

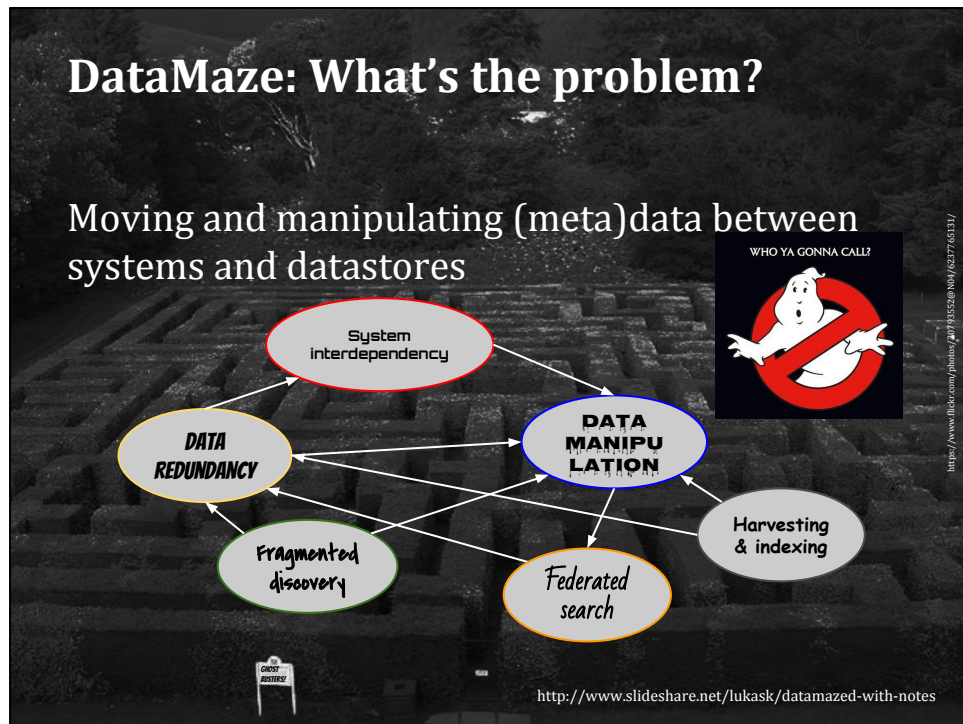
EAD, Primo and a data management hub

Lukas Koster

Library of the University of Amsterdam

IGeLU 2016

Developers Day



The problem is the heterogeneous and complicated information infrastructure of the library.

Huge amounts of time and effort are spent on moving data from one system/database to another and shoehorning one record format into the next, only to fulfill the necessary everyday services of the university library.

This creates critical system interdependencies, data redundancies and proprietary data transformation configurations.

See my earlier ELAG 2015 presentation "DataMazed".

ETL: Extract Transform Load

Transformation is mostly done with utilities that belong to a specific system/tool.

Proprietary transformation and storage

Multiple data formats and transformation tools

The process of getting data from one system/database to another is referred to as ETL: Extract - Transform - Load.

All three parts are usually executed with proprietary, system-specific tools and utilities. ETL is the most time and resource consuming portion of the total data infrastructure management organisation.

DataMap - Data Exchange

This is only a small part, diagram of a subset of this DataMap repository. DataMap makes very visible, that there are numerous individual, parallel, partly overlapping system and/or data format dependent ETL processes.

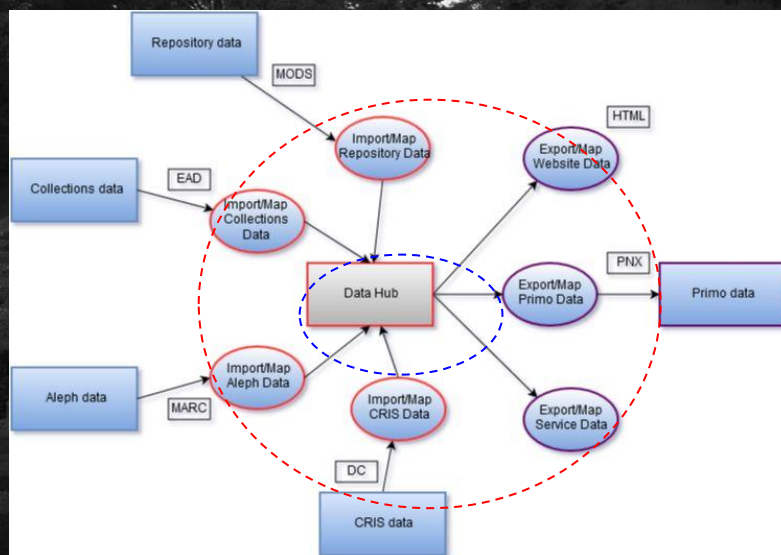
Dependency on systems provided by one or a few companies is sometimes referred to as “vendor lock-in”, something that we need to avoid, we are told. In reality however, there is not much else. We are always dependent on one or more system and database providers (not only commercial ‘vendors’, but also local developers and free and open source software communities). Better to speak of “systems lock-in” (also referred to as “silos”).

Anyway, from a system management and efficiency perspective ‘vendor lock-in’ appears to be better than the chaotic multi system and silo environment. This does not mean that you won’t have any problems, but you don’t have to solve them

yourself. From the innovation perspective however, this is not the case. But in my view there is not much difference here with a fragmented infrastructure.

It would however be great if we can free the data, in such a way as to minimize (not eliminate) these dependencies, which would also lead to more efficient and innovative investment of people, expertise and funding.

Alternative: central data hub



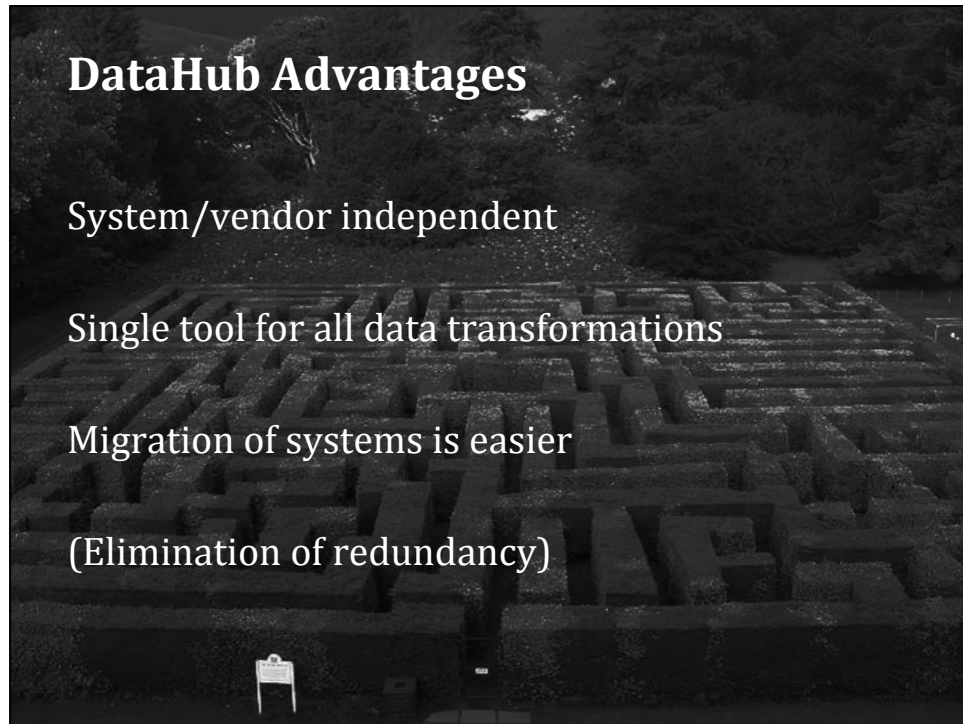
<http://commonplace.net/2015/08/maps-dictionaries-guidebooks/>

One of the possible alternatives, which we are testing now in a separate pilot project, is a central Data Hub, with some kind of universal and open data format. The idea is that all transformations are done using one universal tool, and all resulting data is stored and linked in one universal data format. The Extract and Load procedures may partly still be dependent on the source and target systems/databases/services.

One of the advantages would be that it will be a lot easier to migrate systems.

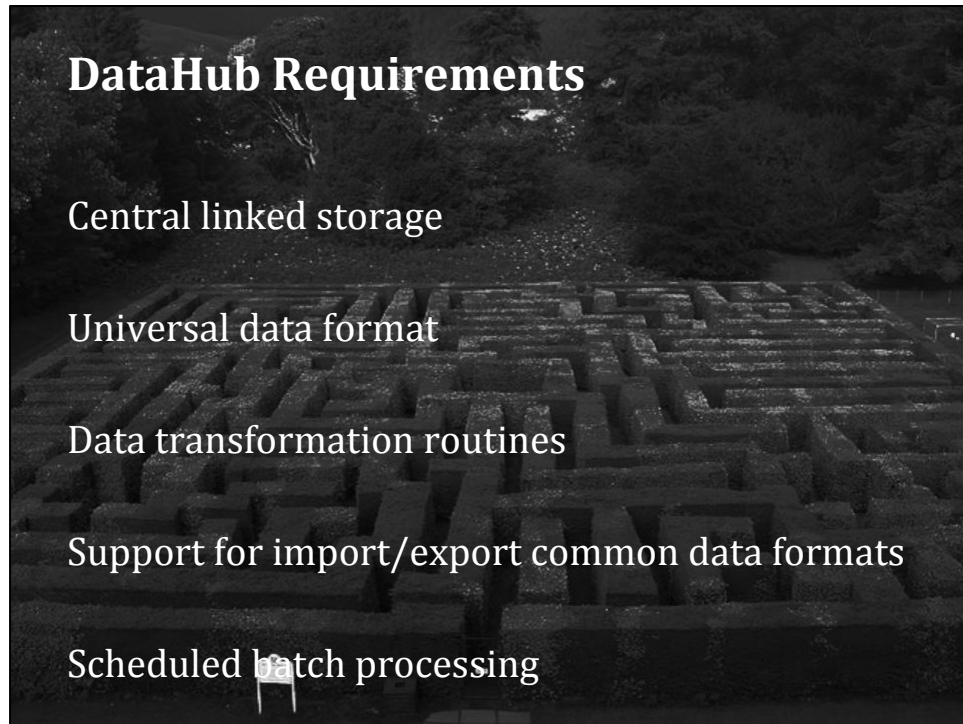
Not only existing legacy systems can then get the data from this hub, but also many new services.

A far-away future might be that libraries will use only one central data hub with sources and targets plugged in.



Theoretically, the advantages of such a universal Data Hub are:

- It is independent of systems/vendors (except of course of the Data Hub itself....)
- You only need knowledge of and expertise in one transformation tool
- Migration of systems would be easier because the data transformations will remain available, without having to redo these in the new systems
- On the long run: it should be possible to store all your data of all your content in one place, once, in a reusable format.



DataHub Requirements

Central linked storage

Universal data format

Data transformation routines

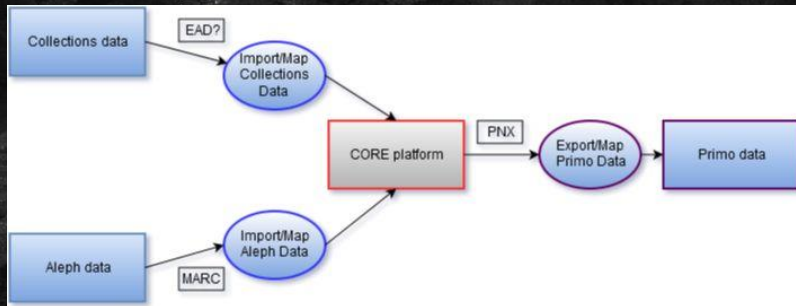
Support for import/export common data formats

Scheduled batch processing

As mentioned before, the requirements of a central Data Hub are:

- Central linked storage
- Universal data format
- Data transformation routines
- Support for import and export (in many ways) of common and proprietary data formats
- Last but not least: scheduling all these ETL tasks in batch: no need for manual fiddling

CORE Pilot: Use Case



*Connected Open Relations Exchange
Central Online Reference Engine
Controlled Object Registry Environment
Consolidated Ontology Range Ecosystem*

We started a pilot project, as mentioned.

Name: CORE (because everybody else already uses “HUB”, like our new library file sharing environment based on SharePoint). CORE can mean anything, whatever you like.

Focusing on a real life practical challenge: getting EAD formatted collection/archives data into Primo PNX, preferably linked to related information from our Aleph ILS.

The problem is: we’re still in the preliminary phase, testing tools etc. Also our collections system/database environment is changing. So there is not much to show. Only a description of the context, the issues and the possible routes.

EAD - Encoded Archival Description

Hybrid format:

- Collection/Archive level
- Nested sub-levels - Items

Focused on Describing, Displaying Collection structure

Variations within standard structure (MARC!)

EAD2 -> EAD3 Differences

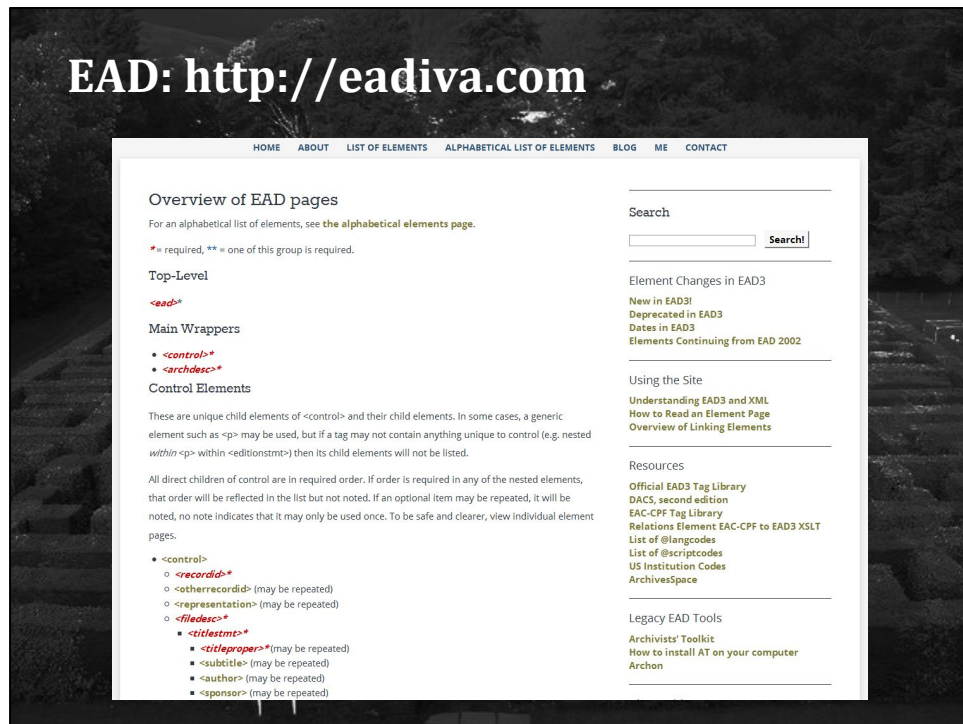
EAD is just like MARC a proprietary data format, targeted at just one type of content and just one type of display/ end user environment. Both are not suited for generic discovery tools like Primo. And here we have the core problem of our current library information infrastructure: Introverted systems. Legacy workflows.

Just like MARC, EAD is a display format that is incorrectly used as a storage format.

Actually, EAD does not describe one type of content (although it does for archivists) it is a hybrid format: both a single collection level in one EAD file, and multiple multi-level containers and objects, that make up the collection.

Just like with MARC (AACR2, RDA), you can have cataloguing rules, for instance ISAD(G). And just like with MARC every user can implement their own interpretation.

To make the situation complete, there is currently a major upgrade from EAD2 to EAD3, with a number of substantial differences.



Besides the official LoC EAD pages <https://www.loc.gov/ead/> I find this EADIVA site very helpful, maintained by Ruth Kitchin Tillman, of Notre Dame Hesburgh Library. EAD is written in XML, a hierarchical data format structure

Just like MARC it has a Control section, and then a Content section with separate sections for the Collection, and the Containers/Objects

EAD Collection Level

```
<?xml version="1.0"?>
+ <PUBLICATION id="Export1">
- <ead>
- <eadheader relatedencoding="dublin core" langencoding="iso639-2b" dateencoding="iso8601" countryencoding="iso3166-1">
  <eadid identifier="UBAInv297" encodinganalog="identifier" countrycode="nl">UBAInv297</eadid>
- <filedesc>
  - <titlestmt>
    - <titleproper encodinganalog="title">
      Inventaris van het archief van
      <lb/>
      Hugo de Vries
      <date normal="1866/1935">1866-1935</date>
    </titleproper>
    <author encodinganalog="creator">Erik Zevenhuizen, 2010</author>
  </titlestmt>
  - <publicationstmt>
    <publisher encodinganalog="publisher">
      Universiteitsbibliotheek Amsterdam
      <lb/>
      Universiteit van Amsterdam
    </publisher>
    <date encodinganalog="date" normal="2011">2011</date>
  </publicationstmt>
</filedesc>
- <profiledesc>
  + <creation>
  + <language>
    <language encodinganalog="language" langcode="dut">Nederlands</language>
  </language>
  <desrules>ISAD(G)</desrules>
</profiledesc>
</eadheader>
- <archdesc relatedencoding="dublin core" type="description" level="fonds">
  - <did>
    - <origination>
      <persname role="creator" encodinganalog="creator" normal="Vries, Hugo de">Hugo de Vries (1848-1935)</persname>
    </origination>
    <untititle encodinganalog="title" type="display" label="Vries, Hugo de (archief)">Archief van Hugo de Vries</untititle>
    <unitdate encodinganalog="coverage" normal="1866/1935" type="inclusive">1866-1935</unitdate>
  </physdesc>
  <extent>ca. 8 meter</extent>
</physdesc>
```

An example of an actual University of Amsterdam Collection EAD.

Some remarks:

- relatedencoding="dublin core", meaning "If only the <control> element (not the entire finding aid) is going to be mapped to a particular standard such as MARC or Dublin Code, this is where one would specify it."
- <lb/>: a pure display directive! "Line Break". Confuses Primo Normalization Rules!
- <desrules>ISAD(G)</desrules>: enumerates any "rules, standards, conventions, and protocols used in preparing the description." (Deprecated in EAD3, now: "Use [conventiondeclaration](#) or [localtypeddeclaration](#) for more controlled forms of citing conventions used in writing the description."

Bijzondere Collecties

[Home](#)
[Over ons](#)
[Bezoek & contact](#)
[Zoeken & raadplegen](#)
[Nieuws & agenda](#)
[Winkels, café & zalen](#)
[Steun ons](#)

Collecties en archieven

zoek in deze collectie:

[BC Home](#) / [Collecties](#) / [Archief van Hugo de Vries](#)

Inventaris van het archief van Hugo de Vries 1866-1935

Gebruik dit adres om naar deze collectie te linken: <http://dpc.uba.uva.nl/inventarissen/ubainv297>

overzicht

printversie

Herkomst

Hugo de Vries (1848-1935)

Titel

Archief van Hugo de Vries

Datering

1866-1935

Omvang

ca. 8 meter

Taal (materiaal)

Diverse talen, waaronder Nederlands, Duits, Frans, Engels en Spaans

Samenvatting

Archief van Hugo de Vries, hoogleraar plantkunde aan de Universiteit van Amsterdam 1878-1918. Bevat correspondentie, aantekeningen van wetenschappelijk onderzoek, materiaal betreffende publicaties en lezingen, eerbewijzen, herinneringen aan buitenlandse studiereizen, stukken over zijn colleges en excursies met studenten, alsmede over zijn directeurschap van de Hortus Botanicus. Het archief geeft inzicht in het werk van een Nederlands botanicus die wereldfaam verwierf met nieuwe inzichten in de mechanismen van evolutie en erfelijkheid.

Collectienummer

UBA297

Instelling

Universiteitsbibliotheek Amsterdam (UvA)

Bewaarplaats

Bijzondere Collecties

Inhoudsopgave

[Beschrijving van het archief](#)
[Beknopt overzicht](#)
[Herkomst en verwerving](#)
[Inhoud en samenstelling](#)
[Aanwijzingen voor de gebruiker](#)
[Verwant materiaal](#)
[Bijlagen](#)
[Gecontroleerde trefwoorden](#)
[Collectieonderdelen](#)
[1. Persoonlijk leven](#)
[2. Openbaar leven](#)






The same data, now in the (home grown) collection system display. In Dutch obviously.

The left side is based on the <eadheader>/<filedesc> and the <archdesc>/<did> etc. parts of the EAD file.

The right side (ToC) is based on the <archdesc> and multi-level container parts.

EAD: Levels

```

<dsc>
  <c01 level="series">
    <did>
      <daogrp>
        <daoloc href="NL-AmU_AF_UBAINV0297_001" linktype="locator">
          <daodesc>
            <p>Persoonlijk leven</p>
          </daodesc>
        </daoloc>
      </daogrp>
    </did>
  </c01>
  <c02 level="subseries">
    <did>
      <daogrp>
        <daoloc href="NL-AmU_AF_UBAINV0297_001" linktype="locator">
          <daodesc>
            <p>Overig</p>
          </daodesc>
        </daoloc>
      </daogrp>
    </did>
  </c02>
  <c03 level="file">
    <did>
      <daogrp>
        <daoloc href="NL-AmU_AF_UBAINV0297_001" linktype="locator">
          <daodesc>
            <p>Aanwijzingen en andere notities betreffende oude ordeningen van het archief.</p>
          </daodesc>
        </daoloc>
      </daogrp>
    </did>
  </c03>
</dsc>

```

An example of the multi-level EAD section:

<dsc> and nested <c0> - <c12> levels.

If the value of the "level" argument = "file" it describes an actual object.

The <daoloc href=" argument contains not a full link/URL/URI, but only the identifier part. The full link has to be constructed with some base-url.

Bijzondere Collecties

[Home](#)
[Over ons](#)
[Bezoek & contact](#)
[Zoeken & raadplegen](#)
[Nieuws & agenda](#)
[Winkels, café & zalen](#)
[Steun ons](#)

▼ Collecties en archieven

▼ zoek in deze collectie:

[BC Home](#) / [Collecties](#) / [Archief van Hugo de Vries](#)

Inventaris van het archief van Hugo de Vries 1866-1935

Gebruik dit adres om naar deze collectie te linken: <http://dpc.uba.uva.nl/inventarissen/ubainv297>

▼ overzicht

▼ printversie

Collectieonderdelen

1.. Persoonlijk leven

1.1.. Overig

1. z.d. (20ste eeuw), 1 omslag

Aanwijzingen en andere notities betreffende oude ordeningen van het archief.

1.2.. Familie- en gezinsleven

2. 1907, 1929, 1934 en z.d., 2 omslagen

Foto's en negatieven van De Vries en E.L. de Vries-Egeling, gemaakt in onder andere de Hortus Botanicus te Amsterdam. De afdrukken zijn alle aangetroffen in: UBA-Artisbibliotheek: Archief Theo J. Stomps, in een doos met opschrift 'Persoonlijke herinneringen aan prof. Hugo de Vries'. De kunststof negatieven zijn aangetroffen in inv. no. 87. De foto van Wegner en Mottu werd

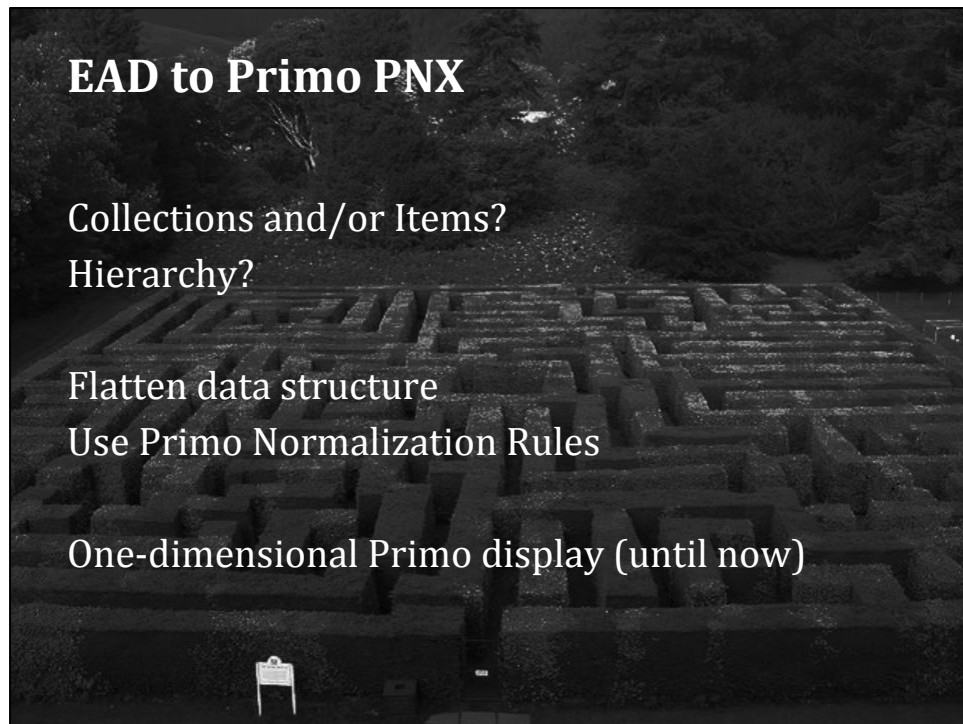
Inhoudsopgave

[Beschrijving van het archief](#)
[Beknopt overzicht](#)
[Herkomst en verwerving](#)
[Inhoud en samenstelling](#)
[Aanwijzingen voor de gebruiker](#)
[Verwant materiaal](#)
[Bijlagen](#)
[Gecontroleerde trefwoorden](#)

[Collectieonderdelen](#)

[1. Persoonlijk leven](#)
[2. Openbaar leven](#)

The EAD data from the multi-level section, first top level: in the collection system display.



In current Primo there is basically a flat record structure, describing individual items. There is a very simple Whole/Parts implementation (parts of a serial) with linked internal ID's.

Getting the collection level into Primo is relatively easy. But that doesn't apply to the multi-level structure.

It is not possible to use Primo Normalization Rules to traverse all XML levels, as far as I know. Most libraries that harvest EAD objects into Primo first flatten the EAD-files to some other format (MODS, etc.).

EAD in Primo: Collection Level

Inventaris van het archief van Hugo de Vries
Erik Zevenhuizen
Universiteitsbibliotheek Amsterdam Universiteit van Amsterdam 1866-1935 Hugo de Vries (1848 1935) UBAinv297 UBA297
[Online access](#)

Details Reviews & Tags

Title: Inventaris van het archief van Hugo de Vries

Author: Erik Zevenhuizen

Subjects: Plantkunde;
Erfelijkheid;
Evolutietheorie;
Botany;
Heredity;
Evolution theory;

Description: Archief van Hugo de Vries, hoogleraar plantkunde aan de Universiteit van Amsterdam 1878-1918. Bevat correspondentie, aantekeningen van wetenschappelijk onderzoek, materiaal betreffende publicaties en lezingen, eerbewijzen, herinneringen aan buitenlandse studiereizen, stukken over zijn colleges en excursies met studenten, alsmede over zijn directeurschap van de Hortus Botanicus. Het archief geeft inzicht in het werk van een Nederlands botanicus die wereldfaam verwierf met nieuwe inzichten in de mechanismen van evolutie en erfelijkheid.
Archive of Hugo de Vries, professor at the University of Amsterdam from 1878 until 1918. Contains correspondence, notes of his scientific research, documents concerning publications and lectures, tokens of honour, souvenirs from his travels abroad, documents relating to his lectures for university students, and documents concerning his directorship of the Hortus Botanicus of the University of Amsterdam. The archive illustrates the work of a Dutch botanist who gained worldwide fame with novel insights into the mechanisms of speciation and heredity.

Publisher: Universiteitsbibliotheek Amsterdam Universiteit van Amsterdam

Creation Date: 1866-1935

Format: ca. 8 meter

Language: Dutch
German
French
English
Spanish

Identifier: UBAinv297
UBA297

Source: Hugo de Vries (1848 1935)

Example of harvested Collection level EAD displayed in Primo.
Note again the <lb/> effect.

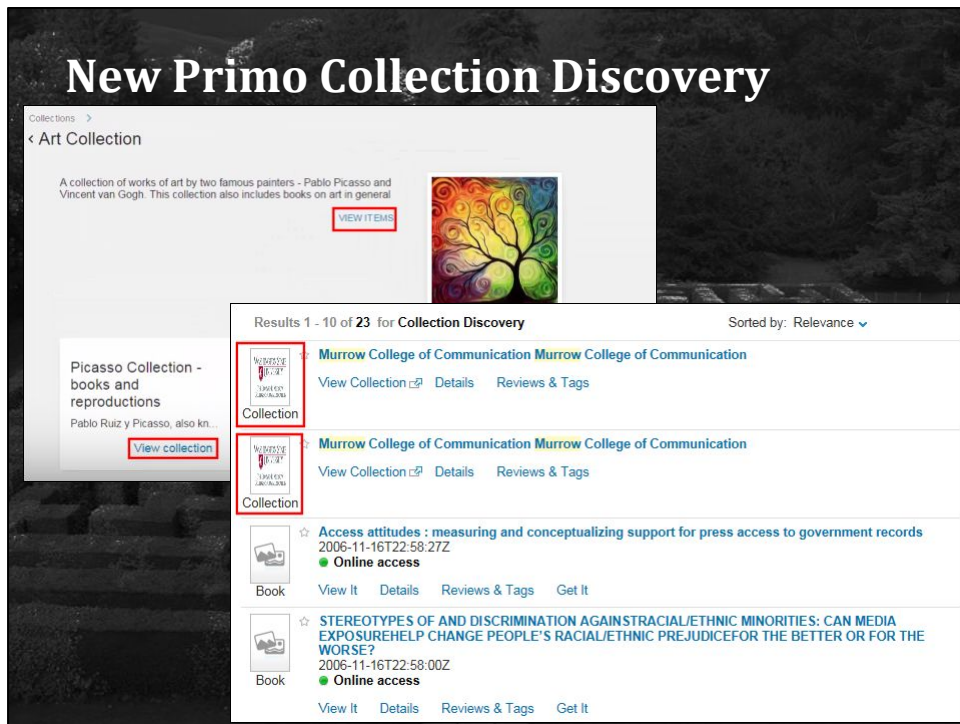
EAD in Primo - Collection: PNX

```

<record>
  <control>
    <sourcecrecordid>UBAIn297</sourcecrecordid>
    <sourceid>UVA_EAD</sourceid>
    <recordid>UVA_EADUBAIn297</recordid>
    <sourcesystem>Other</sourcesystem>
  </control>
  <control>
    <lb/>
  </control>
  <display>
    <type>collection</type>
    <title>Inventaris van het archief van Hugo de Vries</title>
    <creator>Erik Zevenhuizen</creator>
  </display>
  <publisher>
    Universiteitsbibliotheek Amsterdam</publisher>
  <creationdate>1866-1935</creationdate>
  <format>ca. 8 meters</format>
  <identifier>UBAIn297</identifier>
  <identifier>UBA297</identifier>
  <subject>Plantkunde</subject>
  <subject>Erfelijkheid</subject>
  <subject>Evoluтиetheorie</subject>
  <subject>Botany</subject>
  <subject>Heredity</subject>
  <subject>Evolution theory</subject>
  <description>
    Archief van Hugo de Vries, hoogleraar plantkunde aan de Universiteit van Amsterdam 1878-1918. Bevat correspondentie, notulen van buitenlandse studiereizen, stukken over zijn colleges en excursies met studenten, alsmede over zijn directeurschap van de Hortus Botanicus en de mechanismen van evolutie en erfelijkheid.
  </description>
  <description>
    Archive of Hugo de Vries, professor at the University of Amsterdam from 1878 until 1918. Contains correspondence, notes, documents relating to his lectures for university students, and documents concerning his directorship of the Hortus Botanicus and the mechanisms of speciation and heredity.
  </description>
  <description>
    <language>dut</language>
    <language>ger</language>
    <language>fre</language>
    <language>eng</language>
    <language>spa</language>
    <source>Hugo de Vries (1848-1935)</source>
  </description>
  <rights>
    Voor raadpleging is een bezoekers- of lezerspas van de Universiteitsbibliotheek Amsterdam vereist.
  </rights>
</display>

```

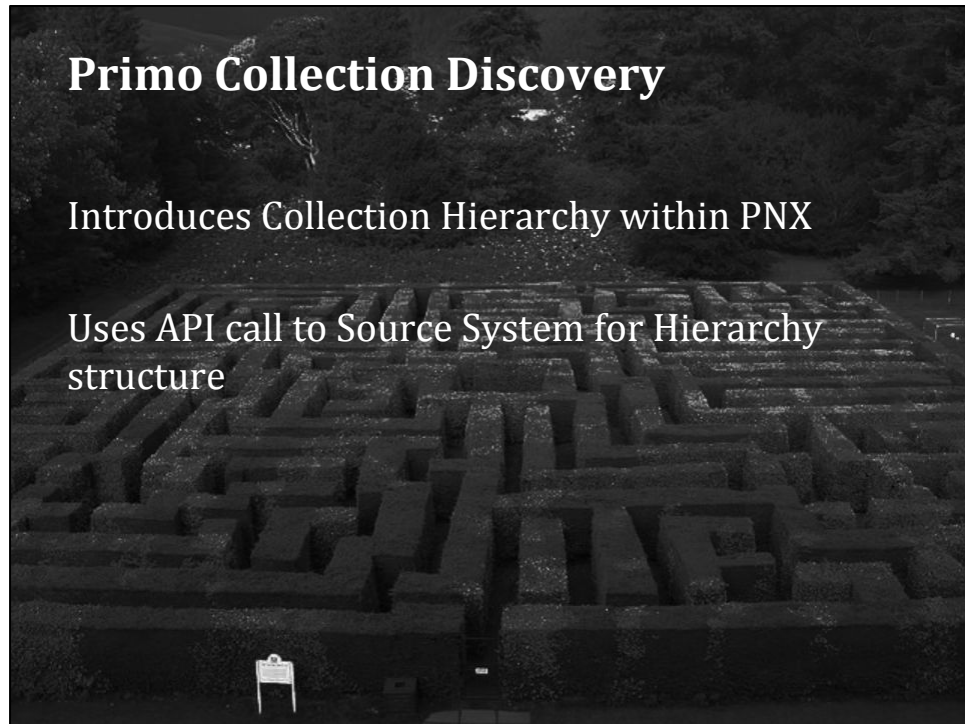
Example of the EAD file collection level in Primo PNX.
See the <lb/> tag that is ignored by Primo Normalization and can't be referenced. So it is just substituted by a NULL.



New in Primo: hierarchical Collection Discovery. Currently only for collection data from Alma and Rosetta. Soon also for third party source systems.

All items are available as individual PNX records, with a new PNX link field that links to the higher level object.

An additional Primo area ("Lobby") is available for displaying the Collections as units.



Primo Collection Discovery still uses the one-dimensional data structure in PNX, but adds display hierarchy using on the fly API calls to the source system.

Primo Collection Discovery: pnx fields

Section	Field name	Code	Description
Control	Collection Discovery	control/coldiscovery	<p>The field has 3 elements that enable Primo to display the collection path in the details tab based on the Collection Hierarchy API:</p> <p>\$\$T – type of record:</p> <ul style="list-style-type: none">- topcollection- collection.- item <p>\$\$D – the collection ID or, for item, the parent collection ID</p> <p>\$\$I – the Primo institution to which the collection/item belongs. Primo needs the institution to know which API to use.</p> <p>Note that an items can belong to more than one collection and can include more than one control/coldiscovery field. A collection should only have one.</p>
Search	Parent ID	search/cdparentid	A new index for the ID of the item's parent ID. Primo uses this index when the "View Items" link is invoked from the Collection Discovery Lobby

For example:

control/coldiscovery \$\$Titem\$\$D8115027510000121\$\$IEXLDEV1_INST

cdparentID 81451426600001021

Implementation of collection hierarchy in PNX.
Field "Collection Discovery": control/coldiscovery.
Holds Type of record (\$\$T), ID (collection or parent collection; \$\$D), Institution (\$\$I).

DataHub Requirements

Central linked storage

Universal data format

Data transformation routines

Support for import/export common data formats

Scheduled batch processing

For EAD/Collections in Primo:

API: hierarchy structure

To come back to our initial DataHub requirements: for our specific Primo Collection hierarchy need we have to add the option of getting the hierarchy structure through an API call.

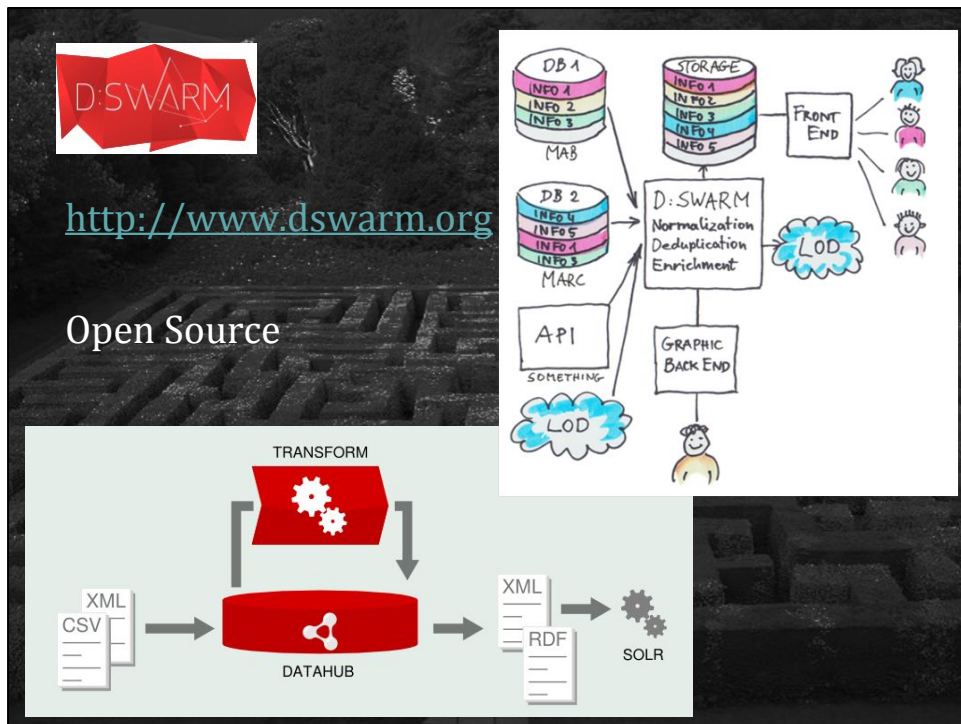


Now, that was the context and background. Next step was: looking for and selecting tools for implementing a Data Hub.

There are a number of tools and projects out there. Some of them dedicated for a specific project, like DPLA, DigitalNZ. Probably also Europeana has a tool.



We decided on looking closer at d:swarm and catmandu, because these tools are open source, not dedicated for one specific project, and relatively easy to use.



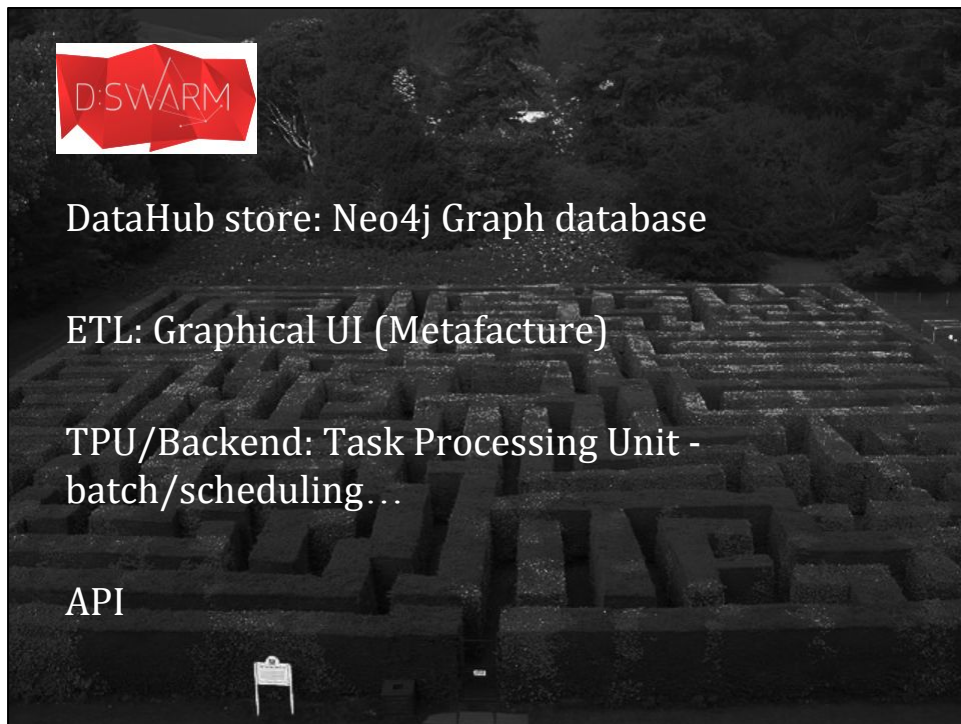
d:swarm is developed by SLUB (Saxon State and University Library) in Dresden, Germany, together with Avantgarde Labs).

Open Source.

It theoretically meets all the requirements for the data hub.

There are two versions:

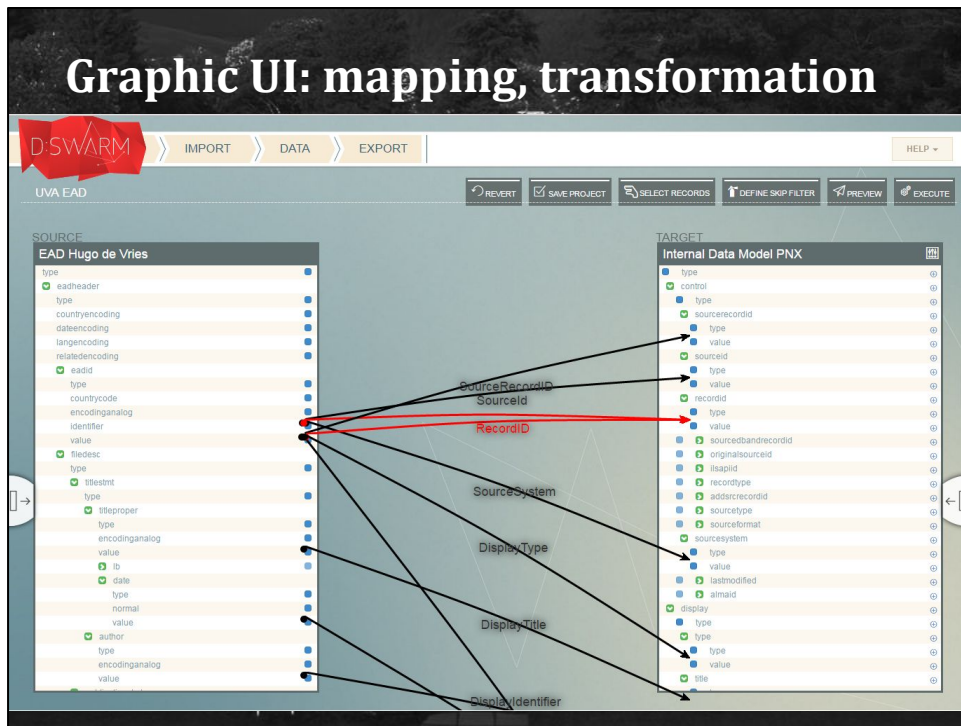
- Streaming: no data storage, just ETL
- DataHub: with data storage



Neo4J Graph database is used as the DataHub storage, where all imported data is transformed to graph relations, and linked where obvious.

For Mapping and Transformation of data the graphical UI Metafactory tool is used.

It has TPU (Task Processing Unit) for batch processing of ETL procedures. And there is an API.



The d:swarm team was so kind to provide us with a temporary virtual installation on their servers for free, for evaluating purposes.

Mapping of data from source to target is done in the graphical UI.
EAD can be handled as XML, with manual mapping.

Graphic UI: mapping, transformation

DSWARM | IMPORT | DATA | EXPORT | HELP

UVA EAD

SOURCE
EAD Hugo de Vries

MAPPING CONFIGURATION
SourceSystem DisplayPersonSubject **DisplayCreator**

DisplayCoverage DisplaySubjectSubject DisplayUseRights SourceId
 DisplayIdentifier DisplayPublisher DisplayDescription RecordID
 DisplayCreationDate DisplayFormat DisplayIdentifier2 DisplayTitle
 DisplaySource DisplayRights SourceRecordID DisplayCorpSubject
 DisplayGenreSubject DisplayLanguage DisplayType

eadheader → split → occurrence → display
 > filedesc >
 > author >
 > creator
 > value

Function: split
 Split string based on a regular expression. Pattern syntax corresponds to Java Regular Expressions.

delimiter

Regular expression, defining the split

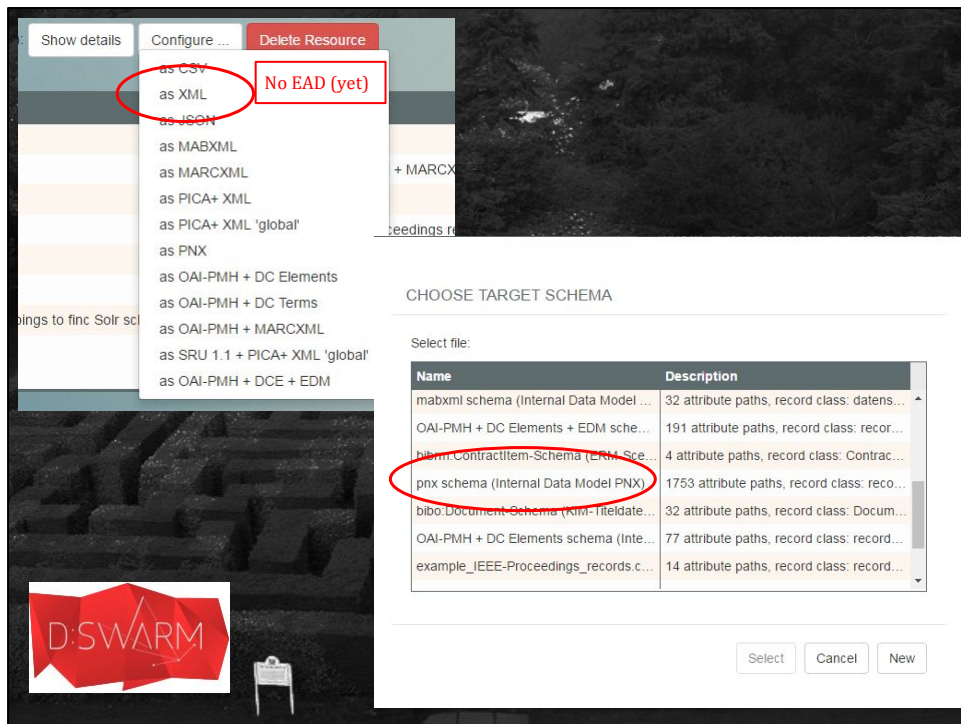
Close

Functions

- all
- any
- blacklist
- case
- choose
- collect
- combine
- compose
- concat
- constant
- convert-value
- count
- dateformat
- dewey
- equals
- htmlanchor
- http-api-request
- ifelse
- isbn
- issn
- lookup
- none
- normalize-urls
- not-equals
- numfilter
- occurrence
- parse-json
- regexlookup
- regex
- replace
- setreplace
- split

el PNX

Data transformation is done in the graphical UI, with adding and specifying routines, much like primo normalization rules.



Many common import and export data formats are available, although not EAD. But there is Primo PNX.



Catmandu

<http://librecat.org/Catmandu/>

Command line tool - Open Source

Items: data units

No EAD (yet)

Importers: CSV, JSON, MARC, XML, ALEPH, etc.

Exporters: MARC, JSON, RDF, CSV, ALEPH, etc.

No PNX(yet)

Stores: MongoDB, CouchDB, ElasticSearch, etc.

Fixes: transformations -

Catmandu, developed by University Libraries of Gent, Lund, Bielefeld. Open Source.
Fully command line operated.

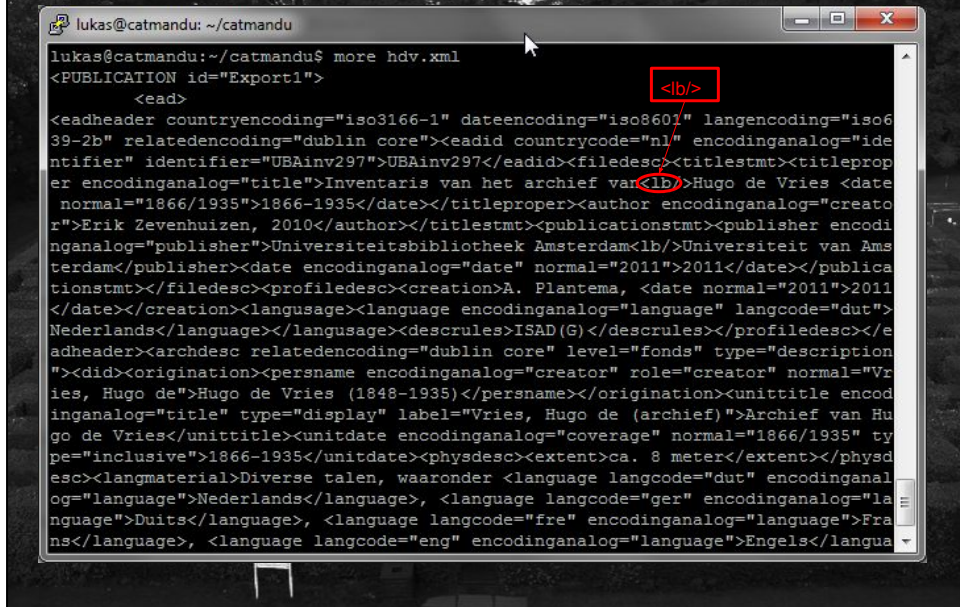
Consists of many optional modules.

Many common import and export data formats, although not EAD and Primo PNX.

Storage is available via a number of tools.

Data transformations are performed using “fixes”, small modules that perform a single operation.

Catmandu - EAD as XML



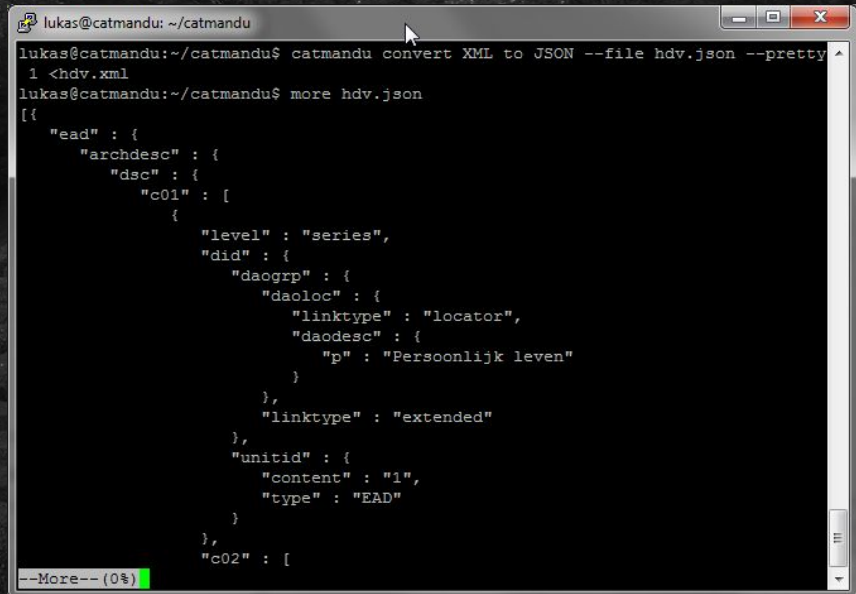
The screenshot shows a terminal window titled 'lukas@catmandu: ~/catmandu'. The user has entered the command 'more hdv.xml'. The terminal displays XML code for an EAD document. A red box highlights the closing bold tag '' in the title 'Inventaris van het archief van **Hugo de Vries**'. Another red circle highlights the opening bold tag '' in the same title. The XML code includes various EAD elements like <eadheader>, <eadid>, <filedesc>, <titlestmt>, <titleproper>, <date>, <author>, <publicationstmt>, <publisher>, <creation>, <language>, <descrules>, <archdesc>, <did>, <origination>, <persname>, <unittitle>, <unitdate>, <physdesc>, <extent>, <physdesc>, <langmaterial>, and <language>.

```
lukas@catmandu:~/catmandu$ more hdv.xml
<PUBLICATION id="Export1">
  <ead>
    <eadheader countryencoding="iso3166-1" dateencoding="iso8601" langencoding="iso6
39-2b" relatedencoding="dublin core"><eadid countrycode="nl" encodinganalog="ide
ntifier" identifier="UBAinv297">UBAinv297</eadid><filedesc><titlestmt><titleprop
er encodinganalog="title">Inventaris van het archief van Hugo de Vries <date
normal="1866/1935">1866-1935</date></titleproper><author encodinganalog="creato
r">Erik Zevenhuizen, 2010</author></titlestmt><publicationstmt><publisher encodi
nganalog="publisher">Universiteitsbibliotheek Amsterdam<b>Universiteit van Ams
terdam</publisher><date encodinganalog="date" normal="2011">2011</date></publica
tionstmt></filedesc><profiledesc><creation>A. Plantema, <date normal="2011">2011
</date></creation><language><language encodinganalog="language" langcode="dut">
Nederlands</language></language><descrules>ISAD(G)</descrules></profiledesc></e
adheader><archdesc relatedencoding="dublin core" level="fonds" type="description
"><did><origination><persname encodinganalog="creator" role="creator" normal="Vr
ies, Hugo de">Hugo de Vries (1848-1935)</persname></origination><unittitle encod
inganalog="title" type="display" label="Vries, Hugo de (archief)">Archief van Hu
go de Vries</unittitle><unitdate encodinganalog="coverage" normal="1866/1935" ty
pe="inclusive">1866-1935</unitdate><physdesc><extent>ca. 8 meter</extent></physd
esc><langmaterial>Diverse talen, waaronder <language langcode="dut" encodinganal
og="language">Nederlands</language>, <language langcode="ger" encodinganalog="la
nguage">Duits</language>, <language langcode="fre" encodinganalog="language">Fra
ns</language>, <language langcode="eng" encodinganalog="language">Engels</langua
```

It is really easy to install Catmandu core and additional modules, using CPAN.

Again, EAD is imported in catmandu with the XML Importer.

Catmandu - XML to JSON



```
lukas@catmandu: ~/catmandu
lukas@catmandu:~/catmandu$ catmandu convert XML to JSON --file hdv.json --pretty
1 <hdv.xml
lukas@catmandu:~/catmandu$ more hdv.json
[{"
  "ead" : {
    "archdesc" : {
      "dsc" : {
        "c01" : [
          {
            "level" : "series",
            "did" : {
              "daogrp" : {
                "daoloc" : {
                  "linktype" : "locator",
                  "daodesc" : {
                    "p" : "Persoonlijk leven"
                  }
                },
                "linktype" : "extended"
              },
              "unitid" : {
                "content" : "1",
                "type" : "EAD"
              }
            },
            "c02" : [
```

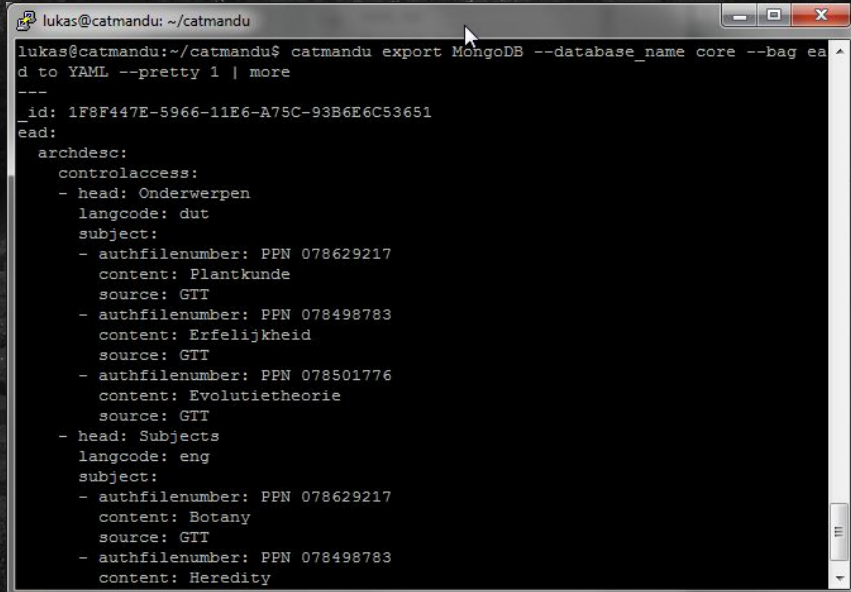
Data transformation from one format to another is easy using the available Exporters. Here an example with JSON output.

Catmandu - XML to MongoDB to JSON

```
lukas@catmandu: ~/catmandu
lukas@catmandu:~/catmandu$ catmandu import XML to MongoDB --database_name core --bag ead <hdv.xml
lukas@catmandu:~/catmandu$ catmandu export MongoDB --database_name core --bag ead to JSON --pretty 1 | more
[[
  "ead" : {
    "archdesc" : {
      "level" : "fonds",
      "dsc" : {
        "c01" : [
          {
            "level" : "series",
            "did" : {
              "daogrp" : {
                "linktype" : "extended",
                "daoloc" : {
                  "linktype" : "locator",
                  "daodesc" : {
                    "p" : "Persoonlijk leven"
                  }
                }
              }
            },
            "unitid" : {
              "content" : "1",
              "type" : "EAD"
            }
          }
        ]
      }
    }
  },
],
```

Also storing data is easy with the Store modules. In this case XML to MongoDB, and then exporting from MongoDB to JSON.

Catmandu - XML to MongoDB to YAML

A terminal window titled 'lukas@catmandu: ~/catmandu' showing the execution of the 'catmandu export MongoDB' command. The output is a YAML document with a single document's details, including its ID, head, and a list of subjects with their respective authfile numbers, content, and sources.

```
lukas@catmandu: ~/catmandu
lukas@catmandu:~/catmandu$ catmandu export MongoDB --database_name core --bag ea
d to YAML --pretty 1 | more
----
_id: 1F8F447E-5966-11E6-A75C-93B6E6C53651
head:
  archdesc:
    controlaccess:
      - head: Onderwerpen
        langcode: dut
        subject:
          - authfilenumber: PPN 078629217
            content: Plantkunde
            source: GTT
          - authfilenumber: PPN 078498783
            content: Erfelijkheid
            source: GTT
          - authfilenumber: PPN 078501776
            content: Evolutietheorie
            source: GTT
      - head: Subjects
        langcode: eng
        subject:
          - authfilenumber: PPN 078629217
            content: Botany
            source: GTT
          - authfilenumber: PPN 078498783
            content: Heredity
```

The same example with YAML output.

Catamandu - Fixes

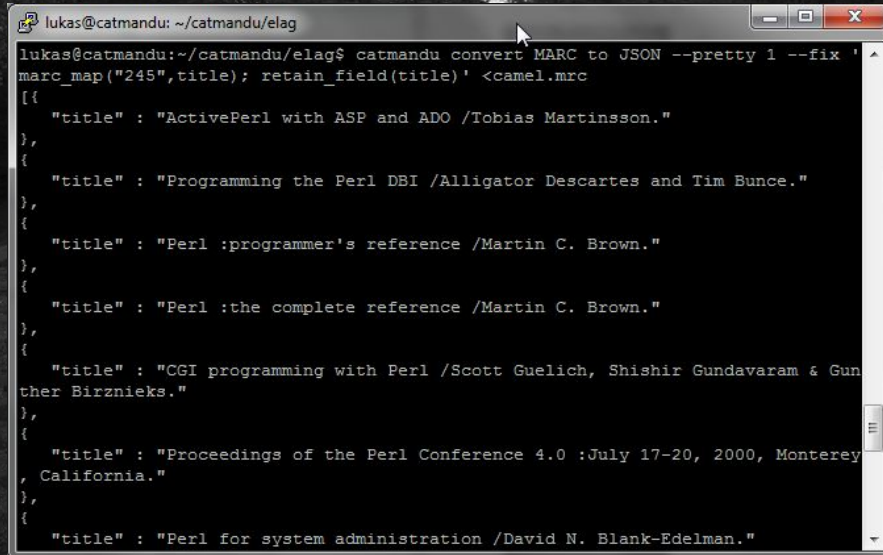
Data transformation: Fixes (functions)

Such as:

```
split_field(author, ",")  
marc_map('245', title)  
remove_field(isbn)  
substring()  
viaf_read('number')  
xml_simple()  
xml_transform()
```

In Catmandu, data transformations are done with available Fixes, or create your own.

Catmandu - Fixes

A terminal window titled 'lukas@catmandu: ~/catmandu/elag' showing a command and its output. The command is 'catmandu convert MARC to JSON --pretty 1 --fix 'marc_map("245",title); retain_field(title)' <camel.mrc'. The output is a JSON array of book titles.

```
lukas@catmandu:~/catmandu/elag$ catmandu convert MARC to JSON --pretty 1 --fix '
marc_map("245",title); retain_field(title)' <camel.mrc
[[{
  "title" : "ActivePerl with ASP and ADO /Tobias Martinsson."
},
{
  "title" : "Programming the Perl DBI /Alligator Descartes and Tim Bunce."
},
{
  "title" : "Perl :programmer's reference /Martin C. Brown."
},
{
  "title" : "Perl :the complete reference /Martin C. Brown."
},
{
  "title" : "CGI programming with Perl /Scott Guelich, Shishir Gundavaram & Gun
ther Birznies."
},
{
  "title" : "Proceedings of the Perl Conference 4.0 :July 17-20, 2000, Monterey
, California."
},
{
  "title" : "Perl for system administration /David N. Blank-Edelman."
}]]
```

An example of data transformations: using the `marc_map` fix to map input fields to MARC tags, and keep just one field using the `retain_field` fix.



CORE Project: Issues

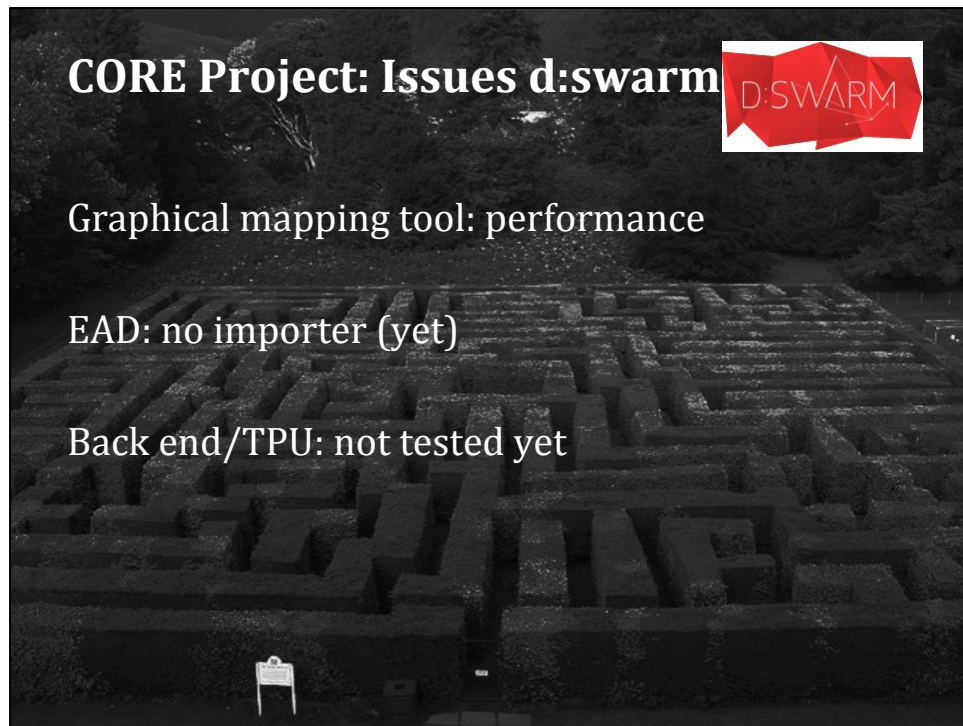
Still in exploratory phase: tools evaluation

Good support from both tools' developers

- d:swarm - Thomas Gängler and colleagues
- Catmandu - Patrick Hochstenbach and friends

Our CORE project issues: we still haven't started the actual pilot. Still in preliminary phase of evaluating and selecting tools.

Both d:swarm and Catmandu have many dedicated developers that are very willing to give support.

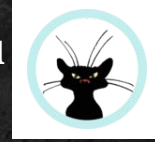


The d:swarm graphical mapping tool currently has some performance and display issues. This makes it hard to work with.

There is no EAD importer (yet). The d:swarm team is of course willing to develop one, given the time etc. The hybrid hierarchical structure makes this not an easy task.

We haven't had the time to test the batch processing yet.

CORE Project: Issues: Catmandu



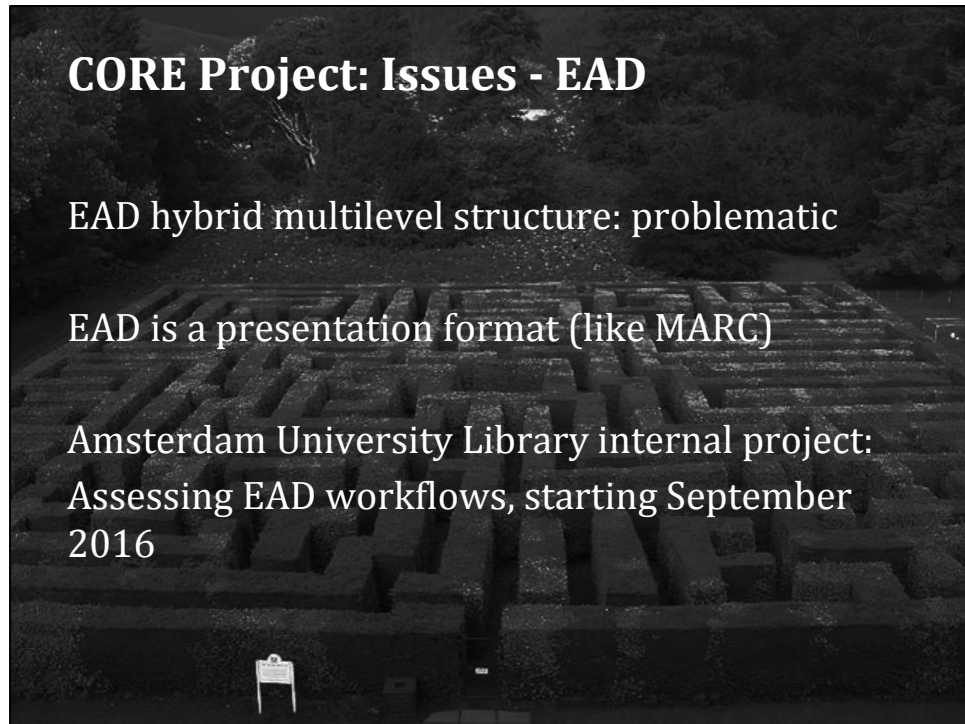
Command line: fast!

Needs a lot of framework, scripting, etc.

EAD: not supported (yet)

PNX: not supported (yet)

Unlike the d:swarm graphical interface, the Catmandu command line is fast. However you have to be very proficient in UNIX/Linux command line work to be able to use it correctly and efficiently. Of course for an actual DataHub this means putting together a vast and complex framework of prepared scripts and batch processes. Which will take a lot of work and resources to set up. Neither EAD nor PNX is supported (yet), but the Catmandu community is willing to develop importers and exporters for these data formats.



CORE Project: Issues - EAD

EAD hybrid multilevel structure: problematic

EAD is a presentation format (like MARC)

Amsterdam University Library internal project:
Assessing EAD workflows, starting September
2016

EAD is a data format has a number of issues too. We have already named most of them. Most importantly: the hybrid multilevel structure and the use of a display format as storage and exchange format.



Primo Collection Discovery is not yet available for third party collection/archive systems, so we can't test that at the moment.



That's all Folks!

Background image http://commons.wikimedia.org/wiki/File%3ATraquair_House_Maze.jpg
<http://outlawjimmy.com/2013/04/24/thats-all-folks/>