SANSA Catalogue Documentation

Tim Sutton, 2008

Contents

1 Introduction **3**
- 1.1 The CSIR . . . 3
- 1.2 SAC . . . 3
- 1.3 SANSA Takover . . . 3
- 1.4 The project . . . 3
- 1.5 The Online Catalogue . . . 6

2 Informix SPOT Catalogue Notes **7**
- 2.1 Overview of the data migration process . . . 7
- 2.2 Technical notes for informix access via python . . . 8
  - 2.2.1 Making a simple python test . . . 14
- 2.3 Trouble shooting and general tips . . . 15
  - 2.3.1 WKT representation of GeoObjects . . . 15
  - 2.3.2 When things go wrong on the informix server . . . 17
  - 2.3.3 File System . . . 19
  - 2.3.4 Schema dump of informix databases . . . 19
  - 2.3.5 Listing system and user functions . . . 20
  - 2.3.6 Problems running functions . . . 20
  - 2.3.7 Accessing the server Interactively . . . 20
- 2.4 Command line batch processing . . . 20
  - 2.4.1 Command line processing using echo . . . 21
  - 2.4.2 Changing geotype to wkt . . . 21
  - 2.4.3 Reverting geotype to informix format . . . 21
  - 2.4.4 Informix environment preparation . . . 21
- 2.5 Backup and Restore of the postgres ACS clone . . . 22

3 Procedures for importing data from various sources into the catalogue **22**
- 3.1 Legacy ACS System . . . 22
- 3.2 SPOT Image Data . . . 22
- 3.3 Sumbandilasat . . . 22
  - 3.3.1 Copying the product folder over to LION . . . 23
  - 3.3.2 Importing the report file . . . 23
  - 3.3.3 Unified product migration . . . 25
  - 3.3.4 Downloadable Products . . . 25
- 3.4 CBERS . . . 25
- 3.5 SACC . . . 25

4 Catalogue Reporting tools **25**

1 Introduction

1.1 The CSIR

This project has been carried out for the CSIR Satellite Applications Center, near Johannesburg, South Africa. The CSIR is the 'Council for Science and Industrial Research' - it is the main national science foundation of the country. They are a big organisation with many divisions of which SAC (Satellite Applications Center) is one.

1.2 SAC

SAC is a satellite ground station. This means they have a big campus with many antennas and collect information from satellites as they pass over our sky window they also do satellite tasking (telling satellites where to go and what to do) and satellite / space craft telemetry (tracking space vehicle orbit information etc).

SAC has two divisions:

1. Telemetry command and control where they do tracking, tasking etc.
2. EO (Earth Observation) where the focus is more software based to do remote sensing and generate products from imagery downloaded from satellites

SAC-EO is the client for this project.

1.3 SANSA Takover

South Africa is busy creating its own space agency - SANSA (South AfricAN National Space Agency). SANSA will aggregate space technology from various gov, parastatal, non-gov organisations to form a new organisation funded by the state. SAC-EO is scheduled to become part of SANSA as of 1 April 2011 and wil become SANSA-EO.

1.4 The project

SAC-EO has been building for the last 3 or 4 years an integrated system before this project (of which we form a small part), the processing of imagery was done manually and ad-hoc which is not very efficient and prone to difficulty if an expert leaves.

Thus they have started to build an integrated system called SAEOS (pronounced 'sigh-os'). The purpose of SAEOS is to create an automated processing environment through all the steps of the EO product workflow i.e.:

- satellite tasking ('please programme spot5 to take an image at footprint foo on date X')
- image processing (level 1a through 3a/b)
- image analysis (level 4)

- image ordering ('can I please get a copy of that SPOT image you took on dec 4 2008 of this area')
- product packaging ('bundle up the stuff that was ordered using a DVD robot, placing on an ftp site, writing to an external HD etc')

To achieve this goal they have a number of software components.

The first components are the 'terminal software'. Terminal software are provided by satellite operators such as SPOT5 (I will use SPOT as an example a lot as its the pilot sensor for their project, eventually to incorporate many more sensors) The terminal software is typically a linux box with the operators own proprietary software on top that lets the operators do the tasking of satellites (to collect an image at a given place and time) and also to extract archived images from their tape library

The second component is 'SARMES'. SARMES is a collection of EASI scripts / routines. EASI is a programming language that runs on top of PCI / Geomatica a proprietary GIS tool that runs on windows and linux. SAC are busy porting SARMES to SARMES II which has the same functionality but uses python language bindings of PCI/Geomatica instead of EASI script. SARMES has all the logic to do things like:

- take a raw image and convert it to a common GIS format e.g. pix, gtiff etc.
- collect GCP's automatically using a reference image
- orthorectify an image using a dem, gcps and other reference data
- reproject the image into different coord systems (typically UTM 33S - UTM 36S in our area but others may apply too)
- perform atmospheric correction to remove effects of the stratosphere interference between lens and ground target
- perform sensor specific correction to e.g. remove effects of lens distortion on a specific camera (using published sensor models)
- perform mosaicking of images to create one big seamless colour corrected dataset
- perform pan sharpening (make a colour image higher resolution by merging it with a pan-chromatic / grey scale band)
- chop up images in various tile schemes (e.g. degree squares, quarter degree sqauares etc)

These jobs are run by manual process - creating config files, placing input files in a specific dir heirachy etc.

The third component is DIMS. DIMS is a software system running on top of linux written in java, corba, and using oracle or postgresql as a backend (at SAC they are using PostgreSQL). DIMS is proprietary software written by a german company called

WERUM. The same software is used by the German Space Agency and others. DIMS provides automated tool chain processing. Basically you set up work flows and run them using an 'operating tool'. Although DIMS uses postgresql, there is no third party access to that db and the whole system should be considered a black box except for a few specific entry and exit points.

DIMS is being extended and customised for SAC-EO including modifications so it wll provide ogc interfaces. Before this had their own catalogue implementation and ordering system using very old standards or proprietary interfaces. So DIMS can process EO data and it builds up a catalogue of products that it has processed or 'knows about' - in its own silo. This catalogue is / will be accessible via CSW and for processing of ordering they are implementing the OGC

The OS4EO (ordering service for earth observation) is an ogc standard. The OS4EO standard is pretty simple and familiar. In essence it allows you to:

- get capabilities
- get quote
- place order

In DIMS it is implemented using SOAP rather than a RESTful service.

Along the process of creating the SAEOS project, SAC-EO have also been investing in high end hardware - particularly storage. They have a petabyte capable heirachical storage system that in short works as follows:

- data is written to local hard drives
- after a certain period of inactivity moved down to slower sata drives (nearline storage)
- and after that its migrated down onto a tape library

The tape library (offline storage) is treated as part of the file system. It has a robot arm that loads tapes automatically. When you browse the file system, it appears that all data is local since all inodes are present in online storage. When you try to read a file that is offline, the robot fetches if from tape and puts it online - typically in under a minute, though that depends on system load.

DIMS is integrated with this file system (this file system is SGI's HFS - Heirachical File System). HFS is also proprietary software running on top of Linux. One of the things DIMS will be doing is de-archiving from old manually loaded tapes and moving them into HFS. De-archiving historically collected raw satellite imagery that is. When DIMS is finished going through that there will be hundreds of thousands (probably millions) of raw images stored in HFS and accessible via DIMS.

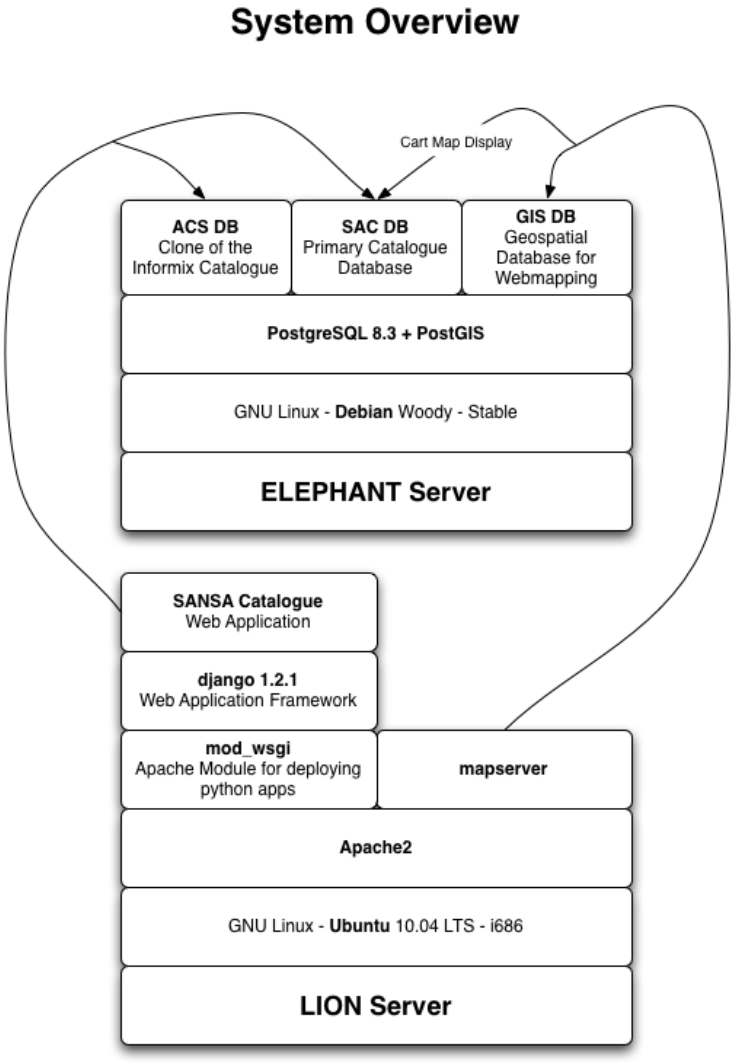
Since DIMS integrates with SARMES so you can do things like:

"Pull out that landsat 5 image from 2002, orthorectify it, correcto for atmospheric interference and lens distortions, reproject it to UTM 35S and clip it to this bounding box, then place the product on a dvd and write this label on it"

Thats the goal of the system - end to end automation with minimal operator intervention.

1.5 The Online Catalogue

Along side of these other packages, Linfiniti has been building a new web catalogue for SAC-EO. The catalogue is django + postgresql + all the other great FOSS tools we can use together to make a rich, interactive site.



Work on this started before any dims software was installed on site and the first major task was to migrate data and thumbnailss from a legacy, proprietary informix catalogue and get it into postgresql.

We also went through a few refactorings since the first version was almost a direct clone of the data model from the legacy informix system. Our current version uses the

concept of a 'generic product' for the data model.

XXXXXXXXX Insert image here XXXXXXXXXXXXXXX

On the left / center part of the diagram you will see an entity name 'Generic product'. This is a generic representation of any product (e.g. optical, radar, atmospheric, derived (like landcover map) and in the future we want to tune this to cater for vector data too.

There are five inherited models:

XXXXXXXXXXX Todo explain inherited models XXXXXXXXXXXXXXXXx

There are a bunch of small tables that provide foreign key dictionaries for the terms described in the models including:

- tables relating to ordering and tasking
- search and search record models are used to store user searches
- temporary tables used during product imports

The Online catalogue has the capability to deliver some products directly if they are held on local storage and also some basic capabilities for visitors to submit tasking requests.

Every time a search is made, its remembered, and the records the user is shown may be added to a cart and then assigned to an order

2 Informix SPOT Catalogue Notes

This section is broken up into the following parts:

1. a description of the ACS port and data migration process
2. technical information on how to connect to informix from python
3. miscellaneous tips and tips relating to using the informix database

2.1 Overview of the data migration process

The process of data migration seeks to largely clone the informix ACS catalogue into a postgresql database, and then use that as the basis of running various import steps in order to migrate key parts of the ACS database into the SAC Catalogue.

There are two things to consider here:

1. migration of metadata
2. migration of product thumbnails

Metadata records are defined in a complex of different tables which need to be queried in order to be able to obtain all the data related to a specific product. The informix acs catalogue stores all thumbnails as whole segments within blobs inside the database and these need to be extracted, georeferenced and clipped into individual scenes.

2.2 Technical notes for informix access via python

This section covers the installation and setup process for the informix python client so that you can connect to the server from a separate linux box using python.

Download the InformixDB driver for python from:

```
http://sourceforge.net/project/showfiles.php?group_id=136134
```

And the Informix client sdk from:

```
http://www14.software.ibm.com/webapp/download/preconfig.jsp?
    id=2007-04-19+14%3A08%3A41.173257R&S_TACT=104CBW71&S_CMP=
```

(write the above url on a single line)

If the above link doesnt work for you (it seems to contain a session id), go to the

```
http://www14.software.ibm.com
```

website and search for

```
3.50.UC4
```

using the search box near the top right of the page. Downloading requires an IBM id etc. which you can sign up for if you dont have one.

Note: You will need to get the appropriate download for your processor type. For Lion, which is running ubuntu server x86_64, I downloaded the sdb bundle called:

```
IBM Informix Client SDK V3.50.FC4 for Linux (x86) RHEL 4, 64bit
clientsdk.3.50.FC4DE.LINUX.tar  (72MB)
```

Note 2: Even though it says Red Hat Enterprise Editition (RHEL) you can use it on ubuntu servers too.

After you have downloaded the client sdk do the following to install (below is a log of my install process).

```
sudo adduser informix
Adding user ‘informix’ ...
Adding new group ‘informix’ (1003) ...
Adding new user ‘informix’ (1003) with group ‘informix’ ...
Creating home directory ‘/home/informix’ ...
Copying files from ‘/etc/skel’ ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for informix
Enter the new value, or press ENTER for the default
```

```
Full Name []: Informix
Room Number []:
Work Phone []:
Home Phone []:
Other []:
Is the information correct? [Y/n] Y
[linfiniti:timlinux:DownloadDirector] sudo ./installclientsdk




         Initializing InstallShield Wizard........
          Launching InstallShield Wizard........


-------------------------------------------------------------------------------
Welcome to the InstallShield Wizard for IBM Informix Client-SDK Version 3.50

The InstallShield Wizard will install IBM Informix Client-SDK Version 3.50 on
your computer.
To continue, choose Next.

IBM Informix Client-SDK Version 3.50
IBM Corporation
http://www.ibm.com


Press 1 for Next, 3 to Cancel or 4 to Redisplay [1] 1

-------------------------------------------------------------------------------
     International License Agreement for Non-Warranted Programs

     Part 1 - General Terms

     BY DOWNLOADING, INSTALLING, COPYING, ACCESSING, OR USING THE PROGRAM
     YOU AGREE TO THE TERMS OF THIS AGREEMENT. IF YOU ARE ACCEPTING THESE
      TERMS ON BEHALF OF ANOTHER PERSON OR A COMPANY OR OTHER LEGAL
      ENTITY, YOU REPRESENT AND WARRANT THAT YOU HAVE FULL AUTHORITY TO
      BIND THAT PERSON, COMPANY, OR LEGAL ENTITY TO THESE TERMS. IF YOU DO
      NOT AGREE TO THESE TERMS,
```

```
- DO NOT DOWNLOAD, INSTALL, COPY, ACCESS, OR USE THE PROGRAM; AND


- PROMPTLY RETURN THE PROGRAM AND PROOF OF ENTITLEMENT TO THE PARTY

Press Enter to continue viewing the license agreement, or, Enter "1" to accept
the agreement, "2" to decline it or "99" to go back to the previous screen, "3"
 Print.

1

Press 1 for Next, 2 for Previous, 3 to Cancel or 4 to Redisplay [1] 1

-------------------------------------------------------------------------------
IBM Informix Client-SDK Version 3.50 Install Location

Please specify a directory or press Enter to accept the default directory.

Directory Name: [/opt/IBM/informix] /usr/informix

Press 1 for Next, 2 for Previous, 3 to Cancel or 4 to Redisplay [1] 1

-------------------------------------------------------------------------------
Choose the setup type that best suits your needs.

[X] 1 - Typical
        The program will be installed with the suggested configuration.
        Recommended for most users.

[ ] 2 - Custom
        The program will be installed with the features you choose.
        Recommended for advanced users.

To select an item enter its number, or 0 when you are finished: [0]


Press 1 for Next, 2 for Previous, 3 to Cancel or 4 to Redisplay [1] 1

-------------------------------------------------------------------------------
IBM Informix Client-SDK Version 3.50 will be installed in the following
```

```
location:

/usr/informix

with the following features:

Client
Messages
Global Language Support (GLS)

for a total size:

 91.8 MB

Press 1 for Next, 2 for Previous, 3 to Cancel or 4 to Redisplay [1] 1

Installing IBM Informix Client-SDK Version 3.50. Please wait...


|-----------|-----------|-----------|------------|
0%         25%         50%         75%        100%
||||||||||||||||||||||||||||||||||||||||||||||||||

Creating uninstaller...
Performing GSKit installation for Linux ...


Branding Files ...
Installing directory .
Installing directory etc
Installing directory bin
Installing directory lib
Installing directory lib/client
Installing directory lib/client/csm
Installing directory lib/esql
Installing directory lib/dmi
Installing directory lib/c++
Installing directory lib/cli
Installing directory release
Installing directory release/en_us
Installing directory release/en_us/0333
Installing directory incl
Installing directory incl/esql
Installing directory incl/dmi
```

```
Installing directory incl/c++
Installing directory incl/cli
Installing directory demo
Installing directory demo/esqlc
Installing directory demo/c++
Installing directory demo/cli
Installing directory doc
Installing directory doc/gls_api
Installing directory doc/gls_api/en_us
Installing directory doc/gls_api/en_us/0333
Installing directory tmp
Installing directory gsk
Installing directory gsk/client
Installing directory gskit
Installing directory gsk
Installing directory gsk/client

IBM Informix Product:       IBM INFORMIX-Client SDK
Installation Directory: /usr/informix

Performing root portion of installation of IBM INFORMIX-Client SDK...


Installation of IBM INFORMIX-Client SDK complete.

Installing directory etc
Installing directory gls
Installing directory gls/cm3
Installing directory gls/cv9
Installing directory gls/dll
Installing directory gls/etc
Installing directory gls/lc11
Installing directory gls/lc11/cs_cz
Installing directory gls/lc11/da_dk
Installing directory gls/lc11/de_at
Installing directory gls/lc11/de_ch
Installing directory gls/lc11/de_de
Installing directory gls/lc11/en_au
Installing directory gls/lc11/en_gb
Installing directory gls/lc11/en_us
Installing directory gls/lc11/es_es
Installing directory gls/lc11/fi_fi
Installing directory gls/lc11/fr_be
Installing directory gls/lc11/fr_ca
```

```
Installing directory gls/lc11/fr_ch
Installing directory gls/lc11/fr_fr
Installing directory gls/lc11/is_is
Installing directory gls/lc11/it_it
Installing directory gls/lc11/ja_jp
Installing directory gls/lc11/ko_kr
Installing directory gls/lc11/nl_be
Installing directory gls/lc11/nl_nl
Installing directory gls/lc11/no_no
Installing directory gls/lc11/os
Installing directory gls/lc11/pl_pl
Installing directory gls/lc11/pt_br
Installing directory gls/lc11/pt_pt
Installing directory gls/lc11/ru_ru
Installing directory gls/lc11/sk_sk
Installing directory gls/lc11/sv_se
Installing directory gls/lc11/th_th
Installing directory gls/lc11/zh_cn
Installing directory gls/lc11/zh_tw

IBM Informix Product:       Gls
Installation Directory: /usr/informix

Performing root portion of installation of Gls...


Installation of Gls complete.

Installing directory etc
Installing directory msg
Installing directory msg/en_us
Installing directory msg/en_us/0333

IBM Informix Product:       messages
Installation Directory: /usr/informix

Performing root portion of installation of messages...


Installation of messages complete.


-------------------------------------------------------------------------------
The InstallShield Wizard has successfully installed IBM Informix Client-SDK
```

```
Version 3.50. Choose Finish to exit the wizard.

Press 3 to Finish or 4 to Redisplay [3]
```

Note that trying to install it to another directory other than /usr/informix will cause the db adapter build to fail (and various other issues). So dont accept the default of /opt/IBM/informix and rather use /usr/informix

Now build the python informix db adapter:

```
cd /tmp/InformixDB-2.5
python setup.py build_ext
sudo python setup.py install
```

Now ensure the informix libs are in your lib search path:

```
sudo vim /etc/ld.so.conf
```

And add the following line:

```
/usr/informix/lib/
/usr/informix/lib/esql
```

Then do

```
sudo ldconfig
```

2.2.1 Making a simple python test

First you need to add a line to informix's sqlhosts file:

```
sudo vim /usr/informix/etc/sqlhosts
```

And add a line that looks like this:

```
#catalog2 added by Tim
#name, protocol, ip, port
catalog2        onsoctcp        196.35.94.210  1537
```

Next you need to export the INFORMIXSERVER environment var:

```
export INFORMIXSERVER=catalog2
```

I found out that it is running on port 1537 by consulting the /etc/services file on the informix server. Now lets try our test connection. This little script will make a quick test connection so you can see if its working:

```
#!/usr/bin/python

import sys
import informixdb  # import the InformixDB module

# -------------------------------------
# open connection to database 'stores'
# -------------------------------------
conn = informixdb.connect('catalogue@catalog2', user='informix', password='')

# ---------------------------------
# allocate cursor and execute select
# ---------------------------------
cursor1 = conn.cursor(rowformat = informixdb.ROW_AS_DICT)
cursor1.execute('select * from t_file_types')

for row in cursor1:

  # ----------------------------------------
  # delete row if column 'code' begins with 'C'
  # ----------------------------------------
  print "%s %s" % (row['id'], row['file_type_name'])
# ---------------------------------------
# commit transaction and close connection
# ---------------------------------------
conn.close()

sys.exit(0);
```

Note that the documentation for the python InformixDB module is available here:

`http://informixdb.sourceforge.net/manual.html`

And the documentation for the Informix SQL implementation is here:

`http://publib.boulder.ibm.com/infocenter/idshelp/v10`

2.3 Trouble shooting and general tips

2.3.1 WKT representation of GeoObjects

Informix uses its own representation of geometry objects. There are two extensions for informix that deal with spatial data : Geodetic and Spatial. It seems we have only geodetic extension at SAC and thus can't use ST_foo functions to work with geometry

fields. For Geodetic we need to alter a value in the GeoParam table in order to change what formats are output / input. From the manual:

```
Converting Geodetic to/from OpenGIS Formats

Geodetic does not use functions to convert data to a specific format.

Instead, the GeoParam metadata table manages the data format for transmitting
data between client and server. If the "data format" parameter is set to "OGC",
then binary i/o is in WKB format and text i/o is in WKT format. (For specific
details, see Chapter 7 in the Informix Geodetic DataBlade Module User's Guide).
```

You can override the representation type that should be returned so that you get e.g. WKT back instead. Consider this example:

```
-- set output format to 3
update GeoParam set value = 4 where id =3;
-- show what the format is set to now
select * from GeoParam where id = 3;
-- display a simple polygon
select first 1 geo_time_info from t_localization;
-- revert it to informix representation
update GeoParam set value = 0 where id =3;
-- display the polygon back in native informix representation
select first 1 geo_time_info from t_localization;
--verify that the format is reverted correctly
select * from GeoParam where id = 3;
```

Which produces output like this:

```
id       3
name     data format
value    4
remarks  This parameter controls the external text & binary format of GeoObjects.
         It is not documented in the 3.0 version of the user's guide.  See release
         notes for more info.



geo_time_info  POLYGON((28.73 -15.35, 28.969999 -13.79, 27.34 -13.55, 27.1 -15.
                       11, 28.73 -15.35))



geo_time_info  GeoPolygon((((-15.35,28.73),(-13.79,28.969999),(-13.55,27.34),
```

```
                (-15.11,27.1))),ANY,(1987-04-26 07:34:45.639,1987-04-26
                07:34:45.639))


id        3
name      data format
value     0
remarks   This parameter controls the external text & binary format of GeoObject
          s.  It is not documented in the 3.0 version of the user's guide.  See
          release notes for more info.
```

2.3.2 When things go wrong on the informix server

Record Lock Issues

If the client does not cleanly disconnect it can leave records locked. You may see a message like this from dbaccess when trying to do an interactive query:

```
244: Could not do a physical-order read to fetch next row.
107: ISAM error:  record is locked.
```

There are probably solutions that are better than this, but the most robust way of dealing with the issue is to restart the informix database:

```
ssh informix@informix
cd /home/informix/bin
onmode -k
```

You will then get prompted like this:

```
This will take Informix Dynamic Server 2000 OFF-LINE -
Do you wish to continue (y/n)? y

There are 1 user threads that will be killed.
Do you wish to continue (y/n)? y
```

Afterwards, you can bring up the database like this:

```
oninit
```

The record locks should have been cleared at this point.

DBAccess Unresponsive

Collect diagnostics:
[101] catalog2:/home/informix> onstat -V Informix Dynamic Server 2000 Version 9.21.UC4 Software Serial Number AAD#J130440
[101] catalog2:/home/informix> onstat -a

Nightly Informix Compact Job

```
ssh informix@informix
crontab -l
[101] catalog2:/home/informix> crontab -l
no crontab for informix
```

So now we make a little bash script:

```
#!/bin/bash

# A simple bash script to  be invoked by CRON on a nightly basis
# To enable add to your crontab like so:
#
# ------------------------------------------------------------
#
# Run informix stats update nightly to keep responsiveness good
# Job will run 5 min after midnight
# 5 0 * * * /home/informix/nightly_cron.sh
#
# ------------------------------------------------------------
#
# Actual script follows:
#
# Tim Sutton, May 2009
#

# Update the db stats on a nightly basis:

date >> /tmp/informix_stats_update_cron_log.txt
echo "Nightly stats update running" >> \
       /tmp/informix_stats_update_cron_log.txt
echo "update statistics high;" | \
       dbaccess catalogue >> /tmp/informix_stats_update_cron_log.txt
```

And then set up a nightly cronjob to run it:

```
crontab -e
```

Now add this:

```
# Added by Tim for others to see how crontab works
#*     *     *     *     *  command to be executed
#-     -     -     -     -
#|     |     |     |     |
#|     |     |     |     +----- day of week (0 - 6) (Sunday=0)
#|     |     |     +------- month (1 - 12)
#|     |     +--------- day of month (1 - 31)
#|     +----------- hour (0 - 23)
#+-------------- min (0 - 59)


# Run a test command every minute to see if crontab is working nicely
# comment out when done testing
#*/1 * * * * date >> /tmp/date.txt

# Run informix stats update nightly to keep responsiveness good
# Job will run 5 min after midnight
5 0 * * * /home/informix/nightly_cron.sh
```

2.3.3 File System

```
root@informix's password:
Last login: Tue Sep  9 12:57:53 2008 from :0
[root@catalog2 /root]# mount
/dev/sda6 on / type ext2 (rw)
none on /proc type proc (rw)
usbdevfs on /proc/bus/usb type usbdevfs (rw)
/dev/sda2 on /boot type ext2 (rw)
/dev/sda10 on /home type ext2 (rw)
/dev/sda8 on /tmp type ext2 (rw)
/dev/sda5 on /usr type ext2 (rw)
/dev/sda9 on /var type ext2 (rw)
none on /dev/pts type devpts (rw,gid=5,mode=620)
/dev/sdb1 on /mnt/disk1 type ext2 (rw)
/dev/sdc1 on /mnt/disk2 type ext2 (rw)
automount(pid458) on /misc type autofs (rw,fd=5,pgrp=458,minproto=2,maxproto=3)
```

2.3.4 Schema dump of informix databases

Its useful to be able to see the schema of databases so you can understand how it was put together. The following command will dump the catalogue2 (SAC Production database)

schema to a text file. **Note:** No data is dumped in this process.

```
dbschema -t all -d catalogue catalogue_schema.sql
```

2.3.5 Listing system and user functions

To see what functions are installed in the database do:

```
select procname from sysprocedures;
```

To see full details of a function:

```
select * from sysprocedures where procname="lotofile";
```

2.3.6 Problems running functions

If you try to run a function that you know exists, but you get an error message like this:

```
_informixdb.DatabaseError: SQLCODE -674 in PREPARE:
IX000: Routine (lotofile) can not be resolved.
```

It probably means you passed the incorrect number or type of parameters to the function.

2.3.7 Accessing the server Interactively

```
ssh 196.35.94.210 -l informix
```

Interactive database access:

```
dbaccess
```

2.4 Command line batch processing

Add some sql commands to a text file:

```
vim /tmp/tim.sql
```

Some commands:

```
select geo_time_info from ers_view;
```

Save and run, redirecting output to another text file:

```
dbaccess catalogue < /tmp/tim.sql >> /tmp/tim.out
```

2.4.1 Command line processing using echo

Handy for quickly running once off commands of from bash scripts.

```
echo "select * from t_file_types" | dbaccess catalogue
```

2.4.2 Changing geotype to wkt

For batch export to the django catalogue the geometries need to be exported as wkt (well known text) which is not the type used internally for the spot catalogue.

```
echo "update GeoParam set value = 0 where id =3;" | dbaccess catalogue
```

2.4.3 Reverting geotype to informix format

To set geometry output back informix representation and restoring normal catalogue functioning do:

```
echo "update GeoParam set value = 4 where id =3;" | dbaccess catalogue
```

2.4.4 Informix environment preparation

```
export INFORMIXSERVER=catalog2
```

from the shell to make sure you have the informix env set up

```
Superclasses - OK 3 Records
DataMode - OK 3 Records
EllipsoidType - OK 2 Records
ErsCompMode - OK 2 Records
FileType - OK 18 Records
HeaderType - OK 7 Records
Satellte - OK 9 Records
Satellite - OK 15 Records
SpotAcquisitionMode - OK 3 Records
Station - OK 110 Records
Medium - OK: 157277 Records, Failed:0 Records.
Localization - OK: 1179257 Records, Failed:0 Records.
SegmentCommon - OK: 157277 Records, Failed:0 Records.
```

Note: Also probably no longer needed!

2.5 Backup and Restore of the postgres ACS clone

To backup the ACS postgres database do:

```
pg_dump -f acs-‘date +%a%d%b%Y‘.sql.tar.gz -x -O -F tar acs
```

To restore the postgres database do:

```
pg_restore -F t acsThu21May2009.sql.tar.gz |psql acs
```

3 Procedures for importing data from various sources into the catalogue

Note: This document should be considered compulsory reading before you attempt to import any data into the catalogue.

Note2: This document *must* be kept up to date when you make changes to import scripts etc.

The catalogue system provides search access to metadata describing acquisitions that have taken place from a variety of sensors. This metadata needs to be lodged in the database in one way or another. Different sensors have different entry points into the system - and this document tries to cover the various permutations and procedures for lodging data into the database.

3.1 Legacy ACS System

3.2 SPOT Image Data

3.3 Sumbandilasat

For sumbandilasat the procedure for import at the moment boils down to this:

- Wolfgang / other SAC staffer performs initial L1Ab processing of imagery
- Products are placed on the SAC storage array and an email is sent to Tim detailing the names of the new product directories.
- The products are copied over to LION into ‘/mnt/cataloguestorage/imagery_processing/sumband
- The .shp project file is then imported into the sac database in the import schema to the 'sumb' table
- The scripts/sort_sumb_imagery.py script is then run. This converts the sumb pix images to tif and then files them under imagery master copies in the L1Ab folder as shown below.

```
imagery_mastercopies
+-- C2B      <-- CBERS
|   +-- 1Aa
|   +-- 1Ab
+-- S-C      <-- SACC
|   +-- 1Ab
+-- ZA2      <-- sumbandilasat
    +-- 1Aa
    +-- 1Ab
```

- The scripts/sort_sumb_raw_imagery.py script is then run. This archives the raw folder and files it into the L1Aa folder as shown above. Note: This step will be merged with the above step for convenience.

After this process the new data should be searchable in the catalogue, thumbnails should be available, and the raw products should be downloadable.

3.3.1 Copying the product folder over to LION

Currently we pull the data over from the storage array to LION. This is carried out using rsync. Here is an example of copying a DIMS project folder over:

```
cd /mnt/cataloguestorage/imagery_processing/sumbandilasat
rsync -ave ssh cheetah:/S/INT/RI/SS1/20100901_20100910 .
```

The copied over project file should have a structure something like this:

```
20100801_20100830
+-- imp
|   +-- ThNl
+-- raw
    +-- I0049
    +-- I0085
    ...etc
```

So the data in imp will be converted from pix into tif and made available as L1Ab products. The data in Thnl will be imported as prodct thumbnails or 'quicklooks'. The folders under raw will be archved using a filename that matches their sac product ID and made available as downloads.

3.3.2 Importing the report file

Once the project folder has been carried over to LION, you need to import the report file into the database. To do this the report file needs to be copied over to ELEPHANT (the database server), the temporary sumb import table cleared and the new report file brought in to populate that table.

There is a django model called 'Sumb' which maps to this temporary import table

- it is not used for anything besides data import and can be safely removed if you do not use Sumbandilasat on your catalogue deployment.

  The report file comes in two forms, a Geomatica 'PIX' file and a 'Shapefile' (which is actually a collection of a number of files).

  ```
  SARMES_SS1_20100409-20100712_rep.dbf
  SARMES_SS1_20100409-20100712_rep.pix
  SARMES_SS1_20100409-20100712_rep.prj
  SARMES_SS1_20100409-20100712_rep.shp
  SARMES_SS1_20100409-20100712_rep.shx
  ```

  To move these files (the name will differ by product folder so this is just an example) to elephant we do:

  ```
  scp -P 8697 SARMES_SS1_20100409-20100712_rep.* elephant:/tmp/
  ```

  You will need login credentials for elephant of course. Once the files are transferred, you need to log in to elephant (196.35.94.197), clear the import.sumb temporary table and import the report file:

  ```
  ssh -p 8697 elephant
  cd /tmp
  ```

  Now open a db session and clear the sumb temporary table:

  ```
  psql sac
  delete from import.sumb;
  \q
  ```

  Now load the report shapefile into the temporary table (lines wrapped for readability):

```
shp2pgsql -a -s 4326 -S \
  /tmp/SARMES_SS1_20100409-20100712_rep.shp \
  import.sumb | psql sac
```

If you have a batch of report files to import in one go you can do it with a bash one liner like this:

```
for FILE in *.shp; do shp2pgsql -a -s 4326 -S $FILE import.sumb | psql sac; done
```

After importing, you can verify that all product records were loaded in the metadata table like this:

```
psql sac
sac=# select count(*) from import.sumb;
\q
```

Which should output something like this:

```
 count
 -------
  352
 (1 row)
```

Now log out of elephant and we will continue with the import on LION.

3.3.3 Unified product migration

Our goal here is to convert the Sumbandilasat data into the SAC Unified Product Model (UPM). The purpose of the UPM is to use a common product table for all sensor types - it includes only cross cutting attributes and does not try to model sensor specific attributes of a product. There are a few UPM specialisations - UPM-O for optical products, UPM-R for radar products and UPM-A for atmospheric products. Since Sumbandilasat is an optical product, metadata records will be lodged as UPM-O.

```
cd /opt/sac_catalogue/sac_live
```

now edit `‘scripts/sumb_importer.py‘` and at the bottom of the file populate the list of project folders to process. Also make sure that `‘mSourcePath‘` at the top of the file is correct (you would typically not need to change it).

Now run the script by typing:

```
python manage.py runscript sumb_importer
```

To achieve this we will run a python script that will do the work for us.

3.3.4 Downloadable Products

3.4 CBERS

3.5 SACC

4 Catalogue Reporting tools

The catalogue provides numerous interactions for users and is continually being updated with new metadata records. It is useful to produce reports that allow SANSA staff to obtain the pulse of the system. These reports cover 4 main areas:

1. Data holdings

2. Search activities
3. Visitor statistics
4. Order and tasking activities

The reports can be obtained in one of two ways:

1. Visiting the 'staff' area of the web site and selecting from the reports presented there
2. By direct email. Here staff can nominate which reports they wish to receive, and with which frequency they receive them

Note: Only 'staff' members are elegible to receive reports.
Reports sent by email will be in either rich html format, or as pdf attachments.