



painting: Fyodor Bronnikov (1827—1902); Pythagoreans Celebrate Sunrise; 1869 | Image: Wikimedia Commons (altered)

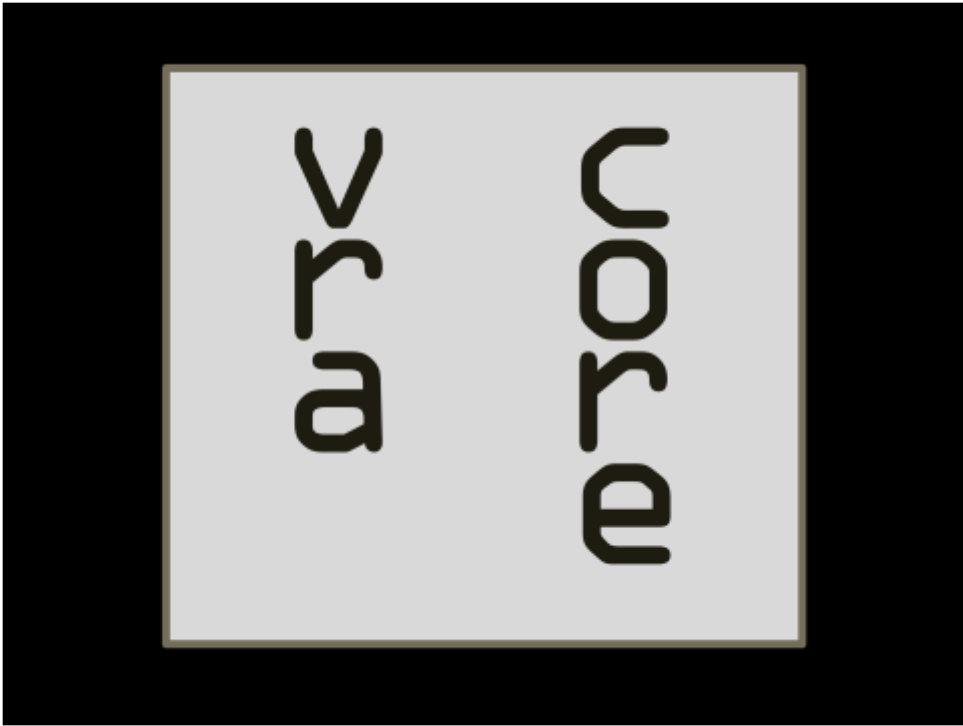
This session isn't about VRA Core worship.



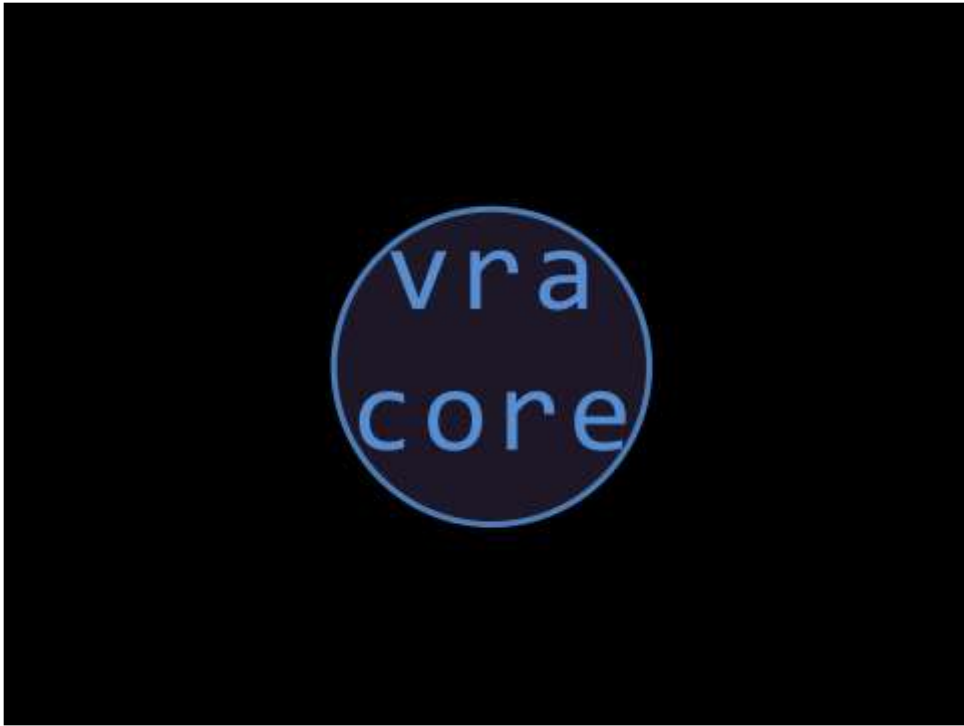
It's about taking the Core...



...and adapting it...



...interpreting it...



...to work within limitations...

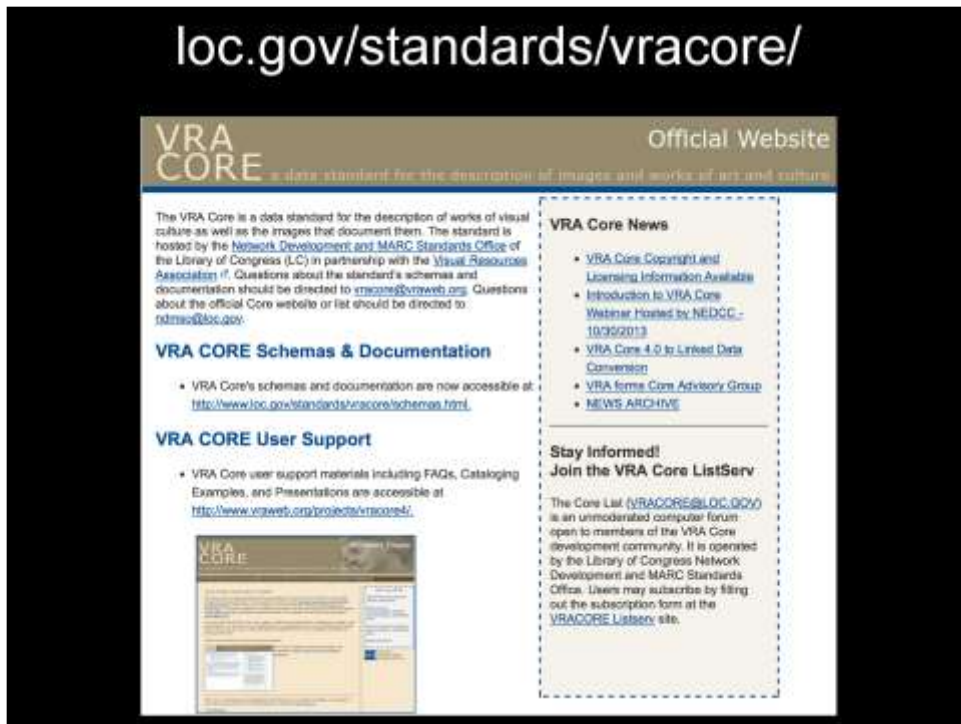
vra core...

...,or extending it, to meet local needs...



...without leaving it unrecognizable as VRA Core.

loc.gov/standards/vracore/



The screenshot shows the official website for VRA CORE. At the top, the URL 'loc.gov/standards/vracore/' is displayed in large white text on a black background. Below this, the website header features the 'VRA CORE' logo on the left and 'Official Website' on the right. A tagline reads: 'a data standard for the description of images and works of art and culture'. The main content area is divided into several sections:

- VRA CORE Schemas & Documentation**: A paragraph explains that the VRA Core is a data standard for describing visual culture, hosted by the Network Development and MARC Standards Office of the Library of Congress (LC) in partnership with the Visual Resources Association (VRA). It provides contact information for questions: vracore@vraweb.org for schemas and documentation, and rdmrao@loc.gov for the official website or list. A bullet point states: 'VRA Core's schemas and documentation are now accessible at: <http://www.loc.gov/standards/vracore/schemas.html>'.
- VRA CORE User Support**: A bullet point states: 'VRA Core user support materials including FAQs, Cataloging Examples, and Presentations are accessible at: <http://www.vraweb.org/projects/vracore4/>'.
- VRA CORE News**: A list of news items:
 - [VRA Core Copyright/Licensing Information Available](#)
 - [Introduction to VRA Core Webinar Hosted by NEDCC - 10/30/2013](#)
 - [VRA Core 4.0 to Linked Data Conversion](#)
 - [VRA Forms Core Advisory Group](#)
 - [NEWS ARCHIVE](#)
- Stay Informed! Join the VRA Core ListServ**: A paragraph explains that the Core List (VRACORE@LOC.DC) is an unmoderated computer forum open to members of the VRA Core development community. It is operated by the Library of Congress Network Development and MARC Standards Office. Users may subscribe by filling out the subscription form at the [VRACORE Listserv](#) site.

A small thumbnail image of the VRA CORE website interface is visible in the bottom left corner of the main content area.

This is the current version of VRA Core - version 4.0, first released in 2007.

Version 1.0 was launched in 1996 and ever since it has adapted in response to developments in technology and community input. The intended community has been cultural heritage visual resource professionals, but that is changing. Notice the URL there, VRA Core is now hosted at the Library of Congress and is more visible to a wider range of users seeking a standard to catalog digital media from many disciplines and within many different systems. The original authors of the Core couldn't have imagined the all the ways people want to use it today. The presentations in this session demonstrate the process of resolving problems and conflicts in local implementations.

VRA Core 4 for Embedded Metadata



Many of you are already familiar with the work of the VRA Embedded Metadata group has done to incorporate Core 4 into digital image embedded metadata, but how many of you know the backstory – the dirty, behind the scenes work that went into making it all work? I'm still working on my memoirs, so I think not many of you. What follows is a little taste of the story.

VISUAL RESOURCES ASSOCIATION BYLAWS

Article I: MEMBERSHIP

Section 1. There shall be three classes of membership in the Association: Individual, Institutional, and Special Honorary Life membership awarded by the Executive Board.

Section 2. Regular Individual Membership shall be available to anyone who has completed an application form and paid the currently stipulated Individual dues. An Individual Member in good standing shall have all the privileges of membership as established by the Executive Board including the right to vote and to hold office in the Association. The Executive Board may at its discretion authorize Contributing and Patron Memberships, which shall be available to anyone eligible to hold a regular Individual or Institutional Membership who makes a monetary contribution to the Association beyond the applicable dues amount at levels determined by the Executive Board. Acknowledgement of Contributing and Patron Members shall be published annually in an official publication or venue of the Association as may be determined by the Executive Board.

Section 3. Reduced rate Individual Membership shall be available to any individual meeting the special needs criteria determined by the Executive board. Reduced Rate Individual Members shall have all of the privileges of membership as established by the Executive Board including the right to vote and to hold office in the Association. Reduced Rate Individual Memberships may include:

a) Student Membership shall be for full-time students enrolled in an accredited degree program who provide appropriate documentation of current enrollment as determined by the

I checked the fine print in the VRA Bylaws – it's not required to use Core 4 or to support it. I think it is still illegal to disparage it or use the Core logo in an unflattering manner, so I'm probably in some sort of trouble here. The EMwg chose to use it in embedded metadata because it has a lot of useful features.

Work / Image



Which users?

Limited access to central database

No IT support

Shared image project

By the way, what users where we trying to help?

Which tools?



Photoshop



Bridge



Excel

Plenty of tools exist for entering basic photo metadata: title, caption, keywords. We wanted VR quality metadata, so that meant creating custom input tools. Most of our users are not programmers, so the tools had to be familiar and as easy to use as possible. Pretty much all our users have Adobe Photoshop, Bridge, and Excel. They do most of their data manipulation in Excel – the lingua franca of data. Our task was to figure out what these tools were capable of in terms of Core 4 metadata. What could they create? What could they import and export? We also wanted to produce data that was **transferrable** to other programs.

Which metadata format?



Currently, the best way to embed metadata in images is Adobe's XMP standard. It is open source, widely supported and extensible. XMP is not perfect though – it has limitations. XMP is serialized in RDF/XML, but it is a subset of RDF, not all RDF properties and attributes are available.

Will Core 4 fit?



Amy Dominello / News & Record (altered)

How do we jam Core into it? What will we loose? What matters?

Software Customization

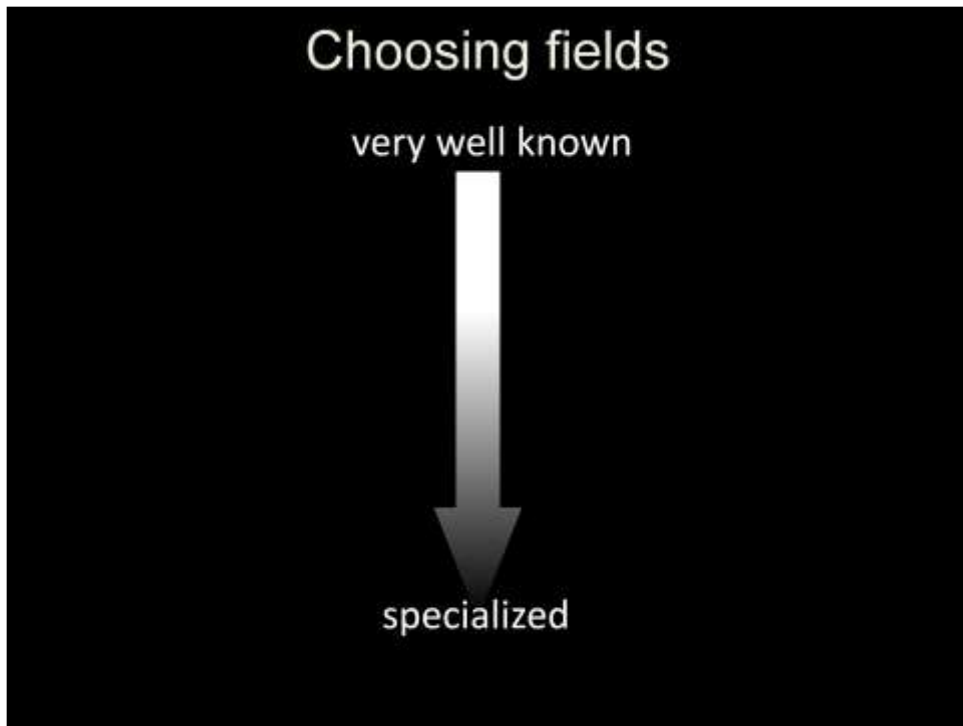


The screenshot shows the Adobe Developer Connection / Adobe XMP Developer Center. The navigation bar includes links for Products, Business solutions, Support & Learning, Download, Company, and Buy. The main content area features the Adobe logo, the title "Adobe Developer Connection / Adobe XMP Developer Center", and a paragraph explaining Adobe's Extensible Metadata Platform (XMP) as a labeling technology for embedding data into files. Below this, there are two SDK download options: "XMP Toolkit SDK CC 2015.06 (ZIP; 28.4 MB)" and "XMP FileIO SDK CS6 (ZIP; 2.75 MB)".



The screenshot shows the Adobe Developer Connection / Bridge Developer Center. The navigation bar includes links for Products, Business solutions, Support & Learning, Download, Company, and Buy. The main content area features the Adobe logo, the title "Adobe Developer Connection / Bridge Developer Center", and a section for the "Adobe Bridge CS6 SDK" dated April 2012. A paragraph below describes the SDK, stating it is updated for CS6 and enables developers to integrate with and extend Bridge, providing details on programming with the JavaScript API and developing plug-ins for Bridge in C/C++.

The first step was to figure out how XMP data works – format, serialization, limitations. Then we had to figure out how to write plugins and info panels for Photoshop and Bridge. What is possible in the UI? What functions can be built in?



When choosing fields, like Work Title or Image Copyright, the idea is to start with schemas that are most widely used by the majority of photo applications and web services and then move down the list, using specialized schemas last. This places as much of the metadata as possible in properties that will be read by common tools

Choosing fields



IPTC Core



IPTC Extension



PLUS



Dublin Core



Other Native XMP



VRA Core 4.0

There's VRA all the way at the bottom looking like the least favored of the children. It turns out however, that it has an important role to fill, stepping when the other schemas can't fulfill our needs.

Work / Image



**IPTC Extension-
Artwork/object**



**EXIF
IPTC
IPTC Extension**

Most of all is the distinction between the original work and the image of the work. Standards have been in place for digital image metadata for a long time. The EXIF and IPTC standards cover the who what and where of the photograph, but they are a bit muddy on the details of creative works shown in the photo.



Extension

Artwork or Object in the Image

Creator

Title

Date Created

Source

Source Inventory Number

Copyright Notice

IPTC Extension does have fields specifically for artworks and, while a good start, they don't meet the needs of the users EMwg was trying to help.



Extension Artwork or Object in the Image

Date Created

single calendar date
no BCE

~~“built 1298 – 1310, restored 1872”~~

For Instance, “Date Created” is a single calendar date only and doesn’t allow for a range of dates or a complex free-text date such as “built 1298 – 1310, destroyed 1673”

Qualifying IPTC Ext. with VRA

<



Extension
Artwork or Object in the Image



Core 4.0
Work

</

XMP allows qualifying one namespace with another

Another thing we tried, and the method that would be the most reliable and computer friendly, would be to nest VRA within IPTC. This keeps all the artwork data together in one structure and makes it possible to describe multiple artworks using multiple arrays, each one being a completely discrete packet. This method is supported by XMP.

Qualifying IPTC Ext. with VRA



Extension
Artwork or Object in the Image

gone



Popular photo apps delete qualifiers

Unfortunately, most applications don't recognize the nested VRA data and delete it. So, I can make a custom info panel that qualifies IPTC with VRA, but if you open that metadata with another tool, the VRA data will be stripped out. This is far too dangerous – we can't risk having VRA metadata erased.

Qualifying IPTC Ext. with VRA

~~Warning:~~

~~Only edit metadata in the VRA
custom info panel!~~

**Metadata must be
interoperable**

One solution would be to tell people that they can only edit embedded metadata in the VRA tool. This would never work. We can't tell people to only open their file in one particular application. The metadata we create has to be safe to open in the tools people are likely to use.

Working with popular software

Understand the limits

Keep all metadata safe

Make it as useful as possible

Just to reiterate: this is a problem with the way software developers have implemented IPTC Extension, not in XMP. Most developers don't think about people customizing IPTC metadata so they don't build their software to handle it. This is a reality that we have to live with which means extending IPTC Extension artwork with VRA work is not practical – VRA data has to be separate.

VRA Info Panel Compromise



IPTC Core

Dublin Core
Photoshop
XMP Rights
PLUS



VRA Core 4.0

All work properties

Our practical solution was to use IPTC Core for the Image and simple Core 4 display values for all the Work.

Too much?



Graeme Newcomb / Flickr (altered)

Core 4 does have a lot of fields. Do we need an image file to carry them all?

Core 4.0 XML

```
<work>
  <dateSet>
    <display> built 1298 – 1310, destroyed 1943</display>
    <date type="creation">
      <earliestDate>1298</earliestDate>
      <latestDate>1310</latestDate>
    </date>
    <date type="destruction">
      <earliestDate>1943</earliestDate>
      <latestDate>1943</latestDate>
    </date>
  </dateSet>
</work>
```

Here is a snippet of Core 4 XML. Could we replicate this structure without losing meaning in XMP? Yes, we can and we did.

Core 4.0 XMP RDF/XML

```
<vra:dateSet rdf:parseType="Resource">
  <vra:display> built 1298 – 1310, destroyed 1943</vra:display>
  <vra:date>
    <rdf:Bag>
      <rdf:li rdf:parseType="Resource">
        <vra:type>creation</vra:type>
        <vra:earliestDate rdf:parseType="Resource">
          <vra:date>1298</vra:date>
        </vra:earliestDate>
        <vra:latestDate rdf:parseType="Resource">
          <vra:date>1310</vra:date>
        </vra:latestDate>
      </rdf:li>
      <rdf:li rdf:parseType="Resource">
        <vra:type>desctruction</vra:type>
        <vra:earliestDate rdf:parseType="Resource">
          <vra:date>1943</vra:date>
        </vra:earliestDate>
        <vra:latestDate rdf:parseType="Resource">
          <vra:date>1943</vra:date>
        </vra:latestDate>
      </rdf:li>
    </rdf:Bag>
  </vra:date>
</vra:dateSet>
```

Here is a snippet of Core 4 XMP RDF/XML. Sure, it looks great. It's got lots of arrows, colons, and slashes and everything is nicely indented, so it should work well. The question is, do we really need this complexity?

Because Excel



Cultural Context	Material
Greek (ancient); Roman; Samnite	stone tess
Netherlandish; Flemish	wool; met

The answer was, no. This is the use case we targeted. This is how many database exports and imports work in Excel. You can parse the data if you want using semicolons as delimiters. Of course there are limitations with Excel – you can't easily include subfields or types, but many people seem to be OK with that because they only display simple lists of values in their user interface. Most importantly, users are comfortable with Excel, they aren't programmers and aren't going to build an XSLT to transform complex data.

Flat Display Fields

```
work.agent  
work.title  
work.date  
work.stylePeriod  
work.culturalContext  
work.worktype  
work.material  
work.technique  
work.measurements  
work.refid  
work.rights  
work.description  
work.subject  
work.inscription  
work.relation  
work.textref  
work.source
```

We decided to keep it simple and use a flattened version of Core 4 display fields. This means that we will eliminate the nested arrays of parsed terms and use single free text display values instead. These are very easy to export to Excel. A spreadsheet is the easiest and most common way people will use it. This does mean that curators will have to do some work before they can ingest the data into their database. For instance, if multiple artworks are present in an image, the display values must reflect this through the use of identifying labels and delimiters.

Locations

work.locationRepository
work.locationSite
work.locationCreation
work.locationDiscovery
work.locationExhibition
work.locationInstallation
work.locationPerformance
work.locationOther
work.locationNotes

Locations were tough. We saw a need for more granularity there because works can have several Locations worthy of mention.

RDF Namespace

vrae = “VRA Essentials”

vrae:work.agent

Choosing a namespace. We went in assuming others would use our version of VRA in XMP. If we were successful, it wouldn't be just us using it, so we wanted to be careful about what it conveyed. We didn't want it to conflict or suggest that it was a complete representation for VRA Core.

We came up with the name vrae – VRA Essentials



TRUCKAUS (altered)

If we wanted to park big 'olCore 4 in XMP...

Adapt



TRUCKAUS (altered)

...we would have to adapt it while leaving it drivable. A lowrider big rig – I guess that is kinda cool. Sure people pointed and laughed, but we didn't mind, were able to go where we wanted. And then, Sheryl Frisch asked me if we could add some complexity to the VRA Panel to accommodate the California State University's Dspace shared image project.

More, more, more

	A	B	C	D	E	F	G
1	dc:title	dc:contributor.displayName	dc:contributor.role	dc:contributor.datesact/vedisplay	dc:contributor.artist	dc:contributor.begindate	dc:contributor.enddate
2							
3							
4							
5							
6							
7							
8							
9							
10							
11							
12							
13							

They used qualified Dublin Core to cover the fields they needed, then they designed a matching Excel template for contributors to use. Our first thought was to just make a completely new XMP panel using their qualified Dublin Core field, but then we thought we could use this as an opportunity to expand the VRA XMP panel and build something everyone could use. Again we were faced with obstacles from XMP and the tools we were using.



The VRA panel was kept simple by design - a single display text field for every Core 4 element. Obviously for the CalState Creator section we had to add a lot of fields...

CSU The California State University
SERVING FOR CALIFORNIA

DISPICES

Creator 2 Label: Durand, Guillaume (French author, approximately 1225-1296)

copy and paste from: [ULAN](#) or build a new Creator below

Type	personal	Vocabulary	LCNAF	ID	n87900857
Last Name	Durand	First Name	Guillaume		
Culture	French	Role	author		
Dates	approximately 1225-1296	Begin	1225	End	1296

... 10 to be exact. We also added an auto-complete feature for the Creator Label so a uniformly formatted ULAN-type Label would be produced.

The screenshot displays the CSU Digital Library System (DLS) interface for creating or editing creators. It features three distinct forms, each with a header, a source link, and a set of input fields.

Creator 1 Label: unknown (French illuminator)
 copy and paste from: [ULAN](#) or build a new Creator below
 Type: personal | Vocabulary: local | ID:
 Name: unknown | No First Name:
 Culture: French | Role: illuminator
 Dates: | Begin: | End:

Creator 2 Label: Durand, Guillaume (French author, approximately 1225-1296)
 copy and paste from: [ULAN](#) or build a new Creator below
 Type: personal | Vocabulary: LCNAF | ID: n87900857
 Last Name: Durand | First Name: Guillaume
 Culture: French | Role: author
 Dates: approximately 1225-1296 | Begin: 1225 | End: 1296

Creator 3 Label:
 copy and paste from: [ULAN](#) or build a new Creator below
 Type: Select | Vocabulary: Select | ID:
 Last Name: | First Name:
 Culture: | Role:
 Dates: | Begin: | End:

We also allowed for multiple Creators.

After we finished this and it was working I thought back on the many times people told me, “No one will ever want to enter granular data in Bridge.”

Not in VRA

Creator 2 Label: Durand, Guillaume (French author, approximately 1225-1296)
copy and paste from [ULAN](#) or build a new Creator below

Type	personal	Vocabulary	LCNAF	ID	n87900857		
Last Name	Durand	First Name	Guillaume	Role	author		
Culture	French	Dates	approximately 1225-1296	Begin	1225	End	1296

CSU wanted a GETTY ULAN-like name display and we thought it would be easier for the user if we provided individual boxes to enter the data and then let the panel build a properly formatted label. Core 4 doesn't have all the individual fields to do this – for instance, it doesn't have first and last name or date display ...

Extentions

Label	cdwalite:descriptive.displayCreator
Type	vrae:nameType
Vocabulary	vrae:vocab
ID	vrae:refid
Last Name	foaf:familyName
First Name	foaf:givenName
Culture	vrae:culture
Dates	cdwalite:vitalDates
Begin	vrae:dateEarliest
End	vrae:dateLateest

...so we used FOAF and CDWA lite properties.

vrae details

```
<vrae:work_agentDetails>
  <rdf:Seq>
    <rdf:li rdf:parseType="Resource">
      <cdwalite:descriptive_displayCreator></cdwalite:descriptive_displayCreator>
      <vrae:nameType></vrae:nameType>
      <vrae:vocab></vrae:vocab>
      <vrae:refid></vrae:refid>
      <vrae:name></vrae:name>
      <foaf:lastName></foaf:familyName>
      <foaf:firstName></foaf:givenName>
      <cdwalite:vitalDates></cdwalite:vitalDates>
      <vrae:dateEarliest></vrae:dateEarliest>
      <vrae:dateLatest></vrae:dateLatest>
      <vrae:culture>
        <rdf:Seq>
          <rdf:li rdf:parseType="Resource">
            <vrae:text></vrae:text>
          </rdf:li>
        </rdf:Seq>
      </vrae:culture>
      <vrae:role>
        <rdf:Seq>
          <rdf:li rdf:parseType="Resource">
            <vrae:text></vrae:text>
          </rdf:li>
        </rdf:Seq>
      </vrae:role>
    </rdf:li>
  </rdf:Seq>
</vrae:work_agentDetails>
```

What ended up with - a very faithful expression of VRA Core 4.0 in XMP. Yeah, it's kind of complex.

Non-repeatable

```
<vrae:work_agentDetails>
  <rdf:Seq>
    <rdf:li rdf:parseType="Resource">
      <odwalite:descriptive_displayCreator></odwalite:descriptive_displayCreator>
      <vrae:nameType></vrae:nameType>
      <vrae:vocab></vrae:vocab>
      <vrae:refid></vrae:refid>
      <vrae:name></vrae:name>
      <foaf:lastName></foaf:familyName>
      <foaf:firstName></foaf:givenName>
      <odwalite:vitalDates></odwalite:vitalDates>
      <vrae:dateEarliest></vrae:dateEarliest>
      <vrae:dateLatest></vrae:dateLatest>
      <vrae:culture>
        <rdf:Seq>
          <rdf:li rdf:parseType="Resource">
            <vrae:text></vrae:text>
          </rdf:li>
        </rdf:Seq>
      </vrae:culture>
      <vrae:role>
        <rdf:Seq>
          <rdf:li rdf:parseType="Resource">
            <vrae:text></vrae:text>
          </rdf:li>
        </rdf:Seq>
      </vrae:role>
    </rdf:li>
  </rdf:Seq>
</vrae:work_agentDetails>
```

An example of how we retained Core structures in XMP. We determined which Core elements were repeatable, then used an RDF structure for these. The fields here are not repeatable, so they are a simple structure: they are just a simple list of non-repeatable properties within the Agent Details. For instance, Name, earliest, latest dates are not repeatable.

Repeatable

```
<vrae:work_agentDetails>
  <rdf:Seq>
    <rdf:li rdf:parseType="Resource">
      <odwalite:descriptive_displayCreator></odwalite:descriptive_displayCreator>
      <vrae:nameType></vrae:nameType>
      <vrae:vocab></vrae:vocab>
      <vrae:refid></vrae:refid>
      <vrae:name></vrae:name>
      <foaf:lastName></foaf:familyName>
      <foaf:firstName></foaf:givenName>
      <odwalite:vitalDates></odwalite:vitalDates>
      <vrae:dateEarliest></vrae:dateEarliest>
      <vrae:dateLatest></vrae:dateLatest>
      <vrae:culture>
        <rdf:Seq>
          <rdf:li rdf:parseType="Resource">
            <vrae:text></vrae:text>
          </rdf:li>
        </rdf:Seq>
      </vrae:culture>
      <vrae:role>
        <rdf:Seq>
          <rdf:li rdf:parseType="Resource">
            <vrae:text></vrae:text>
          </rdf:li>
        </rdf:Seq>
      </vrae:role>
    </rdf:li>
  </rdf:Seq>
</vrae:work_agentDetails>
```

Agent Role and Culture need to be repeatable, so we placed them inside containers – in this case RDF arrays. This allows us to have as many of them as desired.

Repeatable

```
<vrae:role>  
  <rdf:Seq>  
    <rdf:li rdf:parseType="Resource">  
      <vrae:text>painter</vrae:text>  
    </rdf:li>  
  </rdf:Seq>  
</vrae:role>
```

For example, a single role would be written like this.

Repeatable

```
<vrae:role>
  <rdf:Seq>
    <rdf:li rdf:parseType="Resource">
      <vrae:text>painter</vrae:text>
    </rdf:li>
    <rdf:li rdf:parseType="Resource">
      <vrae:text>engraver</vrae:text>
    </rdf:li>
  </rdf:Seq>
</vrae:role>
```

But additional roles can be added to the array.

More details

```
<vrae:role>
  <rdf:Seq>
    <rdf:li rdf:parseType="Resource">
      <vrae:text>painter</vrae:text>
      <vrae:vocab>AAT</vrae:vocab>
      <vrae:refid>300025136</vrae:refid>
    </rdf:li>
    <rdf:li rdf:parseType="Resource">
      <vrae:text>engraver</vrae:text>
      <vrae:vocab>AAT</vrae:vocab>
      <vrae:refid>300025165</vrae:refid>
    </rdf:li>
  </rdf:Seq>
</vrae:role>
```

And if desired, VRA attributes, like vocab and refid, can be added.

Preferred Title

Core 4XML

Preferred Title

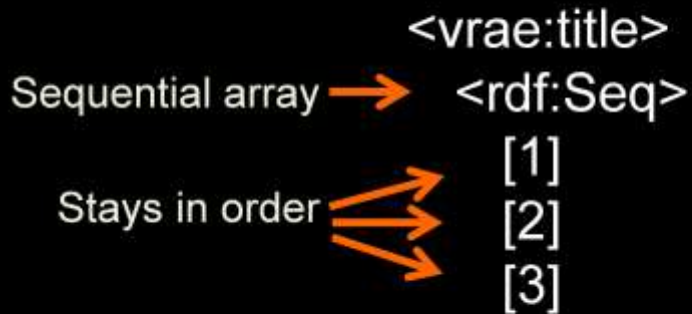


```
<title pref="true">Mona Lisa  
<title pref="false">La Gioconda
```

OK, I'm going to lay some technical XMP information on you now. I'm doing it to illustrate how you can replicate meaning from a schema to a new data format. We wanted to keep our XMP Core 4 data as lean as possible so we looked at use inherent meaning in XMP RDF to express Core 4 concepts such as preferred Title. Core 4 uses the XML attribute pref=true/false

Preferred Title

vrae XMP



XMP uses RDF arrays (lists) to hold repeating data values. These lists must be given a type to indicate if the order of the items is meaningful. Why? So the the UI knows if the list should be displayed in order. A list that is meant to always be displayed in a certain order is called Sequential

Preferred Title

```
<vrae:title>
  <rdf:Seq>
    1 → <rdf:li rdf:parseType="Resource">
        preferred <vrae:text>Mona Lisa</vrae:text>
    </rdf:li>
    2 → <rdf:li rdf:parseType="Resource">
        <vrae:text>La Gioconda</vrae:text>
    </rdf:li>
  </rdf:Seq>
</vrae:title>
```

For example, if you enter “Mona Lisa” in the top position of the Title, it always stays there and will be displayed on top every time. We used this inherent meaning in XMP RDF to express the Core 4 attribute `pref="true"`. Sequential array item 1 = XML `pref="true"`. This simplifies the data because we didn't have to add a new Preference property.

Technical Triumph

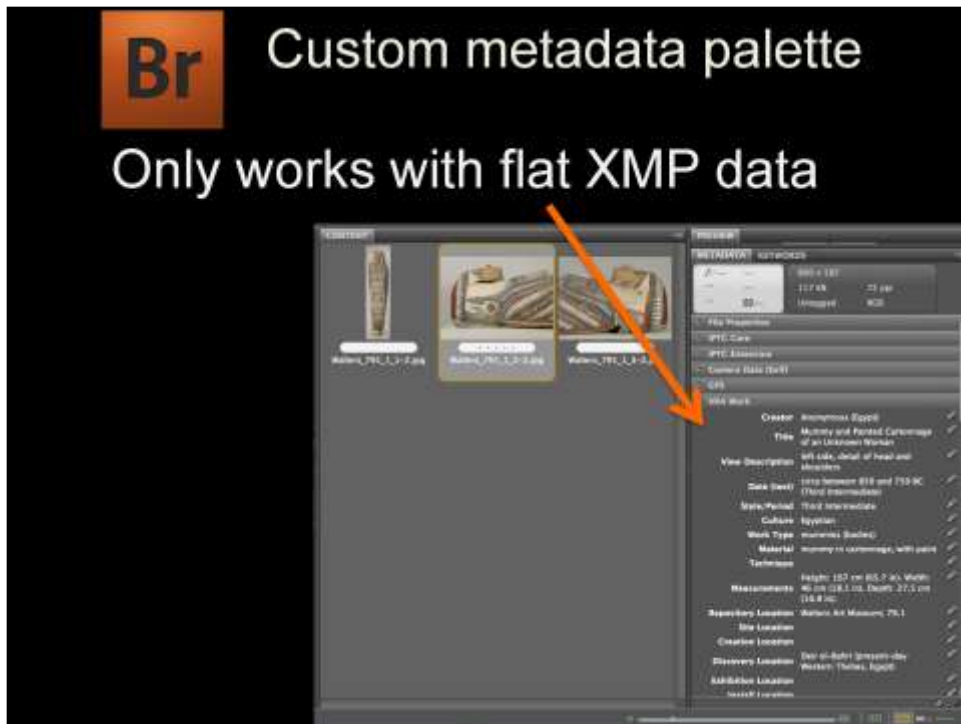


The vrae structure accurately reflected Core 4 and the custom file info panel was beautiful, and the export-import tool worked. There was just one problem...

Users Underwhelmed



...the users. Yes, we want to make the users happy and what made them happy was ease of use and speed.



They found the metadata palette in Bridge faster and easier to use than the file info panel window. We set out adapting the CSU panel and vrea to the Bridge palette and quickly discovered that it was not possible. It turned out that custom palettes are very limited, most crucially for us, they cannot create structured data...

vrae details

```
<vrae:work_agentDetails>
  <rdf:Seq>
    <rdf:li rdf:parseType="Resource">
      <odwalite:descriptive_displayCreator></odwalite:descriptive_displayCreator>
      <vrae:nameType></vrae:nameType>
      <vrae:vocab></vrae:vocab>
      <vrae:refid></vrae:refid>
      <vrae:name></vrae:name>
      <foaf:lastName></foaf:lastName>
      <foaf:firstName></foaf:firstName>
      <odwalite:vitalDates></odwalite:vitalDates>
      <vrae:dateEarliest></vrae:dateEarliest>
      <vrae:dateLatest></vrae:dateLatest>
      <vrae:culture>
        <rdf:Seq>
          <rdf:li rdf:parseType="Resource">
            <vrae:text></vrae:text>
          </rdf:li>
        </rdf:Seq>
      </vrae:culture>
      <vrae:role>
        <rdf:Seq>
          <rdf:li rdf:parseType="Resource">
            <vrae:text></vrae:text>
          </rdf:li>
        </rdf:Seq>
      </vrae:role>
    </rdf:li>
  </rdf:Seq>
</vrae:work_agentDetails>
```

...like vrae. Those nested structures that match VRA Core? Forget them, they can't be created in the Bridge palette.

vrae flat

```
<vrae:work_agent_1.nameType></vrae:work_agent_1.nameType>  
<vrae:work_agent_1.nameVocab></vrae:work_agent_1.nameVocab>  
<vrae:work_agent_1.nameRefid></vrae:work_agent_1.nameRefid>  
<vrae:work_agent_1.foaf.lastName></vrae:work_agent_1.foaf.lastName>  
<vrae:work_agent_1.foaf.firstName></vrae:work_agent_1.foaf.firstName>  
<vrae:work_agent_1.culture></vrae:work_agent_1.culture>  
<vrae:work_agent_1.role></vrae:work_agent_1.role>  
<vrae:work_agent_1.cdwalite.vitalDates></vrae:work_agent_1.cdwalite.vitalDates>  
<vrae:work_agent_1.dateEarliest></vrae:work_agent_1.dateEarliest>  
<vrae:work_agent_1.dateLatest></vrae:work_agent_1.dateLatest>  
<vrae:work_agent_1.name></vrae:work_agent_1.name>
```

A somewhat faithful expression of VRA Core 4.0 in XMP. It is very easy to program and to import and export to Excel. But, it has some strict limitations.

Repeating elements

```
vrae:work.agent_1.name  
vrae:work.agent_1.nameVocab  
vrae:work.agent_2.nameRefid
```

```
vrae:work.agent_3.name  
vrae:work.agent_2.nameVocab  
vrae:work.agent_2.nameRefid
```

```
vrae:work.agent_3.name  
vrae:work.agent_3.nameVocab  
vrae:work.agent_3.nameRefid
```

To handle repeating fields, like multiple Agents, we numbered them, 1, 2, and three in this case, then added extensions for the subproperties and types we needed.

What if?

```
vrae:work.agent_1.name  
vrae:work.agent_1.nameVocab  
vrae:work.agent_1.nameRefid
```

```
vrae:work.agent_2.name  
vrae:work.agent_2.nameVocab  
vrae:work.agent_2.nameRefid
```

```
vrae:work.agent_3.name  
vrae:work.agent_3.nameVocab  
vrae:work.agent_3.nameRefid
```

agent 4?

But can the flat version work for other people? We predefined three Agents, but what if someone needs a fourth?

Excel works well

	vms:work_agent	vms:work_agent_1:role	vms:work_agent_1:date	vms:work_agent_1:name	vms:work_agent_1:startedDate	vms:work_agent_1:endedDate
1						
2						
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						

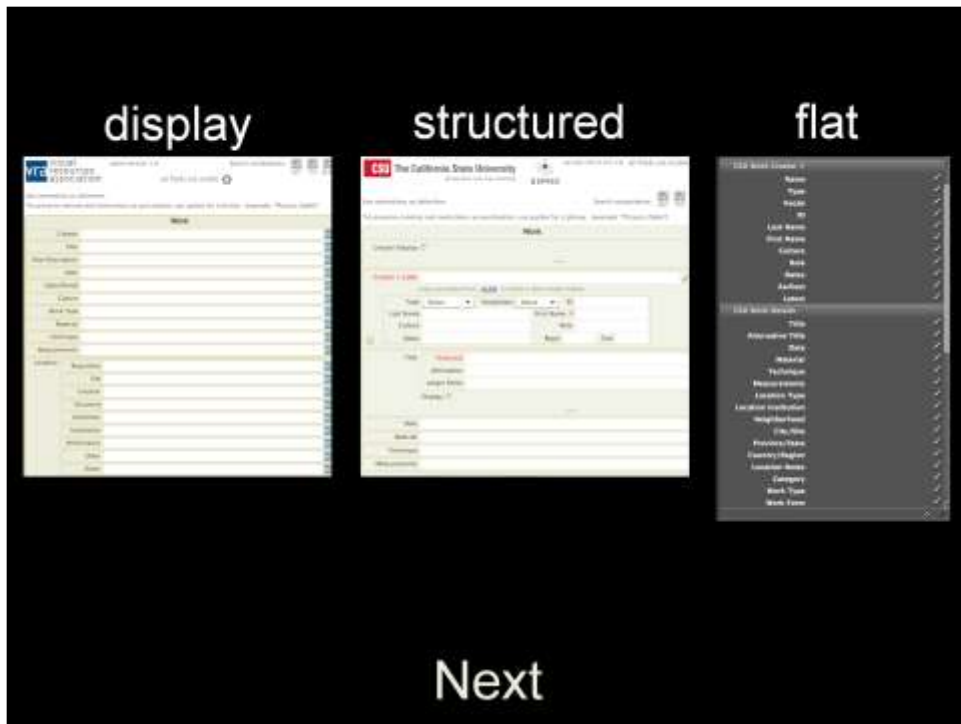
This flat expression of Core 4 works very well for CSU's project because there is a known template. The flat fields drop right into their Excel template. Import and export work very well and the data entry tool was easy to code.

Considerations



theory vs. reality
ideal vs. practical

Scale designed by Nikhil Dev from the Noun Project.



So we have a stable, simple, Core 4 display values info panel which has gained a following.

We have sorted out how to express complex Core 4 in XMP and have a working info panel for it.

We have adapted that to a flat format that is easier to work with, but has some strict limitations. So what do we do with these, choose one and make it public? We could, but now there is another complication (a good one): widespread interest in setting an XMP standard for cultural heritage images. The VRA EMwg doesn't want to put out an XMP standard and tools that might become obsolete or might compete with an international effort – we want to be part of that effort.

IPTC/SCREM

Add fields to IPTC

Establish SCREM

more granular

pick up where IPTC

leaves off

Artwork or Object Depicted

Field Label	Value
16	Source Description Code
17	Medium or City
18	Title
19	Image
20	Image URL
21	Image Date
22	Image Attribution
23	Contributor
24	Contributor Role
25	Contributor Name
26	Contributor Location
27	Image
28	Image Attribution Number
29	Image Number
30	Image Date
31	Image Description
32	Image Title
33	Image URL
34	Image Date
35	Image Attribution
36	Image Number
37	Image Date
38	Image Description
39	Image Title
40	Image URL
41	Image Date
42	Image Attribution
43	Image Number
44	Image Date
45	Image Description
46	Image Title
47	Image URL
48	Image Date
49	Image Attribution
50	Image Number
51	Image Date
52	Image Description
53	Image Title
54	Image URL
55	Image Date
56	Image Attribution
57	Image Number
58	Image Date
59	Image Description
60	Image Title
61	Image URL
62	Image Date
63	Image Attribution
64	Image Number
65	Image Date
66	Image Description
67	Image Title
68	Image URL
69	Image Date
70	Image Attribution
71	Image Number
72	Image Date
73	Image Description
74	Image Title
75	Image URL
76	Image Date
77	Image Attribution
78	Image Number
79	Image Date
80	Image Description
81	Image Title
82	Image URL
83	Image Date
84	Image Attribution
85	Image Number
86	Image Date
87	Image Description
88	Image Title
89	Image URL
90	Image Date
91	Image Attribution
92	Image Number
93	Image Date
94	Image Description
95	Image Title
96	Image URL
97	Image Date
98	Image Attribution
99	Image Number
100	Image Date

First, IPTC have acknowledged the need for more heritage fields and have approved a group to put forward a list of candidate fields. We are in the process of doing this right now. The museums, libraries and visual resource collections we surveyed expressed an interest in a more granular XMP standard, one that picks up where IPTC leaves off, so we are also pursuing that through the SCREM project.

What is SCREM?

It stands for Schema for Rich Embedded Metadata for Media Files and is *the working name* for a wider schema to enable interchange of heritage metadata between and within heritage organisations, to be supported by a custom XMP interface.

SCREM

VRA Core?

LIDO?

schema.org?

Structured?

Flat?

Artwork or Object Depicted

Property	Value Label	Color	Color	Color	Color
16	Accession Number	Blue	Blue	Blue	Blue
19	Accession or ID	Blue	Blue	Blue	Blue
17	Title	Blue	Blue	Blue	Blue
20	Image	Blue	Blue	Blue	Blue
21	Image URL	Blue	Blue	Blue	Blue
22	Image Alt	Blue	Blue	Blue	Blue
23	Image Attribution	Blue	Blue	Blue	Blue
24	Image Description	Blue	Blue	Blue	Blue
25	Image Type	Blue	Blue	Blue	Blue
26	Image Date	Blue	Blue	Blue	Blue
27	Image Location	Blue	Blue	Blue	Blue
28	Image Medium	Blue	Blue	Blue	Blue
29	Image Material	Blue	Blue	Blue	Blue
30	Image Size	Blue	Blue	Blue	Blue
31	Image Weight	Blue	Blue	Blue	Blue
32	Image Height	Blue	Blue	Blue	Blue
33	Image Width	Blue	Blue	Blue	Blue
34	Image Depth	Blue	Blue	Blue	Blue
35	Image Length	Blue	Blue	Blue	Blue
36	Image Volume	Blue	Blue	Blue	Blue
37	Image Area	Blue	Blue	Blue	Blue
38	Image Perimeter	Blue	Blue	Blue	Blue
39	Image Circumference	Blue	Blue	Blue	Blue
40	Image Diameter	Blue	Blue	Blue	Blue
41	Image Radius	Blue	Blue	Blue	Blue
42	Image Thickness	Blue	Blue	Blue	Blue
43	Image Weight	Blue	Blue	Blue	Blue
44	Image Volume	Blue	Blue	Blue	Blue
45	Image Area	Blue	Blue	Blue	Blue
46	Image Perimeter	Blue	Blue	Blue	Blue
47	Image Circumference	Blue	Blue	Blue	Blue
48	Image Diameter	Blue	Blue	Blue	Blue
49	Image Radius	Blue	Blue	Blue	Blue
50	Image Thickness	Blue	Blue	Blue	Blue
51	Image Weight	Blue	Blue	Blue	Blue
52	Image Volume	Blue	Blue	Blue	Blue
53	Image Area	Blue	Blue	Blue	Blue
54	Image Perimeter	Blue	Blue	Blue	Blue
55	Image Circumference	Blue	Blue	Blue	Blue
56	Image Diameter	Blue	Blue	Blue	Blue
57	Image Radius	Blue	Blue	Blue	Blue
58	Image Thickness	Blue	Blue	Blue	Blue
59	Image Weight	Blue	Blue	Blue	Blue
60	Image Volume	Blue	Blue	Blue	Blue
61	Image Area	Blue	Blue	Blue	Blue
62	Image Perimeter	Blue	Blue	Blue	Blue
63	Image Circumference	Blue	Blue	Blue	Blue
64	Image Diameter	Blue	Blue	Blue	Blue
65	Image Radius	Blue	Blue	Blue	Blue
66	Image Thickness	Blue	Blue	Blue	Blue
67	Image Weight	Blue	Blue	Blue	Blue
68	Image Volume	Blue	Blue	Blue	Blue
69	Image Area	Blue	Blue	Blue	Blue
70	Image Perimeter	Blue	Blue	Blue	Blue
71	Image Circumference	Blue	Blue	Blue	Blue
72	Image Diameter	Blue	Blue	Blue	Blue
73	Image Radius	Blue	Blue	Blue	Blue
74	Image Thickness	Blue	Blue	Blue	Blue
75	Image Weight	Blue	Blue	Blue	Blue
76	Image Volume	Blue	Blue	Blue	Blue
77	Image Area	Blue	Blue	Blue	Blue
78	Image Perimeter	Blue	Blue	Blue	Blue
79	Image Circumference	Blue	Blue	Blue	Blue
80	Image Diameter	Blue	Blue	Blue	Blue
81	Image Radius	Blue	Blue	Blue	Blue
82	Image Thickness	Blue	Blue	Blue	Blue
83	Image Weight	Blue	Blue	Blue	Blue
84	Image Volume	Blue	Blue	Blue	Blue
85	Image Area	Blue	Blue	Blue	Blue
86	Image Perimeter	Blue	Blue	Blue	Blue
87	Image Circumference	Blue	Blue	Blue	Blue
88	Image Diameter	Blue	Blue	Blue	Blue
89	Image Radius	Blue	Blue	Blue	Blue
90	Image Thickness	Blue	Blue	Blue	Blue
91	Image Weight	Blue	Blue	Blue	Blue
92	Image Volume	Blue	Blue	Blue	Blue
93	Image Area	Blue	Blue	Blue	Blue
94	Image Perimeter	Blue	Blue	Blue	Blue
95	Image Circumference	Blue	Blue	Blue	Blue
96	Image Diameter	Blue	Blue	Blue	Blue
97	Image Radius	Blue	Blue	Blue	Blue
98	Image Thickness	Blue	Blue	Blue	Blue
99	Image Weight	Blue	Blue	Blue	Blue
100	Image Volume	Blue	Blue	Blue	Blue

Obviously we have solved a lot of the problems with vrae and it could serve a starting point for SCREM. However, we want buy-in from as many people as possible, so we will consult with a wide range of experts on what schema will serve as the foundation and how it will be structured. We will also get a broad consensus on what fields to include.



Matthias Arnold, Heidelberg Research Architecture – Visual Resources
Marta Bustillo, National College of Art and Design (Ireland)
Heidi Eyestone, Carleton College
Sheryl Frisch, Cal Poly, San Luis Obispo
Stephen Jennings, Fine Arts Library, Harvard
Heather Lowe, California State University San Bernardino
Heidi Raatz, Minneapolis Institute of Arts
Amanda Rybin, The University of Chicago Department of Art History
Greg Reser, UCSD
Steve Tatum, Virginia Tech