Prephase :

Take from the dCacheToGo image.
USB stick or download from
http://www.dcache.org/downloads/virtual/dCacheToGo-V06_pnfs.vdi.gz
or
http://www.physik.rwth-aachen.de/~oleg/dCacheToGo-V06_pnfs.vdi.gz

Some useful accounts for the dCacheToGo image (user /password):

root/.school
ui_user/ .grid
dCache admin : admin/dickerelch

Installing Chimera

Start dCacheToGo image and login as root user.
Make sure that neither dCache nor PNFS running.

 # /opt/d-cache/bin/dcache stop

Stop and unmount PNFS:

 # /etc/init.d/pnfs stop

The following is the installation procedure described in the wiki page:
http://trac.dcache.org/projects/dcache/wiki/ChimeraSetup

Create Chimera database:

 # createdb chimera -U postgres

Create tables in Chimera database,

 # psql chimera < /opt/d-cache/libexec/chimera/sql/create.sql -U postgres

And the store-procedures:

 # createlang plpgsql chimera -U postgres

 # psql chimera < /opt/d-cache/libexec/chimera/sql/pgsql-procedures.sql -U postgres

Use your favorite text editor to modify the file: /opt/d-cache/config/dCacheSetup
Insert the line:

PermissionHandlerDataSource=diskCacheV111.services.PnfsManagerFileMetaDataSource

and add the line to enable the companion information into Chimera :

cacheInfo=pnfs

enable Chimera as namespace in the file: /opt/d-cache/etc/node_config

NAMESPACE=chimera

Configure NFS server:

Modfy */etc/exports* file to manage exports.

/ localhost(rw)

/pnfs

Start Chimera:

```
# /opt/d-cache/libexec/chimera/chimera-nfs-run.sh start
```

Mount admin view to configure Chimera

```
# mount localhost:/ /mnt
```

Create root Chimera namespace in the same as directorzstructure PNFS

```
# mkdir -p /mnt/pnfs/dcache.org/tape
# mkdir -p /mnt/pnfs/dcache.org/dcms
```

Add tags for disk and tape directories.

```
# echo "TAPE" > /mnt/pnfs/dcache.org/tape/'.(tag)(sGroup)'
# echo "StoreName TAPE" > /mnt/pnfs/dcache.org/tape/ '.(tag)(OSMTemplate)'
# echo "SE" > /mnt/pnfs/dcache.org/dcms/'.(tag)(sGroup)'
# echo "StoreName SE" > /mnt/pnfs/dcache.org/dcms/'.(tag)(OSMTemplate)'
```

Enable DCap for mounted system (e.g. *dccp /pnfs/dcache.org/dcms/file1 /tmp/file1*), execute:

```
# mkdir /mnt/admin/etc/config/dCache
# touch /mnt/admin/etc/config/dCache/dcache.conf
# touch /mnt/admin/etc/config/dCache/'.(fset)(dcache.conf)(io)(on)'
# echo "hal9000.dcache.org:22125" > /mnt/admin/etc/config/dCache/dcache.conf
```

unmout NFS

```
# umount /mnt
```

Now we have installed Chimera namespace an all what we need to check functionality of Chimera

Check Chimera functionality

Mount Chimera:

```
# mount localhost:/pnfs /pnfs
```

Run install script (yes recommended to start installation script after each modification of the dCache configuration files) :

```
# /opt/d-cache/install/install.sh
```

and start dCache

```
# /opt/d-cache/bin/dcache start

....
Starting chimeraDomain  Done (pid=xxxx)
....
```

Vrify that PnfsManager is working and is using Chimera:

```
# ssh -c blowfish -p 22223 -l admin localhost
admin@hal9000's password:   (dickerelch)

   dCache Admin (VII) (user=admin)

(local) admin > cd PnfsManager
(PnfsManager) admin > info
```

Leave the admin console

```
(PnfsManager) admin >..
(local) admin > logoff
```

Launch some test data transfer

```
#/opt/d-cache/dcap/bin/dccp /etc/group /pnfs/dcache.org/dcms/some.file.on.chimera
635 bytes in 0 seconds
```

And check Chimera pnfsid ID tag

```
#cat '/pnfs/dcache.org/dcms/.(id)(some.file.on.chimera)'
00009DAAAA1096FF407C9E8CB5BBF272FF40
```

Migration

Stop dCache, Chimera and PNFS services
Make sure that the dCache and PNFS not running.

```
# /opt/d-cache/bin/dcache stop
Using CATALINA_BASE:   /opt/d-cache/libexec/apache-tomcat-5.5.20
Using CATALINA_HOME:   /opt/d-cache/libexec/apache-tomcat-5.5.20
Using CATALINA_TMPDIR: /opt/d-cache/libexec/apache-tomcat-5.5.20/temp
Using JRE_HOME:       /usr/java/jdk1.6.0_12
Stopping srm-hal9000 (pid=14004) .....
.... Done
```

Shutdown Chimera-NFSv3 interface

```
# umount /pnfs
# /opt/d-cache/libexec/chimera/chimera-nfs-run.sh stop
```

Install the migration tool pnfsDump:

```
# rpm -ivh "http://www.dcache.org/downloads/pnfs/RPMs/pnfs-dump-1.0.11-1.i386.rpm"
```

http://www.dcache.org/downloads/pnfs/RPMs/pnfs-dump-1.0.11-1.i386.rpm

Prepare Chimera for migration.

```
# psql -U postgres -f /opt/pnfs/share/sql/prep-chimera-for-migration.sql chimera
```

Migrate contain of dcms:
To generate SQL migration script you need do determinate our source directory in PNFS and destination directory in Chimera with usually has a same name but different ID

Discover the source ID pnfsid (from pnfs)
Start Pnfs:

```
# /etc/init.d/pnfs start
```

Get pnfsid dcms directory.

```
# cat '/pnfs/dcache.org/.(id)(dcms)'
```

000200000000000000001060

Stop pnfs

```
# /etc/init.d/pnfs stop
```

Discover the source ID pnfsid (from Chimera)
Start Chimera

```
# /opt/d-cache/libexec/chimera/chimera-nfs-run.sh start
```

Mount Chimera

```
# mount localhost:/pnfs /pnfs
```

Determine destination Chimera pnfsid for dcms .

```
# cat '/pnfs/dcache.org/.(id)(dcms)'
```

0000EF3B9272D7A6465784C843402981710B (looks a similar but not the same)

Unmount and stop Chimera

```
# umount /pnfs
# /opt/d-cache/libexec/chimera/chimera-nfs-run.sh stop
```

Start PNFS (now we need only PNFS shared memory)

```
#/etc/init.d/pnfs start
```

And build the SQL migration script

```
# /opt/pnfs/tools/pnfsDump -r 000200000000000000001060 -o /tmp/dcms_pnfs-2-chimera.sql -vv -d0 chimera -2  -p 00000EF3B9272D7A6465784C843402981710B
```

```
Final stats:

inodes:
  nDir:          354
  nFile:       64225

5
```

```
  nUnknown:       0
  nSkipped:       0
  ----------------
  Total:      64579

Major DB errors: 0

Cache queries: 2124 (hits: 1062, misses: 0)

Time elapsed:  205.5s (00:03:25.5)
    dbserver:  200.5s (00:03:20.5)
    overhead:    2.0s (00:00:02.0)
    pnfsDump:    3.0s (00:00:03.0)
Average inode processing rate: 315 Hz
```

Generate md5sum verification file

```
# /opt/pnfs/tools/pnfsDump -r 00020000000000000001060 -o /tmp/dcms_pnfs-verify-md5sum -vv -d0 verify -r
```

And storage verification files

```
# /opt/pnfs/tools/pnfsDump -r 00020000000000000001060 -o /tmp/dcms_pnfs-verify-storageinfo -vv -d0 files -f
```

With the same method generate SQL script, and the verification files for the /pnfs/dcache.org/tape directory

Stop PNFS

```
# /etc/init.d/pnfs stop
```

Inject data to Chimera.

```
# psql -U postgres -f /tmp/dcms_pnfs-2-chimera.sql chimera  | grep error
```

This make take approximately 6 minutes to complete.

```
# psql -U postgres -f /tmp/tape_pnfs-2-chimera.sql chimera| grep error
```

Convert StorageInfo locations to URIs
Download script:

```
# wget “http://www.dcache.org/downloads/osm2chimera.sql”
```

Convert how StorageInfo locations are stored.

```
# psql -U postgres -f osm2chimera.sql chimera

# echo "select osm2chimera();" | psql -U postgres chimera
```

Verifying the data

Download the storage info check script

```
# wget “http://www.dcache.org/downloads/pnfs/migration-check-0.0.3.tar.gz”
# tar xzvf migration-check-0.0.3.tar.gz
#./migration-check.sh /tmp/dcms_pnfs-verify-storageinfo
```

Stop PNFS

```
# /etc/init.d/pnfs stop
```

Start and mount Chimera

```
# /opt/d-cache/libexec/chimera/chimera-nfs-run.sh start
# mount localhost:/pnfs /pnfs
```

Verify md5sum info

```
# cd /pnfs/dcache.org/dcms
# md5sum -c /tmp/dcms_pnfs-verify-md5sum | grep -v :\ OK$
```

Register the pools

Make a simply copy of cache-location data
Copy the companion data into the Chimera database.

```
# pg_dump -U  postgres -t cacheinfo companion | psql -U postgres chimera.
```

With help of conversions script companion2chimera.sql populate t_locationinfo table with imported data:

```
# wget “http://www.dcache.org/downloads/osm2chimera.sql”
# psql -U postgres -f companion2chimera.sql chimera
# echo "select companion2chimera();" | psql -U postgres chimera
```

List all databases

```
psql -l -U postgres
      List of databases
```

```
   Name    |   Owner   | Encoding
-----------+-----------+----------
 admin     | postgres  | UTF8
 billing   | srmdcache | UTF8
 chimera   | postgres  | UTF8
 companion | srmdcache | UTF8
 dcache    | srmdcache | UTF8
 dcmsdb    | postgres  | UTF8
 postgres  | postgres  | UTF8
 replicas  | srmdcache | UTF8
 tape      | postgres  | UTF8
 template0 | postgres  | UTF8
 template1 | postgres  | UTF8
(11 rows)
```

On the final step drop unused databases, admin, companion, dcmsdb, tape databases with command dropdb

```
# dropdb –U postgres admin
```

End---