Extractor UI Parameter Passing (defaults)

I've encountered a number of situations where the ability to influence the execution of an extractor beyond what is available via file/dataset information and metadata would be convenient.

Having a way to pass additional parameters to an extractor could increase their utility and usefulness while allowing a single extractor to do more without having to be duplicated.

An example could be the image metadata extractor. ImageMagick currently extracts a "ton" of information about an image. Everyone seems to like the idea of having automatic image metadata extraction happening. However, many folks would prefer not get all of the metadata but rather a subset. Imagine if we could pass a set of parameters to image metadata extractor to instruct it to only extract certain terms that we were interested in. Another user could use the same extractor but pass a different list of terms. There are of course many other examples where parameters could be useful.

I believe others have talked about adding this capability so this isn't a new idea.

One question would be, how would it be best to implement the passing of parameters, since there could be different reasons you want to pass them.

1. You may want to pass specific parameter(s) that applies to the auto invocation of an extractor across the entire Clowder instance.
2. You may want to pass specific parameter(s) that applies to the auto invocation of an extractor within a space.
3. You may want to pass specific parameter(s) that applies to the auto invocation of an extractor within a dataset.
4. You may want to pass specific parameter(s) when you manually submit an extractor.
5. others?

Also, where is the logical place to provide these parameters?

1. You could provide an additional link/button next to the extractor in the extractor list that allows you to pop-up a text box and add and save parameters.
2. You could provide a pop-up text box that button/link associated with the instance/space/dataset that allowed you to add and save parameters.
3. You could have a special dataset or file that a user can edit that contains the parameters that is made available to extractors.
4. You could pop-up a dialog when the user submits an extractor manually that allows for parameters to be specified.
5. You could mix some of the options to create a parameter hierarchy (e.g. manually provided parameters override space default parameters, etc.)

Personally, I like the idea of having the concept of "parameters" that can be specified at the instance and space level (possibly user level too?) for more than just extractors. Extractors could be one of the uses, but the use could be extended to other configuration options/overrides within the Clowder app itself. Similar to CSS.

For example (sorry, but I always visualize things like this as XML files)...

(Instance Level)

```
<extractor name="ncsa.image.metadata" >
     <parameters>
          <term name="WinXP-Author" metaname="Author" />
          <term name="Make"/>
          <term name="Model"/>
     </paramaters>
</extractor>
```

(Space Level - Space1234)

```
<extractor name="ncsa.image.metadata" override="true">
     <parameters>
          <term name="Artist" metaname="Author"/>
          <term name="ImageDescription" metaname="Description"/>
     </paramaters>
</extractor>
```

Would define that for ALL spaces the ncsa.image.metadata extractor should extract WinXP-Author (stored as Author), Make and Model. HOWEVER for space (Space1234) this is overridden and Artist (stored as Author) and ImageDescription (stored as Description) is extracted. If override had been false or omitted, all parameters would have been passed to the extractor and it would have been up to the extractor logic to decide how to handle the conflicting "Author" parameters.

Please share your comments and ideas!