Faceted Search

Faceted Search

To aid in discovery and to take advantage of rich fielded metadata, many modern search applications have some concept of faceted searching or browsing. This typically manifests itself as a list of categories into which some subset of the current search results belong.

Terminology:

Facet - (also called "facet field") is a field or category for which search results have one or more values. In general, this maps on to one or more indexed fields.
Facet value - is a possible value that one or more search results express in the given **facet**

SRU

There are no published or popular faceted search extensions for SRU or CQL, so in order to accommodate the needs of our various collections we must establish our own conventions or context sets.

Implementation details

Facet Request Syntax

For our first pass implementation, to meet the immediate needs of our IN Harmony collection, we must be able to convey the following information in an SRU searchRetrieveRequest along with all of the details about the search:

1. the facets to calculate
2. paging information for each facet (currently this only includes an optional bound on the number returned)

After receiving the information and performing the search and facet calculation, the following information must be conveyed back in the response:

1. the facets that were calculated (probably mirroring all categories in the order requested)
2. the facet values for each facet that had at least one hit in the search result set (constrained by the limit set in the request)
3. the number of hits for each facet value in the result set
4. the cql query clause that must be ANDed with the original query in order to search for the hits for the given facet value (this can simply be derived "[facetField]=[facetValue]")
5. A normalized version of the requested facet parameters, to prevent the client from having to maintain that state, and to show what default paging parameters were applied by the server (in the even that they weren't specified).

In SRU/W custom extensions may be created using a limited syntax (SRU 1.2 extension information).

Proposed Syntax:

Because the syntax has to fit into a single URL, complex structures can't easily be encoded and we may run up against URL length limits if we're too wordy.

Parameter Name	x-iudl-requestFacetInformation
Value syntax	[facetName1],[maxValueCount],[offset] [facetName2],[maxValueCount],[offset]
Options	The maxValueCount and offset are optional paging parameters. If excluded, the server decides the paging rules which will be explicitly listed in the response.
Example value	dc.format dc.date,10, dc.subject,10,10
Explanation	A request for all of the represented dc.format values, the first 10 represented dc.date values and the second page (of length 10) of the dc.subject values represented in the search results. (Sort order for paging is left up to the server, but is generally expected to be alphabetical. If sort order is a problem, the client always has the option to request all values, and handle sorting and paging at their end.

Example response

```
<extraResponseData xmlns:ns4="http://www.loc.gov/zing/srw/">
  <ns5:facetInformation xmlns="http://www.dlib.indiana.edu/xml/sruFacetedSearch/version1.0/"
    xmlns:ns5="http://www.dlib.indiana.edu/xml/sruFacetedSearch/version1.0/">
    <ns5:field name="dc.format">
      <ns5:value hits="14421">35mm slide</ns5:value>
      <ns5:value hits="14421">image/jpeg</ns5:value>
    </ns5:field>
    <ns5:field name="dc.date">
      <ns5:value hits="26">1938-09</ns5:value>
      <ns5:value hits="1">1938-09-01</ns5:value>
      <ns5:value hits="5">1938-09-03</ns5:value>
      <ns5:value hits="4">1938-09-04</ns5:value>
      <ns5:value hits="1">1938-09-09</ns5:value>
      <ns5:value hits="16">1938-09-10</ns5:value>
      <ns5:value hits="7">1938-09-17</ns5:value>
      <ns5:value hits="3">1938-09-18</ns5:value>
      <ns5:value hits="3">1938-09-21</ns5:value>
      <ns5:value hits="6">1938-09-22</ns5:value>
    </ns5:field>
    <ns5:field name="dc.subject">
      <ns5:value hits="1">Abraham Grapheus</ns5:value>
      <ns5:value hits="2">Abutments</ns5:value>
      <ns5:value hits="19">Acacia</ns5:value>
      <ns5:value hits="15">Acadia National Park (Me.)</ns5:value>
      <ns5:value hits="1">Acanthi</ns5:value>
      <ns5:value hits="2">Acanthopanax ricinifolius</ns5:value>
      <ns5:value hits="1">Acanthus mollis</ns5:value>
      <ns5:value hits="36">Accidents</ns5:value>
      <ns5:value hits="1">Accordions</ns5:value>
      <ns5:value hits="1">Accreted terrain</ns5:value>
    </ns5:field>
    <ns5:requestInfo>
      <ns5:originalRequest>dc.format dc.date,10 dc.subject,10,10</ns5:originalRequest>
      <ns5:resolvedRequest>
        <ns5:facet maxValues="-1" name="dc.format" offset="0"/>
        <ns5:facet maxValues="10" name="dc.date" offset="0"/>
        <ns5:facet maxValues="10" name="dc.subject" offset="10"/>
      </ns5:resolvedRequest>
    </ns5:requestInfo>
  </ns5:facetInformation>
</extraResponseData>
```

Usage Notes

Some of the specific collection needs can't be implemented generally enough in the SRU/W server but instead must be implemented by the web application. If a reasonably general way of implementing the needed functionality can be devised, it would be ideal to move it to the SRU/W layer, but for complex cases that aren't reusable, it is a bad precedent to implement the feature in SRU/W.

The following are strong reasons to implement something at the SRU level:

1. The feature is general enough that it can be defined fully and made useful for other applications.
2. Required performance is unable to be accomplished from a less central location.

The following are strong reasons to implement something at the web application level:

1. The feature is difficult or impossible to define in general terms.
2. The feature isn't generally useful.
3. The feature requires extensive, project-specific configuration to the search system

Needed Feature	Implementation suggestion
Grouping of 5 year periods in a copyright date facet.	The web application should request all facet information for copyright date (this will rarely be even 100s of searches and shouldn't kill performance) and then consolidate by half-decade, creating queries by ORing the query fragments for each result.
Knowing when to display a button to request all facets (ie, knowing when there are more, undisplayed results)	The web application should request x + 1 facets for each field where x is the number to display. The (x + 1)th result shouldn't be displayed, but if it's present, a button should exist saying "Display all".