If I knew then what I know now….

Building the new database for your migrated data

Susan Jane Williams,
Data Specialist, Scholars Resource; Independent Consultant and Developer

Building the target for your data

- In 2007, Many VR professionals will still be using off-the-shelf products to facilitate data entry to meet local needs
  - Lack of access to larger institutional computing resources
  - Specialized needs, including using specific standards (VRA Core, CCO)
The cataloging utility as a bridge

- Need to create local specialized data, but also the need to hook that to teaching tools.
- The local institutional choice of DAM or DAP might not support the standard that you wish to use and may require exporting data from a cataloging utility to that system at this time.

A “cataloging utility”

- That is, not only the table structure that holds the data....
- And not just the data structure that can be used in other applications....
- But, an understanding of the user interface that facilitates and guides data entry.
So, you have done the first steps....

- Planning documents involving partners across your institution
- Data dictionaries
- Crosswalks of local collections
The “art” of database design

- Cataloging is the single most expensive component of creating the digital asset
- A balance must sometimes be struck between choices one might make in guiding efficient and accurate data entry and the degree of standards adherence
What you bring to database design

- The understanding of your own workflow, local needs, patron concerns, level of expertise of your cataloging staff (professionals or students) will be key to building the right user interface for your cataloging utility—this is the next step beyond data dictionaries and other planning documents.

Flat versus Relational Databases

- “Flatfile” data is what we are used to seeing in spreadsheets
  - Multiple values are either expressed in separate columns: “Subject 1”, “Subject 2” or are run together in the same column with punctuation or other dividers: “Subject 1; Subject 2; Subject 3”
Relational Databases

- Relate information stored in multiple tables
- Ideally, there is no redundancy of data entry—each value that might be reused in data entry is only entered once and stored in one table that is *related* for use everywhere else in the database (made available anywhere needed in the data entry workflow)
- Numeric keys are normally used in this process

<table>
<thead>
<tr>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title_Variant_Work</td>
<td>LOCATION</td>
<td>Classification</td>
<td>Artist_Sortname1</td>
<td>Artist_Sortname2</td>
</tr>
<tr>
<td>Rotonde de Chartres</td>
<td>Paris, France</td>
<td>Architecture</td>
<td>Ledoux, Claude-Nicolas</td>
<td></td>
</tr>
<tr>
<td>Ducal Palace</td>
<td>Dijon, France</td>
<td>Architecture</td>
<td>Mansart, Jules Hardouin</td>
<td>Gabriel, Ange-Jardine</td>
</tr>
<tr>
<td>Ducal Palace</td>
<td>Dijon, France</td>
<td>Architecture</td>
<td>Mansart, Jules Hardouin</td>
<td>Gabriel, Ange-Jardine</td>
</tr>
</tbody>
</table>
Sample of a table of related data

<table>
<thead>
<tr>
<th>fk_AgentID</th>
<th>fk_WorkID</th>
<th>AgentRoles</th>
</tr>
</thead>
<tbody>
<tr>
<td>4467</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>4470</td>
<td>8</td>
<td>director</td>
</tr>
<tr>
<td>2906</td>
<td>9</td>
<td>designer</td>
</tr>
<tr>
<td>4315</td>
<td>9</td>
<td>designer</td>
</tr>
<tr>
<td>4471</td>
<td>11</td>
<td>designer</td>
</tr>
<tr>
<td>3876</td>
<td>8</td>
<td>costume designer</td>
</tr>
<tr>
<td>4468</td>
<td>9</td>
<td>designer</td>
</tr>
<tr>
<td>4469</td>
<td>7</td>
<td>designer</td>
</tr>
<tr>
<td>3562</td>
<td>14</td>
<td>architect</td>
</tr>
<tr>
<td>2257</td>
<td>15</td>
<td>architect</td>
</tr>
<tr>
<td>4467</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>4266</td>
<td>16</td>
<td>architect</td>
</tr>
<tr>
<td>2727</td>
<td>3</td>
<td>painter</td>
</tr>
<tr>
<td>2727</td>
<td>5</td>
<td>painter</td>
</tr>
<tr>
<td>2926</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

GUI interfaces to data tables

- Obviously, looking at rows and columns of related numeric keys is not user friendly, so most commercial databases allow you to build graphic user interfaces (GUI)—forms—for data entry
Work and Image entry layouts

- A clipped sample of 2 forms:

A more sophisticated approach that may be more flexible
Possible in either Filemaker 7+ or MS Access
Possible with other tools such as Servoy (provides the functional interface to SQL or Oracle tables)
(Sample screenshot, next slide)
Relational databases such as Filemaker Pro (FMP) or MS Access also use graphic tools to show specific fields in tables and tables related to each other in the entire database.
Portals and subforms

- Using forms/layouts, you can create “windows” looking into tables of related data (tables with relationships established between them) and showing multiple data values—more than one “answer” per record (multiple locations, agents, titles, etc. for each work). In FMP these are known as portals, in Access, subforms.

VireoCat Locations Portal (on Works form)
Understanding the new role of XML in data mobility

- XML facilitates pushing data between all sorts of applications
- CCO and VRA Core 4 were both formed with an eye to XML
What is XML?

- Extensible Markup Language (XML) is a universal language for sharing data between applications. XML is most appropriate for situations where the volume of data is generally small, as the data is transmitted as text, and controlling the structure of the data is important.

How does XML work?

- It “tags” data—identifies what that data is (what meaning it holds).
  - MARC tags by using numeric designators:
    - for instance a “245” field is always a title, a “700” or “7xx” field is a personal name (creator)
XML tags with natural language — easy to see what the information (the data value) is within the “chicken lips”
XML example

<!-- AGENT -->
<set>
<display>Jasper Francis Cropsey (American painter, 1823-1900)</display>
=index>
<agent>
<name type="personal" vocab="ULAN" refid="500012491">Cropsey, Jasper Francis</name>
<dates type="life">
<earliestDate>1823</earliestDate>
<latestDate>1900</latestDate>
</dates>
<culture>American</culture>
<role vocab="AAT" refid="300025136">painter</role>
</agent>
</index>
</set>

Schema: Where the data standard and XML meet

Once a data standard like VRA Core 4.0 is devised, with all the elements and qualifiers laid out, the standard can then be expressed in one XML document called the schema—a road map to then apply to a specific XSLT style sheet that tells a database how to export data into XML.
VRA Core 4.0 XML schema (a small sample)

```xml
<!-- Agent -->
<xsd:complexType name="agentType">
  <xsd:annotation>
    <xsd:documentation>
      VRA Agent element. Subelements are used for different types of data (names, roles, dates, etc.). At least one subelement must be provided.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:sequence minOccurs="1" maxOccurs="unbounded">
    <xsd:element name="attribution" type="basicString" minOccurs="0"/>
    <xsd:element name="culture" type="basicString" minOccurs="0"/>
    <xsd:element name="dates" type="agentDateType" minOccurs="0"/>
    <xsd:element name="name" type="agentNameType" minOccurs="0"/>
    <xsd:element name="role" type="basicString" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attributeGroup ref="vraAttributes"/>
</xsd:complexType>
```

What is XSLT?

- You can export XML data from FileMaker or Access (and many other programs) to an assortment of applications simply by applying the appropriate Extensible Stylesheet Language Transformation (XSLT) stylesheet.
XLST Sample—how the XML is actually exported from a database

```xml
<!-- Agent -->
<set>
<display>
<xsl:value-of select="fm:AgentDisplay" />
</display>
<index>
<xsl:for-each select="fm:AgentSortName/fm:DATA">
<xsl:variable name="i">
<xsl:value-of select="position()" />
</xsl:variable>
<agent>
```

4 Screenshots of an XML export (VireoCat)

- First slide—work and image screens, perform find
- Second slide showing the work record and 4 related image records and export selected from menu bar
- Third slide, resultant XML document (excerpt)
- Fourth slide, showing the database folder containing database, 2 stylesheets and XML document
<xml version="1.0" encoding="UTF-8"/>
<work xmlns="http://www.wraweb.org/wrakey1.dtd">
<display>unknown (sculptor, Greek, ancient )</display>
<agentSet>
<agent>
<name type="Personal">unknown</name>
</agent>
</agentSet>
<dateSet>
<display>ca 490-480 BCE</display>
<date type="creation">
<earliestDate>4690</earliestDate>
<latestDate>4490</latestDate>
</date>
</dateSet>
<locationSet>
<display>Repository: Glyptothek, Munich, Bavaria, Germany (85): Former: Temple of Aphaia, Aegina, Central Greece and Euboea, Greece ()</display>
<location type="formerGeographic">
<name type="geographic" vocab="DOA" refid="" extent="building">Temple of Aphaia</name>
</location>
</locationSet>
</work>
My better late than never epiphany….

- Appreciating the symmetry of the elements in the work, image and collection records (work title, image title and so forth)
- Appreciating the distinction between indexed and display values, and how that can give you flexibility in data entry

Creating Display values

- Can be created from the indexed values “on the fly” in the xml via the stylesheet, or
- Can be “pre-created” within the database by scripting/programming
- Both approaches are likely to be used in different fields—gives flexibility in design choices
Location in VireoCat:
multiple table design requires “extent” in XML;
Display created on the fly in the XML

Locations in IRIS
Showing CCO display form “pre-created” in the database from parsed values
Using scripting and design to guide cataloging

- First slide shows set-up in VireoCat—as cataloger chooses continent, then next level (nation) is constrained to only the choices in that continent, and so forth down to building/site level.
- Second slide shows multi-table design necessary to do this, with 2 sets of additional linking tables to constrain and display parent/child lists.

Constraining/ guiding entry
Hobbesian choices?

- The multi-table design guides entry and may be a good solution, especially for student workers, but....
- It complicated the design of XML export and import
- Good to look at all these issues holistically, which means acquainting yourself with XML and scripting and design, at least conceptually
And what is my next stylesheet?

- Taking relational XML export and running it through a stylesheet so that it becomes flattened CSV to use in DAPs such as CONTENTdm that are not yet XML ready!

;-)