VERY LARGE TELESCOPE

Meteorological Prediction Software User and Maintenance Manual

Document No.: VLT-TRE-ESO-17443-1678

Issue No.: 01.2

Date : October 19, 1998

Prepared by: P. Labit

Released by: M. Sarazin

Issue/Rev.	Date	Section/Page affected	Reason/Initiation/Document/Remarks
1.0	11.09.1998		
1.1	06.10.1998	6.2.1	IR image transfer more robust (satellite)
1.2	19.10.1998	6.1.3	access to SQL database changed (getlastmeteo)

Change Record

Contents

List of Figures

1 Introduction

1.1 Scope

This document describes the software currently installed on the various ESO Workstations for the purpose of producing forecasts of meteorological observing conditions at ESO Observatories

1.2 Description

The efficiency of a ground based observatory is considerably increased if observations can be planned in a flexible way, taking into account the expected atmosphere environment of the night, in particular with respect to standard meteorological quantities, cloudiness and water vapor.

After the positive conclusion of a feasibility study made by the CRS4 organisation ([CRS4 96] and [CRS4 97]) under contract with ESO, it was decided to initiate a pre-operational phase. To do so, data obtained from external providers have to be processed adequately to show that an operational forecast for the observatories is possible. The aim of all scripts, applications and utilities presented in this report is to publish and to archive data extracted from the different sources for the forecasts of meteorological parameters for the sites of La Silla[1], Paranal [2]and Chajnantor[3]. To understand properly the path followed by the data, a graphic is presented here, which will be used as a reference all along the report.

We shall first study how to use the different utilities developed to display or manipulate the data. The second part is more detailed, with the description of all the tools developed and of their maintenance which provide ESO with a functional and quasi-operational way to forecast the meteorological conditions above the three sites of interest. After a verification period of a few months, the system will then integrate regular operations of the observatories.

[1]located at 70W42',29S16'
[2]located at 70W24',24S37'
[3]located at 67W45',23S01'

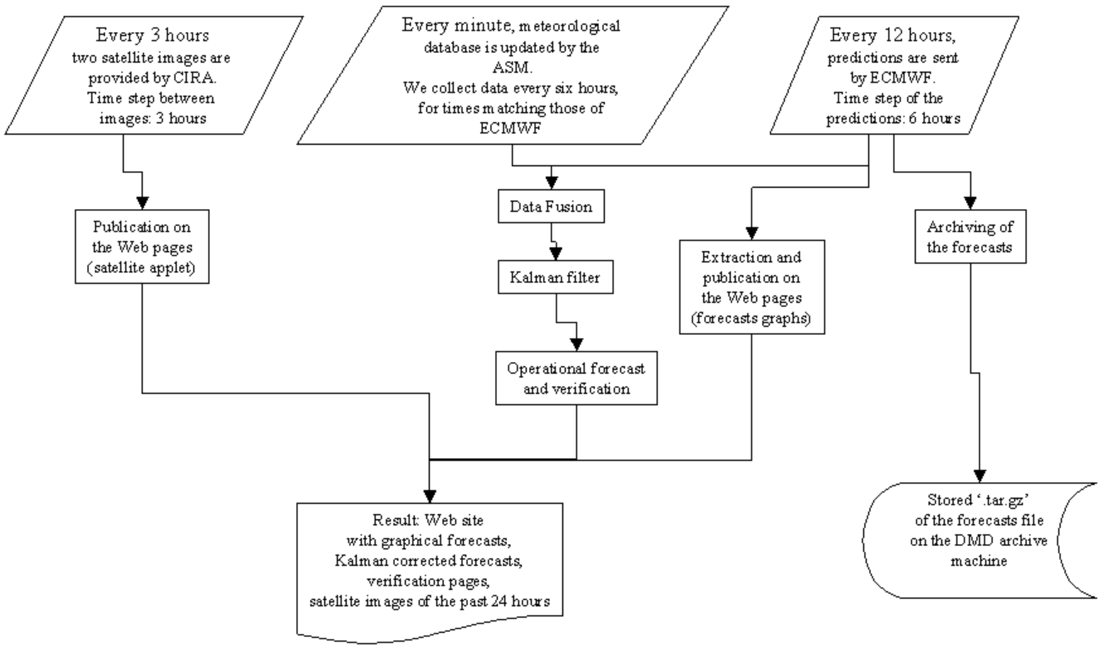
Every 3 hours two satellite images are provided by CIRA. Time step between images: 3 hours
Every minute, meteorological database is updated by the ASM. We collect data every six hours, for times matching those of ECMWF
Every 12 hours, predictions are sent by ECMWF. Time step of the predictions: 6 hours
Publication on the Web pages (satellite applet)
Data Fusion
Kalman filter
Operational forecast and verification
Extraction and publication on the Web pages (forecasts graphs)
Archiving of the forecasts
Result: Web site with graphical forecasts, Kalman corrected forecasts, verification pages, satellite images of the past 24 hours
Stored '.tar.gz' of the forecasts file on the DMD archive machine

Figure 1: Path followed by the data

1.3 Reference Documents

-VLT-SPE-ESO-17410-1174	VLT-ASM Seeing and Coherence Monitor, Dimm Upgrade Plan V-2.3, Aug.98
-VLT-TRE-CRS-17443-0001	Feasibility Study of a Meteorological Prediction Model for ESO Observatories in Chile (Phase A1-A2) 11 Jul. 96
-VLT-TRE-CRS-17443-0002	Feasibility Study of a Meteorological Prediction Model for ESO Observatories in Chile (Phase A3-A4) 11 Apr. 97
-VLT-TRE-ERA-17440-1038	Feasibility Study and Development of Software Computer Code for Cloud Cover and Water Vapour Forecasts (+appendix) Jan. 96
-VLT-TRE-ERA-17440-1039	Feasibility Study and Development of Software Computer Code for Cloud Cover and Water Vapour Forecasts. Phase II: Parameter Definition and Requirements for an Operational System 27 Jul. 97
-VLT-TRE-UNI-17440-0004	Feasibility Study of Seeing Prevision using the MESO-NH meteorological model Dec. 97
-The Messenger No:89	Predicting Observing Conditions at ESO Observatories - Reality and Perspectives Sep. 97

Part I

User's Manual

2 ECMWF Products

2.1 Description of the products

Every twelve hours, ESO receives from ECMWF[4]a set of files. These files arrive on the opus3 machine, which is supposed to be one of the most reliable of the whole ESO network, for the user *asmpredi* in the directory */home/asmecmwf/*. They are all the same length, which is 34,816 bytes. In those files we find the value of 4 parameters (temperature, u-component of wind, v-component of wind and geopotential height) for nine levels of pressure (100 mb, 200 mb, 300 mb, 400 mb, 500 mb, 700 mb, 850 mb, 925 mb and 1000 mb).

During the night, around 4:00 local time,ESO receives 7 files extracted from the 00Z dissemination cycle of ECMWF. Those files cover a range of 36 hours of prediction, with time steps of six hours. It is to say these are analysis for 00UT (T + 0), and predictions for T+6, T+12, T+18, T+24, T+30, T+36. There is one file per time step. The name of the files are ESDxxyyy0. The first two numbers (xx) represent the time step, with respect to the formula : xx = forecast time step / 3 + 8. The three following numbers (yyy) are the day of the year (000 to 366).

During the day, around 8:30 local time, ESO receives 20 files extracted from the 12Z dissemination cycle of the ECMWF. Those files cover a range of 192 hours with a time step of 6 hours until T+36 and 12 hours after. It is to say we have the analysis for 12UT (T + 0), and the forecasts for T+6, T+12, T+18, T+24, T+30, T+36, T+48, T+60, T+72, T+84, T+96, T+108, T+120, T+132, T+144, T+156, T+168, T+180, T+192. There is one file per time step, and the name of the files are ESDxxyyy4. The first two numbers (xx) represent the time step, with respect to the formula : xx = forecast time step / 6 + 4. The three following numbers (yyy) are the day of the year (000 to 366).

In each file we find either the analysis or the forecast for three gridded region : the first range from 65W,10S to 110W,45S with a step of 3 degree. The two others use a step of .5 degree, and range respectively from 67W,22S to 71.5W,26.5S and from 67W,27S to 71.5W,31.5S.

These files respect the GRIB 92 ECMWF format [ECMWF 97], and thus some utilities are needed to decode them in order to get the data in a binary format readable by such utilities as *GrADS*[5] (see section 2.3).

2.1.1 Archiving the products

The first thing to do when we receive the files is to archive them on the DMD archive machine olasg. Two scripts were then written, which names are *ECMtoOlasg* and *ECMtoOlasg2*. They are located in */home/asmecmwf* on opus3 and execute themselves once per hour. The aim of these scripts is in a

[4]European Center for Medium range Weather Forecast, Shinfield Park, Reading, Bershire, RG2 9AX, England. Web site : *http://www.ecmwf.int*

[5]GrADS, Grid Analysis and Display System, is a freeware. An homepage can be found at the URL: *http://grads.iges.org/grads/head.html*

first time to put the files on a big disk on olasg, and next to archive these files in a tar file on another disk as backup of the first one. The two scripts work exactly in the same manner.

To keep integrity, we transfer only entire fileset. That is to say we wait either for the *ESD20???0* file or the *ESD36???4* file before transferring the corresponding set. Then in a first time, the script looks for the last day of transfer from opus3 to olasg. In order to do this, we 'ftp' olasg and we look for the last file transfered. Then we deduce the day of the last transfer. We increment this day variable, and as long as there is a set of files to transfer from opus3, we transfer sets of file in the directories */home2/asmpredi/ECMWF/data/data00UT* or *.../data12UT* according to the case on olasg. Next we produce a tar file with the corresponding set of files, we compress this tar file and we transfer it to the directory */diskb/asmpredi/ECMWF/archive* on olasg. We can then remove the files from the opus3 machine.

2.2 The ECMextrDAT program

2.2.1 Conception of the program

The data provided by ECMWF are in GRIB 92 ECMWF format [ECMWF 97], which is not readable by meteorological utilities like GrADS (see section 2.3). Moreover, there are three grids mixed in the files, and there is one file per time step, instead of one file per grid with all the times steps. Then we need to convert the data in order to be able to visualize them.

ECMWF has developed some libraries of functions in fortran in order to decode this format of files. The heart of the library is the function Gribex. To have explanations about this function and its coding, please report to the file gribex.f, located on olasg in the directory */diskb/asmpredi/ECMWF/software* where the function and its parameters are explained [ECMWF 95]. In fact, this function decodes one record of a GRIB file. The solution was then to use this function to sort the three types of grid, and write, with one set of files from ECMWF, three binary files readable by most utilities.

The conversion utility, *ECMextrDAT*, is based upon a loop which length depend on the number of files from ECMWF (seven for 00Z or twenty for 12Z). In a first step, it scans the GRIB files to sort the records depending on the grid to which they belong. It creates the three tables, in which it puts the exact place of the record in the original file.

In a second time, it calls a subroutine to sort the three arrays to respect the GrADS conventions: the variables must be sorted first by time, next by pressure level, and then by variable. Then the subroutine returns the three sorted arrays to the main loop, in which the three arrays are read, and the binary files written with respect to the place of the records in the arrays.

2.2.2 Use of the utility

The utility *ECMextrDAT* is located on the olasg machine in the directory */home2/asmpredi/ECMWF/software*. It must be linked with the two libraries *pbiolib.a* and *griblib.a*.

The set of files to transform must be in the same directory as the utility. the command line is:

```
olasg: ECMextrDAT 125 0
```

The first number is the day of the year of the files, and the second one is the extension of the files (0 for 00Z predictions and 4 for 12Z predictions).

Then *ECMextrDAT* reads the set of files from ECMWF and produces three or six binary files:

- With the 00Z file set (seven files), *ECMextrDAT* produces three files named *ESDyyy0_x.dat* (binary files) and the three associated control files for GrADS. The extension yyy stands for the day of the year, and x is either 1,2 or 3 (1 stand for the general grid (3x3 degree), 2 is for the Paranal-Chajnantor region and 3 means the the La Silla grid).
- With the 12Z file set (twenty files), the utility produces six files named *ESDyyy4_x.dat* (binary files) and the six associated control files for GrADS. The extension yyy stands for the day of the year, and x is between 1 and 6 (1, 2 and 3 stands for the predictions with a time step of six hours extracted from the ECMWF files with the same convention as above. 4, 5 and 6 stands

for the long term predictions with a time step of twelve hours extracted from the ECMWF files with 4 for the general grid, 5 for Paranal-Chajnantor and 6 for La Silla).

2.3 The GrADS front-end application

2.3.1 Introduction

In order to visualize the data provided by ECMWF, an utility designed to display meteorological data, *GrADS*, is used. Its aim is to display meteorological data. This utility is distributed as a freeware, which means that it is free of charges. It was first created at IGES by Mike Fiorino and Brian Doty. It is permanently maintained and updated, and used by most meteorological institues worldwide. One can visit the home-page for this utility which is located at the URL : *http://grads.iges.org/grads/head.html.*

In order to import the data in *GrADS*, a transformation of the file was needed, which is done by the previous utility (see section 2.2). But *GrADS* is not very user friendly. Then a tool was developed in order to browse the data.

2.3.2 Execution of the GrADS application

The visualization script is located on the olasg machine, in the directory */home2/asmpredi/ECMWF/data/tmp.* Its name is *mainMETEO.gs.* It is adapted from a script developed by CRS4 [CRS4 96], in order to display the data sent by ECMWF.

To run the script, one just go into the previous directory and type 'grads'. This starts the *GrADS* utility. After that, one just need to type 'run mainMETEO.gs' in order to execute the script. All the transformation of the data files are done automatically by the application.

One will then be prompted for the day of the predictions one wants to examine. Then, the only requirement is to enter a three digit code corresponding to the number of the day in the year. Then one is prompted for the hour to examine (0 or 4, see above for details on the significance of those codes). If 4 (which means the 12Z predictions) is entered, the application will prompt for either the short term predictions (0) or the long term ones (1). If one entered 0 at the previous step, nothing is to be done as only short term predictions are available.

If all the files exists and are correct, the script will perform a *ECMextrDAT* on the input files to produce the three *GrADS* files needed (plus the three .ctl, see section 2.2). The user interface will then be displayed in the graphic window.

2.3.3 Overview of the screen

When the scripts starts, a title screen is displayed and so are some buttons bars and menus. The screen should look like the one presented on figure 2. There are menus and buttons on three side of the screen. Below is a small explanation of the buttons, but a more precise explanation is in section 2.3.4.

- On the top are the main tools of the utility (zoom, type of projection, type of representation, print or not,...).
- On the right are the buttons used to control the time and the pressure level.
- On the bottom are the buttons used to control the type of data to display and the type of display.

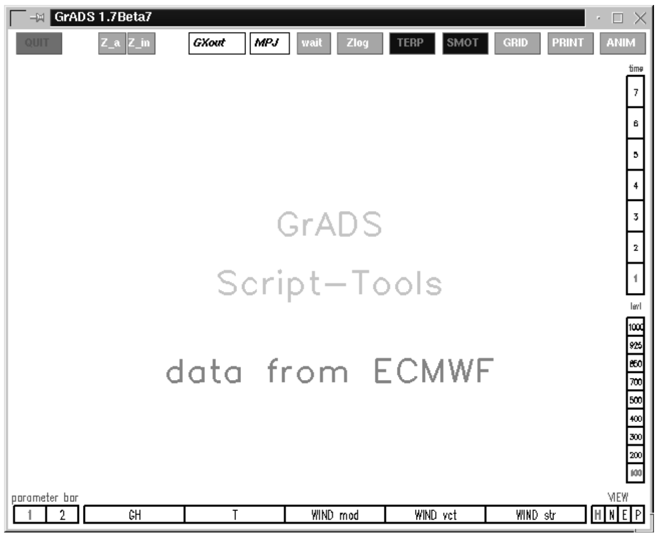

Figure 2: Title screen from mainMETEO.gs

The green button at the top left is the "quit" button. When clicked, it stops the GrADS utility and deletes all the temporary files created by the utility.

2.3.4 Manual use of the Utility

Types of data to display The bottom button bar looks like the one presented on Figure 3. The

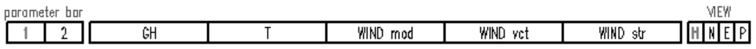

Figure 3: The bottom button bar

bottom button bar allows the user to choose the type of data to display. Since there are too many types of data, those ones are split into two buttons bar. The swap between these two is done with the bottom buttons '1' and '2'.

With the first type of parameter, the geopotential height (GH), the temperature (T), the magnitude of wind (WIND mod), the direction of wind (WIND vec) and the streamlines of wind (WIND str) can be displayed.

With the second type, the horizontal divergence (hdiv) of the wind, the vorticity (hcurl) of the wind can be displayed with or without the streamlines of wind.

The four last buttons are aimed at controlling the type of view that can be obtained from *GrADS*.

- With the first (the H button), one obtains a lat/lon view of the area selected (see below for the different area).
- With the N button, one obtains a two dimension graph with the longitude on the x-axis and the pressure level on the y-axis.
- With the E button, one obtains the same two dimension graph but with the latitude on the x-axis.

- With the P button, one obtains a graph with time on the x-axis and pressure level and y-axis for a given point of the map. For the last three buttons (N,E or P), after clicking on one of them, the program will ask the user to click on a point of the map.

Time and Pressure level: This button bar (right side of the screen) controls the pressure level and the time for which the data are plotted. It is enabled only in the lat/lon view (H mode), since there is no interest of having these two buttons bar in any of the three other display modes.

Controls: The top button bar should look like the one presented on Figure 4. The first button

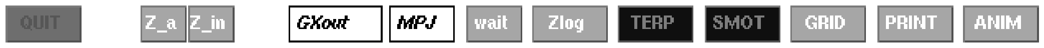

Figure 4: The top button bar

("quit") was explained above. The next two buttons are used to zoom (Z_in) or un-zoom (Z_a). When (Z_in) is clicked, the program asks in which region the user wants to zoom. Then one should click in one of the two rectangles to see the zoomed area. The button (Z_a) is then used to return to the general map.

After that are the two menus controlling the display, GXout and MPJ. GXout handles the way data are presented (contours, shaded surfaces, vectors, streamlines,...) and MPJ controls the type of projection of the base map (lat/lon, SPS, NPS or Robinson).

The 'wait' button is helpful if the user wants to print a hardcopy of the graphic windows. When enabled, *GrADS* draws the map and the data normally, but waits for a user click before displaying the buttons and the menus. It is generally used with the button 'print'. When enabled, this button will make *GrADS* produce a meta-file each time a new data is plotted. The meta-files produced have generic names: *mfxxx.meta*, where xxx stand for the number of the metafile during this session of the application. The user must be aware that *GrADS* will print the whole graphic screen, including the menus and the buttons if 'wait' is not enabled.

The "Zlog" button is useful only in the sections displaying modes (N, E and P). It draws the y-axis (the pressure level), with a logarithmic scale.

The "SMOT" button allows GrADS to perform a smoothing between the points of the grid, to smooth the display.

The "GRID" button draws on the map the grid on which the computation were made.

The 'ANIM' button cycles between the time steps to produce an animation indexed by time (only in H, N and E modes).

2.4 Background applications

Some applications have been developped to be executed in background, in order to produce images and text files for the Web pages. These applications are described more precisely in the the part II. Basically, there are first three scripts designed to produce all the images shown on the Web pages. Those applications, whose names are *script1.gs*, *script2.gs* and *script3.gs*, produce the "lat/lon" maps for the Web pages. The first one deals with the general south pacific region, the second deals with the zoomed region above Paranal and Chajnantor, and produces the reconstructed radio profiles for those two sites, and the third deals with the site above La Silla, and produces the reconstructed radio profiles above this site. Those applications are called by Unix scripts to be executed automatically once per day to update the Web site. Normally, the user should not have to execute these scripts himself. If the update of the Web pages was not made correctly, one should look for the files *scrWeb0* and *scrWeb4* and execute them instead of executing directly the *GrADS* applications.

Another set of background applications contains the applications designed to produce the text files for the forecast verification pages. The first one, *verif0.gs*, is called by an Unix-script every day to produce the data for the graph and verification pages. The second one, *writetxt.gs*, produces the kalman-corrected file, for the java applet presented on the verification pages. Again, the user should not have to call those applications directly. The main script for the verification is called *step2* and is executed once per day. It updates the files *te.log*, *pr.log* and *ws.log*, which are the log files for the temperature, the pressure and the wind speed. If there was a malfunction in those scripts, one should know that there are some backup of those three critical files: the names of the backup are *te.log.bak*, *pr.log.bak* and *ws.log.bak*.

3 Satellite products

3.1 Introduction

In order to display a complete set of tools for the weather forecasts for the three site of interest, ESO has an external contract with an american institue, CIRA [6]. This institue provides ESO with two types of images, one taken at the wavelength of 10.7μm, and which enables to visualize the cloudiness, and the other is taken at 6.7μm, and enables to measure the precipitable water vapor present in the atmosphere (PWV) and the amount or cirrhus clouds as well.

The second type of images is the object of an external contract with Andre Erasmus (*erasmus@saao.ac.za*). By merging the images with the predictions of wind from ECMWF, it produces a forecast of which pixel will be above the sites of La Silla and Paranal in the next hours. By converting its gray level into PWV, one gets the forecasts of PWV. With the first type of images, we just extract some zone of interest and publish it on the Web as it will be explained in the next section.

3.2 The McIdas IDL front end utility

3.2.1 The execution of this utility

The images, provided by the CIRA institute, are coded with the McIdas format. In fact, it contains more information than in a simple image. As the images are a view from earth taken at 36.000 km, the projection is not a linear one so each point contains, in addition of its gray level, the coordinates it represents on the Earth. One can not just take an "image file" and try to open it with an image browser. One needs an utility, just like *GView*, the one provided by CIRA and which runs under *IDL*[7].

To start this utility, one needs first to start the *IDL* software on olasg in any directory. The path is in the environement variables, so one just need to type 'idlde' at the command prompt to start *IDL*. A Graphical interface will appear, with a bottom line used as a command line. So in this one, one just type 'gview' and *IDL* reads the program files, compile it and then execute the *GView* Utility.

3.2.2 The manual use of the GView utility

The use of the *Gview* utilty is pretty simple. Nevertheless, one can access an online documentation of the software in the HTML format. The head file is *GVIEW.HTM*, and is located in the directory

/diskb/asmpredi/CIRA/software/html/. Meanwhile, an abstract is presented here to help the user in the main functions of the utility.

When the utility starts, one should see a menu similar to the one shown on figure 5. From there, one can open a McIdas file, save the image in numerous formats (Raw, Tiff, Jpeg, Jpeg 24 bits, PGM, PPM, Postscript, Color Table) [button "File"], manipulate the color tables [button "Color"], load some graphic files to add to the image [button "Graphic"], diplay the image loaded (enabled only after the reading of a McIdas file) [button "Display"], display only part of the image [button "Sector"], and browse through the different images present on the disk [buttons "Prev" and "Next"].

[6]Cooperative Institute for Research in the Atmosphere, Colorado State Universtity, Fort Collins CO 80523, USA, URL: *http://www.cira.colostate.edu*

[7]Interactive Data Language, Research Systems, URL:*http://www.rsinc.com*

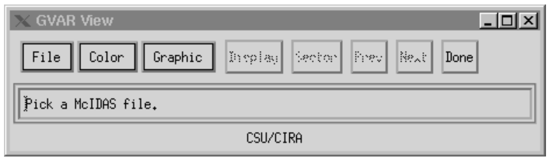

Figure 5: The Gview main button bar

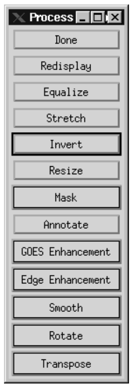

Figure 6: The Gview display button bar

After the loading of an image and its display with the button "Display", one can see the button bar presented on figure 6. This button bar allows the user to apply some treatments to the image file. Those special effects are described in the online documentation of the products. One just needs to know that it enables to user to invert the colors [button "Invert"], to manipulate the color histogram [button "Equalize"] and to apply some enhancements. Meanwhile, in this version provided by CIRA, some buttons are ineffective, like the "Resize" button, the resize function not being yet implemented.

3.3 Bakground applications

Some applications were developped to go to the site of CIRA, download the images and apply some treatments on them for later publishing on the Web. Those applications consist in one script file on olasg, *satellite* and one main IDL application, *tryRead.pro*, which uses a startup IDL file *satellite.pro*, designed to set some global variables.

The experience shows that the sensitive point in those operations is to take the images from the CIRA ftp site to the ESO 'olasg' machine. The link being sometimes very slow, the 'ftp' sometimes cuts down the transmission, and the file is not transmitted properly, which results in a "hole" in the satellite image sequence. A "log" file is located in the directory */diskb/asmpredi/Verif*, under the name *satellite.report*, to present the correct transmission and the failures.

The other parts of the process, like the processing of the images and their publishing on the web pages, seems not to pose any problem. The path followed to process the images is then the following: *satellite* (unix script) which goes to get the images on the CIRA site, *satellite.pro* (IDL applications), which sets global variables and calls *TryRead.pro* (IDL application) which processes the image. Those three step are located on olasg. The Unix script located on the web4 machine is used to get the processed images on olasg and publish them on the web pages.

4 The Local products

4.1 Introduction

To verify the forecast, one needs to access some measured data to compare with the forecasts. ESO has installed meteo-monitors on top of the sites of interest (for now La Silla and Paranal). Those meteo monitors take mesures every minute and update a database containing meteorological data. This database is located on the olasg machine at ESO, and maintained by Benoit Pirenne (bpirenne@eso.org).

The goal for us is to access those real data for times matching times of predictions by ECMWF. In doing so we have for each time of "prediction" one triplet of points: the forecasts of ECMWF, the analysis of ECMWF (run of their model for time T=0) and the measured data (20 minutes average).

4.2 Background applications

To insure the update of the files containing the triplets of points as seen before, some Unix scripts are used to get the local data, to extract the forecasts and the analysis and to run the Kalman filter on the predictions. The first script, which goes in the meteo database and get the data for time matching those of ECMWF (00 UT,06 UT,12 UT,18 UT), is located on the olasg machine in the directory */diskb/asmpredi/Verif/LaSilla* or

/diskb/asmpredi/Verif/Paranal (for now the data for Paranal is not available, but the scripts have been readied, to insure the correct functionning as soon as data become available). This script, named *getlastmeteo*, gets the measured data of the ground temperature, the wind speed at ground level and the ground pressure, and record them in 'log' files nammed *te.log*, *ws.log* and *pre.log*. Those files are located in the directories mentionned above.

The structure of the files is quite easy to understand. For each record (each line), it contains the date, the hour, the measured data, the analysis of ECMWF, the closest forecast from ECMWF and the kalman corrected forecast (see later for explanations concerning the kalman filter). Those log files are maintained by a set of scripts: *step1*, which calls the getlastmeteo script, *step2*, which fills the analysis and forecasts column and calls *step3*, which fills the kalman corrected column by calling a GrADS application: *verif0.gs*. The latter calls the two user defined functions *cuspLaSi* or *cuspPara*, for LaSilla and Paranal. The second step, *step2*, calls also another GrADS application, *writetxt.gs*, which deals with the production of a text file needed by the verification java applet. Then it calls at the end a gnuplot application to produce some graphics presented on the verification pages (*step4.gp*).

5 The internet pages

5.1 Overview

In order to suit the needs of the astronomer, the data sent by ECMWF had to be postprocessed and displayed graphically. This has been done on a web site dedicated to those data and automatically updated twice a day with new images and graphs.

The URL of the web site is :

http://www.eso.org/gen-fac/pubs/astclim/forecast/meteo/

Browsing the site is quite intuitive, since all is well explained and all the links are well defined.

On the site, the user will be able to see the predictions for the chilean coast, for the regions of Paranal and La Silla, both for short and long term. The user will find some time indexed graphics, to follow and anticipate the evolution of the situation above the three sites of Paranal, La Silla and Chajnantor.

On the first page, the user can choose to see some more quantitatives pages dealing with what is called the corrected forecasts and the forecast verification. For the first topic, the corrected forecasts, the basic reason is that ECMWF uses a grid for their predictions which is too coarse. It means for example that for them the site of Paranal is located on the sea (they use a 0.5 degree grid, which corresponds to 60 km). So we apply a kalman filter, which analyses the difference between the forecasts and the measures over the past 14 days, and corrects in an appropriate way the next forecasts. The results are in agreement with the ones CRS4 obtained during the feasability study [CRS4 97].

For the second topic, the forecasts verification pages contain statistics about the accuracy of the raw predictions, compared to the accuracy of the kalman-corrected predictions.

5.2 Web publishing scripts

5.2.1 ECMWF products

The update of the site is made automatically by dedicated scripts which are described in the following pages. These scripts are on the olasg machine, in the directory */home2/asmpredi/ECMWF/data/tmp/*. The top level scripts are named *scrWeb0* and *scrWeb4*, each one taking care of one of the time period (0 for the 00Z predictions and 4 for the 12Z ones).

Basically, these scripts look for the last set of files received from ECMWF. Then they process those files with a *GrADS* application in order to produce the images which will be published on the web. These images, first in a metafile format, are processed to produce .gif files which are then transferred to the web4 machine through a ftp canal by scripts named *publish0* and *publish4* located on the web4 machine.

The GrADS applications used are *script1.gs*, *script2.gs* and *script3.gs*, which are described in this document (see section 2.4).

5.2.2 CIRA products

As seen in the introduction, CIRA is providing satellite images to ESO in order to follow the evolution of the cloud cover on the south pacific region and on the chilean coast. Those images are processed to produce eight images which are published on the Web. On the first page of the satellite images,

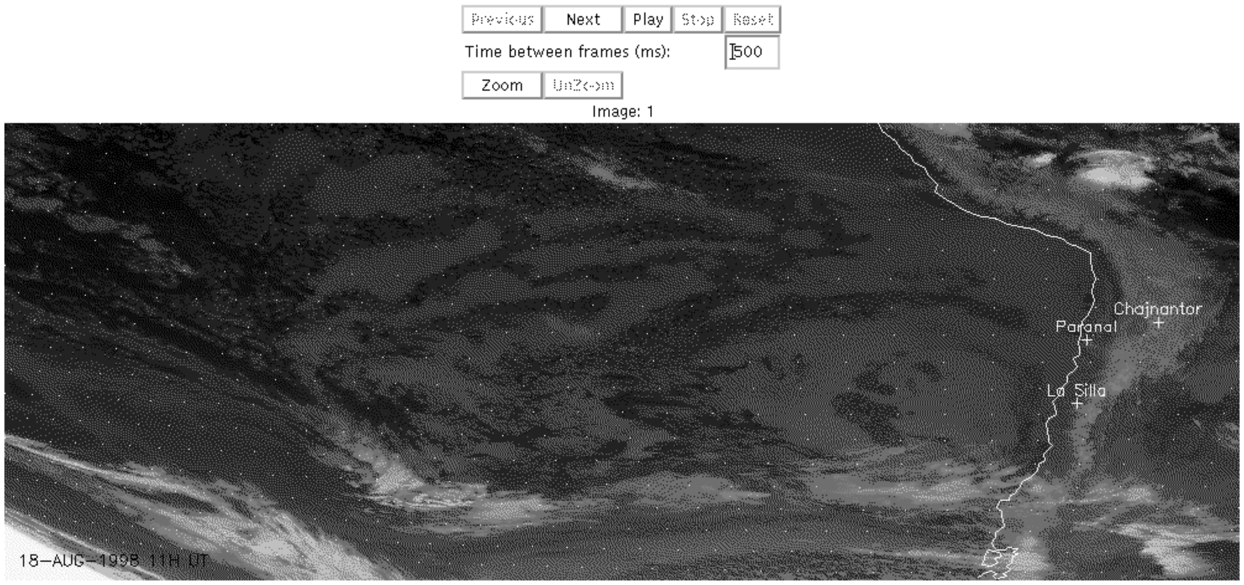

Figure 7: The Satelitte applet

the user can view the last updated image. By clicking on the link just above the image, the user has access to a java applet, which lets him see the last eight images (24 hours). The use of the applet is really simple (see fig.7, the user can browse between the images (buttons next and previous). He can also start an animation of the eight images (the time between each frame can be regulated by the text box below the first button bar). He can stop the animation and reset the images (it displays the first image and resets the image counter).

Another thing the user can do is to click on the button zoom to display a more accurate view of the Chilean coast in the area of the observatories.

5.2.3 Verification pages

The verification pages are provided to present the results of the corrected forecasts. As said in the introduction, the corrections are made by a kalman-filter, initiated by 14 days of predictions, local measures and already corrected forecasts. The program is described in the users manual (see sec. 6).

The navigation on the verification pages is really easy, the first pages are made with three graphs (pressure, temperature and wind speed) to show the evolution of the three type of meteorological data (forecasts, kalman corrected forecasts and measures). There are two versions of this first page. On the first is added a small java applet to show the starting and ending dates of the graphs, and then to show the next kalman corrected forecasts. The user can choose the date, the hour of prediction and the type of parameter to display.

On the second page are presented three more graphs showing the matches with real data for the three parameters and the accuracy of the kalman corrected forecasts with respect to the raw forecasts. These pages should provide the user with some informations about the accuracy of the predictions during the last 15 days. It might be useful to determine if the predictions are reliable or not on the long term, therefore a verification archive has been provided (in the directory */diskb/asmpredi/Verif/LaSilla* on the olasg machine).

6 Fortran programs used to produce the forecasts

To improve the forecasts quality for a better accuracy with the real measured data, some programs were developped. Those programs were written in FORTRAN to be consistent with the ECMWF data processing software. In fact, three programs were developped.

- The first set of *GrADS* user defined functions (*biliLasi*, *biliPara* and *biliChaj*) is aimed at refining the predictions. As the reader has seen before, the predictions are made for equidistant points on a grid. When we ask for the prediction for a point which is not on the grid, GrADS performs a round-up with the nearest point on the grid. To improve that, CRS4 proposed to use a bilinear interpolation. That is to say that when the user requests one point, *GrADS* takes the four surrounding points and performs an interpolation with respect to the distance between the requested point and the grid points. This type of interpolation is known as a bilinear interpolation and can be described with the following formulas: Given the following stencil,

  $$F4(x1, y2) F3(x2, y2)$$

  $$F(x, y)$$

  $$F1(x1, y1) F2(x2, y1)$$

  With F1, F2, F3 and F4 being the forecasts made by ECMWF at the 4 nearest points and F the desired value, we use the equation

  $$F(x, y) = (1-t) * (1-u) * F1 + t * (1-u) * F2 + t * u * F3 + (1-t) * u * F4$$

  with

  $$t = \frac{x - x1}{x2 - x1}; u = \frac{y - y1}{y2 - y1}$$

- The second set of *GrADS* user defined functions (*cuspLaSi*, *cuspPara* and *cuspChaj*) is quite similar to the first one, except that it performs an interpolation on the altitude to get the correct forecast for the given altitude. The interpolation retained for the altitude is a cubic spline one, which seems to give better result than the bilinear one in this case. Those functions also write down the 'log' files with those exact results. Then it should never be used by the user, because a bad use couls corrupt the log files for the three parameters (temperature, wind speed and pressure).

  One needs to know that, as the ECMWF may be sometimes corrupted or may contain some errors, there is a protection in this interpolation. That means that if the interpolated data is out of ranges (defined after), it will be markes as unavailable (999.) in the log files. Moreover, as the pressure is a key for the interpolation for the temperature and the wind speed, if the pressure value is out of ranges, the three data are marked as unavailable. The ranges are:

  - $0 \leq windspeed(m/s) \leq 99$
  - $99 \leq pressure(mB) \leq 1099$
  - $-99 \leq temperature(^oC.) \leq 99$

  The cubic spline interpolation is based on the following formulas (more accurate explanations may be found in the Numerical Recipes [Press et al. 94]). Suppose that $f(p_i)$ is the vertical profile of ECMWF data. We compute $f(p)$, with $p \in [p_j, p_{j+1}]$:

  $$f = A * p_j + B * p_{j+1} + C * p''_j + D * p''_{j+1}$$

with

$$A = \frac{p_{j+1} - p}{p_{j+1} - p_j}; B = \frac{p - p_j}{p_{j+1} - p_j}$$

$$C = \frac{1}{6}(A^3 - A)(p_{j+1} - p_j)^2; D = \frac{1}{6}(B^3 - B)(p_{j+1} - p_j)^2$$

$$p'' = \frac{d^2 f}{dp^2}$$

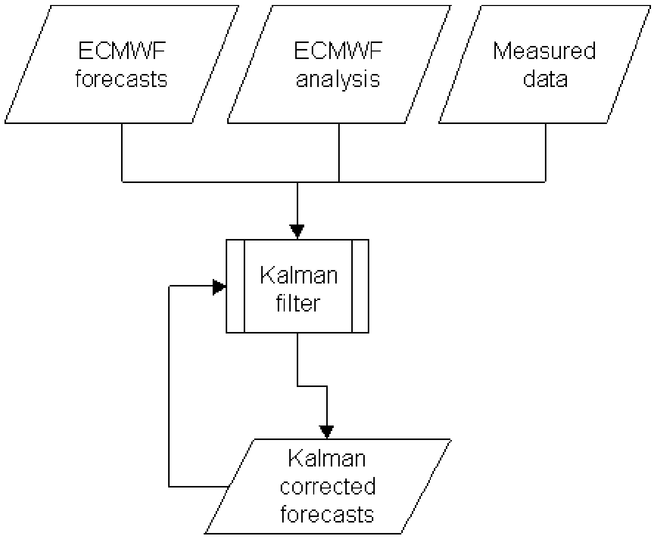

Figure 8: Principle of the Kalman filter

- The third program (*Kalman*) was developped upon a basis provided by CRS4. It performs a Kalman correction. The Kalman filter is a non linear learning filter, which means that basically it computes the difference between the forecasts or other parameters (see below) with the measured data, and then corrects the next forecasts assuming a similar trend will be observed. For the current problem, CRS4 showed, in their feasibility study [CRS4 97], that the best results were obtained by taking into account the forecast from ECMWF, the analysis (which are the run of the model for T = 0), the measured data and the last corrected forecasts. The principle of this filter is shown on the Figure 8. The initialization files for this filter are the three '.log' files (*te.log*, *pr.log* and *ws.log*)for temperature, pressure and wind.

  A manual mode is available on olasg. The user should go to directory */diskb/asmpredi/Verif/LaSilla*) and type *kalman*. The user should then provide on the command line the type of data to correct (e.g. *kalman ws*, *kalman te* or *kalman pr*), and then it asks the user for the input value to correct.

  In a background mode, this program is called by the script step2 to produce the statistics shown on the Web pages.

Part II

Maintenance manual

7 Maintenance of the scripts

7.1 Scripts related to ECMWF products

7.1.1 The transfer scripts (ECMtoOlasg and ECMtoOlasg2)

Those two scripts are on the opus3 machine, in the account asmpredi. They are used to transfer the ECMWF data files from the opus3 machine (where ECMWF put them) to the olasg machine (where we process and store them). In this section we will just comment on one of the scripts, the other being quite similar.

The scripts execute each 20 minutes (00, 20 and 40 for *ECMtoOlasg* and 10, 30 and 50 for *ECMtoOlasg2*). The crontab commant is the following :

```
00,20,40 * * * * /home/asmecmwf/ECMtoOlasg >/dev/null 2>&1
10,30,50 * * * * /home/asmecmwf/ECMtoOlasg2 >/dev/null 2>&1
```

The listing of the script follows:

```
EcmtoOlasg

#!/bin/tcsh
# running on opus3, this script uploads the prediction files
# from ECMWF
# and put them on olasg in
# /home2/asmpredi/ECMWF/data/data00UT for
# the 00ut predictions
# and in /home2/asmpredi/ECMWF/data/data12UT for the 12UT
# predictions.
# On opus3 it creates an archive with the files uploaded and
# deletes those ones.
cd /home/asmecmwf/
# goes on olasg to see the last day of transfer
ftp -n olasg << EOD > essai
user asmpredi xxxxxxx
bin
prompt
cd /home2/asmpredi/ECMWF/data/data12UT
ls -al
quit
EOD
#
grep ESD36 essai > essai2
rm essai
# we search the last day of transfer (the last file
# in the list)
@ d = `tail -5c essai2`
set dd = `expr $d - 4`
set d = `expr $dd / 10`
rm essai2
# we must transfer all the files for the completed
# days since this date
```

```
set eend = 0
while ($eend != 1)
# we add 1 to d
set d = `expr $d + 1`
if ($d < 10) set d = 0$d
if ($d < 100) set d = 0$d
# if d = 367 then d = 001
if ($d == 367) set d = 001
# is there a complete block for the d-day ?
@ f = `ls -al ESD36${d}4 | wc -l`
if ($f != 0) then
ftp -n olasg << EOD
user asmpredi xxxxxxx
bin
hash
cd /home2/asmpredi/ECMWF/data/data12UT
mput "ESD??${d}4"
quit
EOD
tar c ESD??${d}4 > ESD${d}4.tar
gzip ESD${d}4.tar
ftp -n olasg << EOD
user asmpredi xxxxxxx
bin
hash
cd /diskb/asmpredi/ECMWF/archives/12UT
mput "ESD${d}4.tar.gz"
quit
EOD
rm -f ESD??${d}4
rm -f ESD${d}4.tar.gz
else
set eend = 1
endif
end
```

Those scripts are quite simple : ECMWF transmits the files in a sequence. Once the file *ESD20???0* (for the 00Z products) and *ESD36???0* (for the 12Z products) are there, we have a complete set of files which we can transmit to olasg.

On olasg we put the files in the directory

/home2/asmpredi/ECMWF/data/data00UT or *data12UT*. After doing that we create an archive with the complete fileset, we 'gzip' it and we put it on olasg in */diskb/asmpredi/ECMWF/archives/00UT* or

/diskb/asmpredi/ECMWF/archives/12UT. In fact the script loops until it has treated the complete fileset, incrementating on the number of the day.

7.1.2 The display scripts (scrWeb0 and scrWeb4)

Those two scripts (one for each type of product) are located on the olasg machine, in the directory */home2/asmpredi/ECMWF/data/tmp*. They are used to process the ECMWF files located on olasg to produce the images needed by the Web pages.

Those scripts are executed every 30 minutes (0 and 30 for *scrWeb0* and 15 and 45 for *scrWeb4*). The crontab command is:

```
00,30 * * * * /home2/asmpredi/ECMWF/data/tmp/scrWeb0 >/dev/null 2>&1
15,45 * * * * /home2/asmpredi/ECMWF/data/tmp/scrWeb4 >/dev/null 2>&1
```

The listing of the *scrWeb0* script follows :

```
scrWeb0

#!/bin/tcsh
#
# Script designed to produce gif files to
# update web page on forecast.
#
set echo
cd /home2/asmpredi/ECMWF/data/tmp
#
ls -al /home2/asmpredi/ECMWF/data/data0OUT/ > essai
grep ESD20 essai > essai2
rm essai
# we search the last day of transfer (the last file in the list)
@ d = `tail -5c essai2`
set d = `expr $d / 10`
rm essai2
# we loook for the last day of the update
set lastd = `more day0.txt`
if ($lastd < $d || (($lastd == 366 || $lastd == 365)\
 && $d == 1)) then
if ($d < 10) set d = 0$d
if ($d < 100) set d = 0$d
cp /home2/asmpredi/ECMWF/data/data0OUT/ESD??${d}0 .
ECMextrDAT ${d} 0
rm ESD??${d}0
grads -lb -c 'run script1.gs '${d}' 0 0'
foreach file (*.meta)
gxgif -r -i ${file}
end
rm *.meta
#
grads -lb -c 'run script2.gs '${d}' 0 0'
foreach file (*.meta)
gxgif -r -i ${file}
end
rm *.meta
#
grads -lb -c 'run script3.gs '${d}' 0 0'
foreach file (*.meta)
gxgif -r -i ${file}
end
rm -f *.meta
rm -f ESD${d}0_*
rm day0.txt
cat > day0.txt <<EOD
${d}
EOD
set date = `date +%C%y.%m.%d.%H:%M`
awk '{print > "ECMWF.result"} END {printf("%s : \
Successfull traitment of day %d, product of 0OUT\n",\
"'$date'","'$d'") > "ECMWF.result"}' ECMWF.result
endif
set currd = `date +%j`
if ( $lastd < ($currd - 1)) then
set date = `date +%C%y.%m.%d.%H:%M`
set lastd = `expr ${lastd} + 1`
awk '{print >"ECMWF.result"} END {printf("%s : There has \
been an error, no traitment for day %d, 0OUT\n","'$date'",\
"'$lastd + 1'") > "ECMWF.result"}' ECMWF.result
rm day0.txt
cat > day0.txt <<EOD
${lastd}
EOD
endif
#
```

```
cp ECMWF.result /diskb/asmpredi/Verif
```

The *scrWeb4* script is quite similar to the first one. It looks in a file *day0.txt* (or *day4.txt* for the 12Z product) for the last day which has been processed. If the current day is superior and if there is a complete set of files for this day, it processes the set of files : it uses the utility *ECMextrDAT* to extract the information from the files, and then calls the *GrADS* utility with the name of scripts designed to perform meteorological treatments. Thoses scripts produce '.meta' files which we process with the utility *gxgif* designed to convert meta files to gif files. Then we can remove all the temporary files, and write in a file (*ECMWF.result*) if there has been an error or not.

7.1.3 The verification scripts (step1, step2, step3, getlastmeteo, step4.gp)

These are scripts dedicated to the forecasts verification. They are on the olasg machine, in the directory */diskb/asmpredi/Verif* and their aim is to produce some data files for gnuplot. This utility is used to draw the plots used in the Web pages, under the link 'forecasts verification'.

The first script, *step1*, is used to call an utility, *getlastmeteo* designed to go into the archeso database and collect the corresponding real measured data to compare with the forecasts of ECMWF. The listing of *getlastmeteo* follows :

```
getlastmeteo

#! /bin/tcsh -f
# modified  by M.Albrecht 19/10/98
cd /diskb/asmpredi/Verif/

if ( $1 == "" ) then
echo "Usage: getlastmeteo <db-password> hour site [date]"
echo "If no date is supplied, today is assumed"
exit
endif
set hour = $2
set pswd = $1

set site = $3
if ( $site != "lasilla" && $site != "paranal" ) then
echo "Wrong site: must be either 'lasilla' or 'paranal'"
echo "Usage: getlastmeteo <db-password> hour site [date]"
exit
endif

if ($4 == "") then
    set date = `date -u +%C%y.%m.%d`
else
    set date = $4
endif

set datehour = $date.$hour
set hoursec=`~flowmgr/bin/cvdate -${datehour}`
set hourstart = `expr ${hoursec} - 600`
set hourend = `expr ${hoursec} + 600`

isql -Uwww -P$pswd << EOF > temp
use ambient
go
select avg(t3)/10. from meteo_$site
  where (start_date between (-${hourend}) and (-${hourstart}))
  and (t3 != -32768)
select avg(pr)/10. from meteo_$site
```

```
  where (start_date between (-${hourend}) and (-${hourstart}))
  and (pr != -32768)
select avg(ws3)/10. from meteo_$site
  where (start_date between (-${hourend}) and (-${hourstart}))
  and (ws3 != -32768)
go
EOF
# If seeing is also wanted uncomment the lines below
#isql -Uwww -P$pswd << EOF >> temp
#use ambient
#go
#select avg(fwhm)/100. from seeing_$site
#  where (start_date between (-${hourend}) and (-${hourstart}))
#EOF

set t1 = `awk 'NR == 3 {print}' temp`
if ($t1 != "NULL") then
    awk '{print > "te.log"} END {if ('$hour' < 10) printf("%s     0%d     %6.1f     %6.1f     \
%6.1f     %6.1f\n","'$date'",'$hour','$t1',999,999,999) > "te.log"; else printf("%s     %d     \
%6.1f     %6.1f     %6.1f     %6.1f\n","'$date'",'$hour','$t1',999,999,999) > "te.log"}' te.log
else
    awk '{print > "te.log"} END {if ('$hour' < 10) printf("%s     0%d     %6.1f     %6.1f     \
%6.1f     %6.1f\n","'$date'",'$hour',999,999,999,999) > "te.log"; else printf("%s     %d     \
%6.1f     %6.1f     %6.1f     %6.1f\n","'$date'",'$hour',999,999,999,999) > "te.log"}' te.log
endif

set pr = `awk 'NR == 8 {print}' temp`
if ($pr != "NULL") then
    awk '{print > "pr.log"} END {if ('$hour' < 10) printf("%s     0%d     %6.1f     %6.1f     \
%6.1f     %6.1f\n","'$date'",'$hour','$pr',999,999,999) > "pr.log"; else printf("%s     %d     \
%6.1f     %6.1f     %6.1f     %6.1f\n","'$date'",'$hour','$pr',999,999,999) > "pr.log"}' pr.log
else
    awk '{print > "pr.log"} END {if ('$hour' < 10) printf("%s     0%d     %6.1f     %6.1f     \
%6.1f     %6.1f\n","'$date'",'$hour',999,999,999,999) > "pr.log"; else printf("%s     %d     \
%6.1f     %6.1f     %6.1f     %6.1f\n","'$date'",'$hour',999,999,999,999) > "pr.log"}' pr.log
endif

set ws = `awk 'NR == 13 {print}' temp`
if ($ws != "NULL") then
    awk '{print > "ws.log"} END {if ('$hour' < 10) printf("%s     0%d     %6.1f     %6.1f     \
%6.1f     %6.1f\n","'$date'",'$hour','$ws',999,999,999) > "ws.log"; else printf("%s     %d     \
%6.1f     %6.1f     %6.1f     %6.1f\n","'$date'",'$hour','$ws',999,999,999) > "ws.log"}' ws.log
else
    awk '{print > "ws.log"} END {if ('$hour' < 10) printf("%s     0%d     %6.1f     %6.1f     \
%6.1f     %6.1f\n","'$date'",'$hour',999,999,999,999) > "ws.log"; else printf("%s     %d     \
%6.1f     %6.1f     %6.1f     %6.1f\n","'$date'",'$hour',999,999,999,999) > "ws.log"}' ws.log
endif

rm temp
```

It goes into the archeso database to extract the data corresponding to the 00UT, 06UT, 12UT and 18UT measurments of the day and append them in the three log files (*pr.log*, *te.log*, *ws.log* and their backup) containing the data. In a first step, only the first row of the log file is filled with the measured data, the others are filled with '999.'.

The script *getlastmeteo* is called by another script, *step1*, for which the listing follows :

```
step1

#!/bin/tcsh
# This shell script performs the first step in building
# the data file for Kalman-Verif.
# It uses the script getlastmeteo to build the files
# pr.log, temp.log and ws.log
#
```

```
cd /diskb/asmpredi/Verif/
#
#mv te.log temp.te
#tail -n 100 temp.te > te.log
#mv pr.log temp.pr
#tail -n 100 temp.pr > pr.log
#mv ws.log temp.ws
#tail -n 100 temp.ws > ws.log
#
if ( $1 == "-help" || $1 == "-h") then
echo "Usage: step1 [date]"
echo "If no date is supplied, today is assumed"
echo "Date must be written yyyy.mm.dd"
exit
endif
#
set date = $1
getlastmeteo ####### 0 lasilla $date
getlastmeteo ####### 6 lasilla $date
getlastmeteo ####### 12 lasilla $date
getlastmeteo ####### 18 lasilla $date
```

The script *step1* is located on the olasg machine, in the directory */diskb/asmpredi/Verif.* It is executed once per day, at 21h local time (MET DST). The crontab line follows:

```
00 21 * * * /diskb/asmpredi/Verif/LaSilla/step1 >/dev/null 2>&1
```

One must notice that the three log files are not limited in size to allow for long term drifts detection.

After creating the log files with the *step1* script, we need to complete them with the analysis and the predictions by ECMWF. Those jobs are done by a second script, *step2*, which complete the log files. The listing of this script follows :

```
step2

#!/bin/tcsh
#
# This script will add the second part of the .log files,
# i.e. the analysis and the predictions of ECMWF.
# It uses the user defined function cuspLaSi (for La Silla)
# to extract "exact" analysis and predictions for the site
# (It performs a bilinear interpolation on lat lon and a
# cubic spline interpolation on the altitude).
#
cd /diskb/asmpredi/Verif
#
# This part looks for the last set of files from ECMWF
#
ls -al /home2/asmpredi/ECMWF/data/data00UT/ > essai
grep ESD20 essai > essai2
rm essai
# we search the last day of transfer (the last file in the list)
@ d = `tail -5c essai2`
set d = `expr $d / 10`
rm essai2
if ($d < 10) set d = 0$d
if ($d < 100) set d = 0$d
cp /home2/asmpredi/ECMWF/data/data00UT/ESD??${d}0 .
ECMextrDAT ${d} 0
rm ESD??${d}0
#
# We look for the day before in oder to be able to add the predictions
#
set echo
set dbef = `expr $d - 1`
set year = `date -u +%C%y`
if ($dbef == 0) then
if (`expr $year % 4` == 0 && (`expr $year % 100` != 0 || `expr $year % 1000` == 0)) then
set dbef = 366
else
set dbef = 365
endif
endif
if ($dbef < 10) set dbef = 0$dbef
if ($dbef < 100) set dbef = 0$dbef
cp /home2/asmpredi/ECMWF/data/data00UT/ESD??${dbef}0 .
ECMextrDAT ${dbef} 0
rm ESD??${dbef}0
#
# Now we have the files, we can process them to produce the log files
#
grads -lb -c 'run verif0.gs '${d}' 0'
grads -lb -c 'run verif0.gs '${dbef}' 0'
rm ESD*
#
# Second passage for the predictions of 12 hour
#
ls -al /home2/asmpredi/ECMWF/data/data12UT/ > essai
grep ESD36 essai > essai2
rm essai
# we search the last day of transfer (the last file in the list)
@ d = `tail -5c essai2`
set dd = `expr $d - 4`
set d = `expr $dd / 10`
rm essai2
if ($d < 10) set d = 0$d
if ($d < 100) set d = 0$d
cp /home2/asmpredi/ECMWF/data/data12UT/ESD??${d}4 .
ECMextrDAT ${d} 4
rm ESD??${d}4
#
# We look for the day before in oder to be able to add the predictions
#
set echo
set dbef = `expr $d - 1`
set year = `date -u +%C%y`
if ($dbef == 0) then
if (`expr $year % 4` == 0 && (`expr $year % 100` != 0 || `expr $year % 1000` == 0)) then
set dbef = 366
else
set dbef = 365
endif
endif
if ($dbef < 10) set dbef = 0$dbef
if ($dbef < 100) set dbef = 0$dbef
cp /home2/asmpredi/ECMWF/data/data12UT/ESD??${dbef}4 .
ECMextrDAT ${dbef} 4
rm ESD??${dbef}4
#
# Now we have the files, we can process them to produce the log files
#
grads -lb -c 'run verif0.gs '${d}' 4'
grads -lb -c 'run verif0.gs '${dbef}' 4'
cp te.log te.log.bak
cp pr.log pr.log.bak
cp ws.log ws.log.bak
#
# Call for the third part of the script
#
/diskb/asmpredi/Verif/step3
#
# Third party script to create the data file for the java pages
#
rm data.txt
```

```
grads -lb -c 'run writetxt.gs '${d}' 4'
/diskb/asmpredi/Verif/data
#
rm ESD*
/home2/sofi/tools/bin/gnuplot ./step4.gp
```

In *step2*, a call is made with the current and the previous day to fill the column of the analysis (of the current day) and of predictions (of the day before for the current day). *step2* calls the *ECMextrDAT* utility with the correct sets of files, and processes them through *verif0.gs*, to extract the correct information.

The third step is to add the kalman corrected forecasts in the three log files. This is the task of the script *step3*. The listing follows :

```
step3

#!/bin/sh
#
# Part 3 of the verification script, designed to
# compute the kalman results for the verification
# phase.
#
# First part : file te.log
#
while read -r date hour real ana predi kal
do
    if [ $kal -eq 999. ] && [ $predi -ne 999. ]
    then
    cat << EOD > temp
$predi
EOD
    `kalman te < temp > temp2`
    kal=`tail -8c temp2`
    rm temp
    rm temp2
fi
if [ $hour -lt 10 ]
then
    printf "%s     0%d     %6.1f     %6.1f     %6.1f     %6.1f\n" $date $hour $real $ana $predi $kal
else
    printf "%s     %d     %6.1f     %6.1f     %6.1f     %6.1f\n" $date $hour $real $ana $predi $kal
fi
done < te.log > temp3
mv temp3 te.log
#
# Second part : file pr.log
#
while read -r date hour real ana predi kal
do
    if [ $kal -eq 999. ] && [ $predi -ne 999. ]
    then
    cat << EOD > temp
$predi
EOD
    `kalman pr < temp > temp2`
    kal=`tail -8c temp2`
    rm temp
    rm temp2
fi
if [ $hour -lt 10 ]
then
    printf "%s     0%d     %6.1f     %6.1f     %6.1f     %6.1f\n" $date $hour $real $ana $predi $kal
else
    printf "%s     %d     %6.1f     %6.1f     %6.1f     %6.1f\n" $date $hour $real $ana $predi $kal
fi
```

```
done < pr.log > temp3
mv temp3 pr.log
#
# Third part : file ws.log
#
while read -r date hour real ana predi kal
do
    if [ $kal -eq 999. ] && [ $predi -ne 999. ]
    then
    cat << EOD > temp
$predi
EOD
    `kalman ws < temp > temp2`
    kal=`tail -8c temp2`
    rm temp
    rm temp2
fi
if [ $hour -lt 10 ]
then
    printf "%s     0%d     %6.1f     %6.1f     %6.1f     %6.1f\n" $date $hour $real $ana $predi $kal
else
    printf "%s     %d     %6.1f     %6.1f     %6.1f     %6.1f\n" $date $hour $real $ana $predi $kal
fi
done < ws.log > temp3
mv temp3 ws.log
```

This script calls the *kalman* utility to correct the forecast. It reads the second data column of the log file (Pr.log, te.log or ws.log) corresponding to the type of data to correct and extracts the value of the forecast, then writes the Kalman corrected value in the last column. It does this for the three parameters, hence the three parts in the script.

7.1.4 The publication scripts (publish0,publish4)

To end the tour of the ECMWF scripts, we must cite the publication scripts, located on the web4 machine, which download the image files from olasg, merge them to produce some 'animated gifs' and then copy those files on the web machine (web1 and web2) with the utility *webcp*. There are two scripts on web4, *publish0* and *publish4* for the 00UT and the 12UT products.

We print here only one of the two scripts, the other is quite similar to this one.

```
publish0

 #!/bin/tcsh
# this script is designed to produce animated gifs from gif files
# and to publish both of them on the web for the 12UT predictions
cd /disks/diskwa/webdocs/esonew/gen-fac/pubs/astclim/
cd ./forecast/meteo/ECMWF/short/images
#
set echo
ftp -n olasg.hq.eso.org << EOD
user asmpredi xxxxxxx
bin
prompt
cd /home2/asmpredi/ECMWF/data/tmp/
mget temp?.gif
mdelete temp?.gif
mget temp?_zoom1.gif
mdelete temp?_zoom1.gif
mget temp?_zoom2.gif
mdelete temp?_zoom2.gif
mget gh?.gif
```

```
mdelete gh?.gif
mget gh?_zoom1.gif
mdelete gh?_zoom1.gif
mget gh?_zoom2.gif
mdelete gh?_zoom2.gif
mget wind?.gif
mdelete wind?.gif
mget wind?_zoom1.gif
mdelete wind?_zoom1.gif
mget wind?_zoom2.gif
mdelete wind?_zoom2.gif
mget *_modwin.gif
mdelete *_modwin.gif
mget *_vecwin.gif
mdelete *_vecwin.gif
mget *_temp.gif
mdelete *_temp.gif
mget *_gh.gif
mdelete *_gh.gif
quit
EOD
#
rm temp.gif temp_zoom1.gif temp_zoom2.gif gh.gif gh_zoom1.gif gh_zoom2.gif wind.gif wind_zoom1.gif wind_zoom2.gif
../../../scripts/gifmerge temp?.gif > temp.gif
../../../scripts/gifmerge temp?_zoom1.gif > temp_zoom1.gif
../../../scripts/gifmerge temp?_zoom2.gif > temp_zoom2.gif
../../../scripts/gifmerge gh?.gif > gh.gif
../../../scripts/gifmerge gh?_zoom1.gif > gh_zoom1.gif
../../../scripts/gifmerge gh?_zoom2.gif > gh_zoom2.gif
../../../scripts/gifmerge wind?.gif > wind.gif
../../../scripts/gifmerge wind?_zoom1.gif > wind_zoom1.gif
../../../scripts/gifmerge wind?_zoom2.gif > wind_zoom2.gif
/usr/server/bin/webcp *.gif
```

Another script used for the publication is the *verif* script located on the web4 machine. It is used to download all the images and data files (*lasi.txt*, and later *para.txt* and *chaj.txt*) to the Web pages. The listing is not included because this script only performs ftp.

7.2 Scripts related to CIRA products

7.2.1 The satellite scripts (satellite, goesGetSat)

The *satellite* script is located on the olasg machine, in the directory

/home2/asmpredi/CIRA/utils/. It goes on tanager (a CIRA machine - one must notice that the password has been disabled in the listing !) and looks for the last image. If there is on tanager a more recent image than the one we have on olasg, it downloads it and performs some treatments on it with the help of the *IDL* utility. The *IDL* scripts produces two images, *clouds.gif* and *zoom.gif*. The aim of the web pages being to show eight images of each type, we create a sequence by shifting at each execution the sequence number of the image.

The listing follows :

```
satellite

#!/bin/tcsh
#
# This script is designed to get the last image from
# the goes8 satellite and to publish it on the web
```

```
#
cd /home2/asmpredi/CIRA/utils
#
# We check if there is a new file to transfert
#
cd /home2/asmpredi/CIRA/utils
# we check if there is currently a transfer running
set currFile = `tail -n +1 current`
if ($currFile != "") goto end
# No transfer running
# We check if there is a new file to transfert
#
rm temp
ftp -n tanager.cira.colostate.edu << EOD > temp
user eso goes8
cd andre
bin
prompt
ls
quit
EOD
#
# we look for the last .c04 file
#
rm temp2
grep "c04" temp > temp2
set temp = `tail -c 55 temp2`
rm tempString
cat > tempString <<EOD
${temp}
EOD
set size = `cut -c1-8 tempString`
set filegz = `tail -c 26 tempString`
set lastFile = `tail -n +1 last`
set currFile = `tail -n +1 current`
#
if ( $filegz != $lastFile && $filegz != $currFile) then
#
rm current
cat > current <<EOD
${filegz}
EOD
set file = `cut -c1-22 current`
#
# we go and get the last file
#
ftp -n tanager.cira.colostate.edu << EOD
user eso goes8
cd andre
bin
prompt
get ${filegz}
quit
EOD
#
# we check if the file was transfered correctly (sizenew = sizeold)?
#
rm temp
ls -al ${filegz} > temp
set sizeNew = `cut -c36-44 temp`
if ($size == $sizeNew) then
#
/vlt/gnu/bin/gunzip ${filegz}
mv ${file} temp.c04
#
set date = `date '+%C%y.%m.%d.%H:%M'`
awk '{print > "satellite.report"} END {printf("%s : succesfull loading of %s\n","'$date'","'$filegz'") >
 "satellite.report"}' satellite.report
rm last
cat > last <<EOD
```

```
${filegz}
EOD
rm current
touch current
# now we run the idl script to transform this image
# in a gif file and to add coasts and points for
# the three sites
#
rm namefile
cat > namefile << EOD
${file}
EOD
idl satellite.pro
#
mv clouds2.gif clouds1.gif
mv clouds3.gif clouds2.gif
mv clouds4.gif clouds3.gif
mv clouds5.gif clouds4.gif
mv clouds6.gif clouds5.gif
mv clouds7.gif clouds6.gif
mv clouds8.gif clouds7.gif
mv clouds.gif clouds8.gif
#
mv zoom2.gif zoom1.gif
mv zoom3.gif zoom2.gif
mv zoom4.gif zoom3.gif
mv zoom5.gif zoom4.gif
mv zoom6.gif zoom5.gif
mv zoom7.gif zoom6.gif
mv zoom8.gif zoom7.gif
mv zoom.gif zoom8.gif
#
set date = `date '+%C%y.%m.%d.%H:%M'`
#
awk '{print > "satellite.report"} END {printf("%s : succesfull processing of %s\n","'$date'","'$filegz'") > "satellite.report"}' sa
else
#downloading was aborted by ftp timeout
rm ${filegz}
rm current
touch current
set date = `date '+%C%y.%m.%d %H:%M'`
awk '{print > "satellite.report"} END {printf("%s : ftp abort of  %s\n","'$date'","'$filegz'") > "satellite.report"}' satellite.rep
#
endif
else
# no new file posted on tanager
endif
#
#mv satellite.report satellite.temp
#tail -n 50 satellite.temp > satellite.report
cp satellite.report /diskb/asmpredi/Verif
#
end:
```

Due to the unreliability of the internet link between ESO and CIRA, we had to implement some functions to verify, for example, that the size of the uploaded file was correct. Sometimes the transmission is so slow that the ftp gets a timeout before transmitting the whole file. So we need to check if the size of the uploaded file is correct. If not, we write a message in a log file and try again 10 minutes later. This script runs every 10 minutes. If the file has not been successfully transfered before the next image is posted on tanager, there is no recovery procedure currently implemented.

The log file (*satellite.report*) is located on the olasg machine, in the directory */diskb/asmpredi/Verif*.

7.2.2 The goesGetSat script

This script is located on the web4 machine, goes to olag to get the full size binned images (and the corresponding unbinned zoom images) and publish them on the web.

The listing follows:

```
goesGetSat

#!/bin/tcsh
#
# This script goes on olasg to get the satellite images and the
# corresponding animated gifs.
#
cd /disks/diskwa/webdocs/esonew/gen-fac/pubs/astclim/forecast/meteo/CIRA/imag
es
#
ftp -n olasg.hq.eso.org <<EOD
user asmpredi xxxxxxx
bin
cd /home2/asmpredi/CIRA/utils/
prompt
mget clouds*.gif
mget zoom*.gif
quit
EOD
#
cp clouds8.gif clouds.gif
/usr/server/bin/webcp clouds*.gif
/usr/server/bin/webcp zoom*.gif
```

8 The GrADS applications and utilities

8.1 The front-end utility (mainMeteo.gs)

This utility is located on the olasg machine in the

/home2/asmpredi/ECMWF/data/tmp/ directory. As a final product, it does not require a lot of maintenance. It must be in the same directory as a copy of the *ECMextrDAT* utility. It must have access to the data files from ECMWF, too. In fact at the startup of the script, it goes to the directory *../data00UT* or *../data12UT* and copies the data files to the current directory. So if the utility is moved, one must update the paths at the beginning of the script.

Another problem that may appear is an incompatibility with a future version of *GrADS*. Like all the *GrADS* tools and scripts, it has been developped with the version 1.7 beta 7 and tested succesfully with the version 1.7 beta 9. So if a new version of *GrADS* is installed , one must check the backward compatibility with all the tools developped.

8.2 The 'images' scripts (script*.gs)

There are three scripts used to produce the meteorological predictions images for the web, one for each region of interest. The first, *script1.gs* deals with the complete area: the South Pacific and the Chilean coast. The second, *script2.gs*, deals with the region above Paranal and Chajnantor. The third, *script3.gs*, deals with the region above La Silla.

For each region, the script produces the lat-lon maps for the temperature at 700 mb, the wind streamlines at 200 mb and the geopotential height at 700 mb. The second and third scripts produce the reconstructed radio profiles above the sites of interest. Here follows the listing of the third script, the first is the same without the sites profiles, and the second has one more site for the profiles.

If one wants to produces new graphs for other levels, one just needs to change the 'set lev xxx' lines for the new levels.

```
script3.gs

function main(args)

  'reinit'

   _day = subwrd(args,1)
   _hour = subwrd(args,2)
   _type = 0
   if (hour != 0)
      _type = subwrd(args,3)
   endif
   file_ctl='ESD'_day''_hour'_'_type*3+3'.ctl'

  'open 'file_ctl
  'set display color white'
  'set mpdset hires'
  'c'
  'set grads off'
  'set clab forced'
  'set parea 0.6 9.4 0.75 7.5'
  'set gxout shaded'
  'set csmooth on'
*** Drawing of the 7 or 17 images in the general region
```

```
if (_hour = 0 | (_hour = 4 & _type = 0))
   _nbFiles = 7
else
   _nbFiles = 17
endif
'set lev 700'
i = 1
while (i <= _nbFiles)
   'c'
   if (_hour = 0 | (_hour = 4 & _type = 0))
      'enable print temp'i'_zoom2.meta'
   else
      'enable print temp'i'bis_zoom2.meta'
   endif
   'set rbrange 275 290'
   'set grads off'
   'set clab forced'
   'set parea 0.6 9.4 0.75 7.5'
   'set gxout shaded'
   'set csmooth on'
   'set t 'i
   'd temp'
   'run cbarn.gs'
   'set gxout contour'
   'd temp'
   'set gxout shaded'
   drawSta(1)
   'query dims'
   str = sublin(result,5)
   date = subwrd(str,6)
   subtitl = 'Temperature (K at 700mb - 'date')'
   'draw title 'subtitl
   i = i + 1
   'print'
   'disable print'
endwhile

i = 1
while (i <= _nbFiles)
   'c'
   if (_hour = 0 | (_hour = 4 & _type = 0))
      'enable print gh'i'_zoom2.meta'
   else
      'enable print gh'i'bis_zoom2.meta'
   endif
   'set rbrange 3050 3200'
   'set grads off'
   'set clab forced'
   'set parea 0.6 9.4 0.75 7.5'
   'set gxout shaded'
   'set csmooth on'
   'set t 'i
   'd gh'
   'run cbarn.gs'
   'set gxout contour'
   'd gh'
   'set gxout shaded'
   drawSta(1)
   'query dims'
   str = sublin(result,5)
   date = subwrd(str,6)
   subtitl = 'Geopotential height (m at 700mb - 'date')'
   'draw title 'subtitl
   i = i + 1
   'print'
   'disable print'
endwhile

i = 1
while (i <= _nbFiles)
```

```
      'c'
      if (_hour = 0 | (_hour = 4 & _type = 0))
         'enable print wind'i'_zoom2.meta'
      else
         'enable print wind'i'bis_zoom2.meta'
      endif
      'set rbrange 0 80'
      'set lev 200'
      'set grads off'
      'set clab forced'
      'set parea 0.6 9.4 0.75 7.5'
      'set gxout stream'
      'set csmooth on'
      'set t 'i
      'd u;v;mag(u,v)'
      'run cbarn.gs'
      drawSta(1)
      'query dims'
      str = sublin(result,5)
      date = subwrd(str,6)
      subtitl = 'Wind colored by mag. (m/s at 200mb - 'date')'
      'draw title 'subtitl
      i = i + 1
      'print'
      'disable print'
   endwhile

*** drawing of the curves time x mb
*** these functions use the bilinear interpolation
*** implemented in fortran (dir $(GADDIR)/udf))$
   'c'
   'set t 1 '_nbFiles
   'set lev 100 1000'
   'set grads off'

   'set lon -71'
   'set lat -30'
   'temp1 = temp'
   'gh1 = gh'
   'u1 = u'
   'v1 = v'
   'set lon -70.5'
   'temp2 = temp'
   'gh2 = gh'
   'u2 = u'
   'v2 = v'
   'set lat -29.5'
   'temp3 = temp'
   'gh3 = gh'
   'u3 = u'
   'v3 = v'
   'set lon -71'
   'temp4 = temp'
   'gh4 = gh'
   'u4 = u'
   'v4 = v'

   'set yflip on'
   'set zlog on'
   if (_hour = 0 | (_hour = 4 & _type = 0))
      'enable print lasil_temp.meta'
   else
      'enable print lasil_temp2.meta'
   endif
   'd biliLaSi(temp1,temp2,temp3,temp4)'
   'draw title Temperature (K) above La Silla (772 mB)'
   'run cbarn.gs'
   'set gxout contour'
   'd biliLaSi(temp1,temp2,temp3,temp4)'
   'print'
   'disable print'

   'c'
   'set t 1 '_nbFiles
   'set lev 100 1000'
   'set grads off'
   'set yflip on'
   'set zlog on'
   if (_hour = 0 | (_hour = 4 & _type = 0))
      'enable print lasil_gh.meta'
   else
      'enable print lasil_gh2.meta'
   endif
   'set gxout shaded'
   'd biliLaSi(gh1,gh2,gh3,gh4)'
   'run cbarn.gs'
   'draw title Geopotential Height (m) above La Silla (772 mB)'
   'set gxout contour'
   'd biliLaSi(gh1,gh2,gh3,gh4)'
   'print'
   'disable print'

   'c'
   'set t 1 '_nbFiles
   'set lev 100 1000'
   'set grads off'
   'set yflip on'
   'set zlog on'
   if (_hour = 0 | (_hour = 4 & _type = 0))
      'enable print lasil_modwin.meta'
   else
      'enable print lasil_modwin2.meta'
   endif
   'set gxout shaded'
   'd mag(biliLaSi(u1,u2,u3,u4),biliLaSi(v1,v2,v3,v4))'
   'run cbarn.gs'
   'draw title Magnitude of wind (m/s) above La Silla (772 mB)'
   'set gxout contour'
   'd mag(biliLaSi(u1,u2,u3,u4),biliLaSi(v1,v2,v3,v4))'
   'print'
   'disable print'

   'c'
   'set t 1 '_nbFiles
   'set lev 100 1000'
   'set grads off'
   'set yflip on'
   'set zlog on'
   if (_hour = 0 | (_hour = 4 & _type = 0))
      'enable print lasil_vecwin.meta'
   else
      'enable print lasil_vecwin2.meta'
   endif
   'set gxout vector'
   'd biliLaSi(u1,u2,u3,u4);biliLaSi(v1,v2,v3,v4)'
   'draw title Evolution of wind above La Silla (772 mB)'
   'print'
   'disable print'

   'quit'


***
***
***
function drawSta(state)
   'set line 1 1 1'
   'set strsiz 0.1 0.1'
   'set string 1 bl 3'
   'q w2xy -70.'42*100/60' -29.'16*100/60
   x = subwrd(result,3)
   y = subwrd(result,6)
   'draw mark 3 'x' 'y' .1'
   y = y + 0.1
   x = x- 0.5
   'draw string 'x' 'y' La Silla'
   'q w2xy -70.'24*100/60' -24.'37*100/60
   x = subwrd(result,3)
   y = subwrd(result,6)
   'draw mark 3 'x' 'y' .1'
   y = y + 0.1
   x = x - 0.5
   'draw string 'x' 'y' Paranal'
   'q w2xy -67.'45*100/60' -23.'1*100/60
   x = subwrd(result,3)
   y = subwrd(result,6)
   'draw mark 3 'x' 'y' .1'
   y = y + 0.1
   x = x - 0.5
   'draw string 'x' 'y' Chajnantor'
   if (state = 0)
     'query w2xy -71.5 -26.5'
     xLo = subwrd(result,3)
     yLo = subwrd(result,6)
     'query w2xy -67 -22'
     xHi = subwrd(result,3)
     yHi = subwrd(result,6)
     'draw rec 'xLo' 'yLo' 'xHi' 'yHi
     'query w2xy -71.5 -31.5'
     xLo = subwrd(result,3)
     yLo = subwrd(result,6)
     'query w2xy -67 -27'
     xHi = subwrd(result,3)
     yHi = subwrd(result,6)
     'draw rec 'xLo' 'yLo' 'xHi' 'yHi
   endif
```

8.3 The verification scripts (verif0.gs, writetxt.gs)

Two *GrADS* scripts are dedicated to the forecasts verification. The first, *verif0.gs*, is called by the UNIX-script *step2* (see section 7.1.3). It deals with the site of La Silla, as the local data for Paranal and Chajnantor are not available in real time mode from Garching. It calls some user defined FORTRAN functions (*biliLaSi* and *cuspLaSi*), which will be described later in this maintenance manual.

The script is quite simple, it uses the four points nearest to the La Silla site to perform a bilinear interpolation for the latitude and longitude, and it uses a cubic spline interpolation for the altitude of the site.

When the local data for Paranal will be available, one just needs to update the script *getlastmeteo* in the directory */diskb/asmpredi/Verif/Paranal* and to add to the crontab file on olasg the same lines than for La Silla.

The listing of *verif0.gs* follows :

```
verif0.gs

function main(args)

  'reinit'
   _day = subwrd(args,1)
   _hour = subwrd(args,2)
   _type = 0
   file_ctl='ESD'_day''_hour'_3.ctl'

  'open 'file_ctl
  'set t 1 7'
  'set lev 100 1000'
  'set lon -70.5'
  'set lat -29'
  'temp1 = temp'
  'gh1 = gh'
  'u1 = u'
  'v1 = v'
  'set lat -29.5'
  'temp2 = temp'
  'gh2 = gh'
  'u2 = u'
  'v2 = v'
  'set lon -70'
  'temp3 = temp'
  'gh3 = gh'
  'u3 = u'
  'v3 = v'
  'set lat -29'
  'temp4 = temp'
  'gh4 = gh'
  'u4 = u'
  'v4 = v'

  'd cuspLaSi(biliLaSi(gh1,gh2,gh3,gh4),temp,biliLaSi(temp1,temp2,temp3,temp4))'
  'd cuspLaSi(biliLaSi(gh1,gh2,gh3,gh4),ws,\
     mag(biliLaSi(u1,u2,u3,u4),biliLaSi(v1,v2,v3,v4)))'

  'quit'
```

The second script, *writetxt.gs*, is dedicated to the java applet located in the verification pages on the Web. It produces a data file in a format recognized by the java applet. For now, this applet only deals with the La Silla site. It calls the *writPred* function, which is a FORTRAN user defined function for *GrADS*.

In fact, as we will see later, it is a version slightly modified of the cubic spline interpolation program. The *GrADS* script itself is quite simple and similar to the first one, it only uses the four point nearing La Silla and calls the *writPred* function.

The listing follows :

```
writetxt.gs

function main(args)

  'reinit'
   _day = subwrd(args,1)
   _hour = subwrd(args,2)
   _type = 4
   file_ctl='ESD'_day''_hour'_3.ctl'

  'open 'file_ctl
  'set t 1 7'
  'set lev 100 1000'
  'set lon -70.5'
  'set lat -29'
  'temp1 = temp'
  'gh1 = gh'
  'u1 = u'
```

```
 'v1 = v'
 'set lat -29.5'
 'temp2 = temp'
 'gh2 = gh'
 'u2 = u'
 'v2 = v'
 'set lon -70'
 'temp3 = temp'
 'gh3 = gh'
 'u3 = u'
 'v3 = v'
 'set lat -29'
 'temp4 = temp'
 'gh4 = gh'
 'u4 = u'
 'v4 = v'

 'd writPred(biliLaSi(gh1,gh2,gh3,gh4),biliLaSi(temp1,temp2,temp3,temp4),\
     mag(biliLaSi(u1,u2,u3,u4),biliLaSi(v1,v2,v3,v4)))'
 'close 1'
file_ctl='ESD'_day''_hour'_6.ctl'

 'open 'file_ctl
 'set t 1 17'
 'set lev 100 1000'
 'set lon -70.5'
 'set lat -29'
 'temp1 = temp'
 'gh1 = gh'
 'u1 = u'
 'v1 = v'
 'set lat -29.5'
 'temp2 = temp'
 'gh2 = gh'
 'u2 = u'
 'v2 = v'
 'set lon -70'
 'temp3 = temp'
 'gh3 = gh'
 'u3 = u'
 'v3 = v'
 'set lat -29'
 'temp4 = temp'
 'gh4 = gh'
 'u4 = u'
 'v4 = v'

 'd writPred(biliLaSi(gh1,gh2,gh3,gh4),biliLaSi(temp1,temp2,temp3,temp4),\
     mag(biliLaSi(u1,u2,u3,u4),biliLaSi(v1,v2,v3,v4)))'
 'quit'
```

9 Maintenance of the Fortran programs

9.1 The ECMextrDAT utility

This utility was coded in Fortran in order to decode the GRIB92 format used by ECMWF to disseminate its products [ECMWF 97]. The source code is located on olasg, in the */home2/asmpredi/ECMWF/software* directory. In this directory must be located the two librairies (*pbiolib.a* and *gridlib.a*) needed by *ECMextrDAT* for the compilation, the makefile and the source file *ECMextrDAT.f.* The source file being quite long, it will not be printed in the manual. Here is the makefile for this utility :

```
#
#                             configuration file for hp
#
CC       = cc
CFLAGS   = -g -Aa -D_INCLUDE_HPUX_SOURCE -D_INCLUDE_XOPEN_SOURCE \
-D_INCLUDE_AES_SOURCE -D_INCLUDE_POSIX_SOURCE -D_
XPG2 -Dhp
FC       = f77
FFLAGS   = -g +U77 -v -WF,"-P"  +z -Dhp
RANLIB   =
CT = /bin/true
RANLIB = /bin/true

.SUFFIXES:     .f

.f.o:
	$(FC) $(FFLAGS) -c $<

GRIBLIB = griblib.a
PBIOLIB = pbiolib.a

PROGS = ECMextrDAT
PROGS_OBJ = ECMextrDAT.o

all     : $(PROGS)

$(PROGS) : $(PROGS_OBJ)
	$(FC) $(FFLAGS) $@.o $(GRIBLIB) $(PBIOLIB) -o $@

clean :
	rm -f $(SOURCES)
```

The aim of this utility is to produce some files understandable by the *GrADS* utility. Then it must produce binary files without headers for the data, and text files (the ctl files) to inform *GrADS* about the content of those binary files. To produce the binary files, *ECMextrDAT* must read all the files from ECMWF composing a set of prediction files, and sort the information before recording it in new binary files. The challenge is that the files from ECMWF are sorted by time, and not in geographical region as they should be. Another problem comes from the fact that the products are not ordered in a logical manner. Sometimes the file uses the sequence 'general-paranal-la silla' and on another occasion is 'general-la silla-paranal'.

So the *ECMextrDAT* utility looks for patterns (in fact, those patterns are the coordinates of the origin point of the grid, the type of data and the level of forecast) in the headers to identify the geographical regions, the time and the level. It fills an array with the index of the records. It calls then a procedure to sort out this array, to order the records. It then reloads all the records and produces the three files needed by *GrADS.* The sensitive point is the pattern that the program is looking for in the ECMWF headers. If the patterns change (ECMWF could decide to use another coding), one must re-enter the new patterns in the program for it to work properly. If it is not the case the program will produce corrupted files for *GrADS*.

9.2 The user-defined functions for GrADS

9.2.1 The bilinear interpolation (biliLaSi, biliPara, biliChaj)

The first user-defined function (UDF) for the GrADS utility is a small UDF which performs bilinear interpolation in latitude and longitude for a given point. The present version of *GrADS* does not allow to pass to an UDF the number of arguments we want. For instance we could not pass the the coordinates of the four points of the grid nearing the considered point. For this reason, we had to develop three versions of this utility, one for each point of interest. In the directory */diskb/asmpredi/GrADS/udf*, one could find three utility for the interpolation, *biliPara*, *biliLaSi* and *biliChaj*.

Those UDF are quite simple to understand. They take from a temporary file written by GrADS the data concerning the four points, and perform the bilinear interpolation and write the result in a temporary file read by *GrADS*. The listing of the *biliLaSi* utility is given here :

```
      PROGRAM BiliLaSi
c
c
c
      integer nbTime
      real vals(20),ovals(20)
      real dummy(9)
      real x1(153),x2(153),x3(153),x4(153)
      real result(153)
c
c
c
      open (8,file = '/diskb/asmpredi/GrADS/udf/biliLaSi.in',
     x     form = 'unformatted')
c
      read (8)
      read (8) vals
c
      nbTime = vals(5)
c
      read (8) (x1(i),i=1,9*nbTime)
      read(8) dummy
c
      read (8)
      read (8) vals
      read (8) (x2(i),i=1,9*nbTime)
      read(8) dummy
c
      read (8)
      read (8) vals
      read (8) (x3(i),i=1,9*nbTime)
      read(8) dummy
c
      read (8)
      read (8) vals
      read (8) (x4(i),i=1,9*nbTime)
      read(8) dummy
c
      do 5 i = 1,9*nbTime
         result(i) = interL(x1(i),x2(i),x3(i),x4(i))
 5    continue
c
      ovals(1) = 0.0
      open (10,file = '/diskb/asmpredi/GrADS/udf/biliLaSi.out',
     x     form = 'unformatted')
      write(10) ovals
      write(10) vals
      write(10) (result(i),i=1,9*nbTime)
      write(10) dummy
      close (8)
      close (10)
      end
c
c
c
      subroutine interL(val1,val2,val3,val4)
c
      real val1,val2,val3,val4
c
      real t,u,result
c
      t = (-70.7 + 71) / (-70.5 + 71)
      u = (-29.7 + 30) / (-29.5 + 30)
c
      result = (1 - t) * (1 - u) * val1
     x     + t * (1 - u) * val2
     x     + t * u * val3
     x     + (1 - t) * u * val4
c
      return result
      end
```

To modify this script for a fourth site, one must notice that the only things to change are the coordinates of the nearest points in the grid, and the coordinates of the site.

9.2.2 The cubic spline interpolation (cuspLaSi, cuspPara and cuspChaj)

Those UDF which are again three, one for each site, perform a double cublic spline interpolation on the altitude in order to extract the best approximation of the considered parameter at the site ground level. The problem is that we do not have the parameter as a function of the altitude but standard level. The solution was then to call a first time the cubic spline procedure on the geopotential height parameter (gives a correlation between the altitude and the pressure) at the altitude of the site, which delivers the pressure value, and then to call again the cubic spline interpolation at this pressure value on the requested parameter to extract the best approximation of this parameter.

9.2.3 The writPred and writPre2 functions

This user defined function, located on olasg, is used to produce one of the data file for the java applet in the verification pages. It is a *GrADS* user function because it needs to have access to the ECMWF predictions. In fact it opens the predictions given by *GrADS*, it processes them through a Kalman filter and then writes a report file (*lasi.txt* or *Para.txt*) in the directories */diskb/asmpredi/Verif/LaSilla* or */diskb/asmpredi/Verif/Paranal.*

The second function, *writPre2*, is the same as the first but deals with the Paranal site.

The sources are in the directory */diskb/asmpredi/GrADS/udf.*

10 Maintenance of other programs

10.1 The IDL programs

The main IDL program being used is the one which deals with the satellite images, *TryRead.pro*. It is located on the olasg machine, in the directory */home2/asmpredi/CIRA/utils*. It is launched by the script *satellite* which calls the *IDL* initialization file, *satellite.pro*, and which launches the main program with the correct filename.

The crontab line for this script follows:

```
00 * * * * /bin/tcsh /home2/asmpredi/CIRA/utils/satellite >/dev/null 2>&1
```

Basically, it is based on the function *Gview.pro* developped by CIRA. We have taken the core program of the function (the reading and the decoding of a satellite image file), and added some functionalities, for example writing the name of the three sites or the date and the hour at which the image was taken.

This application was developped using the version 5.1 of *IDL*, installed on olasg. This installation hass a single user license.

10.2 The Java applets

Two applet have been developped for the Web astro-climatology site. The first one deals with the satellite images, and allows the user to browse through the last 8 images. The java definition file is named *Satellite.java*, the binary version being *Satellite.class*. This applet needs 16 image files to function properly (8 files *clouds*x*.gif* and 8 files *zoom*x*.gif*). In order to produce the animation loop while being able to interact with the user interface, we needed to use a special thread for the animation, and that is the only difficulty of the applet.

The second applet is behind the link verification pages, and allows the user to extract the corrected forecasts for the next 8 days. It reads on the server (web1 and web2 in this case) three URL's corresponding to the three data files for the applet (*lasi.txt*, *para.txt* and *chaj.txt*). If the files are not present, the applet will throw an exception but will continue to execute properly (it is due to the fact that the ground data for Paranal and Chajnantor are not available). The source file for this applet is *Forecast.java*, the binary file being *Forecast.class*.

Those two applets were developed with the JDK 1.0, to ensure their compatibility with most of the recent browsers (i.e. Netscape 3.0 or later and Internet Explorer 3.0 or later). One could port them to JDK 1.1 in the future, but as they are only basical applets there is no obligation to do so.

References

[CRS4 96] R. Benzi,R. Deidda, M. Marrocu, A. Speranza, Feasibility study of a meteorological prediction model for ESO observatories in Chile (Phases A1,A2), 1996 *ESO contract number 45060/VLT/95/6952/GWI*

[CRS4 97] R. Benzi,R. Deidda, M. Marrocu, A. Speranza, Feasibility study of a meteorological prediction model for ESO observatories in Chile (Phases A3,A4), 1997 *ESO contract number 45060/VLT/95/6952/GWI*

[ECMWF 95] ECMWF, User Guide to ECMWF Products 2.1, 1995

[ECMWF 97] D. Jokic, The dissemination of ECMWF products to Member States, 1997

[Press et al. 94] W. Press, S. Teulkosky, W. Vetterling, B. Flannery, Numerical Recipes in FORTRAN (second edition), 1994

A Appendix

A.1 Location of the Unix-scripts and their related files

This table summarizes the location of the UNIX-scripts and their related files, and shows the use of each of the file. One should notice that each script file uses some temporary files (temp$x$) which are not mentionned here because of their temporary existence.

File	Machine	Location	Use
ECMtoOlasg	opus3 (account asmpredi)	/home/asmecmwf/	transfers complete set of ECMWF files (product of 12UT)
ECMtoOlasg2	opus3 (account asmpredi)	/home/asmecmwf/	transfers complete set of ECMWF files (product of 00UT)
scrWeb0	olasg	/home2/asmpredi/ECMWF/data/tmp	process set of 00UT files from ECMWF to produce images for the Web pages
scrWeb4	olasg	/home2/asmpredi/ECMWF/data/tmp	process set of 12UT files from ECMWF to produce images for the Web pages
day0.txt	olasg	/home2/asmpredi/ECMWF/data/tmp	contains the last day for which the 00UT product has been processed
day4.txt	olasg	/home2/asmpredi/ECMWF/data/tmp	contains the last day for which the 12UT product has been processed
ECMWF.result	olasg	/diskb/asmpredi/Verif	'log' file for the treatment of the ECMWF files
satellite	olasg	/home2/asmpredi/CIRA/utils	upload the most recent satellite image from CIRA and process it through an IDL script
satellite.report	olasg	/diskb/asmpredi/Verif	'log' file for the satellite script
last	olasg	/home2/asmpredi/CIRA/utils	text file containing the name of the last uploaded image
current	olasg	/home2/asmpredi/CIRA/utils	text file containing the name of the current transfer, if there is any
getlastmeteo	olasg	/diskb/asmpredi/Verif/LaSilla	script to extract some data from the base archeso
step1	olasg	/diskb/asmpredi/Verif	first part in the verification process (measures)
step2	olasg	/diskb/asmpredi/Verif	second part in the verification process (forecasts)
step3	olasg	/diskb/asmpredi/Verif	third part in the verification process (kalman corrected forecasts)

File	Machine	Location	Use
te.log and te.log.bak	olasg	/diskb/asmpredi/Verif	log file (and his backup) for the temperature
input.te	olasg	/diskb/asmpredi/Verif	input file (temperature) for gnuplot
pr.log and pr.log.bak	olasg	/diskb/asmpredi/Verif	log file (and his backup) for the pressure
input.pr	olasg	/diskb/asmpredi/Verif	input file (pressure) for gnuplot
ws.log and ws.log.bak	olasg	/diskb/asmpredi/Verif	log file (and his backup) for the wind speed
input.ws	olasg	/diskb/asmpredi/Verif	input file (wind speed) for gnuplot
publish0	web4	/disks/diskwa/webdocs/esonew/gen-fac/ pubs/astclim/forecast/meteo/scripts	uploads the images from olasg to the Web pages (product of 12UT)
publish4	web4	/disks/diskwa/webdocs/esonew/gen-fac/ pubs/astclim/forecast/meteo/scripts	uploads the images from olasg to the Web pages (product of 00UT)
verif	web4	/disks/diskwa/webdocs/esonew/gen-fac/ pubs/astclim/forecast/meteo/scripts	uploads the verification images to the Web pages
data.txt	olasg	/diskb/asmpredi/Verif	text file containing some information for the java applet on the verification pages
goesGetSat	web4	/disks/diskwa/webdocs/esonew/gen-fac/ pubs/astclim/forecast/meteo/scripts	uploads the satellite images and publish them to the web
clouds$x$.gif	olasg	/home2/asmpredi/CIRA/utils	post-processed satellite images
zoom$x$.gif	olasg	/home2/asmpredi/CIRA/utils	corresponding zoom of the post-processed satellite images

A.2 Location of the GrADS applications and their related files

In this table are presented the name of the GrADS scripts and utilities, and their dependent files.

File	Machine	Location	Use
mainMeteo.gs	olasg	/home2/asmpredi/ECMWF/data/tmp	Front-end utility for manipulating the data files from ECMWF
cbarn.gs	olasg	/home2/asmpredi/ECMWF/data/tmp	Utility to draw the colour scale next to the GrADS graphics
usr_colors.gs and usr_sta.gs	olasg	/home2/asmpredi/ECMWF/data/tmp	GrADS data files. The first contains the user defined colors, the second the coordinates of the stations to draw on the maps
par_draw.5.gs and par_bar.5.gs	olasg	/home2/asmpredi/ECMWF/data/tmp	GrADS data files. They contain the parameters and the buttons to draw in the front-end utility
script1.gs	olasg	/home2/asmpredi/ECMWF/data/tmp	called by scrWeb*x*. Produces the images for the generic region
script2.gs	olasg	/home2/asmpredi/ECMWF/data/tmp	called by scrWeb*x*. Produces the images for the Paranal region
script3.gs	olasg	/home2/asmpredi/ECMWF/data/tmp	called by scrWeb*x*. Produces the images for the La Silla region
temp*x*.gif and temp*x*bis.gif	olasg	/home2/asmpredi/ECMWF/data/tmp	gif of the temperature for the generic region - short term and long term
gh*x*.gif and gh*x*bis.gif	olasg	/home2/asmpredi/ECMWF/data/tmp	gif of the geopotential height for the generic region - short term and long term
wind*x*.gif and wind*x*bis.gif	olasg	/home2/asmpredi/ECMWF/data/tmp	gif of the wind streamlines for the generic region - short term and long term
temp*x*_zoom1.gif and temp*x*bis_zoom1.gif	olasg	/home2/asmpredi/ECMWF/data/tmp	gif of the temperature for the Paranal region - short term and long term
gh*x*_zoom1.gif and gh*x*bis_zoom1.gif	olasg	/home2/asmpredi/ECMWF/data/tmp	gif of the geopotential height for the Paranal region - short term and long term
wind*x*_zoom1.gif and wind*x*bis_zoom1.gif	olasg	/home2/asmpredi/ECMWF/data/tmp	gif of the wind streamlines for the Paranal region - short term and long term
temp*x*_zoom2.gif and temp*x*bis_zoom2.gif	olasg	/home2/asmpredi/ECMWF/data/tmp	gif of the temperature for the La Silla region - short term and long term
gh*x*_zoom2.gif and gh*x*bis_zoom2.gif	olasg	/home2/asmpredi/ECMWF/data/tmp	gif of the geopotential height for the La Silla region - short term and long term

File	Machine	Location	Use
wind*x*_zoom2.gif and wind*x*bis_zoom2.gif	olasg	/home2/asmpredi/ECMWF/data/tmp	gif of the wind streamlines for the La Silla region - short term and long term
*xxxxx*_temp.gif and *xxxxx*_temp2.gif	olasg	/home2/asmpredi/ECMWF/data/tmp	gif of the reconstructed radio profile of temperature for site xxxxx (chajn, lasil or paran)- short term and long term
*xxxxx*_gh.gif and *xxxxx*_gh2.gif	olasg	/home2/asmpredi/ECMWF/data/tmp	gif of the reconstructed radio profile of geopotential height for site xxxxx (chajn, lasil or paran)- short term and long term
*xxxxx*_vecwin.gif and *xxxxx*_vecwin2.gif	olasg	/home2/asmpredi/ECMWF/data/tmp	gif of the reconstructed radio profile of the directions of the wind vectors for site xxxxx (chajn, lasil or paran)- short term and long term
*xxxxx*_modwin.gif and *xxxxx*_modwin2.gif	olasg	/home2/asmpredi/ECMWF/data/tmp	gif of the reconstructed radio profile of wind speed for site xxxxx (chajn, lasil or paran)- short term and long term
verif0.gs	olasg	/diskb/asmpredi/Verif	updates the three log files for the verification. Called by the 'step2' script
writetxt.gs	olasg	/diskb/asmpredi/Verif	writes the text file for the java applet in the verification pages

A.3 Location of the Fortran programs

In this table are presented the fortran programs used by the utilities.

File	Machine	Location	Use
ECMextrDAT	olasg	/home2/asmpredi/ECMWF/software	utility to decode the files from ECMWF
ECMextrDAT.f	olasg	/home2/asmpredi/ECMWF/software	source file of the ECMextrDAT utility
griblib.a and pbiolib.a	olasg	/home2/asmpredi/ECMWF/software	librairies needed to compile the ECMextrDAT utility
biliLaSi (biliLaSi.f)	olasg	/diskb/asmpredi/GrADS/udf	bilinear interpolation for the La Silla site
biliPara (biliPara.f)	olasg	/diskb/asmpredi/GrADS/udf	bilinear interpolation for the Paranal site
biliChaj (biliChaj.f)	olasg	/diskb/asmpredi/GrADS/udf	bilinear interpolation for the Chajnantor site
cuspLaSi (cuspLaSi.f)	olasg	/diskb/asmpredi/GrADS/udf	cubic spline interpolation for the La Silla site
cuspPara (cuspPara.f)	olasg	/diskb/asmpredi/GrADS/udf	cubic spline interpolation for the Paranalsite
cuspChaj (cuspChaj.f)	olasg	/diskb/asmpredi/GrADS/udf	cubic spline interpolation for the Chajnantor site
writPred (writPred.f)	olasg	/diskb/asmpredi/GrADS/udf	kalman filtering of the predictions for the java verification applet for the La Silla site
writPre2 (writPre2.f)	olasg	/diskb/asmpredi/GrADS/udf	kalman filtering of the predictions for the java verification applet for the Paranal site

A.4 Location of the IDL applications and their related files

In this table are presented the IDL applications and their related files.

File	Machine	Location	Use
satellite.pro	olasg	/home2/asmpredi/CIRA/utils	header file used to execute the main IDL application and to set some global variables
TryRead.pro	olasg	/home2/asmpredi/CIRA/utils	main IDL application to read and produce the gif files of the satellite images
namefile	olasg	/home2/asmpredi/CIRA/utils	text file containing the name of the satellite image being processed by IDL
current	olasg	/home2/asmpredi/CIRA/utils	text file containing the name of the satellite image file being downloaded fron CIRA (present only during a download)
last	olasg	/home2/asmpredi/CIRA/utils	text file containing the name of the last succesfully transferred satellite image file
clouds$x$.gif	olasg	/home2/asmpredi/CIRA/utils	processed image file of the general region, $x \in [1..8]$
zoom$x$.gif	olasg	/home2/asmpredi/CIRA/utils	processed image file of the zoomed region, $x \in [1..8]$

A.5 Location of the java applets and their related files

In this table are presented the java applets and their related files.

File	Machine	Location	Use
Sat_anim.java	web4	/disks/diskwa/webdocs/esonew/gen-fac/pubs/astclim/forecast/meteo/CIRA	source file of the satellite applet
Sat_anim.class	web4	/disks/diskwa/webdocs/esonew/gen-fac/pubs/astclim/forecast/meteo/CIRA	binary file of the satellite applet
Forecasts.java	web4	/disks/diskwa/webdocs/esonew/gen-fac/pubs/astclim/forecast/meteo/verification	source file of the verification applet
Forecasts.class	web4	/disks/diskwa/webdocs/esonew/gen-fac/pubs/astclim/forecast/meteo/verification	binary file of the verification applet
clouds$x$.gif	web4	/disks/diskwa/webdocs/esonew/gen-fac/pubs/astclim/forecast/meteo/CIRA/images	processed image file of the general region, $x \in [1..8]$
zoom$x$.gif	web4	/disks/diskwa/webdocs/esonew/gen-fac/pubs/astclim/forecast/meteo/CIRA/images	processed image file of the zoomed region, $x \in [1..8]$
lasi.txt	web4	/disks/diskwa/webdocs/esonew/gen-fac/pubs/astclim/forecast/meteo/verification	text file for the verification applet, La Silla site
para.txt	web4	/disks/diskwa/webdocs/esonew/gen-fac/pubs/astclim/forecast/meteo/verification	text file for the verification applet, Paranal site
chaj.txt	web4	/disks/diskwa/webdocs/esonew/gen-fac/pubs/astclim/forecast/meteo/verification	text file for the verification applet, Chajnantor site

B List of crontab files

In this appendix are presented the crontab files present on the machines.

- Machine opus3, account asmpredi

```
00,20,40 * * * * /home/asmecmwf/ECMtoOlasg >/dev/null 2>&1
10,30,50 * * * * /home/asmecmwf/ECMtoOlasg2 >/dev/null 2>&1
```

- Machine olasg, account asmpredi

```
00,30 * * * * /home2/asmpredi/ECMWF/data/tmp/scrWeb0 >/dev/null 2>&1
15,45 * * * * /home2/asmpredi/ECMWF/data/tmp/scrWeb4 >/dev/null 2>&1
00 * * * * /bin/tcsh /home2/asmpredi/CIRA/utils/satellite >/dev/null 2>&1
00 21 * * * /diskb/asmpredi/Verif/LaSilla/step1 >/dev/null 2>&1
00 04,09 * * * /diskb/asmpredi/Verif/LaSilla/step2 >/dev/null 2>&1
```

- Machine web4, account msarazin

```
00 * * * * /disks/diskwa/webdocs/esonew/gen-fac/pubs/astclim/forecast\
/meteo/scripts/publish4 >/dev/null 2>&1
00 * * * * /disks/diskwa/webdocs/esonew/gen-fac/pubs/astclim/forecast\
/meteo/scripts/publish4 >/dev/null 2>&1
00 * * * * /disks/diskwa/webdocs/esonew/gen-fac/pubs/astclim/forecast\
/meteo/scripts/publish0 >/dev/null 2>&1
30 01,04,07,10,13,16,19,22 * * * /disks/diskwa/webdocs/esonew/gen-fac\
/pubs/astclim/forecast/meteo/scripts/goesGetSat >/dev/null 2>&1
00 05 * * * /disks/diskwa/webdocs/esonew/gen-fac/pubs/astclim/forecast\
/meteo/scripts/verif >/dev/null 2>&1
```