



# Single Source Documentation

Open Publish, Sydney, August 2007

# Agenda

---

**Introduction**

**Evolution of documentation**

**Why single source documentation?**

**Components of single source systems**

## Introduction

---

### About Me

- Gareth Oakes (goakes@ptc.com)
- Technical Sales Specialist for Arbortext products
- Working in enterprise publishing since 2001 (Advent3B2, Arbortext, Allette)
- Worked as web developer and web master from 1997 to 2001

## A Provider of Leading PLM\* and Content Management Solutions

PTC helps companies optimize their product development processes and win with superior *physical and information* products.

Employees (as of 9/30/06)	4,309
FY 2006 revenue	\$855 Million
Cash and equivalents (as of 9/30/06)	\$183 Million

\* Product Lifecycle Management



## Introduction

---

### About Arbortext

- Long history – company founded in 1982
- Market leader in enterprise publishing (products such as Epic Editor and E3)
- Heavily involved in standards, eg.
  - Member of XML Core Working Group
  - Author of the XSL-FO Recommendation
  - First vendor to join DITA committee, first vendor to support DITA editing
- Arbortext was acquired by PTC in 2005
- Arbortext is now the dedicated publishing division of PTC



# Evolution of Documentation



## Pictures on cave walls

- ~30000BC

## Logographs on clay tablets and papyrus

- Egypt & Mesopotamia
- ~4000BC



## Alphabets and paper, leading to adoption of moveable type

- Johannes Gutenberg
- circa 1447

## WWW and HTML

- Tim Berners-Lee
- August 6, 1991



## PDF

- Adobe
- circa 1993

## What next?



## Evolution of Documentation Systems

---

The process of creating a book, through the ages:

1. People write books on clay tablets
2. People write books on parchment
3. People write books on paper
4. People write books on typewriter
5. People write books or webpages on computer
6. ???



## Documentation Systems

---

The process of developing documentation really hasn't changed much since the clay tablet days!

OK, OK, so Gutenberg fixed the publishing problem with moveable type

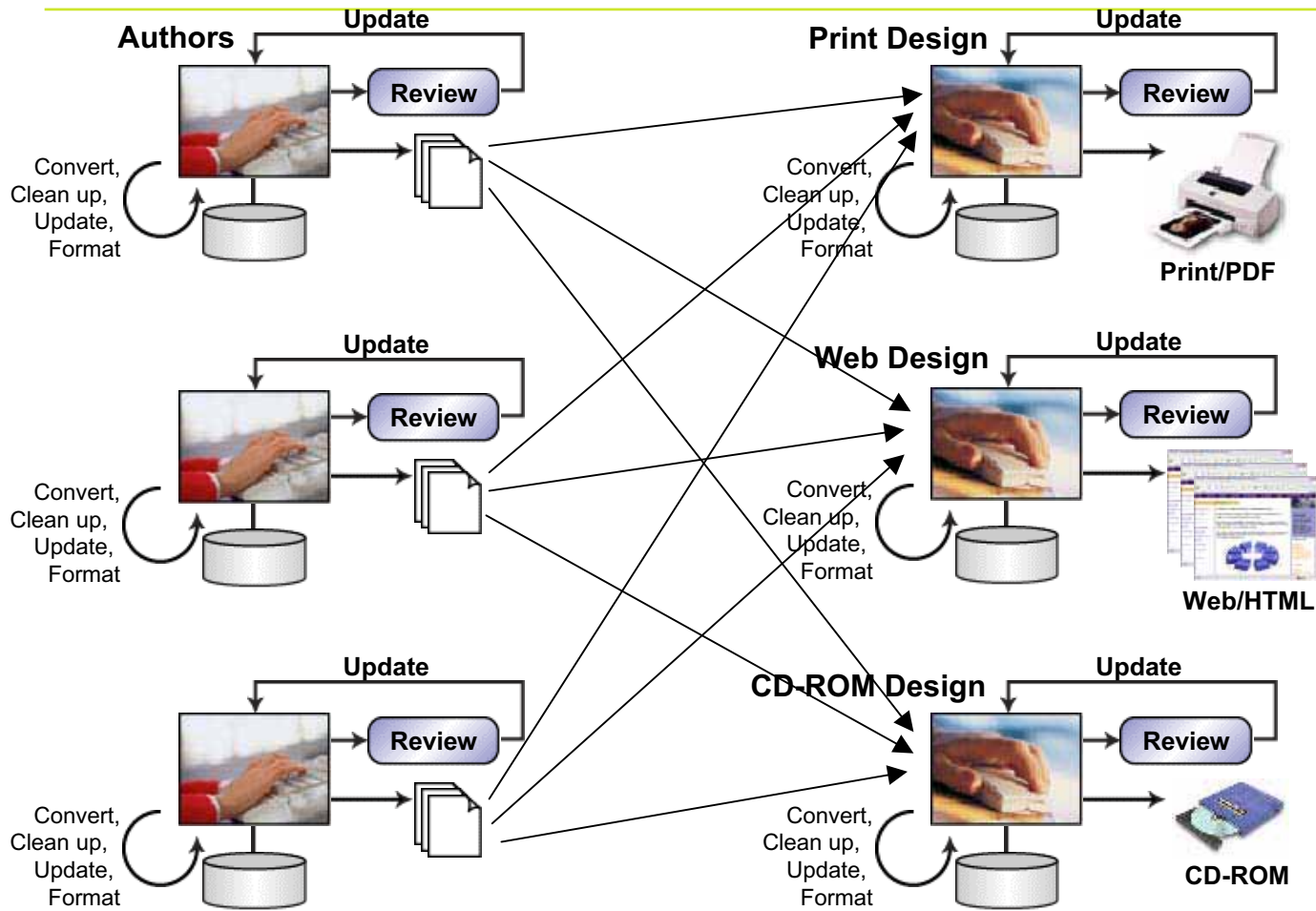
But the basic process hasn't changed:

- Author the content
- Review and approve the content
- Produce or publish deliverable from the content

**Single source documentation is a revolutionary improvement to the tired old process, providing some real benefits:**

- Collaboration between large groups
- Many authors can work on one deliverable simultaneously
- Changes are tracked as a version history
- Simple support for many types of deliverable (PDF, HTML, Online Help, Word...)

# Example Workflow – Traditional Publishing



==

*How can you improve your publishing process?*

## Why single source documentation?

Reduces cost of production (especially for multiple outputs)

Greater efficiency → faster time to market

Reduces effort of content maintenance, eg. a publisher once told me:

*“Across all our books, we have about 3 or 4 different heights for the Eiffel Tower!”*

Improve accuracy and quality of documentation

Saves on translation costs, allowing a more refined & reusable translation memory, eg. another publisher told me:

*“We spend in excess of \$1 million each year on translation!”*



Also helps standardisation, even down to things like Simplified English

**For non-trivial documentation, single source is global best practice**

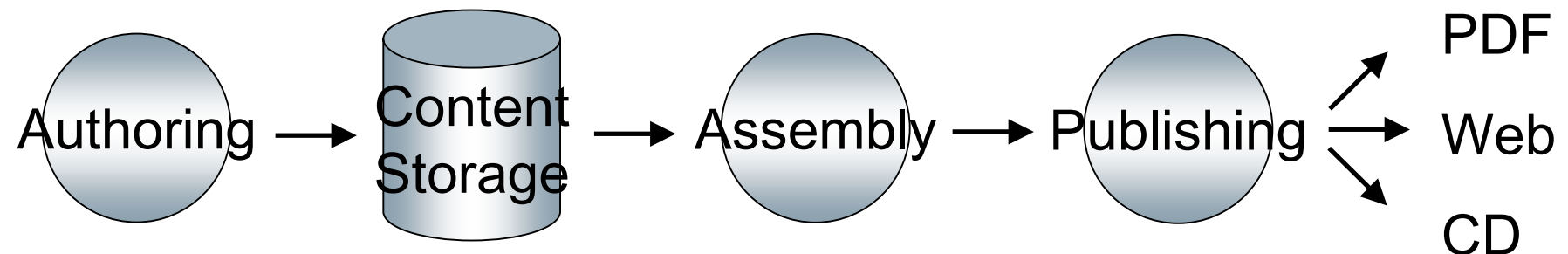
## What is a single source documentation system?

Content is authored once, but is tagged and structured such that it can be reused in many scenarios

A management system is used to store all content versions, plus the structures that define deliverable items

Deliverables are created by linking many content fragments together (similar to a recipe...)

There is one process for publishing to multiple deliverables – reduce redundancy in production process



## Foundations of single source documentation systems

---

### Built on standard data formats

- Structured content (XML or SGML)
- Content format is **commoditised** – plethora of supporting tools and technologies

### Treat content as separate entities, combined only for publishing

- Reuse content by linking – no more copy and paste!
- Manage the **lifecycle** of each unique object

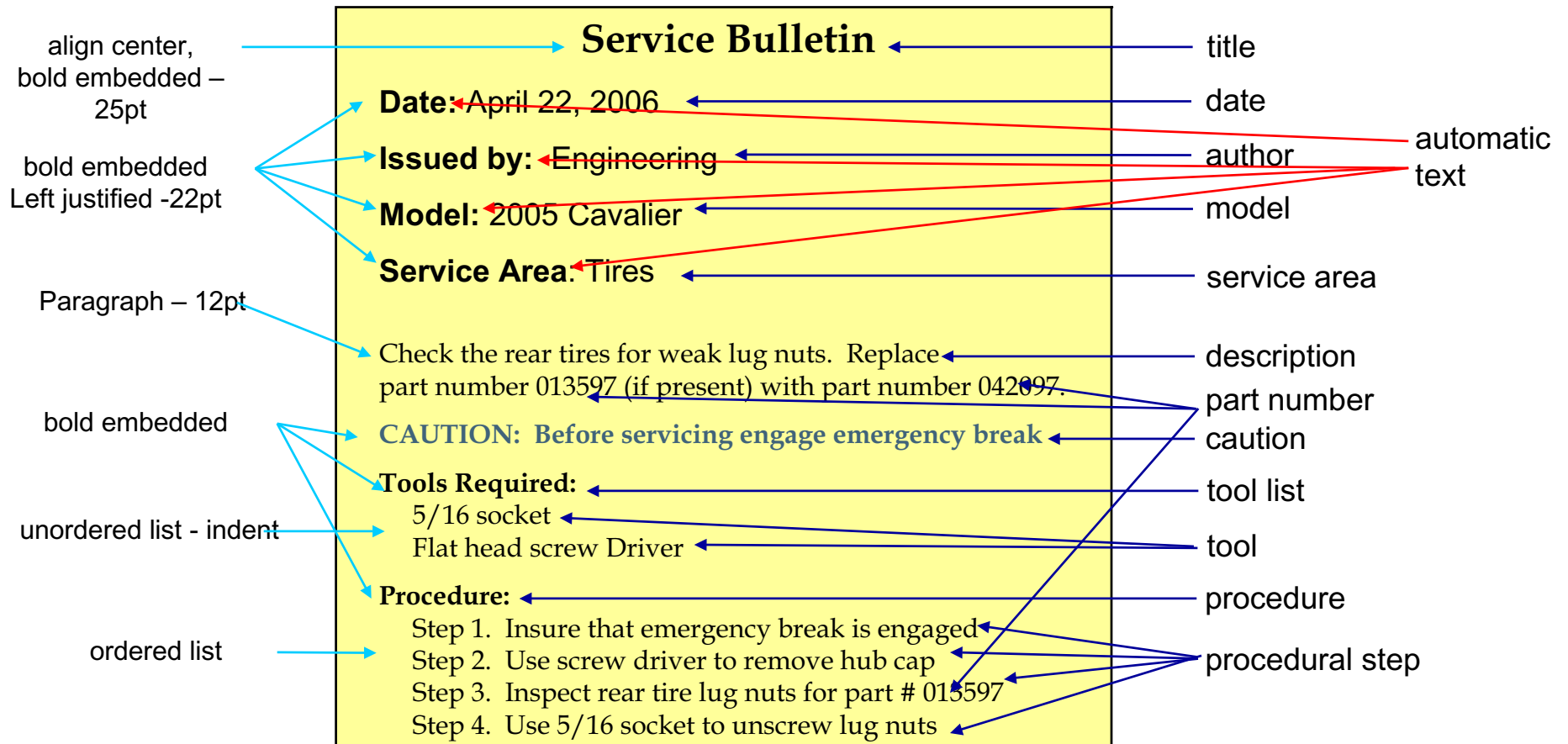
### Separate presentation from information

- Content can be created without concern for its appearance
- Content is authored **once**, then can be easily delivered to all relevant formats (web, PDF, print, CD/DVD, wireless device...)
- Deliverables can be simply re-styled, re-packaged, re-purposed
- Reuse styling templates across many publications – eliminate redundancy

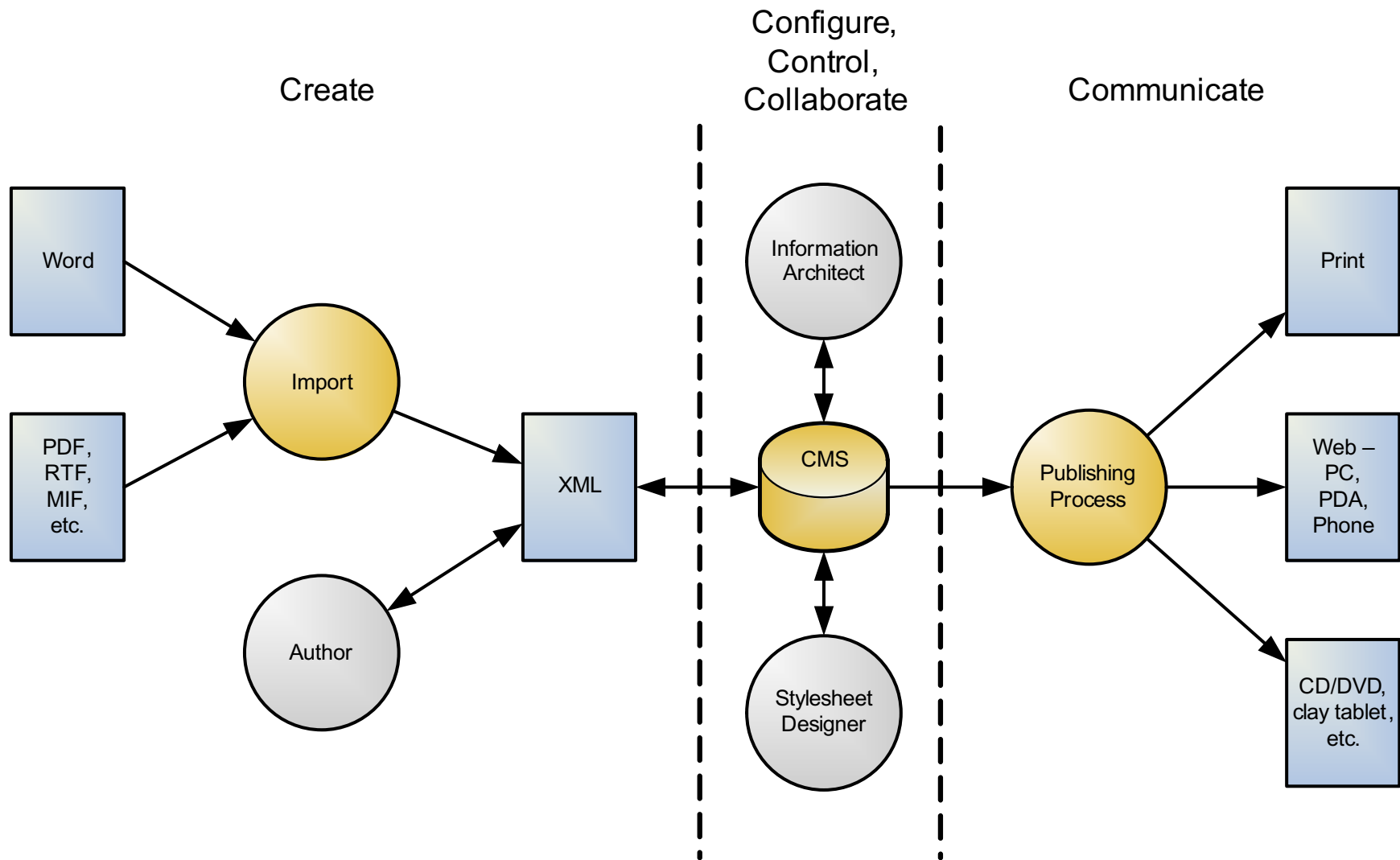
# Separating presentation from information

## Word Processing

## XML Markup



# Example Workflow – Single Source Documentation



## Components of a single source documentation system

### Authoring

- Text
- Photographs
- Illustrations

### Publishing

- PDF
- Web

### Content Management

- Check-out/check-in of content
- Version history
- Auditing/reporting
- Workflow
- Content Review/Approval



## Authoring

---

### How to represent content?

- Unstructured content (such as Word documents) are not suitable for single source documentation
- XML is the best choice due to good architecture and large support base
- There are many different flavours of XML...
  - General books: DocBook and DITA
  - Defense: ATA/100, MILSPEC, S1000D, SCORM, DEF(AUST) 5629A/5629B
  - Customised: catalogues, product information, legislation, etc.
- Recommended: **DITA** for general techpubs, **S1000D** for defense techpubs

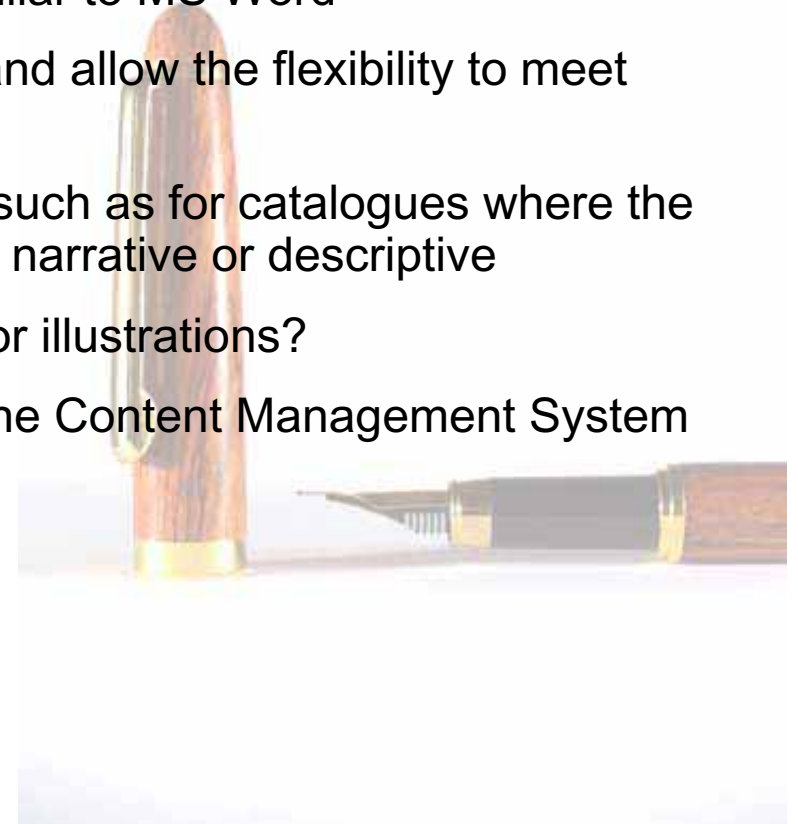


# Authoring

---

## What tools to use?

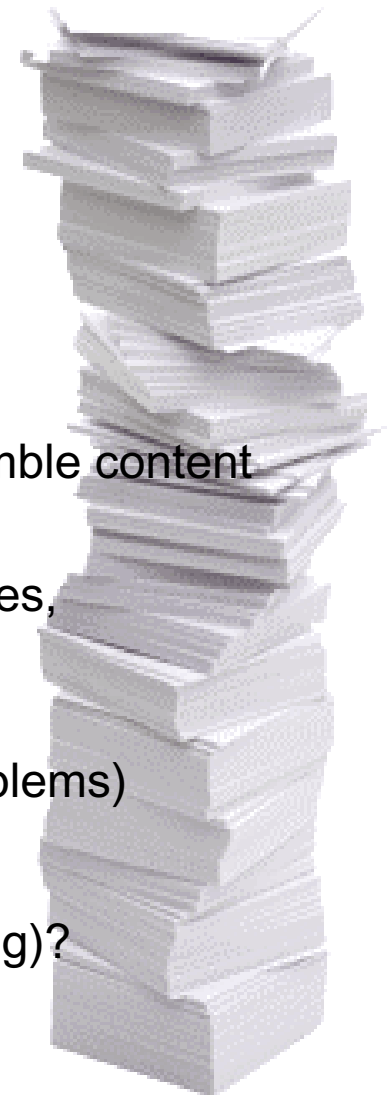
- Authoring is a key activity, the authoring tool can make or break the single source documentation system!
- Most authors want something “familiar”, eg. similar to MS Word
- The authoring tool must natively support XML and allow the flexibility to meet business requirements
- Sometimes authors have very different needs, such as for catalogues where the data is highly structured and factual rather than narrative or descriptive
- Does the authoring tool work well with images or illustrations?
- Authoring tool must integrate seamlessly with the Content Management System



## Publishing

### How to publish content?

- The publishing process will involve three stages:
  - Assemble content fragments, to build a compound document
  - Apply styling to the compound document
  - Publish the styled document to the relevant output media
- Publishing process must integrate with the CMS in order to assemble content automatically, or according to some business logic
- Publishing process must accommodate for all required output types, eg. PDF and web
- Can the publishing tool meet all output requirements?  
(Eg. complex footnotes or rotated text in the PDF may cause problems)
- Is performance important?
- Is manual “touchup” of documents required (usually before printing)?



## Publishing

---

### How to apply styling to the published deliverable?

- Usually a function of the publishing process
- Ideally one stylesheet language for all output types, which can be configured with a GUI rather than by complex scripting
- Support all generated text requirements (Table of Contents, Index, etc.)
- Support for time saving features such as stylesheet modularisation

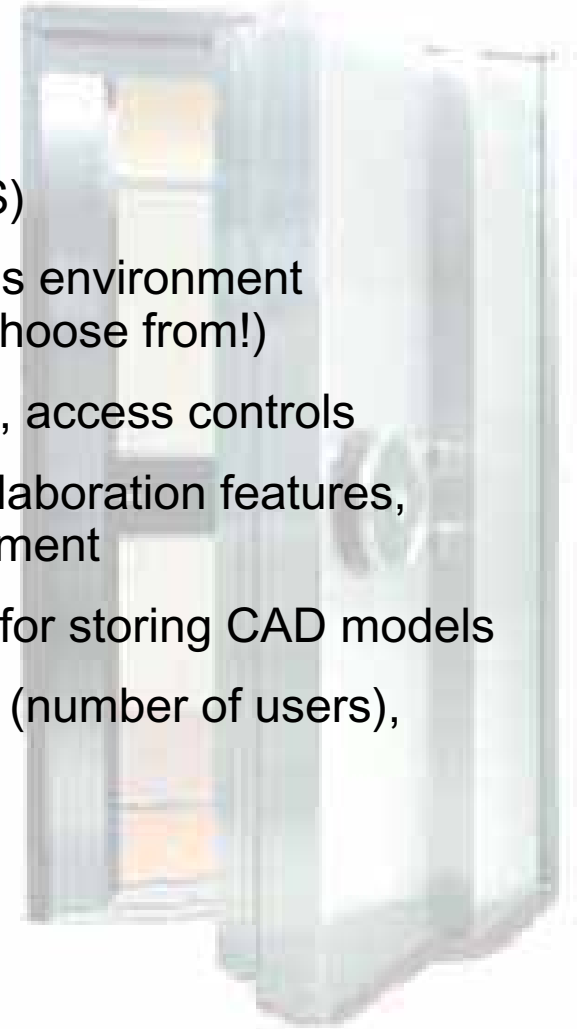


## Content Management

---

### What tool to use?

- Storage and retrieval of content is vital
- May be a network share but more likely a database (CMS)
- Need to select a CMS which is the best fit for the business environment (Easier said than done - there are many, many CMS' to choose from!)
- Basic features: check-in/check-out, versioning, searching, access controls
- Advanced features: **XML support**, workflow features, collaboration features, rendition services, auditing/reporting, translation management
- Other features: eg. DAM for images/illustrations, support for storing CAD models
- Other criteria: cost, hardware requirements, performance (number of users), ease of maintenance, vendor support

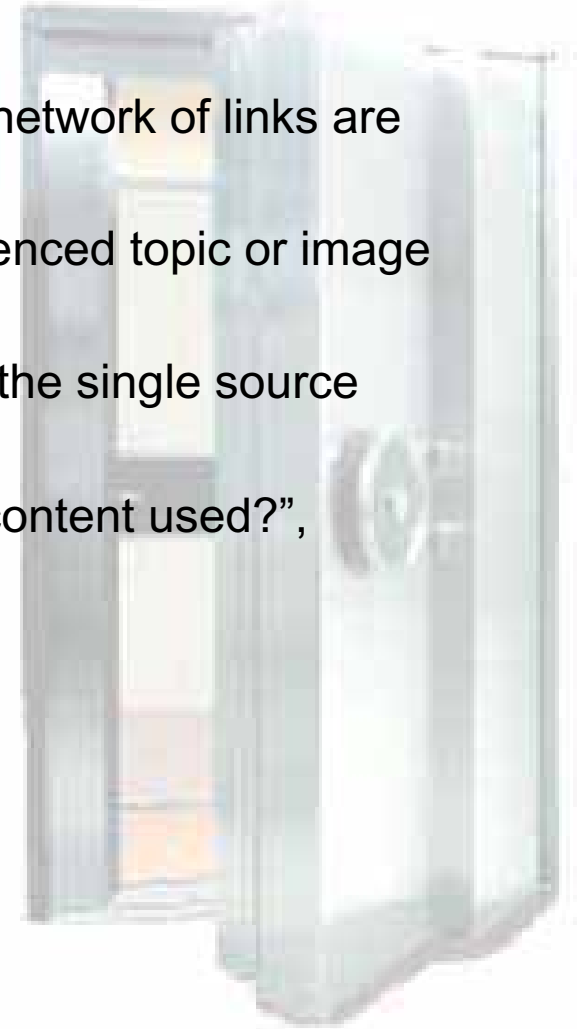


## Content Management

---

### How to maintain referential integrity?

- When storing content as reusable fragments, a complex network of links are built between content fragments
- Need to validate the links - what happens when the referenced topic or image has gone missing?
- Referential integrity features are desirable at all levels of the single source system (authoring, storage, publishing)
- This feature should allow queries such as “where is this content used?”, or “what other content does this content use?”



## The End

---

Questions?

