Purpose

Determine the various implementation points across the RUcore suite of applications that will support the installation and use of Solr/Lucene. Outline in detail the solutions and implementation paths that will be taken to achieve the expected outcome.

dlr/EDIT (Triggs)

Maintenance of the Solr/Lucene index will be done through calls to the solrpost.cgi web service. Solrpost.cgi accepts two parameters: a Fedora PID and a subset of Solr actions, specifically "add", "update", and "delete". The add and update functions are effectively the same. They both open a socket to another CGI script, solrfilter-api.cgi, that gathers a configurable set of current datastreams from a given Fedora object and outputs a search object in valid Solr/Lucene XML with the Fedora PID used also as the Solr ID. Solrpost.cgi then posts this data to the Solr/Lucene index using the add/update method. When the "delete" parameter is passed to solrpost.cgi, it posts the delete method to the Solr/Lucene index purging the object from the index. Solrpost.cgi also posts the "commit" method upon successful completion of these actions. Solrpost.cgi returns messages indicating the success or failure of these actions. The initial indexing of the entire repository can be accomplished by looping solrpost.cgi through a list of Fedora PIDs. For a complete indexing a list of PID tokens drawn from the Fedora database could be used. After the initial complete index, solrpost.cgi only needs to be called after certain "events", specifically when an object is ingested, when an object is edited in any way, when a relationship is created between two objects (in which case both objects should be updated), and when an object is purged from Fedora. The solrpost.cgi delete action should be used when an object is purged from the repository; all other events should trigger update actions. Partner Portal IDs, if they exist, will be gathered as part of the solrfilter-api.cgi process. Most of these should be known at ingest, but if necessary solrpost.cgi could be run periodically as a cron on a set of objects assigned after ingest to a particular portal. Options for securitizing these web services are currently being explored and a solution will be documented when identified.

There will be a two-fold approach to securing the update functions of Solr. First the jetty server will have access restrictions limiting Solr update functions (add, update, and delete) to a solradmin user. The ID and password for this will be configurable in the incs files like the similar user/password combination for Fedora itself. Only privileged applications will have access to the ID and password. The solrpost.cgi program itself will be protected by being placed in a separate CGI directory with access restricted by apache based IP restriction. This way it can still be run from a command line when desired or through web sockets from certain protected local applications only, e.g., dlr/EDIT or WMS.

Currently the ability to create "dynamic collections" based on search terms exists in the dlr/EDIT management tool. This functionality will be moved to the Search Portal tool in this release. Existing "dynamic collections" will need to be migrated to the Search Portal tool and the current functionality will need to be removed from dlr/EDIT.

The Partner Portal code will provide an interface through a PHP web service to the indexing services so it can remove any object to portal associations from its cache tables. Required information will be a Fedora object ID.

WMS (Yu)

Ingesting new object

- **Assign an object to one or many 'thematic' portals**
  The Partner Portal will provide an interface through a PHP web service to the WMS so it can add an object to one or many thematic portals at the time of ingest. Required information will be the object ID and the Partner Portal keys of the portals which the object will be added. The keys will be provided by the Partner Portal code and should be configurable along with the web service information. This feature will not be part of the initial Solr/Lucene release, but will be part of a future release.

- **Trigger indexing of newly ingested object by Solr**
  After an object has been successfully ingested, the ingest script should call solrpost.cgi to add the object to the Solr/Lucene index. This can be done with a simple call such as:
  $posted = file_get_contents("http://$serverbase/cgi-bin/dlr/solrpost.pl?pid=$fedorapid");
  Note: the default action is add/update.

Editing an object

After an object has been edited the script should use the same call to solrpost.cgi as it would for ingest:

```
$posted = file_get_contents("http://$serverbase/cgi-bin/dlr/solrpost.pl?pid=$fedorapid");
```

When a relationship has been created or purged, solrpost.cgi is called for both the object and the subject of the relationship:

```
$indexed = file_get_contents("http://$serverbase/cgi-bin/dlr/solrpost.pl?pid=$object");
```

```
$indexed = file_get_contents("http://$serverbase/cgi-bin/dlr/solrpost.pl?pid=$subject");
```

Purging an object

When an object is purged, the solrpost.cgi script should be called with the delete action:

```
file_get_contents ("http://$serverbase/cgi-bin/dlr/solrpost.pl?action=delete&pid=$fedoraid");
```

Statistics (Geng)

Currently the statistics package supports the granular reporting of search hits per collection based on a single query. When searching a portal comprised of multiple collections there will be no way to discern how many search hits came from the individual collections using Solr.

The statistics database \`stats_search\` table will need to be modified to no longer support this granular logging. Along with that the statistics API will need to remove support when logging a new statistical entry. The \`stats_search\` table will be migrated to the new format of basing searches on portal's not collections. A "best guess" will be made regarding what portal the search entry was triggered from and the resulting table entries will be updated to reflect this.

The administrative interfaces that use and report the granular collection search hit information will need to be removed. The ability to view search hit information based on portal will be available and this might be included in place of the collection search hit capability.

The Search API will need to modify its calls to the statistics API so that it no longer supplies the granular collection hit data.

Partner Portal (Mills)

Managing objects in a portal for the portal owner

The Partner Portal will provide a graphical user interface (GUI) for portal owners to add or exclude objects in their portal based on various criteria. Criteria include; individual object ID, search query, or collection membership.

Managing object to portal association for application administrators

The Partner Portal will provide a GUI for application administrators to view, mange, and refresh object to portal associations.

Methods to manage object to portal associations for developers

The Partner Portal will provide a set of PHP web service methods for the management of object to portal associations. Those methods include:

- **Acquire:** Given a Fedora ID associated portal keys will be returned
- **Add**: Given a Fedora ID and portal key a object to portal association can be added
- **Delete**: Given a Fedora ID all associations will be removed
- **Update**: All or some object to portal associations can be refreshed depending on option criteria supplied
- **List**: Obtain a current list of search portal with their user friendly labels and keys.

The Partner Portal will restrict access to these methods using a key/IP address pair. All applications wishing to interact with any of these methods will be given a unique key. That key will need to be submitted when interacting with any of the methods. The key will be tied to an IP address that must match with the origin of the request. These key/IP address pairs will be stored in an access table in the Partner Portal database.

Functions to refresh the object to portal associations for system administrators

The Partner Portal will provide command line utilities for system administrators to use for refreshing object to portal associations. These command line tools will offer logging and debugging functions along with the ability to be included in a cron job.

Installation of Solr/Lucene (Nakagama)

Initially it has been decided to install Solr running under Jetty, a lightweight Java applet server from Apache. The main reasons behind this are:

- It requires minimal configuration because a working Jetty configuration comes with Solr
- We have not seen (up to this time) that using Tomcat will give us an advantage in performance or reliability
- Most Solr installation run Jetty because of the above two reasons, therefore more help is available
- The available Tomcat runs Fedora as well, and there is currently no reason to make one dependent on the other

The current task is to create a "deployable" installation, which would be a tar of relevant directories and configuration files. For the most part this would be the same on all repository related installations, while the web application side provides machine specific installation.

The PECL Solr PHP module is available for PHP 5.3.x but will not compile on the version of PHP 5.2.x we currently run, and according to documentation is not supported. While the beta version could be configured under PHP 5.2.x, it was agreed to not take this route. Currently the module is compiled and installed on the staging server. The initial release of the repository software onto new hardware will use this module.

Solr itself will be protected by an administrative password for its http access concerning any write/update/modify section.

Its listening port will not be allowed through the firewall, so access will be only from the local machine. Another option is to configure jetty to reject off machine connections, and this may be developed in the future as an extra degree of security.

CGI scripts developed for solr will br protected by an apache based IP restriction, again allowing access from the local machine. For development, an username/password fall back will be implemented for diagnostic/development purposes.