D1.2.1: ITS 2.0 ENABLEMENT IN APACHE JACKRABBIT

Des Oates, Christine Duran, Adobe

Distribution: Public

MultilingualWeb-LT (LT-Web)
Language Technology in the Web

FP7-ICT-2011-7

Project no: 287815

Document Information

Deliverable title:	ITS 2.0 Enablement in Apache Jackrabbit
Contractual date of delivery:	December 2013
Actual date of delivery:	November 2013
Author(s):	Des Oates, Christine Duran

Revision History

Revision	Date	Author	Organization	Description
1	28/10/2013	Des Oates	Adobe	Initial Draft
2				

CONTENTS

1. INTRODUCTION

1.1. Terminology

Term	Definition
Apache Jackrabbit	Java Content Repository (JCR 2.0) Compliant Open Source CMS system
Sling	REST framework used for facilitate URL-based access to JCR content AND functionality
REST	Stands for Representational State Transfer. A protocol for passing data and exercising remote functionality on remote systems over HTTP
JCR	Java Content Repository. A specification that defines interfaces for CMS systems written in Java. Jackrabbit complies with version 2.0 of the specification
Okapi	Java-based framework used for detection, extraction and retrofitting of localisable content published in many document formats
Source Content	Text suitable for translation into other langauges. Often, but not necessarily, the source language is English
Target Content	Text which has been translated from source content

2. BACKGROUND

The purpose of this implementation was to enable users of the Jackrabbit Content Management System – an open source CMS that complies with the Java content Repository 2.0 (JCR 2.0) specification – to utilise the power of ITS 2.0 when managing their localisable content. Adobe is the primary contributor Jackrabbit project and uses the technology as a foundation for several products. Enabling Jackrabbit to create and interpret ITS 2.0-enabled content would be beneficial to both Jackrabbit users, and customers of Adobe, who are interested in publishing content in more than one language on Jackrabbit, or Adobe applications built on the platform.

Users of large scale multilingual content systems often have very specific requirements on how their source content is organized, identified and arranged withiin the content repository. Similarly when there is a need to translate that content, some complex requirements can surface to ensure that translatable (and non-translatable) content can be identified and prepared for translation. As the number of required target languages increase, so does the necesity for adherance to these organizational requirements.

Multilingual content publishers often manage their content by creating a bespoke metadata set and applying it to the content. This helps them map large amounts of target content with its corresponding source content within the repository. It may also identify sections of source content as 'translatable' or not. Useful metadata helps streamline the overall translation process for content.

However because this metadata is customised, it has limited use outside of the context of the context repository. It cannot be used by other systems and agencies, even although translation workflows often require that translatable content be exported out of the CMS to external systems, services and service providers as part of the translation process.

ITS 2.0 helps solve this problem. ITS 2.0 is a standard metadata tag set that can be applied to translatable, and translated content. Because it is standard, disparate systems, technologies and service providers can interpret this

metadata as translatable content moves through each system engaging in a translation workflow. This makes it an attractive to enterprise companies that manage large amounts of translatable content in their CMS'.

In this document it will be shown how we have extended the capability of the Jackrabbit CMS for a subset of the ITS 2.0 metadata specification. This will allow users to tag content with ITS 2.0 metadata and export that content out into a ITS 2.0 enabled translation workflow.

This implementation, although fully functional should not be considered as a fully complete multilingual content management solution, but as a solid standards based foundation on which to build such a solution.

3. Implementation Overview

As mentioned above ITS 2.0 does not mandate that every implementation support every ITS 2.0 data category. This implementation supports 4 data categories:

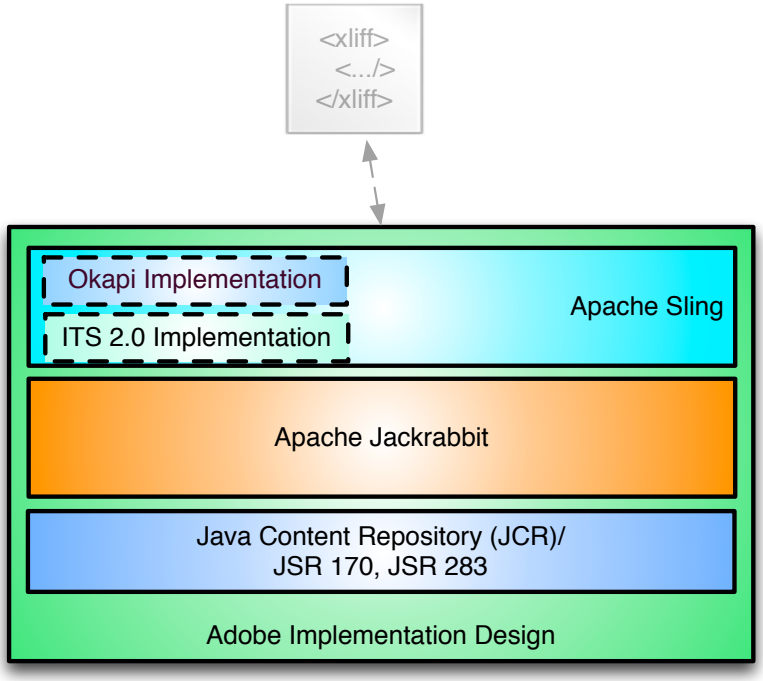
- Translate
- Localization Note
- ID Value
- Target Pointer

These are identified as important areas in the management of large scale multilingual content in translation workflows.

Adobe built its implementation by extending Sling functionality exposed by Jackrabbit. Sling is a REST framework designed to work with JCR repositories. Jackrabbit is easily extensible as it uses a component-centric, OSGi based design at it's core. This enables 3rd parties to easily extend the system by adding their own OSGi Component. Adobe created an OSGi bundle (component) which can be installed into Jackrabbit easily. The component is responsible for processing ITS 2.0 rules, managing ITS 2.0 metadata to the content within the repository, and to prepare content for import and export in and out of the repository.

Okapi is also used in the implementation. Okapi is a mature open source technology used to manage,convert and process translatable content. It supports a wide variety of file formats and crucually, now includes support for ITS 2.0. We decided to use Okapi to do the 'heavy lifting' when import/exporting from the native JCR Content to XLIFF, HTML and other formats.

A schematic overview of the Implementation Design is shown below:



The primary users of the Adobe implementation are content authors and Localisation professionals such as Localisation Project Managers. However the project scope was constrained, and therefore only foundation features were developed. Higher order aspects such as User Interfaces, Workflow Management, and Configuration features were out of scope based on time and resource constraints. Jackrabbit Users wishing to use the ITS 2.0 Enablement functionality will still be required to create out suitable User Experience features on top of this implementation.

That said, the primary features of the implementation are:

- The Management of ITS 2.0 Rules. The ITS rules are the fundamental to identifying content upon which ITS metadata is applicable. This implementation manages global and local rules for all supported ITS 2.0 data categories
- Export Content for Translation: Once rules have been applied, the content is marked up into an intermediate form, ready for processing by Okapi. Okapi then processes the marked up content and transforms it into the desired output format, which can be either XLIFF or HTML5. This content is then suitable for either publication, or (more likely) to initiate a translation process to start it
- Import Translated Content for Publication: When content has been sent for translation, and the Target content is now translated, it can now be imported back into Jackrabbit to be saved in the repository, and published as translated content

How users can apply these features is described in more detail in the next section.

4. Use Cases

4.1. Managing Global ITS2 Rules

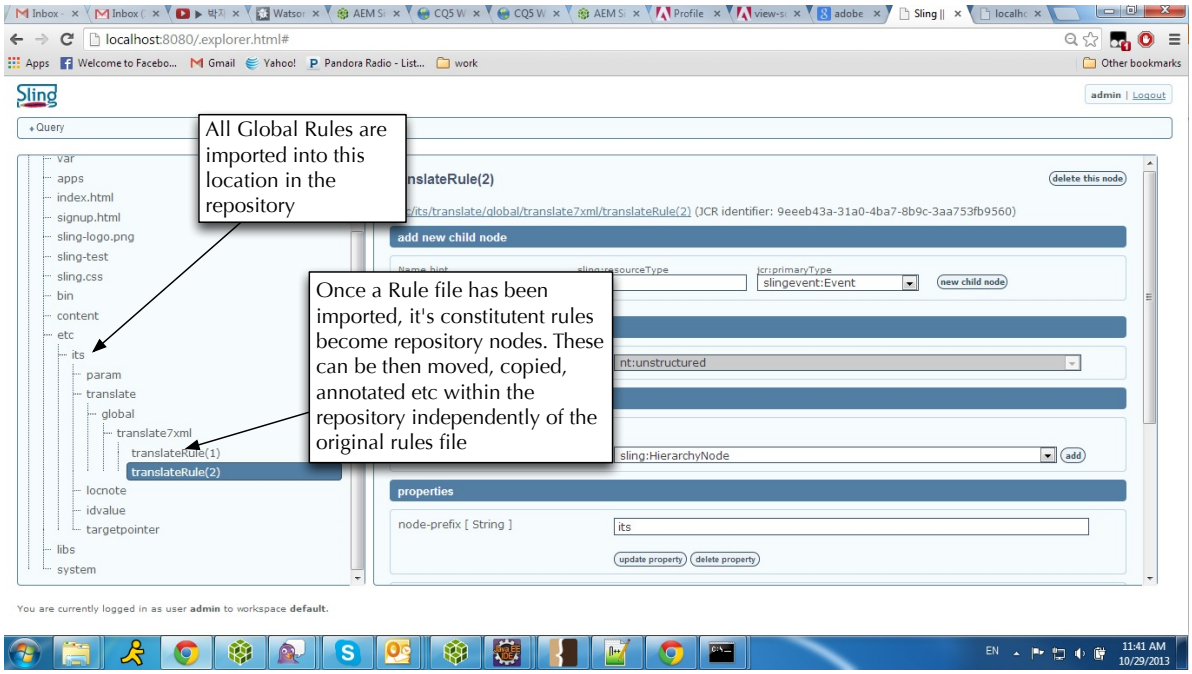
In order for any ITS 2.0 enabled system to work, it must be able to understand and process ITS rules: both Global and Local varieties. We opted to isolate a dedicated area of the JCR repository to manage ITS rules. The ITS processors expect these rules to be in a specific location in the repository which the read from, and in certain circumstances, will update rules at this location.

Users can import ITS rules file into the repository easily via Jackrabbit's content import facility. Once imported the rules contained in the imported file are converted to JCR Nodes. This allows each rule to be manipulatd within the

repository independent of the file it was originally imported from. The ITS parsers use these rules when apply ITS metadata to content. Embedded Global Rules (I.e. Rules embedded within individual documents) will also be honoured by the ITS parsers, along with Local Rules.

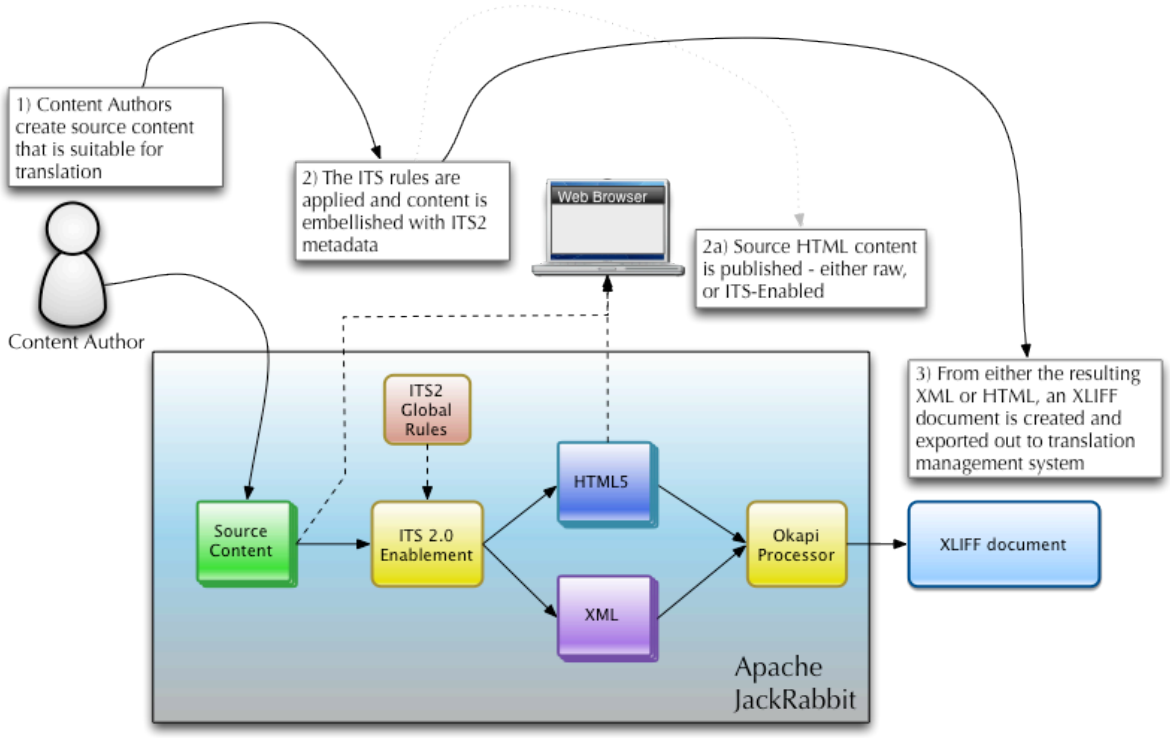
In certain circumstances, if rules have been added to a content file as part of a translation workflow, then the rule will be detected and imported into the global rule set.

New rules can be added to the repository manually simply by adding a Rule Node at the appropriate location. Ideally this would be done in a dedicated Rules Editor application, but such an application was out of scope for this project.



4.2. Exporting ITS 2 Enabled Source Content for Translation

Once all the ITS 2.0 rules are in place, it is then possible to export translatable content out for translation. Under the hood, the process for achieving looks something like this:

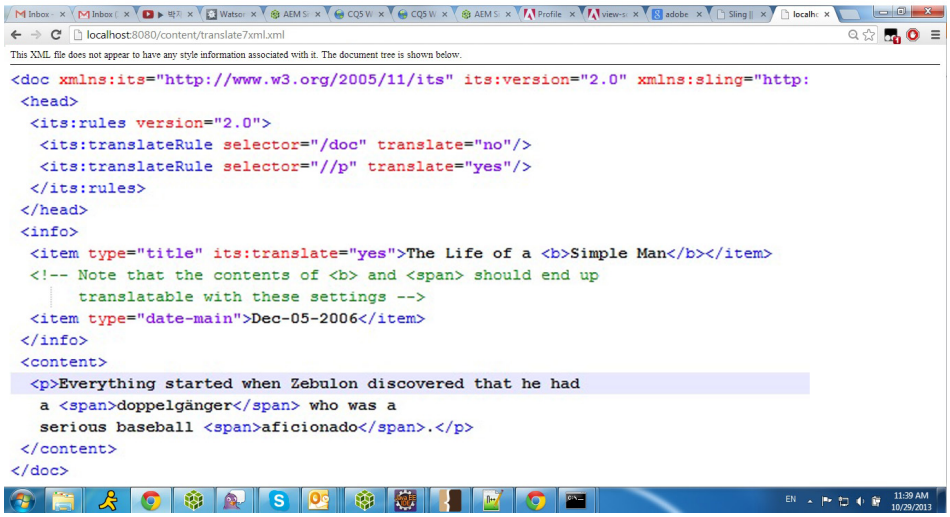


ITS 2.0 metadata is injected into the translatable content. The parsers examine the content, and then compare with the applicable ITS rules to determine if, and which metadata should be applied. This creates an intermediate content form that contains ITS and JCR metadata. Both this metadata is required to roundtrip the content through a full translation workflow.

Once this intermediate stage is complete, it is then passed to Okapi, which processes that content, and transforms it into the required export format, which is usually XLIFF. It creates an XLIFF file on the file system which can then be used to initiate a translation workflow.

The process looks complex, but for the user, it's very simple.

For example, if we want to access the original content file in the repository, we can do so with a simple url in a browser. E.g. in this case it is *"<hostname>/content/translate7xml.xml"* and looks similar to what is shown below:



To invoke the ITS parser and view the ITS enabled markup, all that is required is to add an 'its' selector to the original URL E.g. *"<hostname>/content/translate7xml.its.xml"*

This intermediate markup looks like this:

Finally to export the XLIFF, we add the XLIFF selector to the URL. E.g.

"<hostname>/content/translate7xml.its.xliff.xml"

This creates the XLIFF file and exports it out to the local file system. The XLIFF will contain translatable content as identified by the its translate metadata. If there were any localisation notes embedded in the text, then these will be included. Each translatable segment in the file will be identifiable using the its-idvalue attribute.

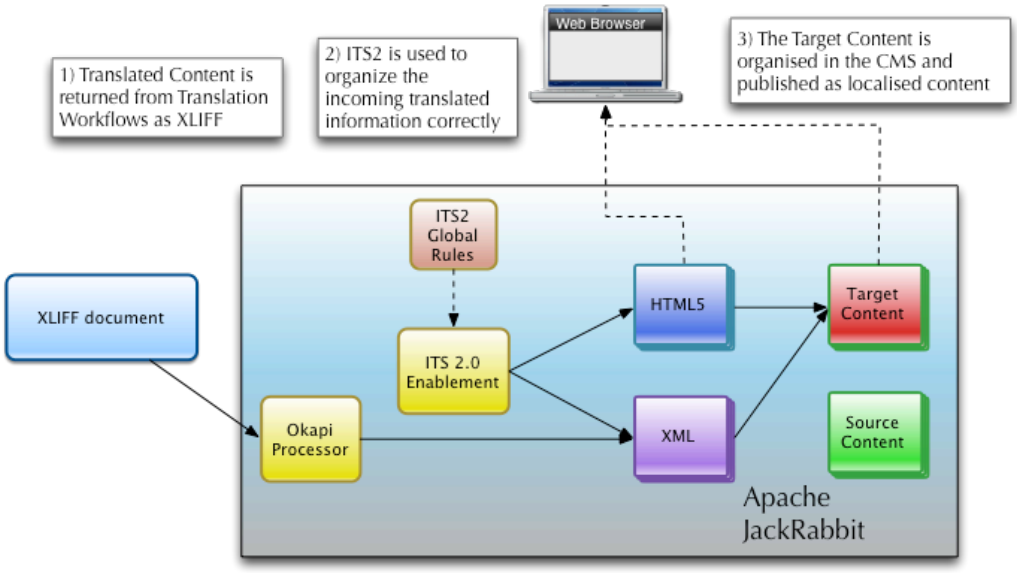
Similarly *"<hostname>/content/translate7xml.its.html"* produces ITS 2.0 enabled HTML5 markup

The process adds applicable global rules definition to the export file. This is to help any downstream systems as it processes the content.

4.3. Importing ITS 2 Enabled Target Content from a Translation Workflow

Once conten has been translated, the process is reversed. Translatable content is identified, imported, and saved to the correct location, (that maps to it's corresponding source location in the repository.) Jackrabbit uses its ID Value and Target Pointer information to determine where to save the translated information.

It will also examine the embedded rules found in the import to determine if there are any new rules added by other systems while the content was out getting translated. If so, and supported, these may be added to the global rules definitions within the repository.



Once the translated content is saved, it can then be reviewed, modified and finally published.

5. AVAILABILITY

The implementation will be available as an open source extension to Jackrabbit. At the time of writing it is going through final phases of quality and legal audit, and will be posted publicly as a project on Github as soon as possible.

Depending on how it is received publicly, it may be proposed for inclusion in the Apache Sling project, .

6. References

Reference	Link
Apache Jackrabbit	*jackrabbit.apache.org*
Apache Sling	*sling.apache.org*
Okapi	*okapi.opentag.com*
XLIFF	docs.oasis-open.org/xliff/xliff-core/xliff-core.html
ITS 2.0	www.w3.org/TR/2013/REC-its20-20131029/