

Building Bridges

Integrative publishing solutions with Plone.
From storages to converters.



Andreas Jung
@MacYET
info@zopyx.com

Plone Conference 2015 Bucharest

/about

20 years in publishing
business since 1995

Integrator

Building generic and
unified solutions

Always interested in
alternative high-quality
components besides
the mainstram





Agenda

- **Storages and services**
 - Integration and federation of external storages and web services into Plone
- **Documents and formats**
 - converters ($A \rightarrow B$)

Plone as Publishing Platform

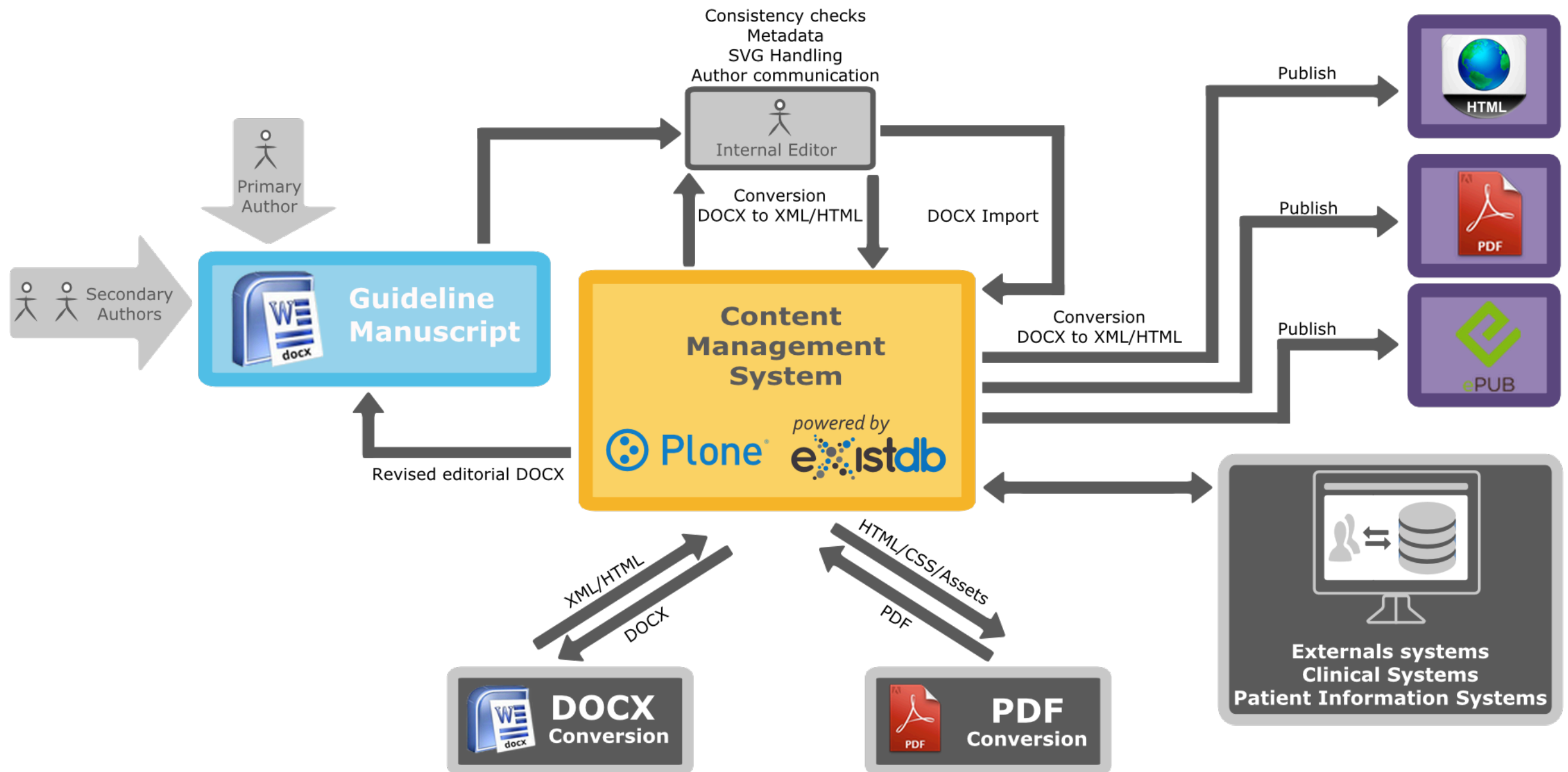
- **Pros**

- Secure
- Workflows
- Extensible

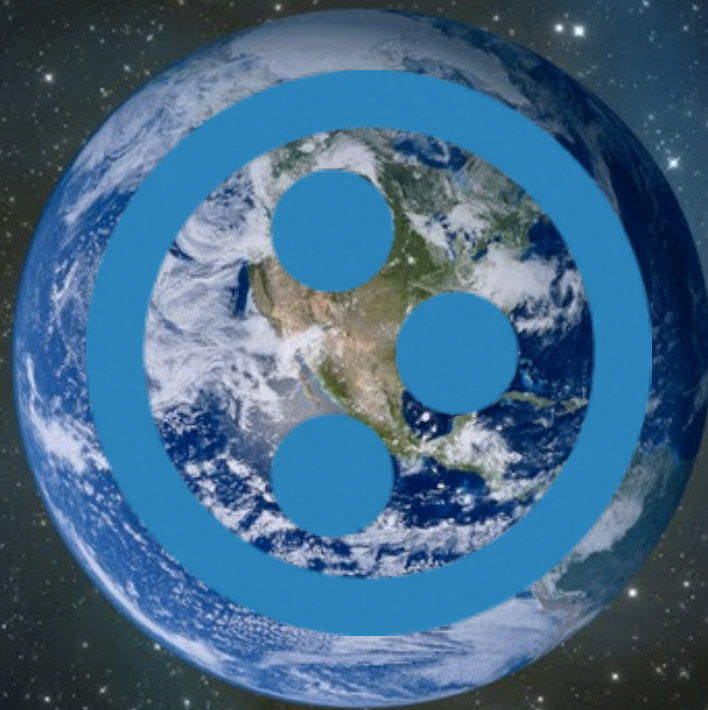
- **Cons**

- self-contained universe (ZODB)
- lack of decent integration with external data sources, cloud storages and cloud services besides relational databases
- focused on HTML as content format (in addition to binary data and assets)

A typical publishing workflow



Our Publishing Universe



Our Publishing Universe



ePUB



EPUB



LATEX



A group of kangaroos is shown in a grassy field under a clear blue sky. There are four kangaroos visible: one on the left, one in the center foreground, one in the center background, and one on the right. The kangaroos are of various shades of brown and grey. The text "(Cloud) storages and web services" is overlaid in the center of the image.

(Cloud) storages and web services

External/cloud storages in Plone

- Current state:
 - Reflecto
 - RDBMS (SQLAlchemy)
 - Dexterity content only stored in ZODB (no dedicated storage layer)
 - Archetypes external storages
- poor integration story
- different integration approaches, different APIs
- most add-ons unmaintained



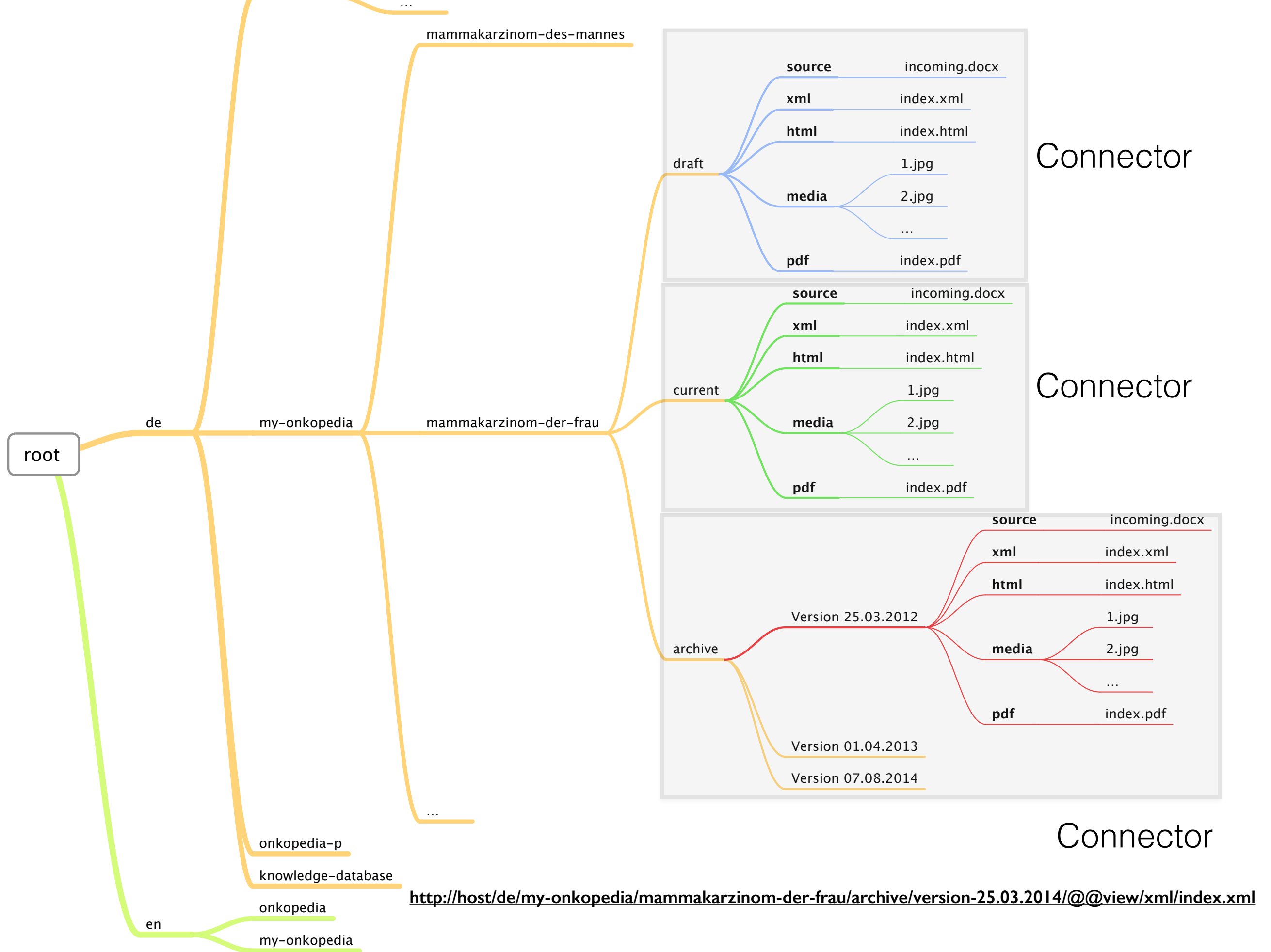
- Plone 4.3/(5.0 compatible) add-on for the integration of other storage systems (other than ZODB) into Plone
- Part of our **XML** publishing toolbox
- Can be used **without** the "**XML**" stuff
- **Unified access** and **API** to external storages and services
- Available modes
 - **Mounting**
 - **Dexterity support**
- 100 tests, Plone 4.3/5.0 against 6 different storage backends



XML Director - Mounting



- Plone "**Connector**" content-type
- parameters: connection URL, username, password
- acts as a *mountpoint*
- URL traversal support
- ZIP import/export, multi-file upload
- basic UI for creating/renaming/deleting collections/folders and resources
- simple view registry
- ACEditor integration for common formats
- minimal, small and extensible
- **no indexing support**
- **no proxy object magic as in Reflecto**
- **intended for applications that need to access external data sources and storages**



XML Director - Dexterity

- three new Dexterity fields
 - **XMLText** (stores, validates XML) with ACEditor widget
 - **XMLImage, XMLFile**
 - **XPath**
- content stored on the configured storage
- flat storage hierarchy based on UID
- dedicated set() and get() methods (due to lack of a DX storage API) as data managers
- DX behaviors not applicable here
- (we need a Dexterity storage API or some wrapper in plone.api)

```
xml_text = XMLText()  
xml_image = XMLImage()  
  
obj.set_xml('xml_text', xml)  
obj.set_xml('xml_img', img_bin)  
  
xml = obj.get_xml('xml_text')  
img_bin = obj.get_xml('xml_img')
```


pyfilesystem

- **abstraction layer** on top of storages, access through a uniform API
- Python 2/3 compatible
- various filesystem/webservices drivers
- **Goal:** your code must not know about the underlying storage system. The backend is just a configuration option.
- extensible (writing a new driver is straight forward)
- sandboxed filesystem operations
- OOTB support for: WebDAV, S(FTP), RPCFS, OSFS, S3, ZIP, Memory, MultiFS, WrapFS

```
handle = fs.opener(some_url)
with handle.open('foo', 'w') as fp:
    fp.write(data)

handle.listdir(dirname)

handle.mkdir('foo/bar/test')

handle.removedir('foo/bar/test')

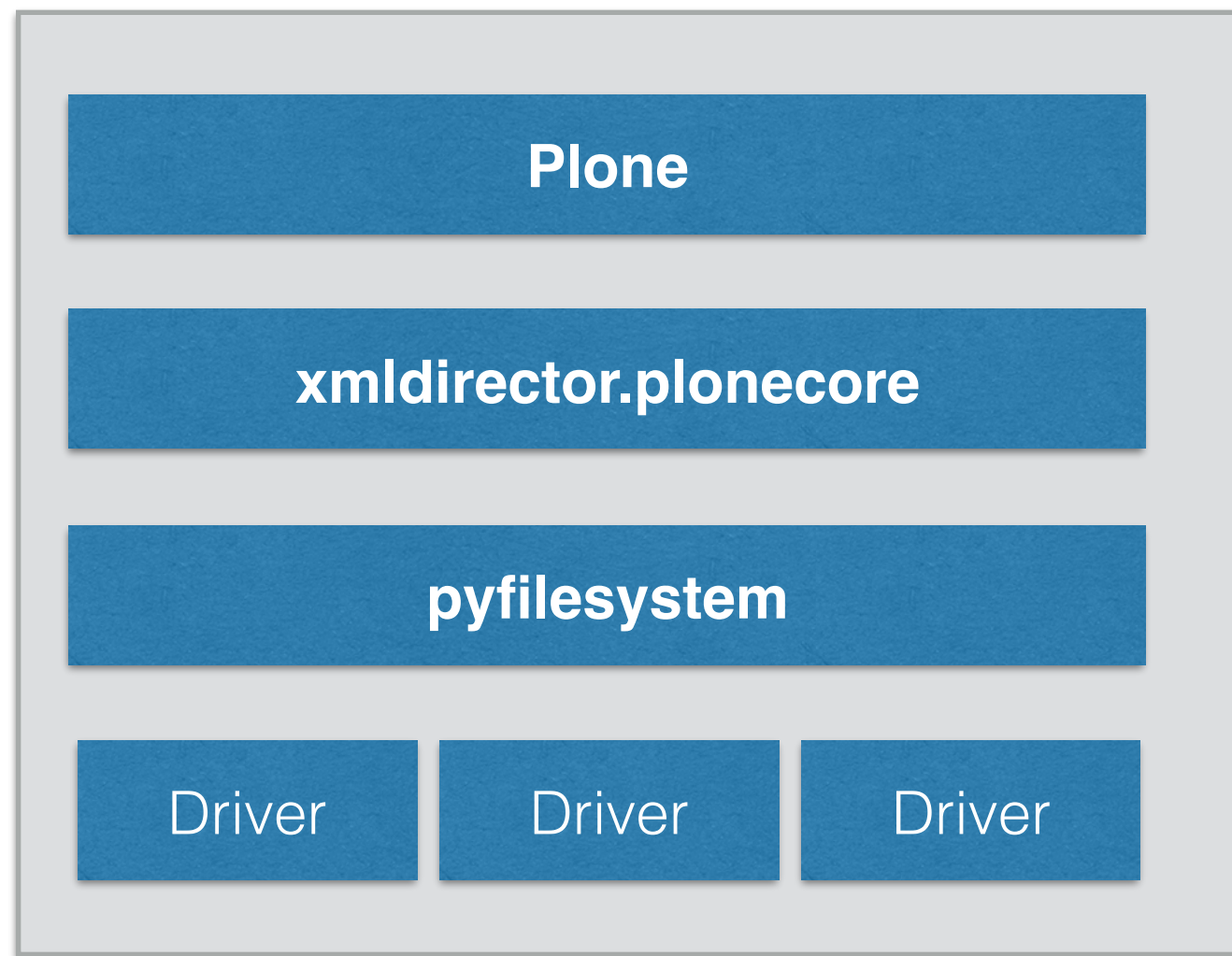
handle.exists(some_filename)

handle.isfile(some_name)

handle.move(src, dst)

handle.copy(src, dst)

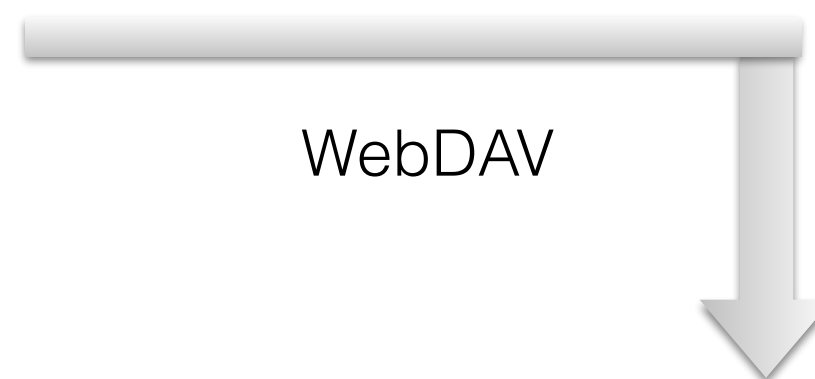
... .
```

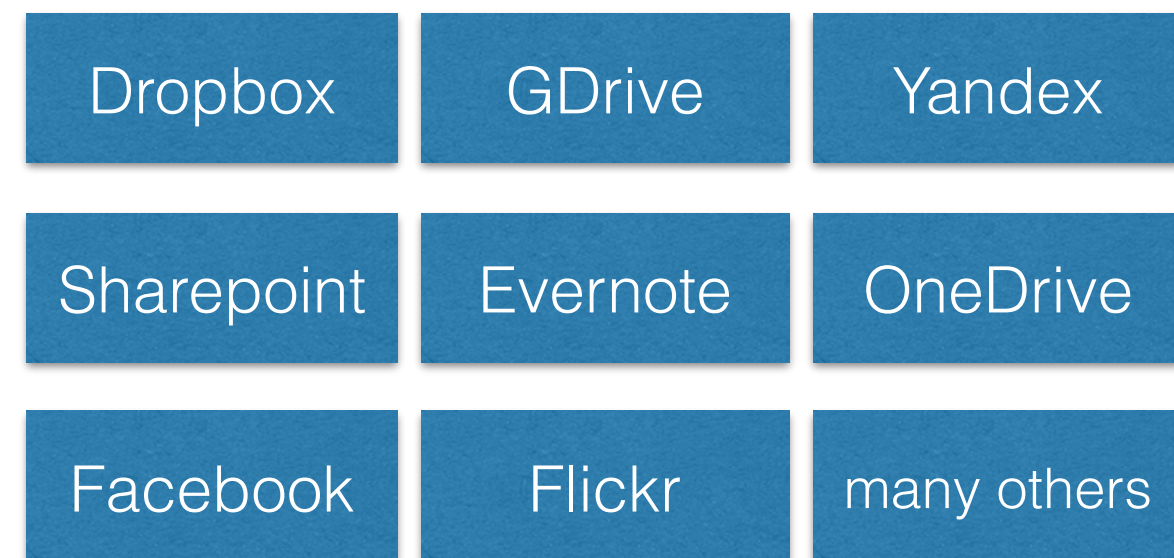
Your setup ↓ native protocols



Architecture



SaaS setup ↓ native protocols



pyfilesystem driver options

Storage/ Web Service	self-hosted (Privacy)	via external SaaS Bridge (limited privacy?)
WebDAV (Owncloud, BaseX, eXist-DB, Alfresco, etc.)	YES	YES
Amazon S3	YES	YES
Local filesystem	YES	NO
Dropbox	(YES, auth token issues)	YES
FTP/SFTP	(YES, V1.4)	YES
4Shared ADrive Alfresco Amazon Cloud Amazon S3 Box CloudMe Copy Cubby Digital Bucket DriveOnWeb Dropbox Dump Truck Evernote FTP Fabasoft Facebook FilesAnywhere Flickr GMX.DE Google Drive HiDrive Huddle LiveDrive Mediacenter MyDrive OneDrive Online FileFolder OwnCloud Picasa SugarSync TrendMicro SafeSync Web.de WebDAV Yandex	NO	YES

Supported services through 3rd party services (example)



Amazon Cloud



Box



Dropbox



FTP



WebDAV



Web.de



Google Drive



Flickr



Evernote



GMX.DE



HiDrive



Digital Bucket



Picasa



Facebook



Copy



Cubby



SugarSync



Amazon S3



MyDrive



Dump Truck



4Shared



CloudMe



Yandex



OwnCloud



Mediencenter



Online FileFolder



Alfresco



Huddle



FilesAnywhere

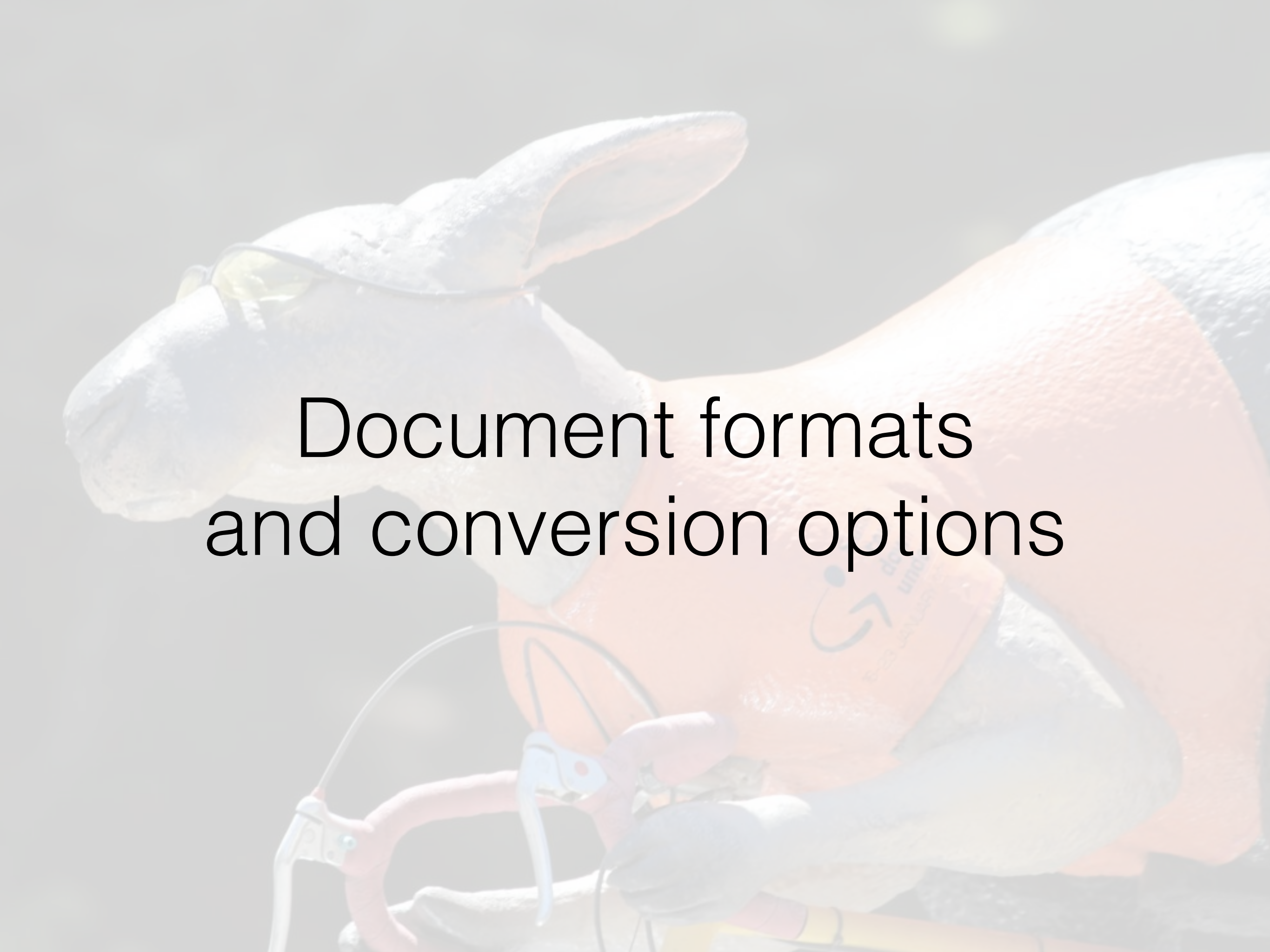


OneDrive



<https://pypi.python.org/pypi/xmldirector.plonecore/1.3.0b1>



A 3D rendered scene featuring a white dog-like character wearing yellow goggles and a life preserver, and a pink rabbit-like character holding a blue tool. The background is a large, textured orange life preserver. The text "Document formats and conversion options" is overlaid in the center.

Document formats and conversion options

Structured data

Structured content

Professional Publishing

Document
relations

Metadata



```
<Books>
  <Book ISBN="0553212419">
    <title>Sherlock Holmes: Complete Novels...
    <author>Sir Arthur Conan Doyle</author>
  </Book>
  <Book ISBN="0743273567">
    <title>The Great Gatsby</title>
    <author>F. Scott Fitzgerald</author>
  </Book>
  <Book ISBN="0684826976">
    <title>Undaunted Courage</title>
    <author>Stephen E. Ambrose</author>
  </Book>
  <Book ISBN="0743203178">
    <title>Nothing Like It In the World</title>
    <author>Stephen E. Ambrose</author>
  </Book>
</Books>
<\Books>
  <\Book>
    <author>Stephen E. Ambrose<\author>
    <title>Nothing Like It In the World<\title>
    <Book ISBN="0743203178">
```

- Industry standard in publishing
- structured data
- structured content
- not many alternatives besides Indesign stuff...

Advantages of XML

- XML structure definition
 - Document Type Definition (DTD)
 - XML Schema (XSD)
 - RelaxNG
- XML business rules
- Schematron

```
<?xml version="1.0"?><!DOCTYPE books [  
  <!ELEMENT title (#PCDATA)>  
  <!ELEMENT author (#PCDATA)>  
  <!ELEMENT authors (author)+>  
  <!ELEMENT subject (#PCDATA)>  
  <!ATTLIST subject class CDATA "">  
  <!ELEMENT book (title,authors,subject)>  
  <!ATTLIST book  
    bookid CDATA #REQUIRED  
    pubdate CDATA #REQUIRED>  
<books name="My books">  
  <book bookid="1" pubdate="03/01/2002">  
    <title>Java Web Services</title>  
    <authors>  
      <author>John Doe</author>  
      <author>Jane Doe</author>  
    </authors>  
    <subject class="novel">The Great Gatsby</subject>  
  </book>  
</books>
```

```
<xs:complexType>  
  <xs:sequence>  
    <xs:element name="SupplierId">  
      <xs:annotation>  
        <xs:documentation>+++ BESCHREIBUNG: Ihre ID</xs:documentation>  
      </xs:annotation>  
      <xs:simpleType>  
        <xs:restriction base="xs:string">  
          <xs:minLength value="1"/>  
          <xs:maxLength value="10"/>  
          <xs:pattern value="[A|P|V|K] [0-9]{9}"/>  
        </xs:restriction>  
      </xs:simpleType>  
    </xs:element>  
    <xs:element name="Timestamp" type="xs:dateTime">  
      <xs:annotation>  
        <xs:documentation>+++ BESCHREIBUNG: Zeitpunkt</xs:documentation>  
      </xs:annotation>  
    </xs:element>  
    <xs:element name="Amount" type="xs:integer">  
      <xs:annotation>  
        <xs:documentation>+++ BESCHREIBUNG: Betrag</xs:documentation>  
      </xs:annotation>  
    </xs:element>  
  </xs:sequence>  
</xs:complexType>
```

XML transformations

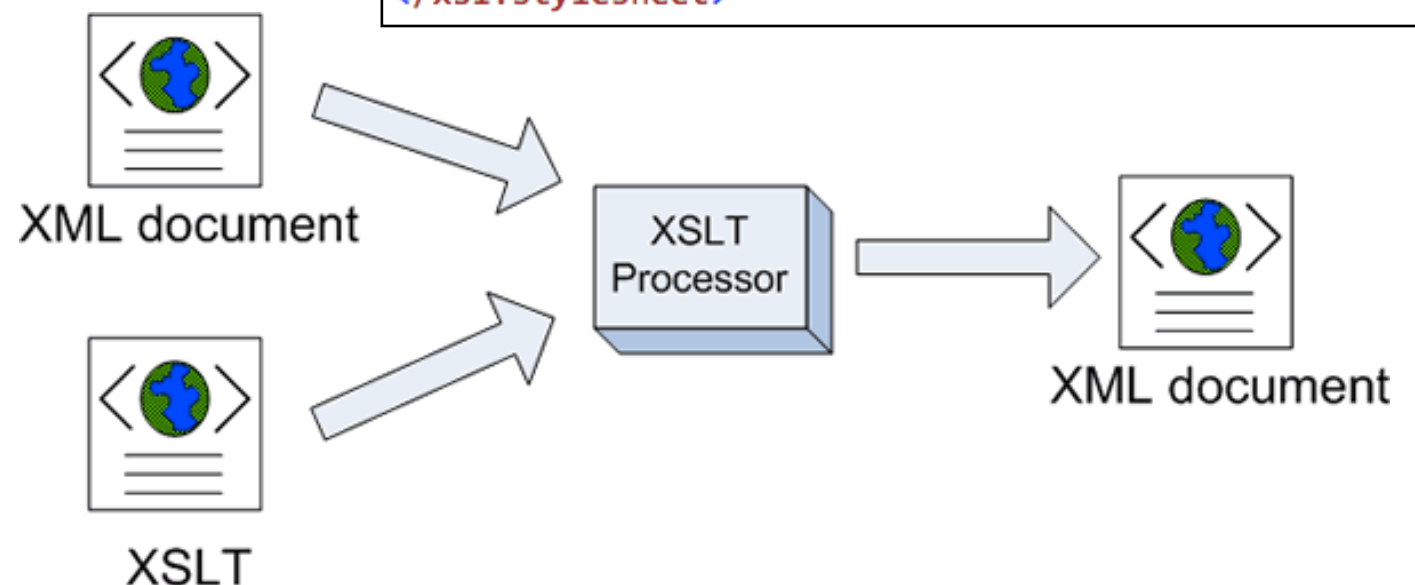
- Transformations
 - XSLT (version 1-3)
 - rule based language
 - Transformation between XML dialects
- or Python
- or

```
<?xml version="1.0" encoding="UTF-8"?>

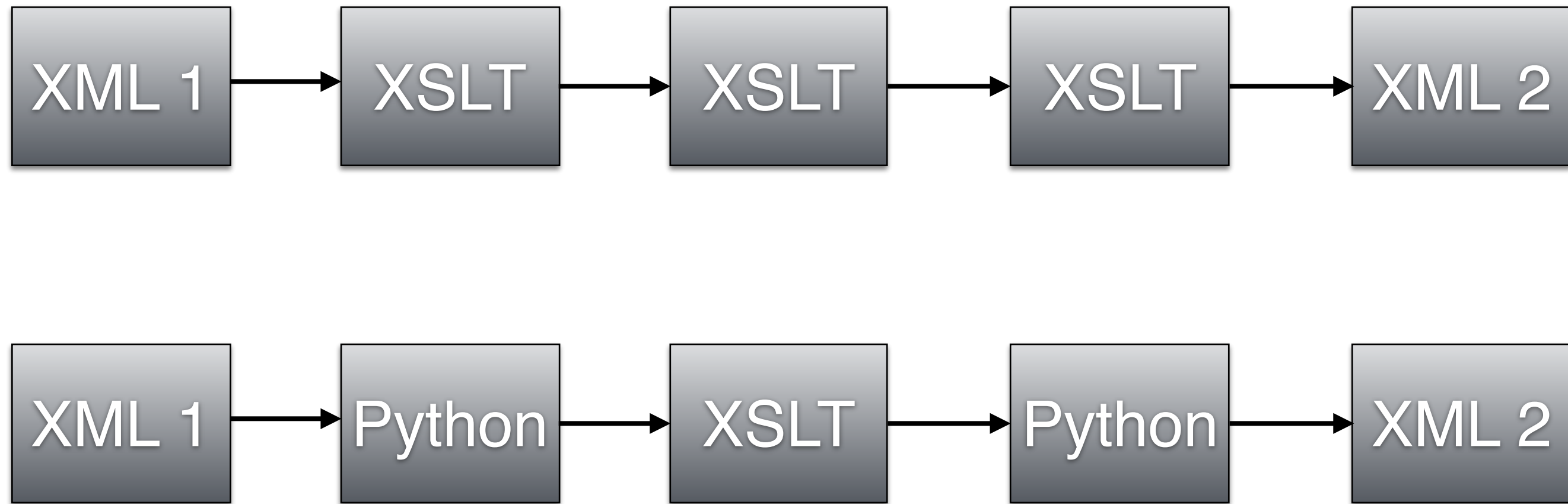
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
  <h2>My CD Collection</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
      <th>Title</th>
      <th>Artist</th>
    </tr>
    <xsl:for-each select="catalog/cd">
      <tr>
        <td><xsl:value-of select="title"/></td>
        <td><xsl:value-of select="artist"/></td>
      </tr>
    </xsl:for-each>
  </table>
  </body>
  </html>
</xsl:template>

</xsl:stylesheet>
```



XML transformation pipelines





Format: DOCX

- DOCX is XML but the same crap as .DOC on a different level
- all DOCX converters suck in their own special ways
- dedicated Word templates require dedicated converters and special treatment
- usually converted to some XML dialect (e.g. Docbook 4/5)
- **Tools**
 - past: **LibreOffice/OpenOffice** (HTML)
 - currently: **c-rex.net** (dedicated XML schema)
 - others: **Transpect (Le-Tex)**



Format: DITA

- DITA = Darwin Information Typing Architecture
- XML model for authoring
- defacto industry standard for technical documentation
- focus on content reuse
- Information typing: Task, Concept, and Reference
- key concepts: topics and maps
- extensive metadata and specialization
- **Tools**
 - **DITA toolkit** for publishing (HTML, PDF, ODT, Docbook)
 - **XMLMind Ditac**



Format: HTML5

- HTML5 as primary source for quality publishing (vs. XML)?*questionable*
- semantic elements **<article>**, **<section>**, **<header>**, **<figure>**, **<nav>**...
- freedom of structure (HTML5) vs. enforced structure and semantics (XML)
- not really suitable for professional high-quality publishing (seen differently by others)
- often used as intermediate format for CSS Paged Media with XML as primary format

PDF



Format: PDF (1/2)

- traditional: XML → XSL-FO → PDF
- CSS Paged Media: HTML + CSS → PDF
- Tools (you get what you pay for, better quality=higher price)

free

\$\$\$\$



- **WKHTMLTOPDF** (free), **Weasyprint**

- **PDFReactor** (RealObjects)

- **PrinceXML** (Prince)

- **PDFChip** (Callas Software)

- **Antennahouse V6.2** CSS Formatter

- Plone integration via Produce & Publish Plone Client Connector, [collective.sendaspdf](#), [abstract.wkhtmltopdf](#), [eea.pdf](#)



Format: PDF (2/2)

- new project: **Vivliostyle** (open-source + commercial)
- "One Source Multi-Use for making eBooks, Web, and Print books"
- based on EPUB Adaptive Layout implementation
<http://www.idpf.org/epub/pgt/>
- fixes many limitations of the CSS Paged Media approach and EPUB limitations



Web (PC, Tablet, Smartphone, etc.)



EPUB ePUB

PDF, Print



Format: ODF

- ODF is completely irrelevant in the publishing world (DOCX is (still) king)
- **Tools:**
 - **Pandoc**
 - **OpenOffice**
 - **LibreOffice**

Format: TeX/LaTeX

- Perfect for text-oriented layouts
- unusable for complex layouts
- **Tools:**
 - **ftw.book**
 - **Pandoc**
 - **Transpect**

Format: E-Books (1/2)

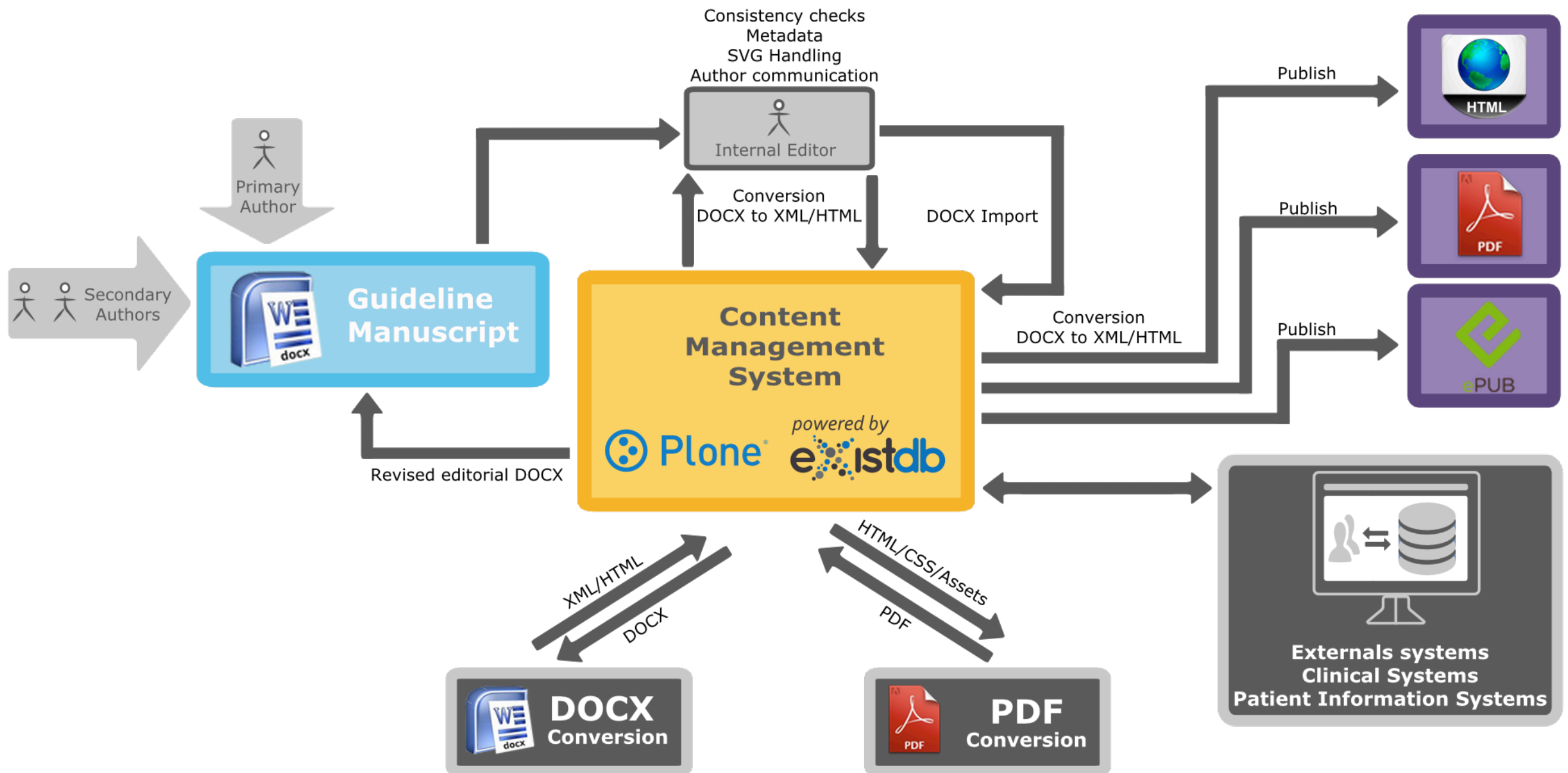
- different ebook formats:
EPUB, EPUB3, Mobi, KF8, Apple's EPUB3
- different hardware and software readers:
Kindle, iOS & Android devices, Kobo, Toliono, Sony, Nook
- fixed format ebooks vs. reflowable ebooks vs. adaptive layouts
- many limitations regarding typography, handling of images and tables

HUGE MESS

Format: E-Books (2/2)

- **Tools:**
 - **Calibre** (Python)
 - **eea.epub**
 - **Produce & Publish server** (via Calibre)
 - **web services like "Bookalope"**

Plone as platform for publishing solutions





Questions?

XMLDIRECTOR
DOCX->XML->PDF/EPUB

www.xml-director.info
demo.xml-director.info
xmldirector.plonecore