Testing XSLT

Tony Graham
Menteith Consulting Ltd
13 Kelly's Bay Beach
Skerries, Co Dublin
Ireland
info@MenteithConsulting.com
http://www.menteithconsulting.com

Version 1.3 – XML Prague 2009 – 21 March 2009

Menteith
CONSULTING

Testing XSLT

Testing XSLT

1

- Introduction
- Testing Source
- Testing Result
- Testing Source–Result
- Testing the Stylesheet

Introductions

Who am I?

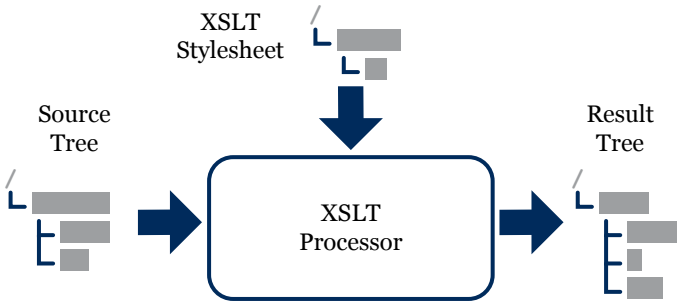
2

- Tony Graham of Menteith Consulting
- XML and XSL/XSLT consultant
- Based in Dublin, Ireland
- Clients in Ireland, USA, France, UK
- Doing XSLT since 1998
- Working with both XSLT 1.0 and XSLT 2.0

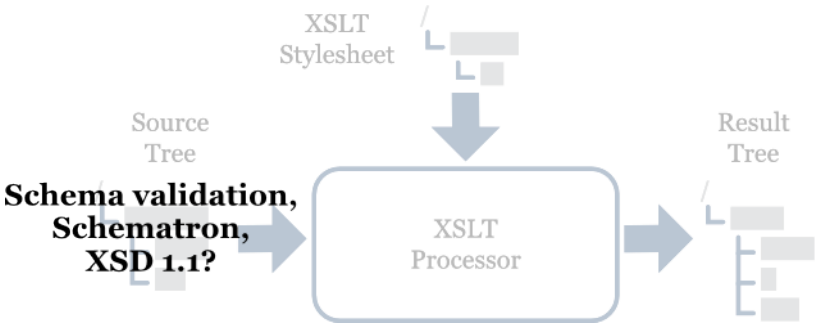
XSLT

3

Transforms source tree into result tree by associating patterns with templates

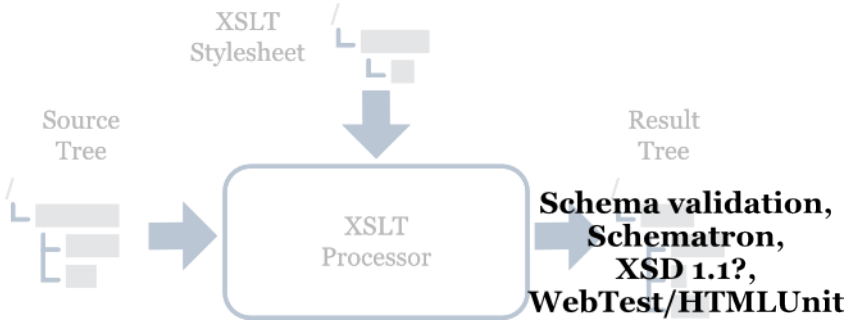


4 Testing Source



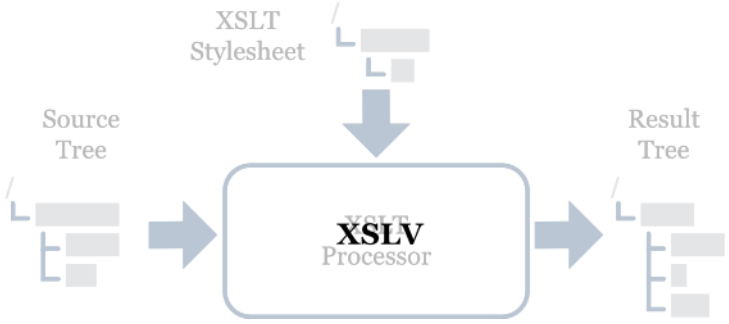
- Good if you can do it
- May be well-formed or very loose schema
- May be constraints that aren't easily checked

5 Testing Result



- Good if you can do it
- May be well-formed or very loose schema
- May be constraints that aren't easily checked, e.g., XSL FO
- If HTML, use HTML testing tools, e.g., WebTest
- Harder to verify text output

6 Testing Source–Result



XSLV Static Validation Tool

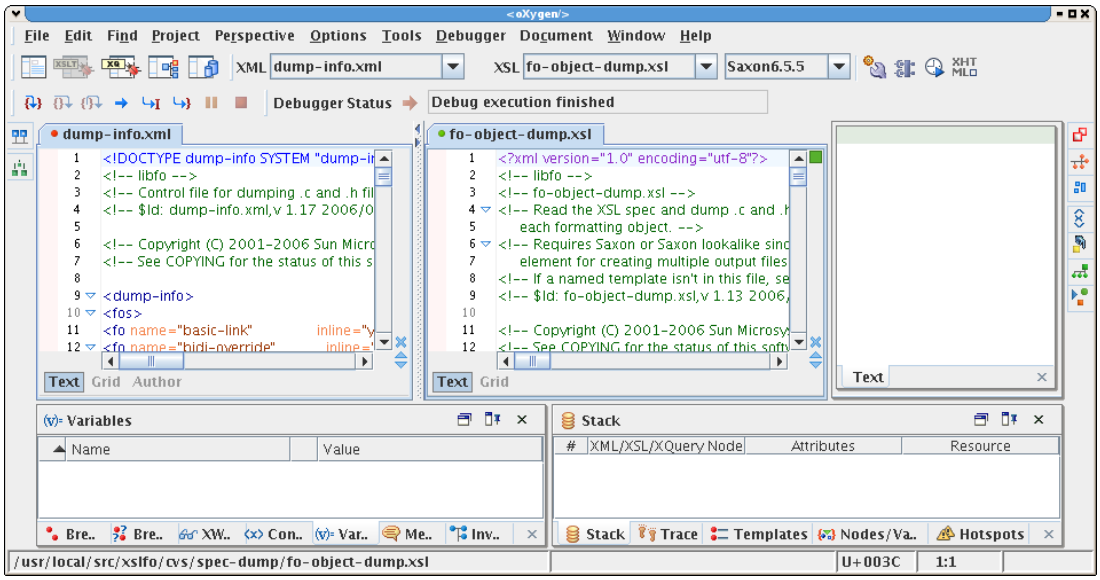
7

- University of Aarhus, Denmark
- `http://www.brics.dk/XSLV/`
- Analyses stylesheet – does not run the transformation
- Determine if output valid to output schema if input valid to input schema
  - DTD, XML Schema, or Relax NG subset
- Makes approximations
  - Some false negatives
  - Never a false positive
- Limited use while developing stylesheet
- Some vocabularies don't have a complete schema, e.g., XSL FO

Testing the Stylesheet

8

9 Debugger



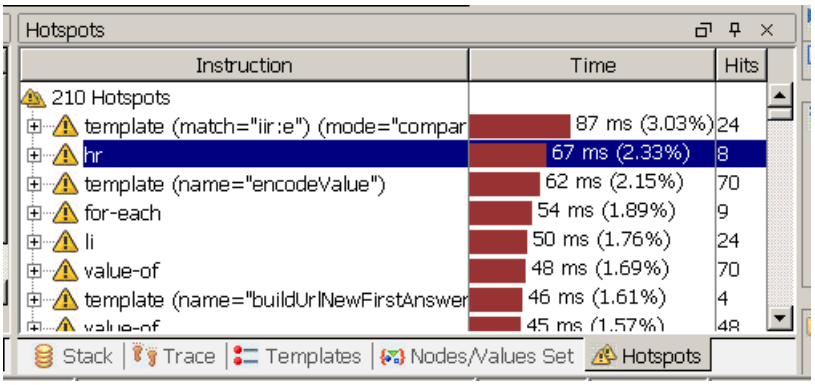
- Step through execution of transformation
- XML IDEs
- Emacs

10 Profiler

- Track activity of a running transformation
- Find “hotspots” where processor spending most time
- Typical scenario:
  - Profile
  - Review
  - Modify
  - Repeat
- XML IDEs
- Saxon
- xsltproc

Profiling XSLT is Inexact

11



- Template execution time depends on execution time of all templates it calls
- Depends on machine state
  - Faster when processor already in memory
- Depends on current node list
- Processor may be optimizing or doing lazy evaluation
- When profiler pauses processor to sample current state:
  - Sampling too seldom gives random results
  - Profiler overhead skews result when sampling too frequently

Profiling with Java (or C) Profiler

12

- Useful if understand internals of XSLT processor
- Last resort when stylesheet is a single template
- `java -Xrunhprof:cpu=samples`
  - Results in `java.prof.txt`
- Used by Michael Kay with Saxon
  - `http://www.biglist.com/lists/xsl-list/archives/200710/msg00192.html`

Unit Testing XSLT

13

- Run all or part of a transformation
- Provide “known” input
- Make assertions about expected result
- “Black box” testing: test complete transform *without* looking into implementation
- “White box” testing: test with knowledge of implementation
- Catching `xsl:message`
- Errors not caught by unit tests

14 A Test By Any Other Name Would Smell As Sweet

- Unit test
  - "Testing of individual hardware or software units or groups of related units" (IEEE)
  - "Execution of a complete class, routine, or a small program…which is tested in isolation from the more complete system" (Code Complete, 2nd Ed.)
  - A.k.a Programmer Test
  - XP usage may (or may not) be different from "professional QA testing community" usage
  - In OOP, only public methods (or not)
- "White box" test
  - Created based on code for the unit
- "Black box" test
  - Created based on specification for the unit
  - A.k.a. Functional Test, Acceptance Test

15 Unit Testing Frameworks for XSLT

Framework	Technology	Test	Report	JUnit?
XSLTunit	XSLT 1.0	XML	XML	
Juxy	Java	Java/XML	Text/XML/HTML/PDF	Yes
XSpec	XSLT 2.0	XML	HTML	
Unit Testing XSLT	XSLT 2.0	XML	XML/HTML	
tennison-tests	XSLT 2.0	XML	XML/HTML	
<XmlUnit/>	Java/C#	Java/C#	Text/XML/HTML/PDF	Yes
utf-x	Java	XML	Text/XML/HTML/PDF	Yes
XTC	XSLT 2.0	XML	XML/HTML	

Sample Unit Test

16

```
<testcase name="UTestJuxyTestCase1">
  <test name="Stylesheet">
    <stylesheet href="juxyx/x2j.xsl"/>
    <document><stylesheet href="href"><root/></stylesheet></
document>
    <assert-error>
      <apply-templates/>
    </assert-error>
  </test>
  <test name="MoreThanOneElementInTheList">
    <stylesheet href="juxyx/xsl/list.xsl"/>
    <document select="/list"><list><item>first item</item>
<item>second item</item><item>third item</item></list></document>
    <call-template name="makeList"/>
    <assert-equals>
      <expected>first item, second item, third item</expected>
    </assert-equals>
  </test>
</testcase>
```

When To Use Unit Testing?

17

- Good for XML and HTML output
  - Easy to make assertions about structure of output
- Harder to verify text output
  - Messier to make assertions
  - Testing individual templates easier than testing complete output
- Still need to validate output
  - Spec may be incorrect
  - Tests may have bugs

18 How Effective is Unit Testing?

Removal Step	Lowest Rate	Modal Rate	Highest Rate
Informal design reviews	25%	35%	40%
Formal design inspections	45%	55%	65%
Informal code reviews	20%	25%	35%
Formal code inspections	45%	60%	70%
Modelling or prototyping	35%	65%	80%
Personal desk checking of code	20%	40%	60%
Unit test	**15%**	**30%**	**50%**
New function (component) test	20%	30%	35%
Integration test	25%	35%	40%
Regression test	15%	25%	30%
System test	25%	40%	55%
Low-volume beta test (<10 sites)	25%	35%	40%
High-volume beta test (>1,000 sites)	60%	75%	85%

Source: *Software Estimation: Demystifying the Black Art, Steve McConnell*

19 Types of Tests

- Black box/white box
- Clean/dirty
- Full documents/fragments

20 "Black Box" (Functional) Testing

- For this input... expect this output
- Need complete documents for input
- Make assertions about complete documents at output
- Unit test for `match="/"` template
- Could also use Schematron, XSD 1.1

"White Box" Testing

21

- For this template, with this context… expect this output
- Complete documents or fragments as input
  - XPath to select part of a document, or
  - Fragment provided as part of test
- Make assertions about complete result from fragment

"Black Box" vs "White Box"

22

	Advantages	Disadvantages
"Black Box"	• Don't need to understand stylesheet • Reusable even if stylesheet refactored	• Need complete documents as input • If input changes, need to revise *all* tests

"White Box"	• Input and output can be simpler
	• Maybe revise fewer tests if input changes
		• Result may be invalid even if all tests pass
		• Best suited to testing named templates
		• Fragile if:
		  • Stylesheet refactored
		  • Selecting templates by context and `match` attribute changes
		• Keys may not work if input is fragment contained in test definition

23 "Clean" and "Dirty" Tests

Clean test:

- "Happy scenario"
- Good input
- Test for correct output

Dirty test:

- Incorrect input
- Test for correct handling of error condition:
  - Message
  - No output
  - Output based on input

24 "Clean" vs "Dirty"

	Advantages	Disadvantages
"Clean"	• Proves it works • Tests most common case	• Optimistic
"Dirty"	• Realistic	• More work • No limit

25 Full documents and Fragments as Test Source

Full document:

- Full, valid source document
- External to test definition

Fragment:

- XML fragment included in test definition
- A.k.a. document fragment, mock object, embedded document
- Enough to exercise test
- Not necessarily valid

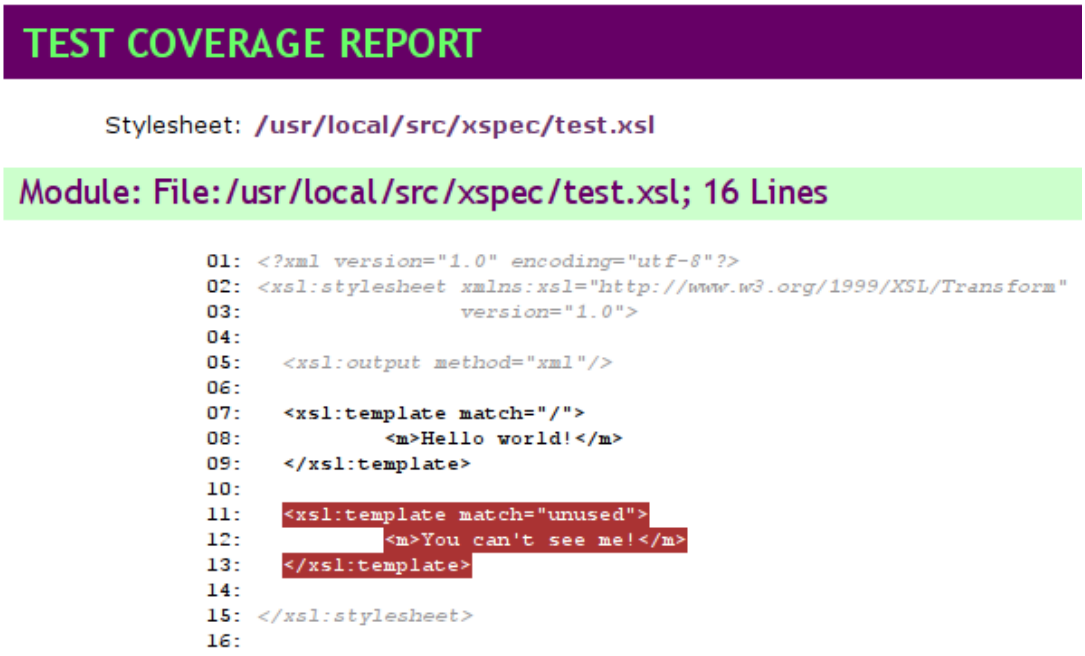
Full vs. Fragment

26

	Advantages	Disadvantages
Full	• Real document	• Longer processing time • Harder to manage • Longer XPaths in assertions • Fragile if schema changes • May not cover enough permutations
Fragment	• Easier to write • Easier to manage • Less to change when schema changes • Easier to add permutations	• May omit important elements, attributes • Whole stylesheet output may be incorrect

Coverage

27



- How much has been exercised by tests
- Extend to one run or multiple runs of “real” documents
- Tools: XSpec, Juxy (future)

28

Metrics

- Proscriptive – what you're doing wrong:
  - Unnecessary //
  - Unused functions, variables
  - Tools: xslqual.xsl, debugxslt, XSLStyle
- Descriptive – what's in your stylesheet:
  - Number and size of templates
  - Ratio of comments to code
  - Tools: XSLT Metrics
- Don't let metrics get in the way of solving the problem

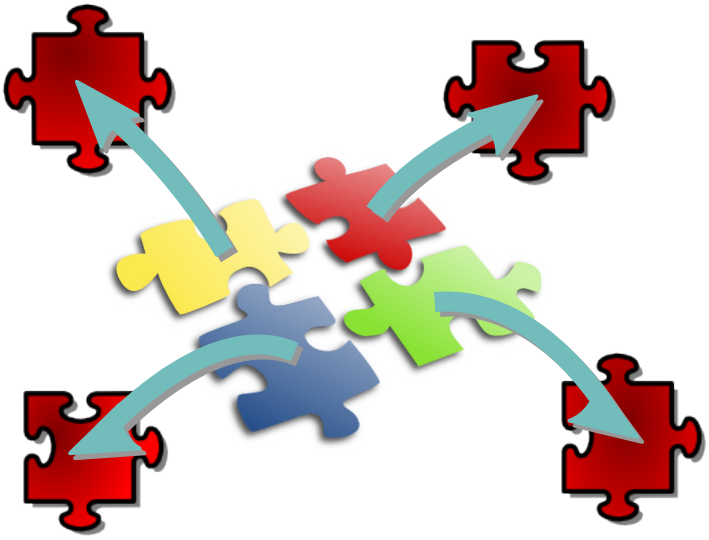
29

Problem Scenarios

- The jigsaw doesn't fit
- `xsl:message`
- `xsl:function`
- Specification errors

The Jigsaw Doesn't Fit

30



- Transform specified in manageable units
- Works according to specification but result not correct

`xsl:message` and Unit Tests

31

- Pure XSLT frameworks can't catch `xsl:message` output
- Can't differentiate between template giving error message and template accidentally giving no result
- Pure XSLT unit tests abort if `terminate="yes"`

`<xsl:template match="/">` and Unit Tests

32

- Pure XSLT frameworks typically stylesheet importing stylesheet under test
- Framework stylesheet uses `<xsl:template match="/">`
  - Unless begins with named template

Specification Errors

33

- DTD:

```
<!ATTLIST  art
             fmt         (art | bkr | ltr)   "art">
```

- Spec:
  - `art/@format` becomes `article/@article-type`
- Errors happen

Errors That Aren't Caught By Other Tests

34 Your Last, Best Defence

35 How Effective is Wetware Testing?

Removal Step	Lowest Rate	Modal Rate	Highest Rate
Informal design reviews	**25%**	**35%**	**40%**
Formal design inspections	**45%**	**55%**	**65%**
Informal code reviews	**20%**	**25%**	**35%**
Formal code inspections	**45%**	**60%**	**70%**
Modelling or prototyping	35%	65%	80%
Personal desk checking of code	**20%**	**40%**	**60%**
Unit test	15%	30%	50%
New function (component) test	20%	30%	35%
Integration test	25%	35%	40%
Regression test	15%	25%	30%
System test	25%	40%	55%
Low-volume beta test (<10 sites)	25%	35%	40%
High-volume beta test (>1,000 sites)	60%	75%	85%

Source: *Software Estimation: Demystifying the Black Art, Steve McConnell*

36 Errors Not Caught By Unit Tests

- Out of date or inappropriate comments
- Template that is never matched
- Specification errors
- Typos in `xsl:strip-space` and `xsl:preserve-space`
- Error in `xsl:when` covered by `xsl:otherwise`

37

Resources

- Testing XSLT – `http://www.menteithconsulting.com/wiki/TestingXSLT`
- Testing XSL FO – `http://www.menteithconsulting.com/wiki/TestingXSLFO`

Tools

38

- XSLTunit – `http://xsltunit.org`
- Juxy – `http://juxy.tigris.org/`
- XSpec – `http://xspec.googlecode.com/`
- Unit Testing XSLT – `http://www.jenitennison.com/xslt/utilities/unit-testing`
- tennison-tests – `http://tennison-tests.sourceforge.net/`
- <XmlUnit/> – `http://xmlunit.sourceforge.net/`
- utf-x – `http://utf-x.sourceforge.net/`
- XTC – `http://www.fgeorges.org/xslt/xslt-unit/`
- xslqual.xsl – `http://gandhimukul.tripod.com/xslt/xslquality.html`
- XSLT Metrics – `http://code.menteithconsulting.com/wiki/XSLTMetrics`
- debugxslt – `http://code.google.com/p/debugxslt/`
- XSLStyle – `http://www.cranesoftwrights.com/resources/xslstyle/index.htm`

Menteith

CONSULTING