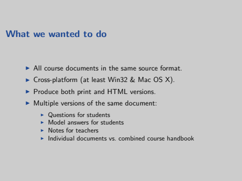
What we wanted to do

- All course documents in the same source format.
- Cross-platform (at least Win32 & Mac OS X).
- Produce both print and HTML versions.
- Multiple versions of the same document:
  - Questions for students
  - Model answers for students
  - Notes for teachers
  - Individual documents vs. combined course handbook



What we wanted to do

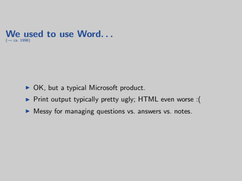
- All course documents in the same source format.
- Cross-platform (at least Win32 & Mac OS X).
- Produce both print and HTML versions.
- Multiple versions of the same document:
  - Questions for students
  - Model answers for students
  - Notes for teachers
  - Individual documents vs. combined course handbook

Cross-platform tends to imply (but doesn't mandate) open source tools.

We used to use Word. . .

(→ ca. 1998)

- OK, but a typical Microsoft product.
- Print output typically pretty ugly; HTML even worse :(
- Messy for managing questions vs. answers vs. notes.



We used to use Word. . .
(→ ca. 1998)

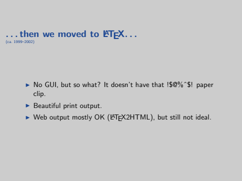
- OK, but a typical Microsoft product.
- Print output typically pretty ugly; HTML even worse :(
- Messy for managing questions vs. answers vs. notes.

1. Supported hyperlinks, styles & hidden text.

. . . then we moved to LaTeX. . .

(ca. 1999–2002)

- No GUI, but so what? It doesn't have that !$@%^$! paper clip.
- Beautiful print output.
- Web output mostly OK (LaTeX2HTML), but still not ideal.



. . . then we moved to LaTeX. . .

(ca. 1999–2002)

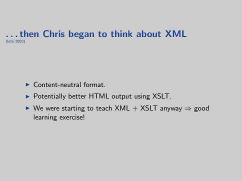
- No GUI, but so what? It doesn't have that !$@%^$! paper clip.
- Beautiful print output.
- Web output mostly OK (LaTeX2HTML), but still not ideal.

It was much better than Word! But there were many niggling issues, such as handling of images under certain circumstances.

. . . then Chris began to think about XML

(late 2002)

- Content-neutral format.
- Potentially better HTML output using XSLT.
- We were starting to teach XML + XSLT anyway ⇒ good learning exercise!



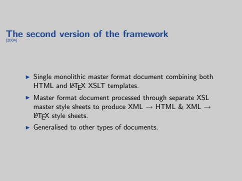
. . . then Chris began to think about XML
(late 2002)

- Content-neutral format.
- Potentially better HTML output using XSLT.
- We were starting to teach XML + XSLT anyway ⇒ good learning exercise!

Different output formats vs. different views of the same document.Or at least, more reliable HTML output.

The second version of the framework
(2004)

- Single monolithic master format document combining both HTML and LaTeX XSLT templates.
- Master format document processed through separate XSL master style sheets to produce XML → HTML & XML → LaTeX style sheets.
- Generalised to other types of documents.

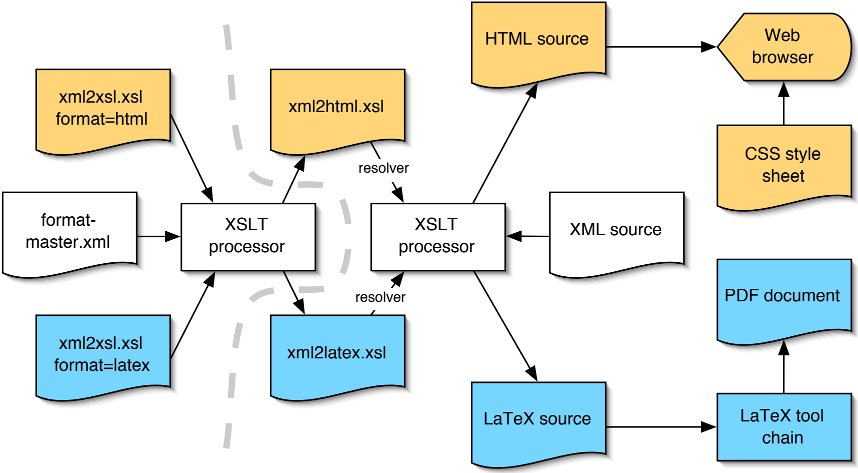


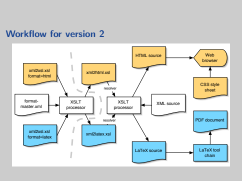
The second version of the framework
(2004)

- Single monolithic master format document combining both HTML and LaTeX XSLT templates.
- Master format document processed through separate XSL master style sheets to produce XML → HTML & XML → LaTeX style sheets.
- Generalised to other types of documents.

Nicely recursive—using a style sheet to generate style sheets!

Workflow for version 2

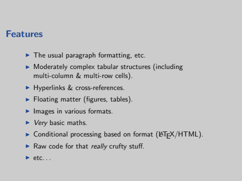



Workflow for version 2

Everything to the right of the thick dashed grey line is part of the day-to-day workflow. The stuff on the left on occurs when the master format is changed.

Features

- The usual paragraph formatting, etc.
- Moderately complex tabular structures (including multi-column & multi-row cells).
- Hyperlinks & cross-references.
- Floating matter (figures, tables).
- Images in various formats.
- *Very* basic maths.
- Conditional processing based on format (LaTeX/HTML).
- Raw code for that *really* crufty stuff.
- etc. . .



Features

- The usual paragraph formatting, etc.
- Moderately complex tabular structures (including multi-column & multi-row cells).
- Hyperlinks & cross-references.
- Floating matter (figures, tables).
- Images in various formats.
- *Very* basic maths.
- Conditional processing based on format (LaTeX/HTML).
- Raw code for that *really* crufty stuff.
- etc. . .

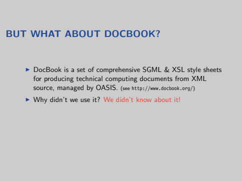
Figures and tables don’t float in HTML.

Examples

1. Show parts of format-master.xml and xml2xsl.xsl.
2. INFO 321 Analogy document (general document).
3. INFO 321 lab 3 is a good example of a lab document, with graphics, etc., plus it's embedded within the INFO 321 course handbook.
4. Do a clean build of the Analogy document and the handbook.

BUT WHAT ABOUT DOCBOOK?

- DocBook is a set of comprehensive SGML & XSL style sheets for producing technical computing documents from XML source, managed by OASIS. (see `http://www.docbook.org/`)
- Why didn't we use it? We didn't know about it!



BUT WHAT ABOUT DOCBOOK?

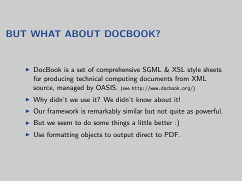
- DocBook is a set of comprehensive SGML & XSL style sheets for producing technical computing documents from XML source, managed by OASIS. (see http://www.docbook.org/)
- Why didn't we use it? We didn't know about it!

1. Also, it was a useful learning exercise to find out how XML + XSLT worked.

BUT WHAT ABOUT DOCBOOK?

- DocBook is a set of comprehensive SGML & XSL style sheets for producing technical computing documents from XML source, managed by OASIS. (see `http://www.docbook.org/`)
- Why didn't we use it? We didn't know about it!
- Our framework is remarkably similar but not quite as powerful.
- But we seem to do some things a little better :)
- Use formatting objects to output direct to PDF.

BUT WHAT ABOUT DOCBOOK?



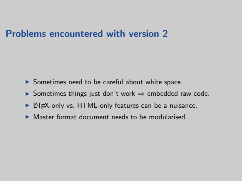
BUT WHAT ABOUT DOCBOOK?

- DocBook is a set of comprehensive SGML & XSL style sheets for producing technical computing documents from XML source, managed by OASIS. (see `http://www.docbook.org/`)
- Why didn't we use it? We didn't know about it!
- Our framework is remarkably similar but not quite as powerful.
- But we seem to do some things a little better :)
- Use formatting objects to output direct to PDF.

1. For example, their tabulars are much more powerful!
2. For example, handling of special characters. Our handling of hyperlinks also seems a bit more flexible and intuitive.

Problems encountered with version 2

- Sometimes need to be careful about white space.
- Sometimes things just don't work ⇒ embedded raw code.
- LaTeX-only vs. HTML-only features can be a nuisance.
- Master format document needs to be modularised.



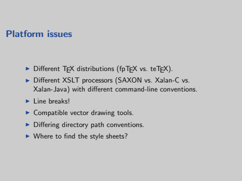
Problems encountered with version 2

- Sometimes need to be careful about white space.
- Sometimes things just don't work ⇒ embedded raw code.
- LaTeX-only vs. HTML-only features can be a nuisance.
- Master format document needs to be modularised.

1. For example, tabulars that followed each other in the source and were intended to be separate paragraphs, ended up on the same line in the PDF because the white space between them got eaten by the transformation.
2. Master format is currently about 2000 lines.

Platform issues

- Different TEX distributions (fpTEX vs. teTEX).
- Different XSLT processors (SAXON vs. Xalan-C vs. Xalan-Java) with different command-line conventions.
- Line breaks!
- Compatible vector drawing tools.
- Differing directory path conventions.
- Where to find the style sheets?



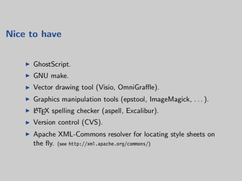
Platform issues

- Different TeX distributions (fpTeX vs. teTeX).
- Different XSLT processors (SAXON vs. Xalan-C vs. Xalan-Java) with different command-line conventions.
- Line breaks!
- Compatible vector drawing tools.
- Differing directory path conventions.
- Where to find the style sheets?

1. We now both use teTeX.
2. Solved the XSLT processor problem by defining an XSLT environment variable as appropriate on each machine, then defining a set of functions in our Makefiles to call the appropriate processor with the correct command line.
3. Line breaks pretty much resolved by storing everything in CVS.
4. Solved the path problem by using Cygwin on the Win32 side.
5. Can resolve the style sheet location issue using either environment variables, symlinks or just copying the files into the local directory, but none of these are ideal. A nicer solution is to use the Apache XML-Commons resolver.

Nice to have

- GhostScript.
- GNU make.
- Vector drawing tool (Visio, OmniGraffle).
- Graphics manipulation tools (epstool, ImageMagick, . . .).
- LaTeX spelling checker (aspell, Excalibur).
- Version control (CVS).
- Apache XML-Commons resolver for locating style sheets on the fly. (see `http://xml.apache.org/commons/`)



Nice to have

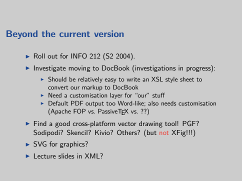
- GhostScript.
- GNU make.
- Vector drawing tool (Visio, OmniGraffle).
- Graphics manipulation tools (epstool, ImageMagick, . . .).
- LaTeX spelling checker (aspell, Excalibur).
- Version control (CVS).
- Apache XML-Commons resolver for locating style sheets on the fly. (see `http://xml.apache.org/commons/`)

1. One nice feature of using CVS is that you can embed the CVS ID tags in the document so that you know which version of a document a student has.
2. The resolver enables you to locate style sheets and DTDs on the fly using catalog files.

Beyond the current version

- Roll out for INFO 212 (S2 2004).
- Investigate moving to DocBook (investigations in progress):
  - Should be relatively easy to write an XSL style sheet to convert our markup to DocBook
  - Need a customisation layer for “our” stuff
  - Default PDF output too Word-like; also needs customisation (Apache FOP vs. PassiveTEX vs. ??)
- Find a good cross-platform vector drawing tool! PGF? Sodipodi? Skencil? Kivio? Others? (but not XFig!!!)
- SVG for graphics?
- Lecture slides in XML?

└Beyond the current version



Beyond the current version

- Roll out for INFO 212 (S2 2004).
- Investigate moving to DocBook (investigations in progress):
  - Should be relatively easy to write an XSL style sheet to convert our markup to DocBook
  - Need a customisation layer for "our" stuff
  - Default PDF output too Word-like; also needs customisation (Apache FOP vs. PassiveTEX vs. ??)
- Find a good cross-platform vector drawing tool! PGF? Sodipodi? Skencil? Kivio? Others? (but not XFig!!!)
- SVG for graphics?
- Lecture slides in XML?

I can see advantages to having the lecture slides in XML, as it's going to be a lot easier to produce an HTML version from an XML original than from the LATEX/Beamer files that I have now.