Embargo

1 What is an embargo?
- 1.1 Embargo model and life-cycle
  - 1.1.1 Terms assignment
  - 1.1.2 Terms interpretation/imposition
  - 1.1.3 Embargo period
  - 1.1.4 Embargo lift
  - 1.1.5 Post embargo
- 1.2 Configuration
- 1.3 Operation
- 1.4 Step-by-Step Setup Examples
- 1.5 Extending embargo functionality
  - 1.5.1 Setter
  - 1.5.2 Lifter

What is an embargo?

An embargo is a temporary access restriction placed on content, commencing at time of accession. It's scope or duration may vary, but the fact that it eventually expires is what distinguishes it from other content restrictions. For example, it is not unusual for content destined for DSpace to come with permanent restrictions on use or access based on license-driven or other IP-based requirements that limit access to institutionally affiliated users. Restrictions such as these are imposed and managed using standard administrative tools in DSpace, typically by attaching specific policies to Items or Collections, Bitstreams, etc. The embargo functionally introduced in 1.6, however, includes tools to automate the imposition and removal of restrictions in managed timeframes.

Embargo model and life-cycle

Functionally, the embargo system allows you to attach 'terms' to an item before it is placed into the repository, which express how the embargo should be applied. What do 'we mean by terms' here? They are really any expression that the system is capable of turning into (1) the time the embargo expires, and (2) a concrete set of access restrictions. Some examples:

"2020-09-12" - an absolute date (i.e. the date embargo will be lifted)
"6 months" - a time relative to when the item is accessioned
"forever" - an indefinite, or open-ended embargo
"local only until 2015" - both a time and an exception (public has no access until 2015, local users OK immediately)
"Nature Publishing Group standard" - look-up to a policy somewhere (typically 6 months)

These terms are 'interpreted' by the embargo system to yield a specific date on which the embargo can be removed, or 'lifted', and a specific set of access policies. Obviously, some terms are easier to interpret than others (the absolute date really requires none at all), and the 'default' embargo logic understands only the most basic terms (the first and third examples above). But as we will see below, the embargo system provides you with the ability to add in your own 'interpreters' to cope with any terms expressions you wish to have. This date that is the result of the interpretation is stored with the item and the embargo system detects when that date has passed, and removes the embargo ("lifts it"), so the item bitstreams become available. Here is a more detailed life-cycle for an embargoed item:

Terms assignment

The first step in placing an embargo on an item is to attach (assign) 'terms' to it. If these terms are missing, no embargo will be imposed. As we will see below, terms are carried in a configurable DSpace metadata field, so assigning terms just means assigning a value to a metadata field. This can be done in a web submission user interface form, in a SWORD deposit package, a batch import, etc. - anywhere metadata is passed to DSpace. The terms are not immediately acted upon, and may be revised, corrected, removed, etc, up until the next stage of the life-cycle. Thus a submitter could enter one value, and a collection editor replace it, and only the last value will be used. Since metadata fields are multivalued, theoretically there can be multiple terms values, but in the default implementation only one is recognized.

Terms interpretation/imposition

In DSpace terminology, when an Item has exited the last of any workflow steps (or if none have been defined for it), it is said to be 'installed' into the repository. At this precise time, the 'interpretation' of the terms occurs, and a computed 'lift date' is assigned, which like the terms is recorded in a configurable metadata field. It is important to understand that this interpretation happens only once, (just like the installation), and cannot be revisited later. Thus, although an administrator can assign a new value to the metadata field holding the terms after the item has been installed, this will have no effect on the embargo, whose 'force' now resides entirely in the 'lift date' value. For this reason, you cannot embargo content already in your repository (at least using standard tools).

The other action taken at installation time is the actual imposition of the embargo. The default behavior here is simply to remove the read policies on all the bundles and bitstreams except for the "LICENSE" or "METADATA" bundles. Also note that since these policy changes occur before installation, there is no time during which embargoed content is 'exposed' (accessible by non-administrators). The terms interpretation and imposition together are called 'setting' the embargo, and the component that performs them both is called the embargo 'setter'.

Embargo period

After an embargoed item has been installed, the policy restrictions remain in effect until removed. This is not an automatic process, however: a 'lifter' must be run periodically to look for items whose 'lift date' is past. Note that this means the effective removal of an embargo is **not** the lift date, but the earliest date after the lift date that the lifter is run. Typically, a nightly cron-scheduled invocation of the lifter is more than adequate, given the granularity of embargo terms. Also note that during the embargo period, all metadata of the item remains visible.This default behavior can be changed.

One final point to note is that the 'lift date', although it was computed and assigned during the previous stage, is in the end a regular metadata field. That means, if there are extraordinary circumstances that require an administrator (or collection editor - anyone with edit permissions on metadata) to change the lift date, they can do so. Thus, they can 'revise' the lift date without reference to the original terms. This date will be checked the next time the 'lifter' is run. One could immediately lift the embargo by setting the lift date to the current day, or change it to 'forever' to indefinitely postpone lifting.

Embargo lift

When the lifter discovers an item whose lift date is in the past, it removes (lifts) the embargo. The default behavior of the lifter is to add the resource policies **that would have been added** had the embargo not been imposed. That is, it replicates the standard DSpace behavior, in which an item inherits it's policies from its owning collection. As with all other parts of the embargo system, you may replace or extend the default behavior of the lifter (see section V. below). You may wish, e.g. to send an email to an administrator or other interested parties, when an embargoed item becomes available.

Post embargo

After the embargo has been lifted, the item ceases to respond to any of the embargo life-cycle events. The values of the metadata fields reflect essentially historical or provenance values. With the exception of the additional metadata fields, they are indistinguishable from items that were never subject to embargo.

Configuration

DSpace embargoes utilize standard metadata fields to hold both the 'terms' and the 'lift date'. Which fields you use are configurable, and no specific metadata element is dedicated or pre-defined for use in embargo. Rather, you specify exactly what field you want the embargo system to examine when it needs to find the terms or assign the lift date.

The properties that specify these assignments live in dspace.cfg:

```
# DC metadata field to hold the user-supplied embargo terms
embargo.field.terms = SCHEMA.ELEMENT.QUALIFIER

# DC metadata field to hold computed "lift date" of embargo
embargo.field.lift = SCHEMA.ELEMENT.QUALIFIER
```

You replace the placeholder values with real metadata field names. If you only need the 'default' embargo behavior - which essentially accepts only absolute dates as 'terms' ,
this is the only configuration required, except as noted below.

There is also a property for the special date of 'forever':

```
# string in terms field to indicate indefinite embargo
embargo.terms.open = forever
```

which you may change to suit linguistic or other preference.

You are free to use existing metadata fields, or create new fields. If you choose the latter, you must understand that the embargo system does **not** create or configure these fields: i.e. you must follow all the standard documented procedures for actually creating them (i.e. adding them to the metadata registry, or to display templates, etc) - this does not happen automatically. Likewise, if you want the field for 'terms' to appear in submission screens and workflows, you must follow the documented procedure for configurable submission (basically, this means adding the field to input-forms.xml). The flexibility of metadata configuration makes if easy for you to restrict embargoes to specific collections, since configurable submission can be defined per collection.

Key recommendations:

1. Use a local metadata schema. Breaking compliance with the standard Dublin Core in the default metadata registry can create a problem for the portability of data to/from of your repository.
2. If using existing metadata fields, avoid any that are automatically managed by DSpace. For example, fields like 'date.issued' or 'date.accessioned' are normally automatically assigned, and thus must not be recruited for embargo use.
3. Do not place the field for 'lift date' in submission screens. This can potentially confuse submitters because they may feel that they can directly assign values to it. As noted in the life-cycle above, this is erroneous: the lift date gets assigned by the embargo system based on the terms. Any pre-existing value will be over-written. But see next recommendation for an exception.
4. As the life-cycle discussion above makes clear, after the terms are applied, that field is no longer actionable in the embargo system. Conversely, the 'lift date' field is not actionable **until** the application. Thus you may want to consider configuring both the 'terms' and 'lift date' to use the same metadata field. In this way, during workflow you would see only the terms, and after item installation, only the lift date. If you wish the metadata to retain the terms for any resaon, use 2 distinct fields instead.

Operation

After the fields defined for terms and lift date have been assigned in dspace.cfg, and created and configured wherever they will be used, you can begin to embargo items simply by entering data (dates, if using the default setter) in the terms field. They will automatically be embargoed as they exit workflow. For the embargo to be lifted on any item, however, a new administrative procedure must be added: the 'embargo lifter' must be invoked on a regular basis. This task examines all embargoed items, and if their 'lift date' has passed, it removes the access restrictions on the item. Good practice dictates automating this procedure using cron jobs or the like, rather than manually running it.

The lifter is available as a target of the DSpace Command Launcher.

Step-by-Step Setup Examples

1. **Simple Dates.** If you want to enter simple calendar dates for when an embargo will expire, follow these steps.
    a. Select a metadata field. For this example, let's use *local.description.embargo*. This field does not exist in in the default DSpace metadata directory, so login as an administrator, go the metadata registry page, select (or create) the 'local' schema, then add the metadata field.
    b. Expose the metadata field. Edit `[dspace]/config/input-forms.xml`. If you have only one form, usually 'traditional', add it there. If you have multiple forms, add it only to the forms linked to collections for which embargo applies:

```
<form name="traditional">
<page number="1">
...
<field>
<dc-schema>local</dc-schema>
<dc-element>description</dc-element>
<dc-qualifier>embargo</dc-qualifier>
<repeatable>false</repeatable>
<label>Embargo Date</label>
<input-type>onebox</input-type>
<hint>If required, enter date 'yyyy-mm-dd' when embargo expires or 'forever'.</hint>
<required></required>
</field>
```

    Note: if you want to require embargo terms for every item, put a phrase in the <required> element. Example: *<required>You must enter an embargo date</required>*
    c. Configure Embargo. Edit `[dspace]/config/dspace.cfg`. Find the Embargo properties and set these two:

```
# DC metadata field to hold the user-supplied embargo terms
embargo.field.terms = local.description.embargo
# DC metadata field to hold computed "lift date" of embargo
embargo.field.lift = local.description.embargo
```

    d. Restart DSpace application. This will pick up these changes. Now just enter future dates (if applicable) in web submission and the items will be placed under embargo. You can enter years ('2020'), years and months ('2020-12'), or also days ('2020-12-15').
    e. Periodically run the lifter. Run the task: `[dspace]/bin/dspace embargo-lifter`. You will want to run this task in a cron-scheduled or other repeating way. Item embargoes will be lifted as their dates pass.
2. **Period Sets.** If you wish to use a fixed set of time periods (e.g. 90 days, 6 months or 1 year) as embargo terms, follow these steps, which involve using a custom 'setter'.
    a. Select two metadata fields. For this example, let's use '*local.embargo.terms*' and '*local.embargo.lift*'. These fields do not exist in the default DSpace metadata registry. Login as an administrator, go the metadata registry page, select (or create) the 'local' schema, then add the metadata fields.
    b. Expose the 'term' metadata field. The lift field will be assigned by the embargo system, so it should not be exposed directly. Edit `[dspace]/config/input-forms.xml`. If you have only one form (usually 'traditional') add it there. If you have multiple forms, add it only to the form(s) linked to collection(s) for which embargo applies. First, add the new field to the 'form definition':

```
<form name="traditional">
<page number="1">
...
<field>
<dc-schema>dc</dc-schema>
<dc-element>embargo</dc-element>
<dc-qualifier>terms</dc-qualifier>
<repeatable>false</repeatable>
<label>Embargo Terms</label>
<input-type value-pairs-name="embargo_terms">dropdown</input-type>
<hint>If required, select embargo terms.</hint>
<required></required>
</field>
```

Note: If you want to require embargo terms for every item, put a phrase in the *<required>* element, e.g. *<required>You must select embargo terms</required>*
Observe that we have referenced a new value-pair list: "embargo_terms'. We must now define that as well (only once even if referenced by multiple forms):

```
<form-value-pairs>
...
<value-pairs value-pairs-name="embargo_terms" dc-term="embargo.terms">
<pair>
<displayed-value>90 days</displayed-value>
<stored-value>90 days</stored-value>
</pair>
<pair>
<displayed-value>6 months</displayed-value>
<stored-value>6 months</stored-value>
</pair>
<pair>
<displayed-value>1 year</displayed-value>
<stored-value>1 year</stored-value>
</pair>
</value-pairs>
```

Note: if desired, you could localize the language of the displayed value.

c. Configure Embargo. Edit `[dspace]/config/dspace.cfg`. Find the Embargo properties and set the following properties (replace "local" with the name of your custom schema; you may need to create one if you don't already have it):

```
# DC metadata field to hold the user-supplied embargo terms
embargo.field.terms = local.embargo.terms
# DC metadata field to hold computed "lift date" of embargo
embargo.field.lift = local.embargo.lift
# implementation of embargo setter plugin - replace with local implementation if applicable
plugin.single.org.dspace.embargo.EmbargoSetter = org.dspace.embargo.DayTableEmbargoSetter
```

d. Now add a new property called '*embargo.terms.days*' as follows:

```
# DC metadata field to hold computed "lift date" of embargo
embargo.terms.days = 90 days:90, 6 months:180, 1 year:365
```

e. Restart DSpace application. This will pick up these changes. Now the submitter can select a value form a drop-down list in web submission and the items will be placed under embargo.
f. Periodically run the lifter. Run the task:
`[dspace]/bin/dspace embargo-lifter`.
You will want to run this task in a cron-scheduled or other repeating way. Item embargoes will be lifted as their dates pass.

Extending embargo functionality

The embargo system supplies a default 'interpreter/imposition' class (the 'Setter') as well as a 'Lifter', but they are fairly rudimentary in several respects.

Setter

The default setter recognizes only two expressions of terms: either a literal, non-relative date in the fixed format 'yyyy-mm-dd' (known as ISO 8601), or a special string used for open-ended embargo (the default configured value for this is 'forever', but this can be changed in dspace.cfg to 'toujours', 'unendlich', etc). It will perform a minimal sanity check that the date is not in the past. Similarly, the default setter will only remove all read policies as noted above, rather than applying more nuanced rules (e.g allow access to certain IP groups, deny the rest).
The DayTable Setter recognizes a table of set time periods as described in the set-up examples above.

Fortunately, the setter class itself is configurable and you can 'plug in' any behavior you like, provided it is written in java and conforms to the setter interface. The dspace.cfg property:

```
# implementation of embargo setter plugin - replace with local implementation if applicable
plugin.single.org.dspace.embargo.EmbargoSetter = org.dspace.embargo.DefaultEmbargoSetter
```

controls which setter to use.

Lifter

The default lifter behavior as described above - essentially applying the collection policy rules to the item - might also not be sufficient for all purposes. It also can be replaced with another class:

```
# implementation of embargo lifter plugin - - replace with local implementation if applicable
plugin.single.org.dspace.embargo.EmbargoLifter = org.dspace.embargo.DefaultEmbargoLifter
```