Migrate to Fedora 6

We are currently supporting migration to Fedora 6 from Fedora 3, 4.7.5 and 5.1.1. This document describes the migration process for each of these pathways.

The Fedora 3 6 Migration Path

Overview

This migration path relies on the **migration-utils** tool.

The migration-utils tool can migrate either exported FOXML (in archive or migration format), or directly migrate the Fedora 3 data stored on-disk. We recommend the direct Fedora 3 filesystem migration.

The migration-utils tool will not make any changes to your Fedora 3 repository – it only reads data from Fedora 3.

You will need to make sure that you have sufficient storage space available in the target Fedora 6 (OCFL) directory, as the migration will effectively create a copy of your Fedora 3 repository in the Fedora 6 format which conforms to the Oxford Common File Layout (OCFL) specification.

Once your Fedora 3 repository has been migrated to Fedora 6, you may start up an instance of Fedora 6 on top of the newly created OCFL directory tree. As Fedora 6 starts up it will automatically rebuild internal indices by scanning the OCFL tree. This index initialization can take a few seconds or several hours depending on the size of your repository.

Note that the Fedora 3 migration utility **will not** migrate the following Fedora 3 features:

- Disseminators (disseminator datastreams will be migrated, but they will no longer function in Fedora 6)
- Redirect (R) and external (E) datastream content (URLs will be copied)
- Solr index
- Resource index
- XACML policies (objects with POLICY datastreams attached will be migrated, but they will no longer control access in Fedora 6)

Prerequisites

- A host machine with filesystem access to both the Fedora 3 disk storage (or exported FOXML files) and the target OCFL disk storage. The target allocated disk space should be roughly the size of your Fedora 3 repository occupied disk space.
    - **Note:** OCFL repositories typically contain many small files, so the allocation block size of your filesystem can make a difference. We recommend using relatively small allocation block sizes if possible to avoid excess bloat in your migrated content.
- Java 11 installed on the host machine.
- Knowledge of the Fedora 3 filesystem layout (legacy or Akubra), if migrating directly from Fedora 3 data on disk.
- Familiarity with the command line: examining files and directories, running commands, redirecting output of commands to files, the `grep` utility, etc.

Running

1. Download the latest version of the migration-utils utility
2. Follow the instructions in the migration-utils README
3. Start up Fedora on top of your newly created Fedora 6 OCFL-compliant repository using the *-Dfcrepo.home* configuration property.

The migration-utils tool is a java command-line tool. Make sure you run the utility with Java 11.

Hints and warnings

- We recommend that you shut down Fedora 3 repository or put it into read-only mode, if you plan on migrating directly from Fedora 3 data on disk, to avoid issues of changes occurring while data is being written.
- When performing a sizable migration, run the utility in a `screen` or `tmux` session that you can detach from. Try not to shut down or reboot the host while the tool is running.
- If the --working-dir isn't specified, the utility will use the current directory as the working directory, so an index directory and pid directory would be created in the current directory.
    - An index of the datastreams will automatically be created in <working_dir>/index, and then that index will be reused for future runs of the utility. If you need to update the index, or don't want it to be used for a new run of the utility, delete <working_dir>/index and the index will be re-created.
- If the migration is interrupted, you can pick up where the migration left off by relaunching the tool with the `--resume` flag (keeping all the other parameters the same).
- Redirect the output of the utility to a log file that you can analyze at your leisure during and after the migration. Example:

  Redirect to log file

  ```
  java -jar migration-utils-6.0.0-driver.jar \
    --source-type=legacy \
    --target-dir=my-fcrepo-6-home \
    --objects-dir=my-fcrepo-3/objects \
    --datastreams-dir=my-fcrepo-3/datastreams > log.txt 2>&1
  ```

- Errors: the migration tool may encounter problems copying some objects or datastreams. The tool by default will halt at the first error; if you wish to migrate as much as possible then go back and address errors, run the tool with the `--continue-on-error` flag. Objects or datastreams with errors will not be written to the OCFL repository; they will be skipped, and the next object in the list will be processed.
  Objects that could not be migrated will provoke a stack dump in the log, marked with the string ERROR. They can be extracted from the log at a later date and fixed, then migrated individually.
  Example:

  Grep ERROR

  ```
  $ grep ERROR log.txt
  ERROR 01:09:09.801 (Migrator) MIGRATION_FAILURE: pid="test:BadPID1", message="Unable to resolve internal
  ID "test:BadPID1+MYDS+MYDS.2"!"
  ERROR 01:29:54.878 (Migrator) MIGRATION_FAILURE: pid="test:BadPID23", message="Unable to resolve
  internal ID "test:BadPID23+MYDS+MYDS.0"!"
  ERROR 02:11:50.644 (Migrator) MIGRATION_FAILURE: pid="test:BadPID617", message="Unable to resolve
  internal ID "test:BadPID617+MYDS+MYDS.1"!"
  ...
  ```

- **Warning**: the migration utility is **not** idempotent! This means that if you run the utility twice over the same content to the same target, you will see an exception when the utility tries to create an object that already exists. For the purposes of testing, delete or move out of the way previous migration attempts before running a new migration.
  However, you **can** migrate new objects to an already-existing OCFL repository, which means you can plan your migration in stages, if you desire. Note that you will need to rebuild your Fedora 6 index after the new additions.
- When adding new or fixed objects to the OCFL repository, make sure to regenerate a fresh datastream index.
- As many files are created per object and per datastream version, make sure to allocate enough inodes on your system to allow for all the files (note: this should only be necessary for
  extremely large repositories, more than 4 million objects, for example).
- See Fedora 3 to 6 Migration Community Updates for examples of medium- and large-scale migrations performed with the migrations tool, with benchmark data and detailed notes.
    - **Note:** If migration speed is an issue, you can speed up the migration by disabled checksum validation in the migration-utils using the --no-checksum-validation flag. Checksums can then be verified using the fcrepo-migration-validator as described below.
- **Warning**: the migration can be run in multiple batches, by passing in a file containing a list of pids to migrate for each run. If you do this, make sure to remove pid/resume.txt once each batch is complete. Also, note that the utility still iterates over all the foxml files in your repository for each run, so you may notice some pauses where the utility is still running, but not logging anything.

Example migration workflow

Select 1000 pids or so from your Fedora 3 repository, and migrate just those objects to an OCFL repository. Start up Fedora 6 on the sample OCFL tree, and examine. You may also look at the contents of the OCFL repository on the filesystem – the structure and contents are easily visible to the naked eye. Use the sample run to determine the disk space required, and time needed to migrate your whole repository.

Full migration:

1. Shut down your Fedora 3 repository or put it in offline mode.
2. For a partial migration, create a file with a list of PIDs to be migrated (one PID per line). You may create several PID list files, one for each batch of objects to migrate. You can migrate just those objects in the file with the tool's `--pid-file` parameter.
3. Run the migration utility, redirecting the output to a log file. Run it with the `--continue-on-error` flag.
4. After the migration has finished, look for objects with errors in the log file. Fix them, if possible, and create a new PID file with just the PIDs of the fixed objects.
5. Delete the datastream index directory, and re-run the migration utility, using the PID file with the fixed PIDs.
6. Repeat steps 4 and 5 until all objects in the PID batch are successfully migrated (or dropped from migration).
7. If running partial migrations in batches, repeat the above steps with the next batch of pids.

Once you are satisfied with the migration, start up Fedora 6 on top of your new OCFL repository.

Migration-utils command line

See the migration-utils README for the different options that can be passed in.

Validating the Results

The results of a migration can be validated against the source Fedora 3.x content using the fcrepo-migration-validator. This tool is capable of performing a number of validations and producing a report. Follow these steps to run the tool and produce a report:

1. Download the latest version of fcrepo-migration-validator
2. Follow the instructions in the README to configure and run the tool
3. Review the report (available as HTML or TSV)

A Sample Migration From 36

Step 1: Run fcrepo-migration-utils

```
java -jar migration-utils-<latest-version>-driver.jar \
  --source-type=legacy \
  --limit=100 \
  --target-dir=my-fcrepo-6-home \
  --working-dir=<tmp working dir> \
  --objects-dir=<path to objects dir> \
  --datastreams-dir=<path to datastreams dir>
```

Step 2: Validate the Results

```
java -jar fcrepo-migration-validator-<latest-version>-driver.jar \
  --source-type=legacy \
  --results-dir=results \
  --exported-dir=my-fcrepo-6-home \
  --ocfl-root-dir=my-fcrepo-6-home/data/ocfl-root \
  --objects-dir=<path to objects dir> \
  --datastreams-dir=<path to datastreams dir> \
  --checksum
```

Step 3: Start up Fedora 6

```
java -Dfcrepo.home=my-fcrepo-6-home -jar fcrepo-webapp-<latest fedora 6 version>-jetty-console.jar --headless
```

The Fedora 4   6 Migration Path

The Fedora 4 6 migration path assumes your source repository is version 4.7.5. If you are running an earlier version of Fedora 4.x please see the Upgrade Notes for the steps to upgrade to version 4.7.5.

Overview

Migrating from Fedora 4 6 is slightly more complicated than the 3 6 path. In a nutshell , you will need to do the following:

1. Export your Fedora 4.7.5 repository to disk using the 0.3.0 release of the Fedora Import ExportTool
2. Upgrade your Fedora 4.7.5 export to a Fedora 5.1.1 export using the latest release of Fedora Upgrade Utility
3. Upgrade the Fedora 5.1.1 export to Fedora 6 compliant OCFL using the same utility except this time using the output of the previous step as the input for this task and adjusting the source and target parameters accordingly.
4. Start up your Fedora 6 pointing to the newly created OCFL in the previous step.

A Sample Migration From 46

Step 1: Export from 4.7.5

Make sure that your Fedora 4.7.5 instance is running. **Also be sure that you are using the 0.3.0 version of the import export tool.** Then run the following command (swapping in appropriate local values):

```
java -jar fcrepo-import-export-0.3.0.jar -b \
  --dir my-4.7.5-export \
  --user fedoraAdmin:fedoraAdmin \
  --mode export \
  --resource http://localhost:8080/rest \
  --binaries \
  --versions
```

Step 2: Upgrade exported 4.7.5 to 5.1.1 export using latest version of fcrepo-upgrade-utils:

```
java -jar fcrepo-upgrade-utils-<latest version>.jar \
  --input-dir my-4.7.5-export \
  --output-dir my-5.1.1-export \
  --source-version 4.7.5 \
  --target-version 5+
```

Step 3: Upgrade 5.1.1 export to Fedora 6 compliant OCFL

```
# create your destination directory for the upgrade
mkdir -p  my-fcrepo-6-home

java -jar fcrepo-upgrade-utils-<latest-version>.jar \
  --input-dir my-5.1.1-export \
  --output-dir my-fcrepo-6-home \
  --source-version 5+ \
  --target-version 6+ \
  --base-uri http://localhost:8080/rest
```

Step 4: Start up Fedora 6

```
java -Dfcrepo.home=my-fcrepo-6-home -jar fcrepo-webapp--<latest fedora 6 version>-jetty-console.jar --headless
```

The Fedora 5   6 Migration Path

Overview

For migrating from Fedora 5.1.1 you will follow a similar process to the previous section, however note that **you will use a different version of the import export tool to export your F5 repository** and you will perform only one upgrade. In other words, here are the steps:

1. Export your Fedora 5.1.1 repository to disk using the latest version of the Fedora Import Export Tool.
2. Upgrade the Fedora 5.1.1 export to Fedora 6 compliant OCFL using the latest release of the Fedora Upgrade Utility
3. Start up your Fedora pointing to the newly created OCFL in the previous step.

Below you will find a sample recipes for this migration path.

A Sample Migration From 56

Step 1: Export from 5.1.1

Make sure that your Fedora 5 instance is running. **Also be sure that you are using the latest version of the import export tool.** Then run the following command:

```
java -jar fcrepo-import-export-1.0.0.jar -b \
  --dir my-5.1.1-export \
  --user fedoraAdmin:fedoraAdmin \
  --mode export \
  --repositoryRoot http://localhost:8080/rest \
  --resource http://localhost:8080/rest \
  --binaries \
  --versions
```

Step 2: Upgrade 5.1.1 export to Fedora 6 compliant OCFL using latest version of fcrepo-upgrade-utils:

```
java -jar fcrepo-upgrade-utils-<latest-version>.jar \
  --input-dir my-5.1.1-export \
  --output-dir my-fcrepo-6-home \
  --source-version 5+ \
  --target-version 6+ \
  --base-uri http://localhost:8080/rest
```

Step 3: Fire up Fedora 6

```
java -Dfcrepo.home=my-fcrepo-6-home -jar fcrepo-webapp-<latest fedora 6 version>-jetty-console.jar --headless
```

FAQs

Why isn't Fedora showing my migrated data?

The very first time Fedora starts it initializes the database and indexes any resources on disk. On subsequent starts these steps are **not** executed. If you started Fedora once and either your migrated content was not in place or there was some other configuration problem that needed addressed, then Fedora initialized to an empty state and it will not see your content until it has been instructed to reindex all of the resources in the repository.

You can force Fedora to reindex your content on startup by starting it with the following argument: `-Dfcrepo.rebuild.on.start=true`