

**PIC**  
port d'informació  
científica

## *Small Files at PIC*

*Emilio Hernández*

**Castor Meeting  
Barcelona, Nov 17th 2004**



**Ciemot**



Generalitat de Catalunya  
Departament d'Universitats, Recerca  
i Societat de la Informació



Universitat Autònoma de Barcelona



**PIC**  
port d'informació  
científica

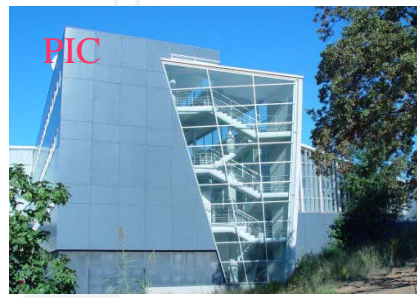
## Agenda

- Definition of requirements
- First experiences storing medical images (medical image=small file) using Castor
- Solutions based on automounting
- We would like...

## UDIAT-PIC collaboration



5 TB of images per year  
Local HD cache for 3 years



Petabyte tape library  
Processing PC farm

Anella científica

## Collaboration goals

- Backup of medical images
  - Format: DICOM
  - File size range from 30K ~ 13MB
  - Daily backup size: 5GB ~ 20GB
  - They upload the images produced the day before
- Retrieval of images
  - Clinical purposes: query for individual images, probably from the same patient (near-line access)
  - Research purposes: creation of datasets by sweeping the whole image database (batch processing)

## Initial solution

- First solution:
  - Images stored as independent Castor files
  - Upload operations via ftp from UDIAT
  - A Castor-ready ftp server used
  - UDIAT defined the directory hierarchy (they chose to have a subdirectory per day)
- Problems:
  - The tape backup system (tape library+Castor) collapsed
  - ~200.000 files waiting for migration
  - Clinical query requirements can be served
  - Upload and research requirements will be difficult to serve

## Automounter solution

- Second solution:
  - First-level directories stored as single archive files (e.g. ISO or tar files), we call these files “volumes”
  - An automounter handles image retrieval from volumes
  - UDIAT is semi-aware of the existence of volumes
    - They upload their images as before
    - They should try to achieve data balance among first-level directories; one directory per day is fine in this case
    - There may be different connection instructions for image upload and image retrieval
  - The amount of files that Castor handles is reduced from hundreds of thousands to a max of 366 per year

# Upload/Retireve Operations

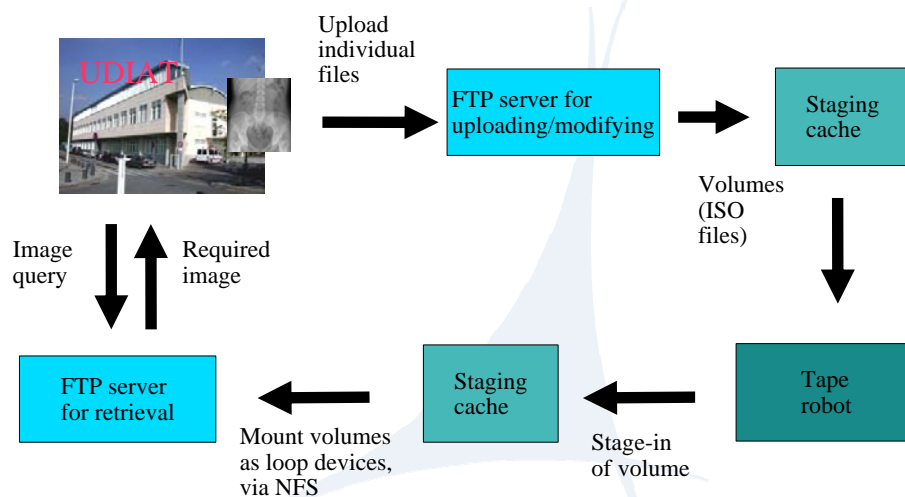
- Uploading/Modifying:

- We expect “modifying” operations to be infrequent
- A “normal” ftp server is used to upload the images into a working directory
- First-level directory upload is followed by a “volume migration” operation; after that, volume information is ready to be retrieved
- Mount/umount scripts are triggered by the automounter
  - mount: “rfcp /castor/.../volume ./volume” + “unpack volume ./working\_dir”  
(an empty working\_dir is created if the volume does not exist in Castor)
  - umount: “pack ./working\_dir > volume” + “rfcp volume /castor/.../volume”

- Image Retrieval:

- Mount/umount scripts are triggered by the automounter
  - mount: “stagein volume” + “protect\_from\_GC volume” + “mount -o loop stage\_server:volume ./working\_dir”
  - umount: “umount ./working\_dir”, “release\_for\_GC volume”

# Operation



## Retrieve/Modify asymmetry

- With this solution
  - Retrieve operations require a single volume copy on open (tape -> staging area) and no copy on close
  - Modify operations require three volume copies on open (tape -> staging area + staging area -> working directory + unpack to a working directory) and three volume copies on close (pack to an archive file, copy file into the staging area + migration)
- Potential problems
  - Modify operation too heavy (a writeable file system can be used to avoid the pack/unpack data copy)
  - Data is not uniformly distributed among volumes

## Achieving data balance with transparency

- Third solution (more flexible for users):
  - Volumes do not correspond to directories (no restrictions on the directory hierarchy)
  - “Volume migration” includes an algorithm for reading the user directory structure and mapping files to volumes. This algorithm would try to balance the amount of data among the volumes
  - Archive files are generated for each volume and written in Castor
  - A persistent hash table could be used to keep the matches between names in the directory structure defined by the user and names in the volume structure
  - The automounter would have to be modified in order to implement the name association, based on the hash table

# Letter to Santa

- Ideal solution (for us):
  - **Third solution implemented within Castor**
  - The name server would have an additional indirection level (corresponding to our hash table)
  - No need of the automounter because the image is retrieved directly from Castor
- Note:
  - A retrieve operation would produce a transfer of the whole volume into the staging area
  - A volume may correspond to a Virtual Tape within Storagetek VSM
  - Tricky but not that much: the data balance algorithm

# Thanks!