Dynamic SOLR Variables - Indexing and Searching Shelving Locator

In the Roman Coins project, we have a requirement to search the shelving locator – either "Catalog Number" or "Box Number" (see the Source XML extract below). Those who are ingesting coin images or maintaining the coins digital collection would typically conduct these searches as opposed to the end user. These fields are not part of the SOLR XML schema and so dynamic fields will have to be created. Other projects are likely to have the same requirement so a generic solution is needed.

We have in the past used the dynamic capability in SOLR to create special variables. This process has been very effective in quickly creating new variables and avoiding the more cumbersome process of modifying the SOLR XML schema. It is also likely that there are many other fields in the various metadata sections that are not yet indexed and will need attention in the future. This draft proposes a solution for Shelving Locator and also suggests a process for handling similar situations.

Implementation Considerations

Dynamic Variables. As a starting point, a generic syntax for creating dynamic variables is proposed. For the shelving locator, consider the following options: 1) creating two SOLR variables – one each for catalog number and box number and 2) creating a single variable that allows a search of the catalog number and box number together. (Alternatively, a user could use a Boolean AND function to accomplish the same search.)

The syntax might look something like the following: [metadata section]_[xml-level1]_[xml-level2]_[type-attribute]. A heuristic could be used for the type attribute as follows: 1) for a two word phrase, use the first three letters of each word or 2) for a single term, just use the first three characters of the term[1]. For the Roman Coins XML example shown below, the two variables could be named as follows: source_shelving_locator_catnum_st and source_shelving_locator_boxnum_st. The "_st" is a SOLR convention to indicate that the variable is a multivalued string[2]. For a combined search of both fields, the variable might be source_shelving_locator_st. The conventions to be used for the metadata section follow the five sections of our METS wrapper and are as follows: mods, source, rights, technical, and digiprov. The generic naming convention provides intelligence to the code reader and also works for all portals, independent of subject content.

A Process for Naming SOLR Variables. Another example is included here from the Rights section that is slightly different in that a type attribute is not included. The objective is to see how the syntax

[1] Obviously this won't work for all cases but we can use a similar approach for other variants.

[2] The primary SOLR suffix conventions used in RUCore are as follows: _txt for a multivalued text variable, _i for uni-valued integer variable, _s for a uni-valued string, _st for a multivalued string, and _dt for a uni-valued UTC date and time field.

and naming convention might be used generally across the multiple sections of metadata in RUcore. In the Rights example (XML shown below), SOLR dynamic variables for searching copyright status and availability could be created as follows: 1) rights_copyright_status_txt and 2) rights_availability_status_txt

Conclusion

This specification proposes a generic solution for naming dynamic SOLR variables and a specific solution for solving the more immediate administrative searching requirement for the Roman Coins project. A task will also be added to software libraries to revisit the SOLR XML schema and consider transforming other variables to the dynamic state. This action will be targeted for the R7.8 release.

XML Example from the Roman Coins Project – Source Metadata

Catalog Number.

```
<rulib:shelving>
        <rulib:locator TYPE="Catalog Number">BAD0451</rulib:locator>
 </rulib:shelving>
```

Box Number.

```
<rulib:shelving>
        <rulib:locator TYPE="Box Number">26</rulib:locator>
</rulib:shelving>
```

XML Example from ETDs – Rights Metadata

Status.

```
<rulib:copyright>
        <rulib:status>Copyright protected</rulib:status>
 </rulib:copyright>
```

Availability.

```
 <rulib:availability>
        <rulib:status>Open</rulib:status>
 </rulib:availability>
```

rcj/jat/cmm – 05/12/2015