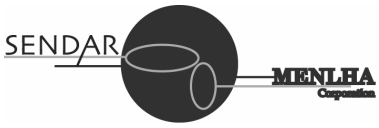


eCTD Specification

Tomas Hansson, Sendar-Menlha Corporation

Presentation available on:
www.eCTD.com

Introduction

- ICH M4 Expert Working Group (EWG)
  - Defined the Common Technical Document (CTD)
- The ICH M2 EWG
  - Specification for the Electronic Common Technical Document (eCTD)
- Version 3.0: September 12, 2002
- Lists the criteria that will make an electronic submission technically valid

Introduction (cont.)

- Interface for industry to agency transfer of regulatory information
- Facilitation of:
  - Creation
  - Review
  - Lifecycle management (LCM)
  - Archival
- Extensible Markup Language (XML)

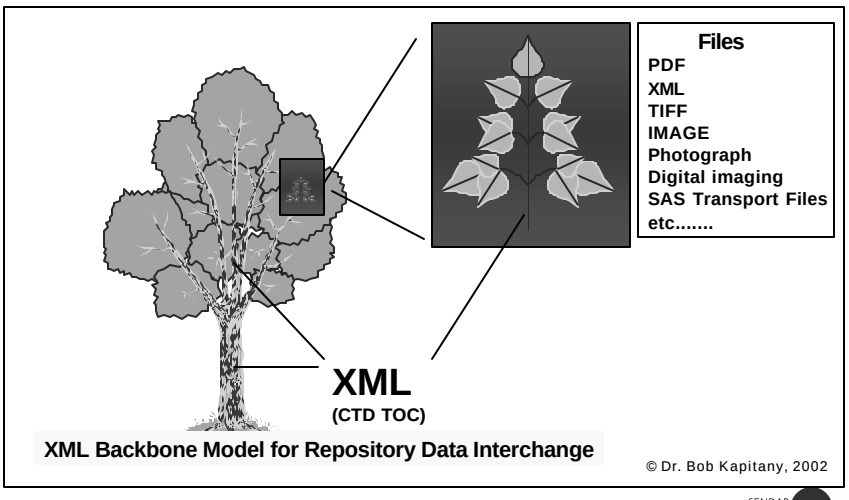
Why XML?

- XML is a markup language for documents containing structured information
- W3C Recommendation (Feb. 1998)
  - Presentation neutral
  - Vendor Neutral
  - Plain ASCII (or Unicode)
  - Self-validating
- Readable today and in 50 years!

Background and Scope

- eCTD is based upon content defined within the CTD
  - CTD describes the organization of modules, sections and documents
  - CTD does not cover the full submission that is to be made in a region
- eCTD is applicable to all modules of initial registration applications
  - The XML backbone has been developed to handle both the regional and common parts of submissions

The Backbone Concept



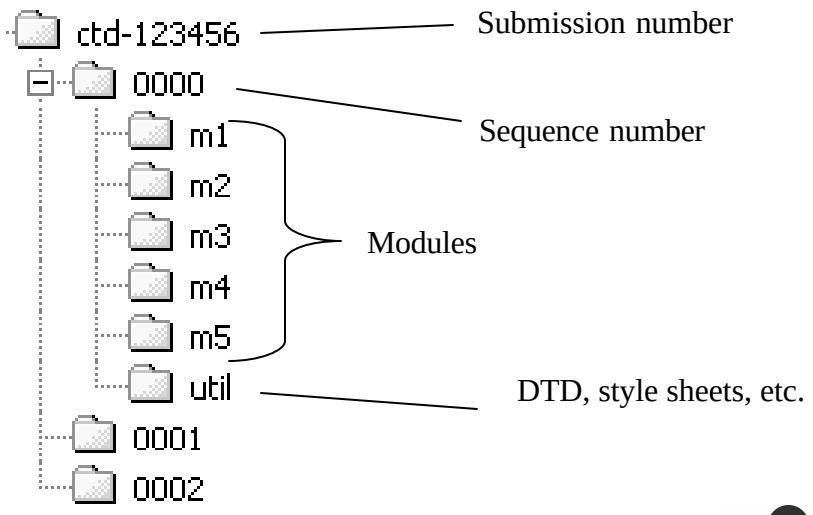
The eCTD Submission

- Collection of data objects
- Main function: data exchange
  - Applications needed to create, verify and review
- The eCTD submission is composed of the following:
  - Directory structure
  - XML eCTD instance
  - Content files

Directory Structure

- Structure of directories and files
- “Reasonable maximum” number of entries
- The name of the files and directories are identifiers:
  - Short
  - Some meaning helps
- Names for directories and files are *recommended* in Appendix 4 of spec.

Directory Structure (cont.)



XML eCTD Instance

- In top-level directory
- Starting file for eCTD processing
- Validity checking
- Checksum verification
- Links from the instance to leaf files
- Meta-data

XML eCTD Instance (cont.)

- What does it look like?

```
<?xml version="1.0"?>
  <!DOCTYPE ectd:ectd SYSTEM "ectd_v300.dtd">
  <ectd:ectd xmlns:ectd="http://www.ich.org/ectd"
    xmlns:xlink="http://www.w3c.org/1999/xlink">
    <m2-common-technical-document-summaries>
    <leaf operation="new" checksum="123" checksum-
    type="md5“ xlink:href=“some-file.pdf”>
      <title>Document title</title>
    </leaf>
    </m2-common-technical-document-summaries>
    </ectd:ectd>
```

Content files

- Common formats:
  - Narrative: Portable Document Format (**PDF**)
  - Structured: Extensible Markup Language (**XML**)
  - Graphic: Whenever possible, use PDF.
    - Joint Photographic Experts Group (**JPEG**)
    - Portable Network Graphics (**PNG**)
    - Scalable Vector Graphics (**SVG**)
    - Graphics Interchange Format (**GIF**)
- Regional formats
  - Word, RTF, WordPerfect

Linking

- Backbone contains *relative* links to Content files
- Content files need *relative* links to other Content files
- Issues:
  - PDF files should be *less than 50Mb*. Splitting breaks links!
  - Moving files from work area to eCTD directory structure breaks links!
  - Changing links after build breaks checksums!

Checksums

- Submission must contain checksums
  - Each individual Content (stored in the Backbone)
  - Checksum file for the eCTD XML instance (the Backbone)
- Initially, the MD5 Message-Digest Algorithm (MD5)
  - The integrity of each file can be verified
  - File has not been altered in the historical archive of the regulatory authority

Checksums

- How (MD5) checksums work:
  - 128-bit “fingerprint” of your document
  - The likelihood of two documents having the same checksum is one in 2^{64} (20 digit number)
  - The MD5 checksum of a simple text document:
    - “Hello World” – One space
      `3579c8da7f1e0ad94656e76c886e5125`
    - “Hello World” – Two spaces
      `978aedb0ff0b7ee6c488230997954d7b`
  - A single space totally alters the checksum of the document!

Media

- Less than 1.4 Mb
  - 3.5 inch DOS Formatted Floppy Disks (EU)
- Less than 10Mb
  - 3.5 inch DOS Formatted Floppy Disks (USA)
- Less than 650 MB
  - CD-ROM ISO 9660 - Joliet (EU, Japan)
- Less than 7 GB
  - CD-ROM ISO 9660 - Joliet (Japan, USA, Canada)

Media (cont.)

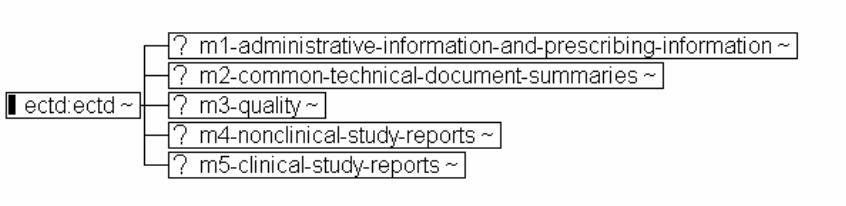
- Greater than 7 GB
  - Digital Tape (USA)
- More than 650 MB
  - DVD (EU, Canada)

Cover Letter

- PDF cover letter
- Paper cover letter
  - Description of submission
  - Listing of the sections of the submission filed as paper, electronic, or both paper and electronic
  - Description of the electronic submission (type and number of electronic media, sizes, etc)
  - A statement that the submission is virus free
  - Printed contents of the index-md5.txt file as an appendix
  - Points of contact for the submission

Backbone technicalities – Nodes

- Can attach leafs or add sub-structure
- Closely matches the CTD
- Structures possible down to four levels
- Descriptive names
- Some nodes have “attributes” and can repeat



Node Extensions

- Possible to extend the definition of an element by creating node extensions
- It is *discouraged* and should only be done when there is no other feasible means to submit information
  - Can only extend the lowest level of defined elements
  - Should not extend the element more than one level
- Updates should be submitted using identical structure

Backbone technicalities – Leafs

- Leafs can appear at any level
- Leafs are used to point to documents
  - All pointed-to documents must be specified using a *relative* path
- Large documents can be split up over many leafs
- Leaf documents can point to other leafs using *relative* paths

Leaf properties (“attributes”)

- Leafs have the following attributes
  - `xlink:href` – Filename of actual content file
  - `xml:lang` – The primary language. (Use ISO-639)
  - `application-version` – The version of the software application that was used to create this file
  - `font-library` – Commercial name of fonts/font set used to create the document
  - `checksum` – The checksum value for the file being submitted
  - `checksum-type` – The checksum algorithm used (intially MD5)

Leaf properties (“attributes”) (cont.)

- `modified-file` – The name of the file to be modified
- `operation` – Operation to be performed on the “modified-file” :
  - new
  - replace
  - append
  - delete
- `version` – The file submitter’s internal version
- `keywords` – Keywords to enhance searches (not currently used)

Life-cycle management

- “operation” attribute enables LCM
  - Amended submissions should only include files that have changed!
  - “modified-file” specifies the file that changes have been made on
    - new – The file has no relationship with files submitted previously
    - append – This file is **current**. Previous file is **current** but has been **appended**

Life-cycle management (cont.)

  - replace – This file is **current**. Previous file has been **replaced**
  - delete – Previous file is no longer relevant
- Submission sequences (folders) start with 0000 and increase
- LCM Issues
  - How to keep track of “current: submission as number of submissions grow?
  - Recreate older submissions
  - Creation of “deltas” for submitting

Submission creation – The hard way

- Steps to create an eCTD submission manually:
  1. Identify all files that will go into the submission
  2. Read through the spec. and create correct eCTD directory structure **(!)**
  3. Copy files from working directories into the eCTD directory structure (need to rename as per spec) **(!)**
  4. Redraw all external PDF/XML links to point to new locations and file names **(!)**
  5. Split up too large PDF files (> 50Mb) and redraw all internal links **(!)**

Submission creation (cont.)

6. Generate MD5 checksums for all files in the submission **(!)**
7. Create the eCTD Backbone **(!!!)**
   - Notepad
   - XML Spy/Adept/XML Writer/...
8. In the backbone, paste in all the MD5 checksums for the files **(!)**
9. Create an MD5 checksum file for the eCTD Backbone
10. Copy in content of the “util” directory (DTDs, style sheets, etc)

Submission creation (cont.)

11. Create the top-level directories
    - Submission number
    - Sequence number
12. Burn the CD
13. Create the cover letter
    - Print out the MD4 checksum
    - Sign and ship

- Estimated time for manual labour
  4 person-weeks

Submission creation – The easy way

- Steps to create an eCTD submission with eCTD dedicated tools
  1. Identify all files that will go into the submission
  2. Drag & drop files into the tool
  3. Use intelligent tools to split PDF files
  4. Hit the “Build” button
  5. Create the top-level directories
     - Submission number
     - Sequence number

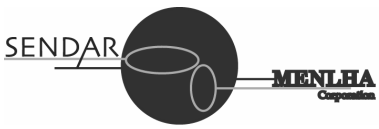
Submission creation (cont.)

  6. Burn the CD
  7. Create the cover letter
     - Print out the MD4 checksum
     - Sign and ship

- Estimated time for manual labour: 4 hours – 2 days

Submission Validation

- Once a submission is received (or before shipping) it needs to check for validity
  - XML Backbone correct as per rules
  - Directory structure correct
  - All referenced files present
  - All MD5 checksums correct
- Manual checking
- Automated tools
  - eCTD Validator (Free!)
  - eCTD Reviewer, others...



eCTD Specification
www.eCTD.com
thansson@sendar.com
(613) 715-9249