Browse & Search

Browse

DSpace-CRIS extends the DSpace browse system to all the CRIS entities.
This means that the user is able to configure browsing on researchers, organization units, projects and second level dynamic objects such as Journals, Events, Laboratories, and so on for any entity that the user has defined.
The configuration syntax follows the same rules of the standard DSpace configuration.
There are two types of browse:

- **Full**, single level browse, which just list in a specific order the instances of an Entity class: Researchers, OrgUnits, etc.
- **Metadata**, two levels browse, that provide a first page listing the values of a configured metadata leading to a second page where the instances that have the clicked value for that metadata are listed

Full, single level browse

The syntax to configure full browse (single level) is

```
webui.browse.index.<n> = <index-name>:<display-type>:<sort-name>[:DESC]
```

index-name is used to refer to further configurations as the list of columns to show and generate the i18n keys for the navigation (menu links, page header, etc.)

display-type can be anything except metadata and metadataAuthority that are reserved word for the two level browse. It is recommended to use the prefix the shortname of the Class with the word cris i.e. crisrp for researchers, crisou for orgunits, crispj for project and finally cris<shortname of the dynamic object>; again, this is just a convention to make the configuration more readable but any word than metadata* or metadataAuthority can be used to define a full browse. A general name is useful when you want create several browse on the same entity (for example org unit) that explore different subsets (i.e. internal structure, external organization, etc.). Indeed, using the browse name it is possible to define filter to apply to the general SOLR query, see “Apply filters to the browse indexes”

sort-name is used to refer to the sorting configuration described below *DESC* if used make the descending order the default for that browse

```
webui.itemlist.sort-option.<n> = <sort-name>:<metadata>:<value-type>
```

metadata to give access to the properties of the cris objects using a DSpace like metadata syntax you need to prefix the shortname of the PropertiesDefinition with the shortname of the entity class (same of above, crisrp, crisou, etc.). So if you want to use for sorting the property defined with shortname “myproperty” in researcher entity you need to write crisrp.myproperty

When the property that you are referring to is a pointer you can use the qualifier to access to property of the pointed object, i.e. crisrp.dept.name will be the name of the orgunit assuming that dept is the shortname of a property that keep the relation between the researcher and the orgunits. You can also use virtual metadata, see “ItemEnhancer: virtual metadata”.

value-type: can be one of title, text, date or any other alias used to configure a Sort Plugin, see Configuration Reference#DefiningSortOptions

To configure which information/columns are displayed for any objects is possible to use the following configuration property

```
webui.itemlist.<entity >.columns = <field1>(rendering), field2 (rendering), … fieldN
```

for example:

```
webui.itemlist.crisrp.columns = crisrp.fullName(cristitle), crisrp.translatedName, crisrp.email
```

the configuration defined for a specific entity (crisrp, item, etc.) can be overridden for a specific browse using the following configuration

```
webui.itemlist.browse.rpname.columns = crisrp.fullName(cristitle), crisrp.email
```

Metadata, two levels browse

As for the DSpace item it is possible to define browse over a metadata (i.e. authors of an item). In this way the system will produce a two levels browse, the first level will show in a paginated way all the values of the metadata (i.e., all the authors) clicking on a single value will show the list of items that match the selection. Applying this concept to the CRIS entities, you can for example build a two level browse showing all the departments of the researchers and for any department the list of researchers affiliated

```
webui.browse.index.<n> = <index-name>:metadata|metadataAuthority|metadataXXXX:<schema.element.qualifier>:
<text|date>
```

for example

```
webui.browse.index.6 = rpdept:metadata:crisrp.dept:text
```

index-name is used as reference in the column definition configuration to apply specific configuration for that browse

metadata is used to build a browse on any values used with or without authority

- *metadataAuthority* limit the browse to only the value with an authority key
- *metadataXXXX* where XXXX can be anything behave as metadata allowing separe definition of default filtering for the browse (see next section)

schema.element.qualifier defines the field upon which the browse is build

text|*date* specify if the values must be interpreted as String or Dates for sorting

Apply filters to the browse indexes

It could be useful to restrict the set of objects for a specific browse applying additional SOLR filter query. To configure a filter for a specific browse you can define the following configuration property

```
browse.solr.bi_<display-type>.filter = <your-solr-filter-query>
```

display-type is the value of the second part of the browse configuration. It is metatadata, metadataAuthority or metadata<Something> for two levels browse or something else for the configuration of a full browse index.
For example

```
browse.solr.bi_item.filter = dateissued:[2000 TO *]
```

It will limit the browse to the items published after the 2000,

```
browse.solr.crisou.filter = crisou.relationwith_authority:"ou00001"
```

It will limit the browse to the OrgUnits that have the field “relationwith” valorized with a pointer to the OrgUnit ou00001.

When you are limiting a two level browse you need to configure, typically the same filter, also for the second level. In such case the browse index is used

```
browse.solr.bi_<n>_dis.filter = <your-solr-filter-query>
```

For example

```
webui.browse.index.2 = author:metadataauthor:dc.contributor.*,dc.creator:text
browse.solr.bi_metadataauthor.filter = dateissued:[2000 TO *]
browse.solr.bi_2_dis.filter = dateissued:[2000 TO *]
```

will limit the browse to contains the authors names of only item published from the 2000 on and to list under such names only these items

ItemEnhancer: virtual metadata

The enhancers allow to access information not immediately available in an object (item or cris object) as it was a direct metadata/field of the object. In this way it is for example possible to access the item author’s affiliation as it was an item metadata.
There are three type of enhancer:

- **org.dspace.app.cris.discovery.CrisItemEnhancer**, used to add dynamically to the item information from linked CRIS entities as it was metadata of the item
- **org.dspace.app.cris.discovery.CrisEnhancer**, used to add dynamically to the CRIS entity information from linked CRIS entities as it was metadata of the current CRIS entity
- **org.dspace.app.cris.discovery.CrisNestedEnhancer**, used to add dynamically to the CRIS entity information from the nested CRIS entities as it was metadata of the current CRIS entity

The enhancers are configured in the spring file **[installDir]/config/spring/cris/cris-metadata-enhancers.xml**
Below are included the snipped of an CrisItemEnhancer in the default configuration

```
<bean class="org.dspace.app.cris.discovery.CrisItemEnhancer">
<property name="alias" value="author" />
<property name="metadata">
<list>
<value>dc.contributor.author</value>
</list>
</property>
<property name="clazz">
<value>org.dspace.app.cris.model.ResearcherPage</value>
</property>
<property name="qualifiers2path">
<map>
<entry key="dept">
<value>dept</value>
</entry>
</map>
</property>
</bean>
```

The properties have the following meaning

- alias, is the “element” of the virtual field
- qualifiers2path, is a map whose key are the “qualifier” of the virtual field and the value the path, evaluated as EL, in the target object
- metadata, is the list of item metadata to consider that linkage the item with the CRIS object
- clazz, is the canonical name of the linked CRIS Entity class

The previous example will create the virtual metadata crisitem.author.dept that will be evaluated as it contains the department name of the researcher linked to the item through the metadata dc.contributor.author as text value and the CRIS ID of the department as authority.

The following snippet show a CRISEnhancer and a CRISNestedEnhancer

```
<bean class="org.dspace.app.cris.discovery.CrisEnhancer">
<property name="clazz">
<value>org.dspace.app.cris.model.Project</value>
</property>
<property name="alias" value="coinvestigators" />
<property name="qualifiers2path">
<map>
<entry key="dept">
<value>dept</value>
</entry>
</map>
</property>
</bean>
<bean class="org.dspace.app.cris.discovery.CrisNestedEnhancer" scope="prototype">
<property name="applicationService" ref="applicationService" />
<property name="nestedClazz" value="org.dspace.app.cris.model.jdyna.ProjectNestedObject" />
<property name="clazz">
<value>org.dspace.app.cris.model.Project</value>
</property>
<property name="alias" value="grant" />
<property name="qualifiers2path">
<map>
<entry key="agencies">
<value>agencies</value>
</entry>
</map>
</property>
</bean>
```

The first one allow to use crispj.dept instead of cripj.coinvestigators.dept in this way the field of the target OrgUnit are available for indexing when working on the Project.
The second one allows access to the agencies (the value) of the nested object named grant (the alias) using the metadata crispj.agencies where agencies is the key of the qualifiers2path map.

> ⚠ **Authority for CRIS**
>
> The enhancers on CRIS object expose the object ID as authority instead of the CRIS ID. This mean that in the previous example the crisph.dept will be a metadata with the name of the department as text value and the Orgunit database ID as authority

Search configuration

Indexes, facets

The discovery module used by DSpace-CRIS has been extended to be able to manage also CRIS Entities. In the [installDir]/config/spring/api/discovery-solr.xml file a new implementation of the SearchService/ IndexingService has been defined

```
<bean class="org.dspace.app.cris.discovery.CrisSearchService" id="org.dspace.discovery.SearchService"/>
```

New special entries can be used in the definition of the DiscoveryConfigurationService in the [installDir]/config/spring/api/discovery.xml file to allow specific configuration for entity type:

```
<bean id="org.dspace.discovery.configuration.DiscoveryConfigurationService"
 class="org.dspace.discovery.configuration.DiscoveryConfigurationService">
 <property name="map">
  <map>
   <entry key="default" value-ref="defaultConfiguration" />
   <entry key="site" value-ref="homepageConfiguration" />
   <entry key="dspacebasic" value-ref="dspaceConfiguration" />
   <entry key="crisrp" value-ref="crisRPConfiguration" />
   <entry key="crisproject" value-ref="crisPJConfiguration" />
   <entry key="crisou" value-ref="crisOUConfiguration" />
   <!-- <entry key="crisjournal" value-ref="crisDOJournalConfiguration" /> -->
   <!-- <entry key="123456789/7621" value-ref="defaultConfiguration"/> -->
  </map>
  …
```

The map containing all the settings, the key is used to refer to the page/scope of the search, the "site", a community/collection handle or an entity type, the value-ref is a reference to a spring bean that actually define the DiscoveryConfiguration format.

- **default** is the configuration key used if a not more specific one exist for the current search/indexing scope
- **dspacebasic** is the fallback configuration key for all the standard DSpace Objects: Items, Collections and Community
- **crisrp** is the configuration key used searching/indexing a ResearcherPage
- **crispj** is the configuration key used searching/indexing a Project
- **crisou** is the configuration key used searching/indexing an OrgUnit
- **cris<shortname>** is the configuration key used searching/indexing a Dynamic Object with <shortname>, for example journal

The searching scope is defined by the UI implicitly when the search is performed from a “specific page” as a community or collection home page or explicitly when the user choose to restrict the search to a specific subset.

During the indexing phase the scope/configuration key is defined as follow

- Standard DSpace objects such Items, Collections and Communities, the configuration key is the handle of the object if not found the parent object handle will be used as fallback until to the dspacebasic or default key
- CRIS Objects such ResearcherPages, OrgUnits, Projects, Dynamic Objects, the configuration key is defined the by the entity type, prefixed with cris (ie crisrp, crisou, crisjournals, etc.), with fallback to the default key

To configure a DiscoverySearchFilter, DiscoverySearchFilterFacet, DiscoverySortFieldConfiguration the “DSpace like metadata syntax” previously explained, see metadata in the Browse customization.

Custom Indexer Plugins

The following IndexerPlugins has been added in the **[installDir]/config/spring/api/discovery.xml** file

Additional indexing plugin to implement the CRIS browse system via SOLR

```
<bean id="crisBrowseIndexer" scope="prototype"
 class="org.dspace.app.cris.discovery.CrisBrowseSolrIndexPlugin">
</bean>
```

Additional indexing plugin to implement CRIS relations preferences, see “Relation Preference Management”, via SOLR

```
<bean id="crisRelationPreferenceIndexer" class="org.dspace.app.cris.discovery.
RelationsPreferencesSolrIndexPlugin">
 <property name="applicationService" ref="applicationService" />
</bean>
```

Additional indexing plugin to add bitstream identifier to item SOLR document, this is used by the extended statistics functionalities as bitstream download are associated to the item using SOLR join query, see “Statistics configuration” for further details

```
<bean id="bitstreamIndexer" class="org.dspace.app.cris.discovery.BitstreamSolrIndexer" />
```

Additional indexing plugin to add resource type in human format readable to SOLR document used to cluster types in the global search

```
<bean id="resourceTypeIndexer" class="org.dspace.app.cris.discovery.ResourceTypeSolrIndexer" />
```

Additional indexing plugin to add cris authoritylookup for authority framework works with dynamic resource type to SOLR document

```
<bean id="researchObjectAuthorityLookupIndexer"  class="org.dspace.app.cris.discovery.
ResearchObjectAuthorityLookupSolrIndexer />
```

Global search

When a generic search is performed in DSpace-CRIS all the entities and the information available in the system are queried. The result are aggregated by type. Inside each type the result are sorted by relevance. The types are defined by a special facet containing the values as derived from the configuration in **[installDir]/config/modules/cris.cfg**

all the configuration are in the format

```
facet.type.<entity> = Value to store in the SOLR document in the format sortvalue\n|||\nDisplayed
value###authority
```

the authority is the key of the Discovery configuration map to use when the search is restricted or the results limited to a specific entity.

Entity can be one of item, community, collection, crisrp, crispj, crisou

```
facet.type.item = 001publications\n|||\nPublications###publications
facet.type.community = 007communities\n|||\nCommunities###community
facet.type.collection = 006collections\n|||\nCollections###collection
facet.type.crisrp = 009researchers\n|||\nResearcher profiles###researcherprofiles
facet.type.crispj = 010projects\n|||\nFundings###fundings
facet.type.crisou = 008orgunits\n|||\nOrganizations###orgunits
```

For additional entities, 2nd level CRIS object (ResearchObject), the shortname of the entity prefixed with crisdo. is used (i.e. crisdo.crisprize)

```
##You can define a clustered type for other research object or set a fine granularity configuration custom -
use @label@ to use default behaviours (<typedefinitionlabellowercase>|||<typedefinitionlabel>)
facet.type.crisdo.default = others\n|||\nOthers###crisothers
#facet.type.crisdo.crisprize = prize\n|||\nHonours, Awards and Prizen###crisprizefacet.type.crisdo.crisjournals
= @label
```

It is also possible to create "virtual" defined entity from DSpace item on a dc.type (or other metadata) basis

The exact metadata to use is defined by

```
global.item.typing = dc.type
```

the metadata value, lowercase, whitespace stripped is used as "entity" in the previous configuration properties, i.e.

```
facet.type.thesis = 005theses\n|||\nTheses###theses
facet.type.patents = 004patents\n|||\nPatents###patents
facet.type.dataset = 003dataset\n|||\nDatasets###datasets
facet.type.conferenceproceeding = 002conferencematerials\n|||\nConference Materials###conferencematerials
facet.type.conferencepaper = 002conferencematerials\n|||\nConference Materials###conferencematerials
facet.type.presentation = 002conferencematerials\n|||\nConference Materials###conferencematerials
facet.type.poster = 002conferencematerials\n|||\nConference Materials###conferencematerials
```

Result rendering and highlighting

The result of a global search are displayed using a new DSpace Tag, **dspace:discovery-artifact**, that show each entry as a list item with a heading and a description. Using spring beans it is possible to customize which fields are displayed for each result, both in the heading than in the description. For each field it is possible to decide if the information are always shown or included it only when in this specific field there is a match with the searched terms.

The default configuration is defined in the **[installDir]/config/spring/api/discovery-global.xml**

```
<bean name="global"
                class="org.dspace.discovery.configuration.DiscoveryViewAndHighlightConfiguration">
                <property name="viewConfiguration">
                        <map>
                                <entry key="community">
                                        <bean
                                                class="org.dspace.discovery.configuration.
DiscoveryViewConfiguration">
                                        ...
```

for each entity type, as defined in the global search, it is possible to specify a separate configuration

```
<bean
        class="org.dspace.discovery.configuration.DiscoveryViewConfiguration">
        <property name="metadataHeadingFields">
                <list>
                        <bean
                                class="org.dspace.discovery.configuration.DiscoveryViewFieldConfiguration">
                                <property name="mandatory" value="true"/>
                                <property name="preHtml" value="&lt;h4 class='text-primary'&gt;" />
                                <property name="postHtml" value="&lt;/h4&gt;" />
                                <property name="field" value="name" />
                                <property name="decorator" value="title" />
                        </bean>
                </list>
        </property>
        <property name="metadataDescriptionFields">
                <list>
                        <bean
                                class="org.dspace.discovery.configuration.DiscoveryViewFieldConfiguration">
                                <property name="field" value="dc.description.abstract" />
                                <property name="preHtml" value="&lt;span&gt;" />
                                <property name="postHtml" value="&lt;/span&gt;" />
                        </bean>
                </list>
        </property>
</bean>
```

Examples

Add facets to all Communities' or Collections' Browse boxes

The default Browse box that is displayed when viewing a community includes five buttons that determine what will be displayed in the browse list: Issue Date, Author, Title, Subject and Department.

The contents of these browse-lists depend on indexes that are configured in dspace.cfg with the webui.browse.index.<n> label:

```
webui.browse.index.1 = dateissued:item:dateissued
webui.browse.index.2 = author:metadata:dc.contributor.*,dc.creator:text
webui.browse.index.3 = title:item:title
webui.browse.index.4 = subject:metadata:dc.subject.*:text
webui.browse.index.5 = rpname:crisrp:rpname
webui.browse.index.6 = rpdept:metadata:crisrp.dept:text
webui.browse.index.7 = pjtitle:crisproject:crisproject
webui.browse.index.8 = ouname:crisou:ouname
webui.browse.index.9 = itemdept:metadata:crisitem.author.dept:text
webui.browse.index.10 = type:metadata:dc.type:text
```

The buttons displayed in the Browse boxes of the communities and collections are determined by the webui.browse.community.index and webui.browse.collection.index fields:

```
webui.browse.community.index = 1,2,3,4,9
webui.browse.collection.index = 1,2,3,4,9
```

Accordingly, the Issue Date, Author, Title, Subject and Department buttons are displayed in the communities and collections Browse boxes.

Assuming your metadata has an appropriate field defined, it can be used to populate an additional index, which can then be viewed using a Browse button. This example will demonstrate adding a 'Submit Date' button.

First add the webui.browse.index with an appropriate sequential number:

```
webui.browse.index.11 = dateaccessioned:item:dateaccessioned
```

Next, add that number to the community and/or collection indexes:

```
webui.browse.community.index = 1,2,3,4,9,11
webui.browse.collection.index = 1,2,3,4,9,11
```

Restart Tomcat, then recreate the discovery indexes to populate them with data from the dc.date.accessioned field:

```
bin/dspace index-discovery -b -f
```

Add links to the Browse section in the Research Outputs page

The default Research Outputs Browse list consists of the following facets: Department, Author, Title, Type, Issue Date, and Subject. It is possible to add any already defined indexes to this list.

The contents of the list is specified in **config/spring/cris/cris-processor.xml**:

```
        <bean class="org.dspace.app.webui.cris.components.ExploreMapProcessors" id="org.dspace.app.webui.cris.
components.ExploreMapProcessors">
                <property name="processorsMap">
                        <map>
                                <entry key="publications">
                                        <list>
                                                <bean class="org.dspace.app.webui.cris.components.
BrowseProcessor">
                                                        <property name="browseNames">
                                                                <list>
                                                                        <value>itemdept</value>
                                                                        <value>author</value>
                                                                        <value>title</value>
                                                                        <value>type</value>
                                                                        <value>dateissued</value>
                                                                        <value>subject</value>
                                                                </list>
                                                        </property>
                                                </bean>
```

Additional facets can be added to the list, using the name defined in the index (discussed in the 'Add facets to all Communities' or Collections' Browse boxes' example):

```
<list>
        <value>itemdept</value>
        <value>author</value>
        <value>title</value>
        <value>type</value>
        <value>dateissued</value>
        <value>subject</value>
        <value>dateaccessioned</value>
</list>
```

Restart Tomcat for the changes to take effect.