with our initial attempts that were resolved on the forum. If you are interested in these then refer to the Sequencing Service thread of the support forum. See <http://forums.icodeon.com/viewforum.php?f=5>

The main difficulties encountered are in locating the place that TomCat looks for the license.properties file – which varies between different TomCat installations, and ensuring that the paths in the various properties files are correctly edited. It's also important that the correct version of Apache Axis is used.

Exception texts led fairly directly to the cause of the trouble when we hit these problems.

When it's all done SSRun and CMIRun (in the screen shot below this is listed as escorte4JSPort) are available and the player is running.



The Sample Applications

The library contains a 'Hello World' application which is a very useful early test and a handy template from which a working application may be grown.

```
/*
 * Copyright 2005 All Rights Reserved
 *
 * Created on 11-Mar-2005
 *
 * Icodeon Ltd
 * Studio 471
 * 48 Regent Street
 * Cambridge
 * CB2 1FD
 * England
 *
 * Registered in England and Wales No: 5068195
 *
 * @author Icodeon Ltd
 *
```

```
*/
package com.icodeon.ss.test.icodeon;

// Import Java packages
import java.io.*;

// Import the Icodeon Sequencing API
import com.icodeon.ss.state.api.*;
import com.icodeon.ss.engine.api.*;

// Import the utility classes
import com.icodeon.ss.util.*;

public class HelloWorld {

    /*
     * Simple example that shows how to create an activity tree, pass the
     * tree to a sequencing engine and issue some navigation commands.
     *
     * The sequencing engine identifies a current activity that may then be
     * accessed from the activity tree.
     */
    public static void main(String[] args) {

        try{
            // Get the IMS Manifest as a File
            String path = "<path to manifest file>";
            File manifest = new File(path);

            // Create an activity tree from the manifest
            TreeBuilder treeBuilder = new TreeBuilder();
            ActivityTree activityTree = treeBuilder.build(manifest, null, true);

            // Create a sequencing engine
            EngineBuilder engineBuilder = new EngineBuilder();
            SequencingEngine sequencingEngine = engineBuilder.build();

            // Pass the activity tree to the sequencing engine
            sequencingEngine.setActivityTree(activityTree);

            // Issue a START navigation request
            sequencingEngine.setRequestType(RequestType.START, null);
            Activity currentActivity = activityTree.getCurrentActivity();
            //System.out.println(currentActivity.getTitle());

            // Issue a CONTINUE navigation request
            sequencingEngine.setRequestType(RequestType.CONTINUE, null);
            currentActivity = activityTree.getCurrentActivity();
            //System.out.println(currentActivity.getTitle());

            // Issue an EXIT_ALL navigation request
            sequencingEngine.setRequestType(RequestType.EXIT_ALL, null);

        }
        catch(Exception e){
            e.printStackTrace();
        }
    }
}
```

We had less success with the example in the Web Services pack – and to date I have not managed to get this running. I'm not entirely sure if this is because of a problem with the example or with me since I have no reason to make direct use of the web service. Axis and TomCat installation and maintenance is perhaps one of the weaker areas when considering distribution of these materials.

Part 2 – Quality of the implementation

Unit Tests

Development and maintenance tests for the ISIS toolkit are listed at http://www.scormtech.com/icodeon_site_html/html/test/index.html.

The unit test methodology turned out to be a valuable way of verifying not only the behaviour of the library, but also verifying IMS manifests. The kind of content we developed in the demonstrator demanded that these manifests were also tested thoroughly.

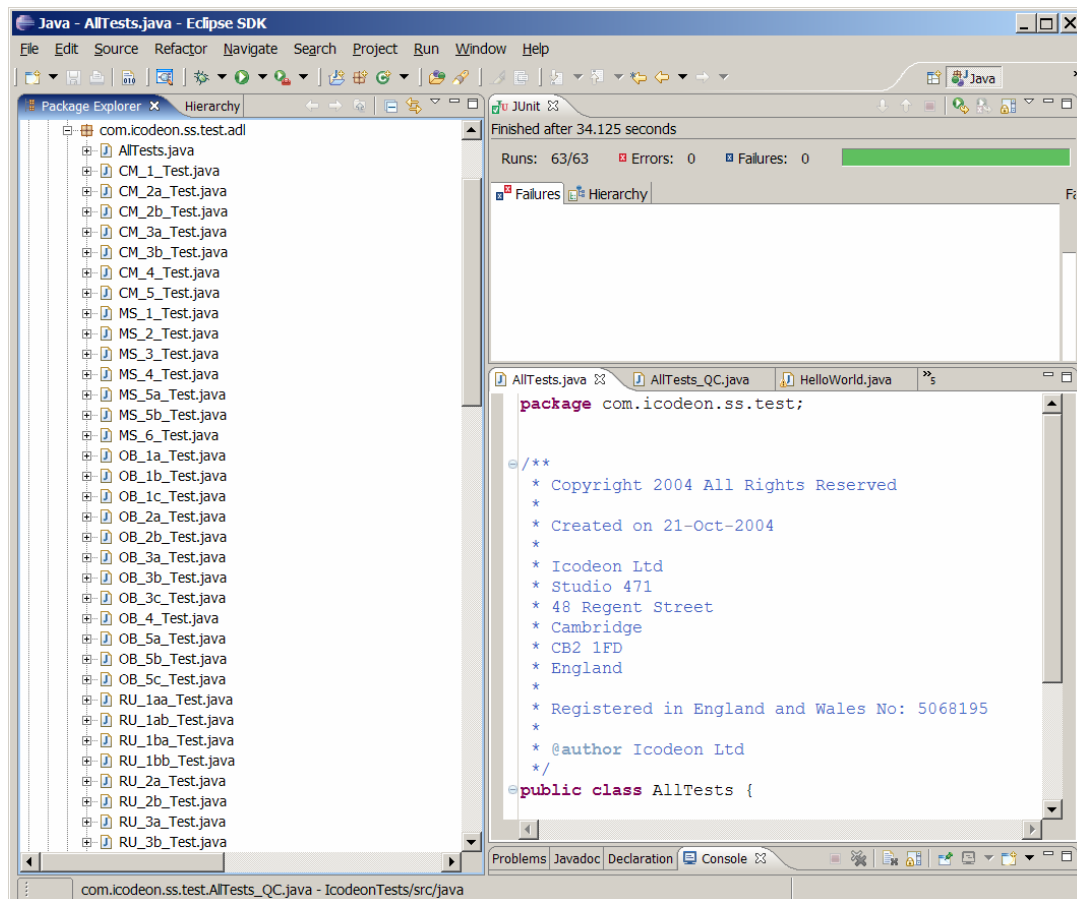
The Flash demonstrator used the SCORM 2004 API to make CONTINUE and PREVIOUS calls on the sequencing service. Since these were made by the SCO rather than by the player, it was useful to be able to test that possible routes were indeed available. It was surprisingly easy to write incorrect manifests that generated sequencing exceptions somewhere, so this formal testing proved very useful.

The Icodeon SCORM Player circumvents this problem when it is allowed to handle the navigation buttons itself, by making advance calls on the sequencer to determine whether or not to enable its navigation buttons. The sequencing state is then rolled back before the screen is presented to the user.

Conformance tests

The conformance tests used by the ISIS project were developed from Appendix A of SCORM 2004 Conformance Requirements (currently at Version 1.3) as published at <http://www.adlnet.org>

The test cases documented there were converted into the JUnit tests and run under Eclipse. All applicable tests were passed and we present a screenshot below of our test run.



Icodeon actively uses these tests to maintain the product in conformance with the SCORM 2004. There is however currently a manual element involved in maintaining this test suite itself as the SCORM 2004 self-test suite cannot be used directly. The SCORM 2004 self-test suite assumes the presence of a complete Learning Management System together with all course upload, deployment and user management facilities.

While building the Making Tracks demonstrator, we discovered that the SCORM conformance suite does not adequately cover functionality defined by the standard where the SCO reads in CMI or objective data. The sequencing tests are all concerned with how sequencing behaves when SCOs *write* out their status. They do not check that data that should be available to the SCO through GetValue calls is indeed accessible. It was also discovered that the conformance tests failed to check retry handling on cluster nodes.

Experience with advanced use of the sequencing in our games-based learning suggests that a more complete conformance test suite for SCORM 2004 sequencing is desirable. We ran off the edges quite a few times, hitting areas where the behaviour in ISIS and the behaviour in our alternative KnowledgeWorks LMS testbed initially differed markedly. Updates were needed for both systems during the course of the project, and some recommendations were passed back to the ADL.

Transfer of state between SCOs and sequencing objectives is a complex area and needs to be supported by a new tests.

The JUnit driver for the conformance test suite identifies a few places where the conformance tests cannot yet be applied. Of these, Suspend and Resume behaviour and inter-

tree global objective persistence are only possible within the context of a more complete CMI implementation and a SCORM LMS with persistent user state. I imagine sub-manifests and sequencing collections are candidates for future updates.

```
//suite.addTestSuite(CM_3b_Test.class); // Known issue in recent ADL addendum 2.3
//suite.addTestSuite(CM_5_Test.class); // Suspend and Resume not yet implemented
//suite.addTestSuite(SX_3_Test.class); // Sub Manifests Not Supported
//suite.addTestSuite(OB_3b_Test.class); // Inter tree global objective persistence not
supported
//suite.addTestSuite(OB_3c_Test.class); // Inter tree global objective persistence not
supported
//suite.addTestSuite(T_1a_Test.class); // Uses sequencing collections, not yet supported
//suite.addTestSuite(T_1b_Test.class); // Uses sequencing collections, not yet supported
```

Performance

It has not proved easy to characterize the performance of the code. Some simple timings may be helpful in predicting relative performance. These were taken from a 3Ghz Pentium 4 Windows system running Java 1.5.

Here is a table of timings of the ADL test CM_1. The test repeats the creation and tear down of the activity tree and includes 6 intermediate navigation requests.

Repeats	Time
1	2.75 s
10	3.67 s
100	12.45 s
1000	108 s

We have not seen any evidence of memory leaks in the library during development and indeed extended runs of tests such as the above show a stable memory footprint.

Repeating the CM_1 test with its payload of 6 navigation requests removed, gave the following times:

Repeats	Time
1	2.5 s
10	2.82 s
100	5.1 s
1000	24.7 s

Repeating RU_9 gives:

Repeats	Time
1	2.6 s
10	2.98 s
100	5.98 s
1000	31.8 s

and without its payload:

Repeats	Time
1	2.4 s
10	2.6 s
100	3.25 s
1000	8.3 s

CM_1 was slower than RU_9, and an inspection of the test code revealed that this was probably because CM_1 contains calls to persist the activity tree after every navigation request.

Conclusion

Overall the ISIS products are excellent. It's a good clean implementation of the IMS Simple Sequencing standard. Even so, the software benefited considerably from being refactored and tested in another context during this project. I believe that it is absolutely essential that the highly technical projects such as this are tested as soon as possible in a real educational context. Doing so has filled the product out considerably and given us a much greater understanding of where it is appropriate and where inappropriate. Conformance tests, while necessary, are not sufficient to ensure that the software is behaving in a useful way.