Test Suite How-To Guide

There are three parts to the test suite:

1. Conformance testing for HTML & XML
2. Conformance testing for NIF
3. Input file validation

1. Conformance testing for HTML & XML

This conformance testing is used to determine whether ITS 2.0 implementations meet the ITS 2.0 specification standard. The test suite has a set of test documents for both XML and HTML which are then used to validate the different ITS 2.0 constructs available for each data category. There are a total of 225 test files spread across all 19 data categories for both HTML and XML.

1.1 How to use the test suite?

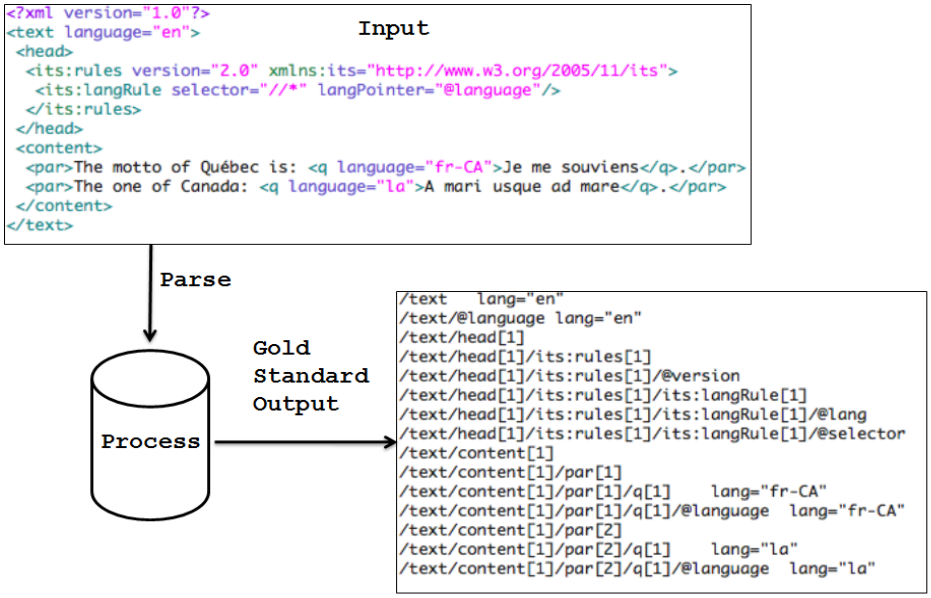

Figure 1: Testsuite processing Input to Output

The validation of the input files is done through processing the input file so that it outputs a file to match a corresponding gold standard output file. The gold standard was develop to be easy to understand and implement for conformance testers. The gold standard output format was developed by the ITS 2.0 working group. Figure 1 above describes the test suite files process.

1.2 Gold Standard output specifications details:

The gold standard output files have the following characteristics:

1. Every element and attribute path from the XML and HTML file are listed (apart from the content within script elements in HTML)
2. The output has to be tab-delimited format:

   /html/body[1]/p[1]/span[2] annotatorsRef="text-analysis|http://enrycher.ijs.si" taConfidence="0.5" taIdent="301467919" taSource="Wordnet3.0"

3. The attributes within elements have to be in alphabetical order:

   /doc/header[1]/its:rules[1]/its:locQualityIssueRule[2]

   /doc/header[1]/its:rules[1]/its:locQualityIssueRule[2]/@locQualityIssueComment

/doc/header[1]/its:rules[1]/its:locQualityIssueRule[2]/@locQualityIssueProfileRef

/doc/header[1]/its:rules[1]/its:locQualityIssueRule[2]/@locQualityIssueSeverity

/doc/header[1]/its:rules[1]/its:locQualityIssueRule[2]/@locQualityIssueType

/doc/header[1]/its:rules[1]/its:locQualityIssueRule[2]/@locQualityIssuesRef

/doc/header[1]/its:rules[1]/its:locQualityIssueRule[2]/@selector

4. The rules output also have to be in alphabetical orders:

   /html/body[1]/p[1]/span[2] annotatorsRef="text-analysis|http://enrycher.ijs.si" taConfidence="0.5" taIdent="301467919" taSource="Wordnet3.0"

5. The rules output can't have Pointer in it but it has to have its equivalent (unless it is in the target pointer data category then it must display targetpointer="...." and the pointer details):

   Incorrect:

   /doc/para[1]/issue[2] locQualityIssueTypePointer="misspelling" locQualityIssuesRefPointer="#l1234"

   Correct:

   /doc/para[1]/issue[2] locQualityIssueType="misspelling" locQualityIssuesRef="#l1235"

6. The rules output for local html rules have to be like their global counterparts:

   Incorrect:

   /doc/p[1] its-loc-quality-issue-type="misspelling" locQualityIssuesRef="#l1235"

   Correct:

   /doc/p[1] locQualityIssueType="misspelling" locQualityIssuesRef="#l1235"

1.3 How gold standard output is compared to implementors output?

The output of the implementors is located in the folder ITS-2.0-Testsuite/its2.0/outputimplementors which can be tested against the gold standard output files located in the ITS-2.0-Testsuite/its2.0/expected folder. This can be done simply through doing a diff of the implementors output files and the corresponding gold standard output files. To automate this process for implementors a test suite dashboard was created which did the following tasks :

1. Help to track the process of implementers in relation to the tests in which they were committed to complete (indicated on the dashboard via N/A = the implementer did not commit to run the test).
2. Help to track if the implementors output file matches the corresponding gold standard output file (indicated on the dashboard via fnf: the output file from the implementer has not been found OK = the output file is identical to the reference output file).
3. Help to track if the output file that is in the output folder for a particular tests doesn't match the corresponding gold standard output file (indicated on the dashboard via error = an error occurred, e.g. the output file is not available or it is not identical to the reference output file. Move the mouse over error to see details).
4. Help to track if the implementor has not committed an output file for a corresponding test (indicated on the dashboard via fnf: the output file from the implementer has not been found).
5. The dashboard can also track how many tests a particular implementor has left to run.

1.4 Validating Output Test Files

To validate the implementors output files the testsuite dashboard has to be compiled so that a diff across all files in the ITS-2.0-Testsuite/its2.0/outputimplementors folder can be done against the corresponding files in the gold standard output ITS-2.0-Testsuite/ITS2.0/expected folder. The test suite dashboard can be compiled by doing the following:

1. Download saxon.jar from here: https://dl.dropbox.com/u/65779171/saxon.jar

2. Then use this command (Linux/Mac/Windows): java -jar /path/of/file/saxon.jar testsuiteMaster.xml testsuiteDashboard.xsl -o:testSuiteDashboard.html
3. Upload newly compiled testsuiteDashboard.html to the git hub
4. Check the state of your files in the related data categories on this web page: http://htmlpreview.github.com/?https://raw.github.com/finnle/ITS-2.0-Testsuite/master/its2.0/testSuiteDashboard.html

The files for the test suite dashboard are as follows:

1. testsuiteMaster.xml - has a list of the implementer's and the tests that they are involved in and aids in the creation of the testSuiteDashboard.html
2. testsuiteDashboard.xsl - does the diffs between the implementer's output and the gold standard output.
3. testSuiteDashboard.xml - gives information on the errors between the diff's for the implementors output and the gold standard output
4. testSuiteDashboard.html - the html test suite dashboard which is located live here

2. Conformance testing for NIF

This part of the conformance testing is used to determine whether ITS 2.0 implementations of NIF 2.0 meet the ITS 2.0 specification standard for NIF usage. The NIF test suite has a set of test documents for HTML for the Localisation Quality Issue data category which are then used to validate the different aspects of ITS 2.0 and NIF against various constructs available to the Localisation Quality Issue data category. There are a total of 11 files only for HTML in the Localisation Quality Issue data category. The input files are located in the ITS-2.0-Testsuite/its2.0/nif-conversion/input folder.

The validation of the input files is done through processing the input file so that it outputs a file to match a gold standard output file. The gold standard for NIF is RDF output using the ITS 2.0 ontology. The gold standard output for NIF can be reached by following the NIF conversion algorithm discussed in the ITS 2.0 specification located here.

2.1 How gold standard NIF output is compared to implementors output?

The NIF output files are compared via the use of SPARQL queries done over the implementers RDF/NIF output files (.ttl). If the SPARQL queries are successful then the NIF output files are correct and meet the gold standard. The implementer's NIF test output files are located in the ITS-2.0-Testsuite/its2.0/nif-conversion/expected folder.

2.2 Validating NIF output files

Prerequisites: Java and Unix Shell

1. create a temporary folder for output files (hence called $datafolder)
2. read ITS files from "its2.0/nif-conversion/input/" one by one, convert to NIF and write output files in turtle to $datafolder
3. go to directory cd its2.0/nif-conversion/sparqltest
4. run : ./executeTests.sh ../relative/pathTo/$datafolder

Explanations of output:

1. If no message appears between "Running: test1.sparql" and "Done: test1.sparql" the test was successful.
2. Otherwise the output filename and additional debug output is shown

3. Input file Validation

This part of the test suite is vital and it is the testing of the HTML and XML input files to see if they are valid in accordance to the ITS 2.0 schema specification for XML and HTML. More information about this validator can be found here and also here.

3.1 Validating Input Test Files

The following sections detail how to validate the test suite input files both HTML and XML.

3.1.1 Validating XML test files

1. Download and install Ant from http://ant.apache.org/
2. Run 'ant validate-xml' command in its2.0 directory

3.1.2 Validating HTML test files

1. Download and install Ant from http://ant.apache.org/
2. Download html5-its-tools from https://github.com/kosek/html5-its-tools
3. Modify its2.0/build.properties to point to your local copy of html5-its-tools
4. Run 'ant validate-html' command in its2.0 directory

3.1.3 Validating all test files

1. Make sure that XML and HTML validation described above works for you
2. Run 'ant' command in its2.0 directory
3. Please note that HTML schema doesn't supports RDFa so RDFa attributes are reported as errors
4. Please note that currently Schematron validation is not performed so some errors are not detected

4. Xliff Sample

In addition to conformance testing and validation of XML and HTML files it provides XLIFF samples. These are not used in conformance testing, but demonstrate the representation of ITS 2.0. metadata in XLIFF. These XLIFF files are located in ITS-2.0-Testsuite/its2.0/xliffsamples folder of the github.