ESO - EUROPEAN SOUTHERN OBSERVATORY

EUROPEAN SOUTHERN OBSERVATORY

Organisation Européenne pour des Recherches Astronomiques dans l'Hémisphère Austral
Europäische Organisation für astronomische Forschung in der südlichen Hemisphäre

VERY LARGE TELESCOPE

Data Flow System
Installation Guide

VLT-SPE-ESO-19000-1781

Issue 21
2008/05/29
54 pages

Prepared: SEG

Name Date Signature

Approved: A. Longinotti

Name Date Signature

Released: M. Peron

Name Date Signature

CHANGE RECORD

Issue	Date	Affected Paragraphs(s)	Reason/Initiation/Remarks
15	2007/03/15	All	Only Scientific Linux 4.3 compliant.
16	2007/03/23	All	Layout corrections.
17	2007/04/30	Here and there	Update after March Commissioning
18	2007/07/01	Chapter 3	OlasWS support Time of Oportunity
19	2008/02/29	New Chapter 8	Adding AstoWS installation
20	2008/03/15	Chapter 6	New references to CPL, esorex
21	2008/05/29		New Role: Data Transfer

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Introduction

1.1 Scope

The document describes the complete installation of the Data Flow System (dfs-5_6) on any machine running any component of the DFS. The installation is applicable to all UT and VLTI installations of the DFS in Paranal, La Silla, VCM Garching and DFO Garching.

1.2 Overview

This guide describes in detail the installation of DFS products on Dataflow workstations. The installation of the DFS is a part of the DFS commissioning plan, which describes HW upgrade plans and time schedules, and is therefore not described in this document.

Furthermore the installation of the following products is not part of this guide:

- OS installation
- VLT Software

Section 2. of the document describes in general terms the DFS machines, their roles and software requirements prior to the installation of the DFS.

- OS requirements.
- VLT requirements
- Required third-party software
- Accounts to be made available on each machine as well as their configuration

Section 3 describes in detail the steps necessary for carrying out successfully a DFS compilation and deployment in the different machines.

Sections 4 to 8, describe final DFS installation and configuration on each machine.

Section 9 is a troubleshooting list for typical system administration tasks on DFS machines.

1.3 Applicable Documents

[1] Science Operations Plan, VLT-PLA-ESO-10000-0441

[2] Scientific Linux 4.3 Installation Manual, VLT-MAN-ESO-17200-2009

[3] DFS High Level User's Guide, VLT-SPE-ESO-19000-1780

[4] P2PP User's Manual, VLT-MAN-ESO-19200-1644

[5] OT User's Manual, VLT-MAN-ESO-19200-1777

[6] OLAS User's Guide, VLT-MAN-ESO-19400-1785

[7] OLAS Operator's Guide, VLT-MAN-ESO-19400-1557

[8] ASTO Operator's Guide, VLT-MAN-ESO-19400-1784

[9] Data Flow Pipeline and Quality Control User’s Manual, VLT-MAN-ESO-19500-1619

[10] FORS Pipeline and Quality Control User’s Manual, VLT-MAN-ESO-19500-1771

[11] ISAAC Pipeline and Quality Control User’s Manual, VLT-MAN-ESO-19500-1772

[12] UVES Pipeline and Quality Control User’s Manual, VLT-MAN-ESO-19500-2019

[13] GASGANO User’s Manual, VLT-PRO-ESO-19000-1932

[14] DFSLog User’s Manual, VLT-MAN-ESO-19000-1827

[15] Data Interface Control Document, GEN-SPE-ESO-19400-794

[16] NG/AMS User’s Manual, VLT-MAN-ESO-19400-2739

[17] NGAS Operations & Troubleshooting Guide, VLT-MAN-ESO-19400-3103

[18] NGAS Acceptance Test Plan & Hands-On Tutorial, VLT-PLA-ESO-19400-3100

1.4 Abbreviations and Acronyms

The following abbreviations and acronyms are used in this document as listed below:

ANSI	American National Standards Institute
ASCII	American Standard Code for Information Interchange
ASTO	Astronomy Storage System
BOB	Broker for Observation Blocks
CCS	Central Control Software
CLI	Command Line Interface
DO	Data Organiser
DFS	Data Flow System
DVD	Digital Versatile Disk
DMD	Data Management and Operations Division
ESO	European Southern Observatory
FITS	Flexible Image Transport System
FTP	File Transfer Protocol
GUI	Graphical User Interface
LTS	Long Term Schedule
NG/AMS	Next Generation/Archive Management System
NGAS	Next Generation Archive System
OB	Observation Block
OCA	Organisation, Classification, Association
OHS	Observation Handling System
OhsWS	Observation Handling System Workstation
OlasW	Online Archive System Workstation
OPC	Observing Programmes Committee
OT	Observing Tool
P2PP	Phase 2 Proposal Preparation Tool
PI	Principal Investigator
PipelineWS	Pipeline Workstation
RB	Reduction Block
RBS	Reduction Block Scheduler
SED	System Engineering Department
SEG	System Engineering Group
SDD	Software Development Division
SM	Service Mode

STS	Short Term Scheduler
TCP/IP	Transmission Control Protocol/Internet Protocol
UserWS	User or Visiting Astronomer Workstation
UT	Unit Telescope
VCM	VLT Control Model
VCS	VLT Control Software
VISTA	Visible Infrared Survey Telescope for Astronomy
VLT	Very Large Telescope
VLTI	Very Larget Telescope Interferometer
VM	Visitor Mode

1.5 Glossary

Astronomical Site Monitor (ASM): hardware and software system which collects observing conditions such as: air temperature, air pressure, humidity, wind speed, wind direction, seeing, sky brightness/emissivity, sky transparency, precipitable water vapour content, and dust content of ambient air. Five-minutes averages of the measurements obtained are logged and periodically transfered to the OLAS system via the VLT Control software in order to be archived into the Ambient database.

Acquisition Template (AT): an *Observation Block* object. An AT is used to specify how a target will be acquired by the telescope. It may also specify any preliminary instrument configuration steps (e.g. set rotator to specific angle). It can contain parameters for interactive as well as automatic acquisitions. This template may define a different instrument/detector configuration from the templates within the Observation Description. Each science OB contains at most one AT.

Archive Storage System (ASTO): Archive Storage System providing means for storing data onto a long-term archive media (CDs or DVDs).

BOB (Broker of Observation Blocks): VCS tool which receives OBs from the OHS applications (OT or P2PP). BOB accepts the incoming OB on the VCS side and begins execution.

Calibration Database: Database containing master calibration data.

Calibration OB: OB used to acquire calibration data. Such OB does not contain any AT.

Central Control Software (CCS): collection of libraries, servers and utilities. The libraries provide a set of services to be used for SW development in a CCS environment covering program-to-program communications, data storage and retrieval, access control to hardware and/or software resources and user interaction with the VLT system. Purpose of the servers is to provide special services on application level.

Condor: a specialized workload management system for compute-intensive jobs. Like other full-featured batch systems, Condor provides a job queueing mechanism, scheduling policy, priority scheme, resource monitoring, and resource management. Users submit their serial or parallel jobs to Condor, Condor places them into a queue, chooses when and where to run the jobs based upon a policy, carefully monitors their progress, and ultimately informs the user upon completion.

Constraint Set (CS): an *Observation Block* object. A CS lists observation conditions required for the OB execution (i.e. requirements for sky transparency, seeing, airmass, lunar illumination, and moon angular distance). Each Observation Block contains at most one CS.

DO: Data Organiser, tool of Pipeline which classifies and analyses the content of any incoming raw frame and creates the corresponding Reduction Block (RB), if appropriate. Assembles calibration frames and raw data to be processed following data reduction recipes (data reduction procedures) specified in a RB.

Exposure: a synonym for the acquisition of a single data frame, typically resulting in a single FITS file.

Instrument Package (IP): set of files containing the TSF and ISF files for a specific instrument. The correct IP must be installed before OBs can be created for that instrument.

Instrument Summary File (ISF): part of the IP, contains a summary of the P2PP addressable optical elements of that instrument.

Master calibration product: a reduced frame used for calibration of science and calibration raw frames.

Next Generation Archive Management System (NG/AMS): the NGAS software.

Next Generation Archive System (NGAS): archive facility with services for archiving files (with on-the-fly checking and processing), retrieving and on-the-fly processing of files, ensuring data consistency, managing data.

OB Repository: Database containing two kinds of Observation Blocks: (1) Service Mode Observations Blocks which are first submitted to ESO for review and scheduled via the OT tool, then for possible execution; (2) Visitor Mode Observation Blocks which are stored only when being submitted to the VLT Control Software for execution.

Observation: a coordinated sequence of telescope, instrument, and detector actions that results in a scientific or technical dataset.

Observation Block: Smallest observational unit within the Data Flow System. It contains a sequence of high-level operations, called 'templates' that need to be performed sequentially and without interruption in order to ensure the scientific usefulness of an observation. Observations Blocks may contain scheduling requirements. They are used both in *Visitor* and *Service* Mode to acquire data.

Observation Tool (OT): Tool used to create queues (sets) of *Observation Blocks* for later scheduling and possible execution.

Observing Run: an approved ESO programme consists of one or more Observing Runs, each of which specify an independent combination of telescope, instrument, and observing operations mode (i.e. *Service Mode* or *Visitor Mode*).

On-Line Archive System (OLAS): System responsible for receiving and distributing all data products generated by the VLT and by the on-line pipeline.

Phase 2 Proposal Preparation Tool (P2PP): Tool used to create and (in visitor mode) execute Observation Blocks.

Pipeline product: Result of the execution of a Reduction Block.

QC0: Quality Control level 0. On-Line tool that checks whether service mode OBs have been executed under the conditions specified by the astronomer. QC0 is executed on raw data.

QC1: Quality Control level 1. QC1 consists of quality checks on pipeline-processed data. The QC1 parameters are used to assess the quality of calibration products and the performance of the instrument.

Raw Frame: Result of OB execution by the VCS, i.e. immediate result of an exposure. Raw frames are delivered to the Science Archive and the Reduction Pipeline as FITS files. The headers (set of keywords) contain all info relevant for reduction, QC and archiving, in particular the identification of OB to which the exposure belongs. As they move through DFS, info is added to the headers (archiving information, seeing conditions, ...). They are stored in directories whose name has the format YYYY-MM-DD, where the date is that of the *night* to which the frame belongs, i.e. the noon preceding the exposure.

Reduction Block Scheduler (RBS): tool which schedules and executes RBs created and sent by the DO. RBS sends the RB to the DRS (MIDAS) which will actually perform the reduction.

Reduction pipeline: Subsystem of the DFS in charge of pipeline processing. Applies reduction recipes and its parameters (calibration frames) on raw frames to generate pipeline products.

Reduction recipe: standard procedure for reducing observational data in a standard way. Recipes are implemented for each of the instrument standard templates. Those scripts take as input raw frames and execute them in a particular Data Reduction System (DRS).

RTD: Acquired data is displayed via Quick-Look tools such as the Real-Time Display directly from the Instrument Control Systems.

Service Mode: observing operations mode where astronomer submits a detailed description of his/her observing programme to ESO for later possible execution. Service mode programmes are executed primarily in order of their OPC assigned priority but only when the astronomer specified observing conditions are achieved on-site.

Template: a high-level data acquisition operation. Templates provide means to group commonly used procedures into well-defined and standardized units. They can be used to specify a combination of detector, instrument, and telescope configurations and actions. Templates have input parameters described by a template signature, and produce results that can serve as input to other templates. As an example, an Acquisition Template takes target coordinates and produces through an interactive procedure the precise positions used later, e.g. to place an object on a slit.

Template Signature files (TSF): files which contain template input parameters used to create OBs.

VCS (VLT Control software): the software and hardware tools that are used to control directly VLT instruments, telescopes, and related hardware. It enables and performs the acquisition of scientific and technical data.

Visitor Mode: observing operations mode where the astronomer is present at the telescope when his/her observing programme is being executed.

2 DFS Installation

2.1 DFS machines and roles

- The Data Flow System (DFS) is a distributed system which runs in several machines and each of them playing a different role. There roles are:
- OlasWS, for Data Handling System role, e.g. machines with names 'wuNdhs'
- OhsWS, for the Observation Handling Ssystem role, generally played on machines with names 'wu*N*oh' (Since 2002, OHS processes are located on OlasWSs).
- PipelineWS, for Pipelines, e.g. machines with names 'wu*N*pl'
- UserWS, for the Visiting Astronomer, e.g. machines with names 'wgsoffN'
- AstoWS, for Archive machines, e.g like 'wgsarc' or 'wvgarc'
- InstWs for VLT Instrument Workstations, e.g. wtatcam, wisaac, wf1ors, w5tcs, etc..
- DataTransferWS, machines handling the Data Transfer to Garching, e.g. wgsdts, wvcdts
- NgasWS, NGAS archive machines, e.g. wgsnau1, wgsnau2, etc..
- SybaseWS, machines running the Sybase Database Server, e.g. wgsdbp, wu0ola, etc..

Telescope/Role	**Olas/ Ohs**	**Pipeline**	**User**	**Asto**	**Ngas Archive**	**Ngas Preimage**	**Sybase**	**DataTransfer**
UT0 (VCM Garching)	wu0dhs	wu0rpl	wgoff2	wg0arc	wg0ngas2 wgongas11	ng0ngas2 ng0ngas2	wu0ola	N/A
UT1 (Paranal)	wu1dhs	wu1pl	wgsoff1	wgsarc	wgsnau1 wgsnau2	ngau1 ngau2	wgsdbp	wgsdts
UT2 (Paranal)	wu2dhs	wu2pl	wgsoff2	wgsarc	wgsnau1 wgsnau2	ngau1 ngau2	wgsdbp	wgsdts
UT3 (Paranal)	wu3dhs	wu3pl	wgsoff3	wgsarc	wgsnau1 wgsnua2	ngau1 ngau2	wgsdbp	wgsdts
UT4 (Paranal)	wu4dhs	wu4pl	wgsoff4	wgsarc	wgsnau1 wgsnau2	ngau1 ngau2	wgsdbp	wgsdts
VLTI (Paranal)	wvgdhs	wvgpl	wvgoff	wvgarc	wgsnau1	ngau1 ngau2	wgsdbp	wgsdts
VISTA (Paranal)	wvcdhs	wvcpl	wvcoff	N/A	N/A	N/A	wgsdbp	wvcdts
NTT (La Silla)	wg5dhs	wg5pl	wg5off	wlsdbs	wlsnau1 wlsnau2	ngau1 ngau2	wlsdbp	wlsdts
2P2 (La Silla)	w2p2dhs	w2p2pl	w2p2off	wlsdbs	wlsnau1 wlsnau2	ngau1 ngau2	wlsdbp	wlsdts
3P6 (La Silla)	w3p6dhs	w3p6pl	w3p6off	wlsdbs	wlsnau1 wlsnau2	ngau1 ngau2	wlsdbp	wlsdts
DFO (Garching)	N/A	dfo21-33 qc01-20	N/A	N/A	ngau1 ngau2	ngau1 ngau2	acdb	pastage_vip

1) All NGAS preimages machines are located in Garching.

2.2 HW installation

Most DFS machines are servers (rackable or desktop) with hot-swapable disks in RAID configuration. For Paranal the following apply before the installation of the OS:

1. Bios configuration. Interrupt the boot sequence with F2 (Entering Setup) and check that:
   - Boot sequence: CDROM, HD, Network
   - Check Data and Time is UTC
   - Disable Hyper-threading
   - Disable Report on keyboard errors
2. RAID configuration. Interrupt the boot sequence with Ctrl-Alt-R (Raid configuration) and check:
   - Alarm is activated
   - The two smallest disk (73GB) will be configured in RAID-1 (mirror). Its volumes will contain the filesystems for the OS and /diska.
   - On OlasWSs and PipelineWSs the four bigger disk (300GB) will be configure in RAID-5. Its complete volume will be later configured as one single partition and mounted as /diskb.
   - On UserWSs the four remaining disks will be configured in RAID-10 (stripped and mirror). Two filesystems will be created later as /diskb and /diskc.

2.3 OS/VLT installation on DFS machines

The OS/VLT installation on DFS machines will be done following the instructions of the document [2], by using the kickstart method, i.e. a file containing all the details of the OS installation (disk partitions, network, OS packages, configuration files, accounts, and post installation packages like VLT).

As summary of the installation, execute:

- Introduce the DVD labelled SL43`FEB2007 dated 15.03.2008
- Connect the machine to the network
- Connect monitor, mouse and keyboard(US101)
- Power-on or reboot the machine with Ctrl-Alt-Delete
- At this point the machine will reboot from the DVD, it will display the prompt *Boot,* and at this point enter:

  ```
  linux ks=cdrom:/isolinux/ks.cfg hostname=<fully_qualified_hostname>
  ip=<IP>    gateway=<IP> nameserver=<IP> ntpserver=<IP>  alias=<email_addr>
  ```

  Example:

  ```
  Boot> linux ks=cdrom:/isolinux/ks.cfg hostname=wu1dhs.pl.eso.org
  ip=134.171.225.5 gateway=134.171.225.254 ntpserver=ogstime.pl.eso.org
  alias=par-unix@eso.org
  ```

With this command the OS installation starts. It also installs the VLT software and it gets finished with the reboot of the machine and the ejection of the DVD.

After reboot and to complete the installation execute the following:

- Login as root (password in the header of the kickstart file)
- Introduce the DVD labelled SL43`FEB2007 dated 15.03.2008. The DVD will be mounted automatically in /media/cdrom
- If this is a new HW, creates the partitions from the data RAID (1xRAID5 on OlasWS and PipelineWS, and 2xRAID10 on UserWS). Be aware that the partitions might already exist if this is an update of an exisiting DFS machine. The partitioning is done identifying the devices (usually */dev/sdb* and */dev/sdc*) and with the command *fdisk.* Only one single partition is created with the complete volume available, i.e. cylinder=1 to cylinder=last)
- Creates ext3 filesystems in the new partitions:
  mkfs.ext3 /dev/sdb1
  mkfs.ext3 /dev/sdc1
- Create new entries in the /etc/fstab for the new filesystems
- Mount the new filesystems
- Recompile a new kernel with the execution of the script:
  /media/cdrom/vlt/InstallNewKerner.sh
- Reboot

2.4 DFS distribution

For the installation and distribution of the DFS, let's make a distinction between the DFS installation machine were DFS is compiled, linked and configured, and the target machines or machines were DFS, in binary format, is deployed. By doing in this way, the source code of the DFS is centrally maintained and can be easily reconfigured if it is necessary.

For the distribution of the DFS two targets are considered:

- Primary DFS machines: OlasWSs, OhsWSs, PipelineWSs, UserWSs and AstoWSs
- Secondary DFS machines: all InstrWSs

On primary targets the DFS is distributed complete while in secondary target only a reduced partial installation is deployed (The secondary DFS distribution is delivered to vltmgr who integrates it in the VLT software under /vlt/FEB2007/DFS).The installation machine is chosen from all the DFS machines (CPU speed and disk space could be arguments) and, of course, the installation machine could be a target machine as well. Best is to use a spare machine not being used in operations.

For binary distribution and monitoring is convenient that flowmgr account at all target machines trust account flowmgr from the installation machine (see .rhosts files).

Source code for DFS releases is delivered as compressed-tar (tar.gz) files, in the DVD DFS, or it can be directly retrieved from the archive repository

The DVD contains also other packages required during the DFS installation (*/opt*, */opsw /scisoft,*). DFS deliveries are installed with the 'flowmgr' account on the installation machine, other deliveries are installed by 'root'.

As a policy, the first delivered release of the DFS is the last beta release called "dfs-**M_m**beta**n**", where **M** and **m** represents the Major and minor release numbers, and **n** the latest beta release. As modifications to the source code of the original release is very likely to happen during the DFS commissioning, we start the DFS installation by making a copy of the delivery release and renaming it without the beta suffix. As an example, if the reference release is dfs-5_6beta7, the commissioning release would be dfs-5_6.

Once all VLT and external software is installed and configured on all DFS machines, the installation procedure compiles the DFS source code and generates binary deliveries.

DFS binary installations in target machines create directories under ~flowmgr as 'dfs-', e.g. ~flowmgr/dfs-5_6. In this way several DFS releases can be kept in the same machine. A symbolic link 'dfs' is created during installation and it is pointing to 'dfs5_0'. The usage of a symbolic link makes it easy to switch back to the previous working release.

After the DFS binary is delivered to a target machine, the DFS installation is completed on a machine by machine basis (chapter 4-9).

2.5 Disk space required

Directory	filesystem	Size (MB)
/opt	/	115
/home/flowmgr/dfs-5_6	/diska	153
/home/flowmgr/dfsSrc/dfs-5_6 [1]	/diska	856
/opsw/python [2]	/	67
/scisoft [3]	/diska	1876

2.6 DFS installation from DVD

The installation of the DFS with these scripts is necessary on all machines always after the previous OS/VLT installation. The DFS installation scripts are executed by root and they complete the OS/VLT installation with the creation of DFS accounts, external software, plus the installation of the latest DFS binary version.. When it finish the machine is fully operational. If an installation of DFS sources is necessary, e.g. a new release newer than the one in the DVD, apply the next chapter "Compiling DFS sources".

- Login as root
- Introduce the DVD labelled DFS dfs-5_6 dated 15.03.2008. The DVD will be mounted automatically in /media/cdrom
- If this is a new machine create the /data directories.
  mkdir /diskb/data
  ln –s /diskb/data /data
- Execute the corresponding script depending of the role of this machine:
  /media/cdrom/dfs/InstallDfsLinuxOlas.sh <UT>
  /media/cdrom/dfs/InstallDfsLinuxOhs.sh <UT>
  /media/cdrom/dfs/InstallDfsLinuxPipeline.sh <UT>
  /media/cdrom/dfs/InstallDfsLinuxUser.sh <UT>
  /media/cdrom/dfs/InstallDfsLinuxAsto.sh <UT>

[1] Only on DFS installation machines
[2] Only on PipelineWSs and UserWSs
[3] Only on UserWSs

Where <UT> is:
UT0 at VCM-Garching
UT1,UT2,UT3,UT4 and UT4(VLTI) at Paranal
UT5(3P6), UT6(NTT) and UT7(2P2) at La Silla
UT9 at DFO-Garching
UT10 (VISTA) at Paranal

- Reboot
- For each DFS account run:

```
dfscheck.sh
```

2.7 Compiling DFS sources

DFS must be installed by flowmgr and in a non-operational machine, in an installation directory

- Login as flowmgr in a non-operational account
- Create a location for the sources, e.g. /home/flowmgr/dfsSrc/dfs-5_6

```
mkdir -p ~dfsSrc/dfs-5_6
cd ~dfsSrc/dfs-5_6
```

- Set a valid CVSROOT environment

```
export CVSROOT=:pserver:user@cvssrv:/project1/CVS
```

  Where *user* is an allowed account in the CVS repository

- Create the $HOME/.cvspass if it does not exist

```
touch .cvspass
```

- Login and retrieve the module dataflow for a given DFS release

```
cvs login
cvs co –r dfs-5_6 dataflow
```

Set the compilation environment:

- INTROOT or final destination. For DFS installation INTROOT must be always /home/flowmgr/dfs. It is important to remove any previous /home/flowmgr/dfs (directory or soft link).

```
export INTROOT=/home/flowmgr/dfs-5_6
```

- CUSTOMER, or definition for OHS deliveries. For the DFS integration, the possible values are: SEG at VCM-Garching, PAR at Paranal and LAS at La SIlla. Example at Paranal:

```
export CUSTOMER=PAR
```

Launch the installation. Sources are compiled, linked and distributed to INTROOT with the script:

```
./dataflow/install.sh
```

The script will stop with any error. A log file is created under the /tmp/install-account.log. Check it to verify the complete installation got completed

Finally re-edit the $INTROOT/admin/dataflowUser.sh. There is a script to do this automatically, execute:

```
cd /home/flowmgr/dfs/admin
```

(or $INTROOT/admin)

```
./EditDataflowUser.sh <UT>
```

Where <UT> is:
UT0 at VCM-Garching

UT1,UT2,UT3,UT4 and UT4(VLTI) at Paranal
UT5(3P6), UT6(NTT) and UT7(2P2) at La Silla
UT9 at DFO-Garching

If this machine is to become operative, create the dfs soft link

```
cd /home/flowmgr
ln –s dfs-5_6 dfs
```

Distribute the binary DFS to target machines (make sure no DFS processes are running)

```
cd /home/flowmgr
tar cf - ./dfs-5_6 | rsh <target> tar xf –
rsh <target> "mv dfs dfs.old; ln –s dfs-5_6 dfs"
```

3 DFS installation on OlasWS

3.1 Preinstallation

- Mount the DVD "DFS dfs-5_6 dated 15.03.2008"
- The /data directory must exist (e.g. as a soft link to /diskb/data)

3.2 Installation

As root with the execution of script

```
cd /media/cdrom/dfs
./InstallDfsLinuxOlas.sh <UT>
```

Where <UT> is:
UT0 at VCM-Garching
UT1,UT2,UT3,UT4,UT5(VLTI)) at Paranal
UT6(NTT), UT7(2P2) and UT8(3P6) at La Silla
UT9 at DFO-Garching
UT10 (VISTA) at Paranal

The InstallDfsLinuxOlas.sh will just execute the following batch:

```
./InstallDfsLinux.sh install_ldsoconf
./InstallDfsLinux.sh install_groups   OlasWS
./InstallDfsLinux.sh install_accounts OlasWS
./InstallDfsLinux.sh install_pecsvltdfs
./InstallDfsLinux.sh install_opt
./InstallDfsLinux.sh install_netperf
./InstallDfsLinux.sh install_flowmgr
./InstallDfsLinux.sh install_dataflowUser OlasWS <UT>
./InstallDfsLinux.sh install_archeso OlasWS
./InstallDfsLinux.sh install_hostreport
```

More information about these options in the headers of these scripts.

3.3 Account archeso

3.3.1 The .bashrc file

The data transfer is done via 'rcp', however the 'rcp' command, as any other 'r' command like 'rlogin' or 'remsh', executes the file .bashrc before the file transfer could start. The execution of this start file may delay the file transfer or it may even refuse the transfer if the execution of .bashrc returns a non-zero value.

So in order to improve the 'rcp' transfer, check that the installation created a .bashrc with the entries:

```
if [ "$TERM" = "dumb" ]; then return; fi
```

3.3.2 The .rhost file

The .rhosts file must be modified to allow access from <inst_userf>@InstrWSs, archeso@UserWS, pipeline@PipelineWS, archeso@OlasWS and archeso@AstoWS. Example for UT1 the following lines must be added:

```
wtatcam  cam
wgsoff1  archeso
wu1pl    pipeline
wu1dhs   archeso
wgsarc   archeso
```

NOTE: Be aware that more than one UserWS (in the so called UT-less UserWS) may want to subscribe to this OlasWS. If so, you may need several archeso@UserWS in the .rhosts file, e.g:

```
wgosff1   archeso
wgsoff2   archeso
wgsoff3   archeso
```

NOTE: In wu1dhs, in addition to the above list, we have to include the account 'asm' from the Ambient Workstation `wasm`:

```
wasm   asm
```

3.3.3 The .dbrc file

Create file .dbrc in the $HOME directory with encrypted passwords:

```
<DBSERVER> observations frameingest <encrypted-password> DPREP
<DBSERVER> ambient frameingest <encrypted-password> AMBIREP
<DBSERVER> asto asto <encrypted-password> ASTOREP
```

where <*DBSERVER*> is ASTOP in Paranal and SEGSRV in the VCM-Garching. And where <*encrtyped password*> is the result of the command:

```
% ~flowmgr/dfs/bin/stcrypt <non-encrypted-password>
```

Another useful command is dbrcGet. Executed with a database server and a database name as arguments,reads the .dbrc file and display the username and password that can be used to access the database. Example:

```
% dbrcGet <DBSERVER> observations
frameingest <non-encrypted-password>
```

3.3.4 The $INS_ROOT environment

The default $INS_ROOT environment as defined by ~flowmgr/dfs/admin/dataflowUser.sh is pointing to ~flowmgr/dfs/data and it contains the dictionaries used by DFS applications.

However frameIngest will report many errors with incoming FITS files when using the standard dictionaries. It is for this reason that La Silla and Paranal prefer to avoid the dictionaries by creating a dummy location e.g.:

```
mkdir –p ~/INS_ROOT/SYSTEM/Dictionary
```

And over-write the standard environment INS_ROOT in the ~/.pecs/misc-all.env with the following entry:

```
export INS_ROOT=~/INS_ROOT
```

Exception to the rule: The OlasWS receiving meteorological PAF and ASM files (i.e. UT1 wu1dhs) requires the corresponding dictionaries for frameIngest to work properly. These are:

```
$INS_ROOT/SYSTEM/Dictionary/ESO-VLT-DIC.ASM
$INS_ROOT/SYSTEM/Dictionary/ESO-VLT-DIC.PAF
```

and can be copied from the DFS distribution, e.g.:

```
cd ~flowmgr/dfs/data/SYSTEM/Dictionary
cp *ASM *PAF $INS_ROOT/SYSTEM/Dictionary
```

3.3.5 Enabling ngamsIngest processes

NgamsIngest is a subscriber to DHS like frameIngest or dhsSubscribe.It is responsible of forwarding files from DHS to a NGAMS server. There are two possible NGAMS servers: ARCHIVE (local) and PREIMAGE (remote). The account acheso is responsible of enabling/disabling the transfer of files to the NGAMS servers, either ARCHIVE, REMOTE, or both, with the definitions of the environment variables:

```
export DFS_NGAMS_ARCHIVE=”YES”
export DFS_NGAMS_PREIMAGE=”YES”
```

If they are set to “YES” the process is enable otherwise disable. These environments can be set permanently in the PECS file:

```
/home/archeso/.pecs/misc-all.env
```

NgamsIngest is launched at boot time by the script:

```
/home/flowmgr/dfs/bin/olasDhsControl
```

But it could also started/stoped by archeso at any time with the commands

```
nmgamsIngestControl archive start
ngamsIngestControl archive stop
ngmasIngestControl preimage start
ngamsIngestControl preimage stop
```

Other environments, defined in /home/flowmgr/dfs/admin/dataflowUser.sh and used by ngamsIngest are:

```
DFS_NGAMS_SERVERS_ARCHIVE
DFS_DATA_NGAMS_ARCHIVE
DFS_BACKLOG_NGAMS_ARCHIVE
DFS_NGAMS_SERVERS_PREIMAGE
DFS_DATA_NGAMS_PREIMAGE
DFS_BACKLOG_NGAMS_PREIMAGE
```

The account archeso is also responsible for the creation of the directories defined by environments DFS_DATA_NGAMS_ and DFS_BACKLOG_NGAMS_ with protections 755.

3.3.6 Enabling Time of Oportunity (ToO)

To enable ToO, 'archeso' has to create the file:

```
~archeso/tooList.dat
```

with a list of two columns 'ProgramIDs' and 'directoriy' separated by newlines. The character '#' comments out any character to the right. An example of a tooList.dat could be:

```
# Created by CG 04.03.04
# List of PIDs to be sent to Garching
60.A-9022(A)    to1
61.A-9022(A)    to2
```

For changes of 'tooList.dat' to take effect, the user archeso will need to stop/start the dhsSubscribe process, with:

```
dhsSubscribeControl stop too
dhsSubscribeControl start too
```

'dhsSubscribe' will read the ~archeso/tooList.dat if such a file exits, is readable and not empty. It will execute the 'tooOrganiser' as the postcommand which organizes the files in subdirectories under $DFS_TOO_STAGE (i.e /data/too)

3.3.7 Other files

DFS does not specify which scripts should be executed with cronjobs. It is known, however , that the account archeso could be used to run scripts which clean regularly the /data directory and other housekeeping tasks. The frequency and times of these cronjobs are responsibility of the operator. These entries may need to be taken into account during upgrades and installations of DFS.

Also, directories bin and scripts in the archeso account are maintained by operators. To be taken into account in upgrades and installations of DFS.

4 DFS installation on OhsWS

4.1 Preinstallation

- Mount the DVD "DFS dfs-5_6 dated 15.03.2008"
- Kernel reinstalled with the script "CreateNewKernel.sh" as described in chapter 2.3. Check it with the output of command:

```
% uname –r
2-6-9-prep
```

4.2 Installation

As root with the execution of script

```
cd /media/cdrom/dfs
./InstallDfsLinuxOhs.sh <UT>
```

Where <UT> is:
UT0 at VCM-Garching
UT1,UT2,UT3,UT4,UT5(VLTI)) at Paranal
UT6(NTT), UT7(2P2) and UT8(3P6) at La Silla
UT9 at DFO-Garching
UT10 (VISTA) at Paranal

The InstallDfsLinuxOhs.sh will just execute the following batch:

```
./InstallDfsLinux.sh install_ldsoconf
./InstallDfsLinux.sh install_groups   OhsWS
./InstallDfsLinux.sh install_accounts OhsWS
./InstallDfsLinux.sh install_pecsvltdfs
./InstallDfsLinux.sh install_flowmgr
./InstallDfsLinux.sh install_dataflowUser OhsWS <UT>
./InstallDfsLinux.sh install_ccslite
./InstallDfsLinux.sh install_hostreport
```

More information about these options in the headers of these scripts.

4.3 Verifying the CCSLite environment

For 'p2pp' and 'ot' to connect via BOB with the InstrWS, one needs to have a CCSLite environment running in the OhsWS.

The installation of the CCSLite is done during the execution of the "InstallDfsLinuxOhs.sh" script, thus we will only describe how to test it.

Preconditions:

- $ACC_HOST is defined and points to the host holding the environment database (wg0acc in VCM-Garching and wgsops in Paranal). Typically it is defined by root in

/vlt/System/config/ 'hostname'.csh and/or /etc/pecs/releases/000/etc/locality/apps-'hostname'.env

- $RTAPENV is defined as the CCSLite environment running in the OLAS/OHS machine (by convention this is the name of the machine).

CCSLite environments are created with the same name of the hosts where they live. First they have to be registered in $ACC_HOST, the host where the databases are kept (wg0acc in VCM-Garching and wgsops in Paranal). At registration time the VLT manager assigns an environment name in a given hostname and with a given port number. Environment names are registered, by default, with the name of the hostname where they live. E.g. wu3dhs wu3dhs 2301/tcp

As root:
and after registration in $ACC_HOST, check the file /etc/services contains:

```
wu3dhs 2301/tcp
```

As vlt:
Check the following entry in the /vltdata/config/CcsEnvList file:

```
wu3dhs  wu3dhs
```

Check /vltdata/ENVIRONMENTS is 777 (writable to everybody) and execute:

```
% vccEnvStop   -e $RTAPENV
% rm -rf /vltdata/ENVIRONMENTS/'hostname'
% vccEnvCreate -e $RTAPENV
```

The command vccEnvCreate, creates a default /vltdata/ENVIRONMENTS/'hostname'/ccsCcsEnv

Table that requires to be modified in order to allow only one process to register itself as 'schedule'. This is done with the following command:

```
% ~flowmgr/dfs/admin/vccEnvDfsUpdate.sh -e $RTAPENV
```

Finally complete the initialization with:

```
% vccEnvInit   -e $RTAPENV
% vccEnvStart  -e $RTAPENV
```

Repeat the previous sequence whenever you change anything in the /etc/services or /vltdata/config/CcsEnvList files.

In successive reboots, the CCSLite environment should be started automatically at boot time, thanks to the ccsLite boot script, located in /sbin/init.d (HPs) or /etc/init.d (Linux)

NOTE: The account running the environment 'vlt' must exist, with the same name and UID/GID, in any remote machine (e.g. Instrument Workstations) that wants to comunicate with OhsWS.

And vice versa: in the OhsWS must exist the same accounts (same UID/GID) as those accounts in the InsWS running the CCS environment that communicates with OhsWS.

NOTE: In the eventuality of the environment being killed improperly, the file /vltdata/ENVIRONMENTS/wg0arc/.wu3dhs.lock may need to be deleted before you try to have this environment running again.

For verification, and as visitor, execute the following command:

```
% msgSend ‘hostname‘ msgServer PING ""
MESSAGEBUFFER:
OK
```

4.4 Accounts visitor, service and instmgr

All these three accounts require running with the VLT LITE mode, i.e. the environment variable VLTSW_CCSTYPE should be set to “lite”. By default this should be set in the file .pecs/apps-all.env:

```
% vi ~/.pecs/apps-all.env
 VLTSW_CCSTYPE=lite
```

The account ’visitor’ is meant for visiting astronomers, who need to handle visitor mode OBs via the command ’p2pp’.

The account ’service’ is meant for ESO astronomers, who prepare OB queues via the command ’ot’. It is also meant to run the ’masktracker’ tool (VIMOS only), and the ’fcviewer’ (viewer of Finding Charts).

Finally the account ’instmgr’ is meant for ESO astronomers in engineering mode, who have to verify the syntax/correctness of Template Signature Files (TSFs) by running ’p2pp’.

Previous to the execution of ’p2pp’ the configuration file .p2pp.cf is required in the HOME directory. Also for accounts ’visitor’, ’service’ and ’instrmgr’, save the previous .p2pp.cf if it exists:

```
% cd
% mv .p2pp.cf .p2pp.cf.<current DD.MM.YYYY>
```

Previous to the execution of ’ot’ the configuration file .ot.cf is required in the HOME directory. Also, for accounts ’service’ and ’instmgr’, save the previous .ot.cf if it exists:

```
% cd
% mv .ot.cf .ot.cf.<current DD.MM.YYYY>
```

The same for the ’masktracker’ under the ’service’ account (only for VIMOS-UT3):

```
% cd
% mv .mt.cf .mt.cf.<current DD.MM.YYYY>

% cd
% mv .ot.cf .ot.cf.<current DD.MM.YYYY>

% cd
% mv .ot.cf .ot.cf.<current DD.MM.YYYY>
```

Update the OHS local configuration files.
Under the 'visitor' account:

```
cp ~flowmgr/dfs/OHS/p2pp/config/VISITOR.p2pp.cf ~visitor/.p2pp.cf
```

Under the 'service' account:

```
cp ~flowmgr/dfs/OHS/p2pp/config/SERVICE.p2pp.cf ~service/.p2pp.cf
cp ~flowmgr/dfs/OHS/ot/config/SERVICE.ot.cf ~service/.ot.cf
and for VIMOS/wu3dhs only:
cp ~flowmgr/dfs/OHS/masktracker/config/SERVICE.mt.cf ~service/.mt.cf
```

Under the 'insmgr' account:

```
cp ~flowmgr/dfs/OHS/p2pp/config/INSTMGR.p2pp.cf ~instmgr/.p2pp.cf
cp ~flowmgr/dfs/OHS/ot/config/INSTMGR.ot.cf ~instmgr/.ot.cf
```

Check that following directories already exist (should be mostly the case).If not, manuallly create them under the related account (e.g. create ~visitor/impex, using the 'visitor' account, and so on):

```
~visitor/impex
~visitor/data
~visitor/dump
~visitor/p2pp-finding-charts

~service/impex
~service/data
~service/dump
~service/p2pp-finding-charts
~service/ot-options
~service/ot-finding-charts
~service/mt-options
~service/mt-finding-charts

~instmgr/impex
~instmgr/data
~instmgr/dump
~instmgr/p2pp-finding-charts
~instmgr/ot-options
~instmgr/ot-finding-charts
```

4.4.1 Final check with dfscheck.sh

Run the script

```
% dfscheck.sh
```

And correct all pending issues

4.5 Installation of instrument templates

Instrument specific products (e.g. templates) must be installed before P2PP and OT can be used.

Retrieve existing instrument packages from the previous DFS installation.

Please check with Paranal Science Operations before updating or removing existing template files in this directory.

Login as instmgr, and retrieve the Instrument packages applicable to the telescope:

Install instrument specific products:

```
% cd
% gunzip -c <instrument>.tar.gz | tar xvof -
```

A directory 'instruments' may be created, otherwise create it manually.

It is very important that each time a new Instrument package is added, the account 'instmgr' has to run 'p2pp' and 'ot' in engineering mode to verify the correctness of TSF and to update the summary.idx and other index files.

```
% cd
% p2pp &
% ot &
```

5 DFS installation on InstrWS

5.1 Introduction

InstrWS is the machine controlling the hardware of the instrument and where raw data files from the instrument are created. There are two interfaces to DFS in the InstrWS: one interface allows the instrument control software BOB to receive OBs from programs like P2PP or OT in the OhsWS. The other interface allows the instrument raw data to be transferred to the OlasWS. This chapter describes the installation and validation of these two interfaces. Please notice that both OhsWS and OlasWS could be located in the same machine (OlasWS/OhsWS).

The interface between BOB and P2PP/OT is made with the proper configuration of VLT environments CCS in the InstrWS to CCSLite in the OhsWS.

Files containing instrument raw data are sent from the InstWS to the OlasWS via an interface called VCSOLAC. The interface is created with the installation of any DFS release and delivered to the InstrWS as a compressed-tarred file called dfs5_0-secondary.tar.gz. The VCSOLAC delivery is installed by 'vltmgr' and executed by the <instrument_user>, an account that changes from InstrWS to InstrWS (e.g. isaac, vimos, etc...) and generally it is the same user that runs BOB.

5.2 Installation

As vltmgr:

```
$ mkdir $VLTSW_ROOT/DFS
$ cd $VLTSW_ROOT/DFS
$ zcat <DEPOT>/dfs/dfs-4_9-secondary.tar.gz | tar xvf -
$ ln -s dfs-<release> dfs
```

As root:

```
# cd sbin/init.d
# $VLTSW_ROOT/DFS/dfs/admin/installBoot.sh vcsolac <instrument_user>
```

where <*instrument_user*> is the account that will run VCSOLAC and BOB.

5.2.1 CCS environment

Two files must be modified to set-up correctly the CCS environment for allowing communication with the P2PP and OT processes running on the OhsWS.

As root, edit the file /etc/services and add:

```
wu<UT>dhs <port>/tcp
```

where *<UT>* is the Unit Telescope number and *<PORT>* is the same tcp port assigned in the wu*<UT>*dhs machine.

As vltmgr, add the following line to /etc/opt/rtap/A.06.70/RTapEnvList

```
wu<UT>dhs  wu<UT>dhs
```

NOTE: For CCSLite to communicate between the InstrWS and the OhsWS, accounts running the CCS or CCSLite environment have to exist with the same username in both machines.

5.2.2 VCSOLAC configuration

VCSOLAC requires the following environment variables to be defined.

If using TCSH then, in $HOME/config/<hostname>.cshrc as:

```
setenv OLAS_ID     <InstrTAG>
setenv PATH        ${VLTSW_ROOT}/DFS/dfs/bin:${PATH}
setenv MANPATH     ${VLTSW_ROOT}/DFS/dfs/man:${MANPATH}
setenv DHS_DATA    ${INS_ROOT}/${INS_USER}/ARCDATA
setenv DHS_LOG     ${DHS_DATA}
setenv BAD_DIR     ${DHS_DATA}
setenv DHS_HOST    <OlasWS>
setenv DHS_CONFIG  archeso@${DHS_HOST}:/data/msg
```

Or in BASH and PECS, the configuration file would be: $HOME/.pecs/misc-<hostname>.env

```
export OLAS_ID=<InstrTAG>
export PATH=${VLTSW_ROOT}/DFS/dfs/bin:PATH
export MANPATH=${VLTSW_ROOT}/DFS/dfs/man:$MANPATH
export DHS_DATA=$INS_ROOT/${INS_USER}/ARCDATA
export DHS_LOG=${DHS_DATA}
export BAD_DIR=${DHS_DATA}
export DHS_HOST=<OlasWS>
export DHS_CONFIG=archeso@${DHS_HOST}:/data/msg
```

Notice that PATH and MANPATH might be defined by VLT if OLAS_ID is defined. OLAS_ID defines a prefix tag to be added to all files delivered to the OlasWS. DHS_DATAdefines the directories in the local machine where VCSOLAC is looking for links containing soft links to files to be transferred, log files; DHS_LOG points to the directory where to write VCSOLAC logs, and BAD_DIR is not used but needs to be defined. DHS_HOST defines the OLAS machine, and DHS_CONFIG the account and directory at the OLAS machine receiving the files.

Also notice that VCSOLAC is based on RCP protocol in order to transfer files to the OlasWS. RCP requires that the account in the remote host (OlasWS) configures its '.rhosts' file to allow permission to the Instrument account in the InstWs.

Example: If the InstrWS is ′wvimos′ and the account running VCSOLAC is ′vimosmgr′, in the Olas

WS and for the 'archeso' account the '~/.rhosts' file should have an entry of this kind:

```
wvimos vimosmgr
```

5.3 Verifying VCSOLAC

5.3.1 VCSOLAC environment

Login in the InstWS as the instrument account and check

$DHS_HOST is valid and you can ping it.
$DHS_LOG directory exists and that you can write on it
$DHS_DATA points to the same directory as $DHS_LOG
$BAD_DIR points to a real and writable direcotry
$OLAS_ID is defined

Execute with no errors:

```
% touch test_file
% rcp test_file $DHS_CONFIG
```

5.3.2 Start VCSOLAC

Check if 'vcsolac' process already initiated at boot time:

```
% ps -ef | grep vcsolac
```

Otherwise you start vcsolac with:

```
% start-vcsolac
```

Check with 'ps -ef | grep vcsolac' that arguments for -dhshost and -dhsdata are as defined by $DHS_HOST and $DHS_DATA environment variables.

Look at the file $DHS_LOG/VCSOLAC-$HOSTNAME-$OLAS_ID.<date> where <date> should be today's date. The file should have no errors.

Keep a window monitoring the transfer of files with the command:

```
% tail -f $DHS_LOG/VCSOLAC-$HOSTNAME-$OLAS_ID.<date>
```

5.3.3 Send test files

Prepare a directory with test fits data, e.g. $HOME/myfits

```
% mkdir $HOME/myfits
% cd $HOME/myfits
```

Move in $HOME/myfits some .fits files and execute

```
% chmod 444 *.fits
% sendFiles $HOME/myfits fits
```

The script 'sendFiles' is a utility which allows to submit files to vcsolac by creating soft link in $DHS_DATA.

In the 'tail -f $DHS_LOG/VCSOLAC-$HOSTNAME-$OLAS_ID.<date>' window you should see an entry for each file sent to OlasWS as:

```
<date>[INFO] VCSOLAC-$HOSTNAME-$OLAS_ID: processing $DHS_LOG/<file>.fits
<date>[INFO] VCSOLAC-$HOSTNAME-$OLAS_ID: delivered file <file>.fits \(NNN bytes) to
DHS-$DHS_HOST
```

5.3.4 Verifying files delivered in the OlasWS machine

In the OlasWS, files from the InstWS are delivered to $DHS_CONFIG (as defined in the InstWS). They are hidden and with the following name:

```
.$OLAS_ID,NNNN,1,<file>.fits,VCSOLAC-$HOSTNAME-$OLAS_ID,DHS-$DHS_HOST.bulk
```

Remark: The files may disappear inmediatelly from $DHS_CONFIG in the OlasWS if 'dhs' process is running.

Remark: $OLAS_ID,$HOSTNAME,$DHS_HOST are the values defined in InstWS, not in the OlasWS
Stop VCSOLAC with the command

```
% cleanup-vcsolac
```

5.3.5 Verifying BOB to P2PP/OT

Verify that the instrument account is running with the VLT LITE mode, i.e. the environment variable VLTSW_CCSTYPE should be set to "lite". Either this is the default otherwise it should be set in the file .pecs/apps-all.env:

```
% vi ~/.pecs/apps-all.env
 VLTSW_CCSTYPE=lite
```

As user running BOB in InstWs try to send a message to the CCSLite environment in OhsWs:

```
% msgSend <CCSLite environment in OhsWs>  msgServer PING ""
```

The output of this command should be always MESSAGEBUFFER-OK.

Example: CCSLite environment in wu1dhs is also called 'wu1dhs', then execute:

```
% msgSend wu1dhs msgServer PING ""
MESSAGEBUFFER:
OK
```

6 DFS installation on PipelineWS

6.1 Preinstallation

- Mount the DVD "DFS dfs-5_6 dated 15.03.2008"
- The /data directory must exist (e.g. as a soft link to /diskb/data)

6.2 Installation

As root with the execution of script

```
cd /media/cdrom/dfs
./InstallDfsLinuxPipeline.sh <UT>
```

Where <UT> is:
UT0 at VCM-Garching
UT1,UT2,UT3,UT4,UT5(VLTI)) at Paranal
UT6(NTT), UT7(2P2) and UT8(3P6) at La Silla
UT9 at DFO-Garching
UT10 (VISTA) at Paranal

The InstallDfsLinuxPipeline.sh will just execute the following batch:

```
./InstallDfsLinux.sh install_ldsoconf
./InstallDfsLinux.sh install_groups   PipelineWS
./InstallDfsLinux.sh install_accounts PipelineWS
./InstallDfsLinux.sh install_pecsvltdfs
./InstallDfsLinux.sh install_netperf
./InstallDfsLinux.sh install_python
./InstallDfsLinux.sh install_opt
./InstallDfsLinux.sh install_flowmgr
./InstallDfsLinux.sh install_dataflowUser PipelineWS <UT>
./InstallDfsLinux.sh install_quality
./InstallDfsLinux.sh install_midas
./InstallDfsLinux.sh install_pipeline
./InstallDfsLinux.sh install_hostreport
```

More information about these options in the headers of these scripts.

6.3 Account pipeline

6.3.1 The .bashrc file

The data transfer is done via 'rcp', however the 'rcp' command, as any other 'r' command like 'rlogin' or 'remsh', executes the file .bashrc before the file transfer could start. The execution of this start file may delay the file transfer or it may even refuse the transfer if the execution of .bashrc returns a non-zero value.
So in order to improve the 'rcp' transfer, check that the installation created a .bashrc with the entries:

```
if [ "$TERM" = "dumb" ]; then return; fi
```

6.3.2 The .rhost file

The .rhosts file must be modified to allow access from archeso@UserWS, pipeline@PipelineWS, archeso@OlasWS. Example for UT1 the following lines must be added:

```
wgsoff1  archeso
wu1pl    pipeline
wu1dhs   archeso
```

6.3.3 The .dbrc file

Create file .dbrc in the $HOME directory with encrypted passwords:

```
<DBSERVER> observations frameingest <encrypted-password> DPREP
```

where <*DBSERVER*> is ASTOP in Paranal and SEGSRV in the VCM-Garching. And where <*encrtyped password*> is the result of the command:

```
% ~flowmgr/dfs/bin/stcrypt <non-encrypted-password>
```

Another useful command is dbrcGet. Executed with a database server and a database name as arguments,reads the .dbrc file and display the username and password that can be used to access the database. Example:

```
% dbrcGet <DBSERVER> observations
frameingest <non-encrypted-password>
```

6.3.4 The .esorex/esorex.rc file

This file is first time created after the execution of the command:

```
%  esorex --link-dir=/data/lists/reduced_olas \
     --log-dir=/data/msg \
     --output-readonly=TRUE \
     --suppress-link=FALSE \
     --create-config
```

This file may need to be recreated with newer releases of esorex. You can check if you need to recreate it by comparing the releases in the header of the file .esorex/esorex.rc and the output of the command esorex –version.

At Paranal and La Silla the value of the entry esorex.caller.recipe-dir is set to "."
This option may also be set using the environment variable ESOREX_PLUGIN_DIR.

At Garching and for DFO and QC blades, the individual pipelines are best configured to set esorex.caller.recipe-dir to the absolute pathname of the pipeline they want to execute. Example for the isaac account to run the ISAAC pipeline release 5.7.0 the entry should look like this:

```
esorex.caller.recipe-dir=/home/quality/pipelines/isaac-5.7.0
```

6.3.5 The pipeline.config file

This is a file containing instructions for the script *startRBS*. The official and standard way to run *startRBS* at the mountain is with:

```
% startRBS pipeline.config
frameingest <non-encrypted-password>
```

where *pipeline.config* should contain the following lines:

```
startDRS
drs @d pipeline.control NODISPLAY overwrite
set auto=yes
```

6.3.6 The gasgasno/gas-scripts directory

This directory should be created and populated with a soft link as follows:

```
% mkdir –p gasgano/gas-scripts
% cd gasgano/gas-scripts
% ln –s ~flowmgr/dfs/gasgano/bin/DO_submit.sh
```

6.3.7 Final check with dfscheck.sh

Run the script

```
% dfscheck.sh
```

And correct all pending issues

6.3.8 Other files

DFS does not specify which scripts should be executed with cronjobs. It is known, however , that the account pipeline could be used to run scripts which clean regularly the /data directory and other housekeeping tasks. The frequency and times of these cronjobs are responsibility of the operator. These entries may need to be taken into account during upgrades and installations of DFS.

Also, directories bin and scripts in the pipeline account are maintained by operators. To be taken into account in upgrades and installations of DFS. Particularly there is a .rtd-remote file generated by RTD/Gasgano.

6.4 Account quality

6.4.1 Final check with dfscheck.sh

Run the script

```
% dfscheck.sh
```

And correct all pending issues

6.4.2 Recreating the calibDB from scratch.

This section describes how to recreate the complete calibDB accumulated in /cal: the data accumulated in the old calibDB directory structure to the new one:

Kill first the msql2d process:

```
% kill `cat /home/quality/MSQL2/msql2d.pid`
```

Remove the old calibDB database:

```
% cd /home/quality/MSQL2/msqldb
% rm –rf calibDB
```

Recreate the calibDB database:

```
% createCalibDB.sh ALL
(type 'y' or 'Y' to all the questions)
```

Copy /cal into /tcal

```
% cd /tcal
% rm –rf *
% cd /cal
% tar cf - . | (cd /cal ; tar xf -)
% cd /tcal
% chmod -R +w .
```

The following command will help you to identify files with bad names that may need to be removed before running 'updateCalibDb -a' :

```
% badnames_in_calibDB.sh
(remove files accordingly to last command)
```

Finally, execute the updateCalibDb script:

```
% updateCalibDb –a
```

6.4.3 Adding a new instrument to the calibDB

It may happens, from time to time, that a new instrument has to be added to an existing calibDB. Re-executing the command createCalibDB.sh <new instrument> may have the risk of removing the content of the calibDB for existing old instruments. It is safer to execute:

```
% msql calibDB < $DFS_HOME/sql/qucSql/<new instrument>.sql
```

6.4.4 Updating the calibDB (not using IPIP)

The update of the calibDB with instrument data (calibration and rule files) is done automatically with each pipeline installation when using the IPIP scripts (see next chapter).
However the update of the calibDB could also be done by hand if necessary, i.e. with deliveries of master calibration frames. Here the instructions:

Deliveries for update of the calibDB should be done as compressed tar files with the following name convention:

```
<instrument>-calibdb-<release>.tar.gz
```

Examples of these deliveries are:

```
naco-calidb-1.1.tar.gz
uves-calibdb-1.6.1.tar.gz
```

To update the calidDB, stop any pipeline activity (e.g. RBS, DO, pipelines, etc) and execute:

```
% cd /tcal
% gunzip –c …/<instrument>-calidb-<release>.tar.gz | tar xf  -
% rm –rf dic offline gasgano
% chmod -R +w .
% badnames_in_calidDB.sh   (remove files accordingly to last command)
% updateCalibDb -a
```

6.4.5 Installing pipelines with IPIP

IPIP is made of three modules, the first 'ipip' checks names and preconditions, the second one 'ipip-m1.pl' compiles the sources and installs the recipes. The second module 'ipip-m2.pl' unpacks the content of the CalibDB tarfile and carries out an updateCalibDB command.

An instrument pipeline delivery consists of, at least, two tarfiles:

```
<instrument>-<rel>.tar.gz
<instrument>-calibdb-<rel>.tar.gz
```

and one optional file:

```
<instrument>-dic-<rel>.tar.gz
```

which can be located anywhere, but in the same directory, under the 'quality' account (E.g. ~quality/pipelines)

A pipeline is installed with the command:

```
 % ipip <instrument>-<rel>.tar.gz
```

The following is a description of the execution of 'ipip':

IPIP module 'ipip' executes preliminary checks :

- the session must be running as <quality> account
- zcat, tar or gtar must exist
- $MIDASHOME and $MIDVERS defined and existing for esomidas
- $VLTROOT must be defined
- /tcal and /cal must exist and point to directories
- $MIDASHOME/$MIDVERS/pipeline must exist and be writable

- <quality> account existing
- <quality> account environment
- Operating system is either of SunOS, Linux or HP-UX.
- calibDb does not contain illegal files (via external script check-calib.pl, which is callable separately)

IPIP module 'ipip-m1.pl' will execute:

- delete any existing directory $MIDASHOME/$MIDVERS/pipeline/<instrument>- <rel>
- create a temporary directory under $MIDASHOME/$MIDVERS/pipeline
- cd to that directory
- untar the source tarfile
- cd to the directory <instrument>-<rel>
- execute ./setup or ./setup.sh providing as one and only command line argument, the path to the deployment directory which will be $MIDASHOME/$MIDVERS/pipeline/ <instrument>-<rel>
- If setup returns a non-zero code, ipip will exit showing the error message from setup delete the temporary directory
- create under $MIDASHOME/$MIDVERS/pipeline a link <instrument> --> <instrument>-<rel>.

IPIP module 'ipip-m2.pl' will execute:

- cd to appropriate /tcal subdirectory
- delete any already existing files which might be overwritten
- untar the tarfile <instrument>.calibdb.<rel>.tar.gz
- copy all the files contained in the dic subdirectory into
- the place specified by the $DFS_QC1_DIC_DIR
- create all needed directories under /cal
- run updateCalibDb -a

If the optional <instrument>-dic-<rel>.tar.gz file exist, IPIP will execute:

- Untar the contents of the <instrument>-dic-<rel>.tar.gz under the directory ~quality/ INS_ROOT.

IPIP will complete the pipeline installation with the execution of

- makeDoDictionary, which generates a dictionary file under ~quality/INS_ROOT_DO/ SYSTEM/Dictionary for the use of the command 'startDO' exclusively.

6.4.6 Installing pipeplines with IPIP for DFO machines

On DFO machines there is now calibDB, msql2d, /cal or /tcal directories, thus the standard IPIP will fails with the first checks it executes. For DFO you can use a modified version of the ipip script called "ipip.dfo". This version of IPIP will not run the checks, however it will try to install the <instrument>-calibdb-<rel>.tar.gz file is it exists. For that reason it is necessary to remove or move away this component of the pipeline distribution before executiong "ipip.dfo":

```
% cd pipeline-deliveries
% mv <instrument>-calibdb-<rel>.tar.gz <instrument>-calibdb-<rel>.tar.gz.org
% ipip.dfo <instrument>-<rel>.tar.gz
```

On DFO machines is becoming very common the need to install a given pipeline against a different DFS version that the default (typically when newer DFS releases integrates newer version of CPL). These installations are possible if the environment is change appropriately. Example

```
% export DFS_RELEASE=dfs-5_6_1
% . /home/flowmgr/$DFS_RELEASE/admin/dataflowUser.sh
% ipip.dfo <instrument>-<rel>.tar.gz
```

6.4.7 Installing MIDAS

Retrieve source files MIDAS <release><patchlevel>.tar.gz (e.g. <release> could be 07FEB, and <patchlevel> could be pl1.0) from the anonymous ftp account ftp::/ ftp.hq.eso.org/pub/midaspub/<release>/sources, and copy it to /midas, then execute:

```
% cd /midas
% gunzip -c <release><patchlevel>.tar.gz | tar xf -
% ln -s <release><patchlevel> <release>
% cd <release>/install/unix
% ./autoconfig
```

and wait until the installation gets completed.

To verify the MIDAS installation, execute:

```
% cd /midas
% mkdir -p tmp
% cd tmp
% rm –f *
% export MIDASHOME=/midas
% export MIDVERS=<release>
% inmidas
Midas> @ vericopy
...
Midas> @@ veriall -nodisplay
...
```

It should finish with no errors.

7 DFS installation on UserWS

7.1 Preinstallation

- Mount the DVD "DFS dfs-5_6 dated 15.03.2008"
- The /data directory must exist (e.g. as a soft link to /diskb/data) if the machine is to be configured none UT-less, otherwise:
- If UT-less is to be used, do not create /data, instead create the /data-utN directories first (real diretories or soft links to real ones).
  ```
  % mkdir /diskb/data-ut1; ln –s /diskb/data-ut1 /data-ut1
  % mkdir /diskb/data-ut2; ln –s /diskb/data-ut2 /data-ut2
  % mkdir /diskb/data-ut3; ln –s /diskb/data-ut3 /data-ut3
  % mkdir /diskb/data-ut4; ln –s /diskb/data-ut4 /data-ut4
  % mkdir /diskb/data-ut5; ln –s /diskb/data-ut5 /data-ut5
  ```
  UT5 represents here the VLTI.
- If this machine is also subscribing to MASCOT data, you also need to create the /data-mascot, e.g.:
  ```
  % mkdir /diskb/data-mascot ; ln –s /diskb/data-mascot /data-mascot
  ```

7.2 Installation

As root with the execution of script

```
cd /media/cdrom/dfs
./InstallDfsLinuxUserWS.sh <UT>
```

Where <UT> is:

UT0 at VCM-Garching

UT1,UT2,UT3,UT4,UT5(VLTI)) at Paranal

UT6(NTT), UT7(2P2) and UT8(3P6) at La Silla

UT9 at DFO-Garching

UT10 (VISTA) at Paranal

The InstallDfsLinuxPipeline.sh will just execute the following batch:

```
./InstallDfsLinux.sh install_ldsoconf
./InstallDfsLinux.sh install_groups   UserWS
./InstallDfsLinux.sh install_accounts UserWS
./InstallDfsLinux.sh install_pecsvltdfs
./InstallDfsLinux.sh install_opt
./InstallDfsLinux.sh install_netperf
./InstallDfsLinux.sh install_python
./InstallDfsLinux.sh install_flowmgr
```

```
./InstallDfsLinux.sh install_dataflowUser UserWS <UT>
./InstallDfsLinux.sh install_archeso UserWS
./InstallDfsLinux.sh install_astroN
./InstallDfsLinux.sh install_scisoft
./InstallDfsLinux.sh install_quality
./InstallDfsLinux.sh install_midas
./InstallDfsLinux.sh install_hostreport
```

More information about these options in the headers of these scripts.

7.3 Account archeso

The account archeso is used, in the UserWS, as a dealer of raw data from the OlasWS and of reduced data from the PipelineWS in one side and the user accounts (astroN and astuserN) in the other side.

7.3.1 The .bashrc file

The data transfer is done via 'rcp', however the 'rcp' command, as any other 'r' command like 'rlogin' or 'remsh', executes the file .bashrc before the file transfer could start. The execution of this start file may delay the file transfer or it may even refuse the transfer if the execution of .bashrc returns a non-zero value.
So in order to improve the 'rcp' transfer, check that the installation created a .bashrc with the entries:

```
if [ "$TERM" = "dumb" ]; then return; fi
```

7.3.2 The .rhost file

The .rhosts file must be modified to allow access from archeso@UserWS, pipeline@PipelineWS, archeso@OlasWS. Example for UT1 the following lines must be added:

```
wgsoff1  archeso
wu1pl    pipeline
wu1dhs   archeso
```

7.3.3 Final check with dfscheck.sh

Run the script

```
% dfscheck.sh
```

And correct all pending issues

7.3.4 Other files

DFS does not specify which scripts should be executed with cronjobs. It is known, however , that the account archeso could be used to run scripts which clean regularly the /data directory and other housekeeping tasks. The frequency and times of these cronjobs are responsibility of the operator. These entries may need to be taken into account during upgrades and installations of DFS.

Also, directories bin and scripts in the archeso account are maintained by operators. To be taken into account in upgrades and installations of DFS.

7.4 Account quality

The account quality in the UserWS implements and executes the same functions as in the PipelineWS. However the only way to cheat the account quality to believe it is running in a PipelineWS is by forcing in PECS file ~/.pecs/apps-all.env the environment PipelineWS to be this host, i.e.:

```
export PipelineWS=$HOSTNAME
```

7.5 Account astro and astroN

The account 'astro' is used to subscribe to OlasWS and PipelineWS (raw and reduced data). The 'astroN' provide the same functionality as the 'astro' accounts, they only change the environment, which is automatically configured with the InstallDfsLinuxUser.sh script by writing them in the ./pecs/apps-all.env file:

```
export DFS_USER=astro
export OlasWS=<OlasWS_to_subscribe_to_raw_data>
export QcWS=<PipelineWS_to subscribe_to_reduced_data>
export DFS_DATA_ROOT=<data_root_area>
```

Example: To implement the previous 'astro1' account enter the following definitions in the ~astro1/pecs/apss-all.env file:

```
export DFS_USER=astro
export OlasWS=wu1dhs
export QcWS=wu1pl
export DFS_DATA_ROOT=/data-ut1
```

7.5.1 The .dbrc file

Create file .dbrc in the $HOME directory with encrypted passwords:

```
<DBSERVER> observations frameingest <encrypted-password> DPREP
<DBSERVER> asto asto <encrypted-password> ASTOREP
```

```
<DBSERVER> ambient archeso <encrypted-password> AMBIREP
```

where <*DBSERVER*> is ASTOP in Paranal and SEGSRV in the VCM-Garching. And where <*encrtyped password*> is the result of the command:

```
% ~flowmgr/dfs/bin/stcrypt <non-encrypted-password>
```

Another useful command is dbrcGet. Executed with a database server and a database name as arguments,reads the .dbrc file and display the username and password that can be used to access the database. Example:

```
% dbrcGet <DBSERVER> observationsframeingest <non-encrypted-password>
```

7.5.2 The .dataSubscriberConfig file

The first time the command 'dataSubscriber' is executed, the ".dataSubscriberConfig" file is created if it does not exists. Please check the contents:

It uses some defaults that may require to be edited to fit your environment, like:

```
DHS_HOST-RAW        : wuNdhs
DHS_CONFIG-RAW      : archeso@wuNdhs:/data-utN/msg
DHS_DATA-RAW        : /data-utN/raw
StartScript-RAW     : dhsSubscribeControl start userRaw
DHS_HOST-RED        : wuNpl
DHS_CONFIG-RED      : pipeline@wuNpl:/data/msg
DHS_DATA-RED        : /data-utN/reduced
StartScript-RED     : dhsSubscribeControl start userReduced
transTables         : {Moon /home/flowmgr/dfs/gui/guiDomino/config /
MoonCalc.ftt} {Spectroscopy /home/flowmgr/dfs/gui/guiDomino/config/spectroscopy.ht}
{Imaging /home/flowmgr/dfs/gui/guiDomino/config/imaging.ht}
```

Please notice that the following setting are not used: DHS_DATA-RAW and DHS_CONFIG-RAW. Instead of these setting the following environment variables are used: OlasWS, DHS_HOST and DHS_CONFIG, all of them defined in the '~flowmgr/dfs/admin/dataflowUser.sh file.

7.5.3 Scisoft setting

In order to make use of the applications available in the 'scisoft' packages, the accounts 'astro' or 'astroN" may need to configure it under PECS files.

In the ~/.pecs/misc-all.env file add the following entries in order to run the packages available on the scisoft installation:

```
. /scisoft/bin/Setup.bash
```

```
export LM_LICENSE_FILE=/scilocal/idl/idl_5/license.dat
```

In the ~/.pecs/misc-all.ali, one should add the following entries:

```
alias cl=" cd ~/iraf; xgterm -sb -e cl"
```

Create the ~/iraf directory and create some configuration files:

```
/scilocal/bin/mkiraf xgterm
```

7.5.4 Accounts astuserN

The astuserN accounts are meant to be working account for the visiting astronomers. Each astuserN account has exclusive access to data coming from its UT PipelineWS and OlasWS machines.

Because accounts astuserN need to be created clean and new for each new visiting astronomer they can be quickly generated from a tar file located in:

```
/usr/server/bin/user_template.tar
```

The creation of the astuserN accounts is done from a sudo script executed by astroN accounts as follows:

```
% remakeUsers astuserN
```

This script will take care of all details for these accounts. Otherwise if they are created by root the following should be checked:

In order to mantain privacy from other astuserN accounts, each account should protect the HOME directory:

```
% cd
% chmod 750 .
```

In the ~/.pecs/misc-all.env file add the following entries in order to run the packages available on the scisoft installation:

```
. /scisoft/bin/Setup.bash
export LM_LICENSE_FILE=/scilocal/idl/idl_5/license.dat
```

In the ~/.pecs/misc-all.ali, one should add the following entries:

```
alias cl=" cd ~/iraf; xgterm -sb -e cl"
```

Create the ~/iraf directory and create some configuration files:

```
/scilocal/bin/mkiraf xgterm
```

7.5.5 Account mascot (only Paranal)

Do not try to use the GUI 'dataSubscriber' to subscribe to MASCOT files. It may work if you set properly the ~/.dataSubscriberConfig file, but it will interfere with other 'dataSubscriber' running in the same machine (e.g. from astroN accounts). Best is to disable it, e.g. by setting an alias in the ~/.pecs/misc-all.ali as follows:

```
alias dataSubscriber='echo "dataSubscriber is disabled for this account. Use instead: " ; echo "
dhsSubscribeControl start/stop mascot"'
```

Subscribing to MASCOT files manually

```
% dhsSubscribeControl start mascot
```

To stop the subscripton, execute:

```
% dhsSubscribeControl stop mascot
```

8 DFS installation on AstoWS

8.1 Preinstallation

- Mount the DVD "DFS dfs-5_6 dated 15.03.2008"
- The /data directory must exist (e.g. as a soft link to /diskb/data)

8.2 Installation

As root with the execution of script

```
cd /media/cdrom/dfs
./InstallDfsLinuxAsto.sh <UT>
```

Where <UT> is:
UT0 at VCM-Garching
UT1,UT2,UT3,UT4,UT5(VLTI)) at Paranal
UT6(NTT), UT7(2P2) and UT8(3P6) at La Silla
UT9 at DFO-Garching
UT10 (VISTA) at Paranal

The InstallDfsLinuxPipeline.sh will just execute the following batch:

```
./InstallDfsLinux.sh install_ldsoconf
./InstallDfsLinux.sh install_groups   AstoWS
./InstallDfsLinux.sh install_accounts AstoWS
./InstallDfsLinux.sh install_pecsvltdfs
./InstallDfsLinux.sh install_opt
./InstallDfsLinux.sh install_netperf
./InstallDfsLinux.sh install_flowmgr (1)
./InstallDfsLinux.sh install_dataflowUser AstoWS <UT> (4)
./InstallDfsLinux.sh install_archeso AstoWS
./InstallDfsLinux.sh install_hostreport
./InstallDfsLinux.sh install_quality
./InstallDfsLinux.sh install_pipeline
./InstallDfsLinux.sh install_hostreport
```

More information about these options in the headers of these scripts.

8.3 Account archeso

The account archeso is used, in the AstoWS to collect data from the OlasWS and generate .

8.3.1 The .bashrc file

The data transfer is done via ’rcp’, however the ’rcp’ command, as any other ’r’ command like ’rlogin’ or ’remsh’, executes the file .bashrc before the file transfer could start. The execution of this start file may delay the file transfer or it may even refuse the transfer if the execution of .bashrc returns a non-zero value.
So in order to improve the ‘rcp’ transfer, check that the installation created a .bashrc with the entries:

```
if [ "$TERM" = "dumb" ]; then return; fi
```

8.3.2 The .rhost file

The .rhosts file must be modified to allow access from archeso@OlasWS, pipeline@PipelineWS.
Example for UT1 the following lines must be added:

```
wu1pl    pipeline
wu1dhs   archeso
```

8.3.3 The .dbrc file

Create file .dbrc in the $HOME directory with encrypted passwords:

```
<DBSERVER> asto2 asto <encrypted-password> ASTOREP
```

where <*DBSERVER*> is ASTOP in Paranal and SEGSRV in the VCM-Garching. And where <*encrtyped password*> is the result of the command:

```
% ~flowmgr/dfs/bin/stcrypt <non-encrypted-password>
```

Another useful command is dbrcGet. Executed with a database server and a database name as arguments,reads the .dbrc file and display the username and password that can be used to access the database. Example:

```
% dbrcGet <DBSERVER> observations
frameingest <non-encrypted-password>

% chmod 600 .dbrc
```

8.3.4 The .netrc file

This file located in the home directory will be used by the curl library. Its goal is to be able to connect to a machine without requiring the user to give its name and password during operations. Each line of this file will be associated to a user for a given machine and will have the following schema:

```
machine <machine name>  login <user name>  password  <non encrypted password>
```

Examples:

```
machine wu1dhs  login archeso password selavi
```

For security reasons, as it contains passwords, only the owner of the file should have the permissions to read and modify it.

```
% chmod 600 .netrc
```

8.3.5 The astoconf directory

It should contain the following files:

```
astoControl.config
dataSubscriberAsto.config
libAd.config
libAm.config
writeControl.config
mkisofs.config
```

8.3.6 Final check with dfscheck.sh

Run the script

```
% dfscheck.sh
```

And correct all pending issues

9 DFS installation on DataTransferWS

9.1 Preinstallation

- Mount the DVD "DFS dfs-5_4 dated 15.03.2008"
- The /data directory must exist (e.g. as a soft link to /diskb/data)

9.2 Installation

As root with the execution of script

```
cd /media/cdrom/dfs
./InstallDfsLinuxDataTransfer.sh <UT>
```

Where <UT> is:
UT0 at VCM-Garching
UT1,UT2,UT3,UT4,UT5(VLTI)) at Paranal
UT6(NTT), UT7(2P2) and UT8(3P6) at La Silla
UT9 at Garching
UT10 (VISTA) at Paranal

The InstallDfsLinuxDataTransfer.sh will just execute the following batch:

```
./InstallDfsLinux.sh install_ldsoconf
./InstallDfsLinux.sh install_groups   DataTransferWS
./InstallDfsLinux.sh install_accounts DataTransferWS
./InstallDfsLinux.sh install_pecsvltdfs
./InstallDfsLinux.sh install_netperf
./InstallDfsLinux.sh install_python
./InstallDfsLinux.sh install_opt
./InstallDfsLinux.sh install_flowmgr
./InstallDfsLinux.sh install_dataflowUser DataTrasferWS <UT>
./InstallDfsLinux.sh install_archeso DataTransferWS
./InstallDfsLinux.sh install_hostreport
```

More information about these options in the header of the InstallDfsLinux.sh script.

9.3 Account archeso

The account archeso is used, in the DataTransferWS, to collect data from the OlasWS, compress it and send them, via network, to the DataTransferWS at Garching, except for VISTA telescope that, due to the high volume of data generated, the process just copies the data to external USB disks.

At Paranal two DataTransferWSs are available. One for VLT(UTs) + VLTI called wgsdts, the other one for the VISTA telescope or wvcdts. The one for VLT+VLTI has to subscribe to each OlasWS. The one for VISTA just subscribe to the VISTA OlasWS.

At La Silla only one DataTransferWS is available and it's called wlsdts. It subscribes to all La Silla OlasWS: NTT, 2P2 and 3P6 telescopes.

9.3.1 The .bashrc file

The dhsSubscriber retrieves data from the OlasWS via 'rcp', however the 'rcp' command, as any other 'r' command like 'rlogin' or 'remsh', executes the file .bashrc before the file transfer could start. The execution of this start file may delay the file transfer or it may even refuse the transfer if the execution of .bashrc returns a non-zero value.
So in order to improve the 'rcp' transfer, check that the installation created a .bashrc with the entries:

```
if [ "$TERM" = "dumb" ]; then return; fi
```

The file .bahsrc should contain the above line by default with the DFS standard installation. But if the account is re-created new and with PECS, it will miss this line, thus it is necessary to proceed as follows:

```
mv .bashrc .bashrc. org
cp .bashrc.org .bashrc
 vi .bashrc (and at line 24 add the following line):
    if [ "$TERM" = "dumb" ]; then return; fi
```

9.3.2 The .rhost file

The .rhosts file must be modified to allow access from archeso@OlasWSs.
The DataTransferWS for VISTA should have just one line:

```
wvcdhs   archeso
```

The DataTransferWS for VLT (all UTs + VLTI) should grant access to each of these archeso@OlasWS:

```
wu1dhs  archeso
wu2dhs  archeso
wu3dhs  archeso
wu4dhs  archeso
wvgdhs  archeso
```

The DataTransferWS for La Silla should grant access to each of these archeso@OlasWS:

```
wg5dhs  archeso
```

```
w2p2dhs  archeso
w3p6dhs  archeso
```

The DataTransferWS in Garching should keep this file empty.

9.3.3 The .ssh/authorized_keys file

This file authorizes Secure Shell connections (scp, ssh,) without password from those pairs of trusted accounts/hostnames, listed in the file and with a valid public key.

The file is only required by DataTransferWS at Garching (pastage1 and pastage2) and in order to receive data, via bbcp, from DataTransferWSs at Paranal and La Silla and fromt the UsbWSs. Therefore the list of trusted accounts/hostnames should be limited to:

ssh-rsa <key> archeso@wgsdts	DataTransferWS for VLT-VLTI at Paranal
ssh-rsa <key> archeso@wvcdts	DataTransferWS for VISTA at Paranal
ssh-rsa <key> archeso@wlsdts	DataTransferWS for NTT,3P6,2P2 at La Silla
ssh-rsa <key> archeso@acmusb1	UsbWS at Garching

The keys are generated at the DataTransferWSs at Paranal and La Silla in the file ~/.ssh/id_rsa.pub after the execution of the command:

```
ssh-keygen -t rsa        (press CR to all questions)
```

Transfer this file to the DataTrasferWS at Garching (pastage1 and pastage2), e.g. /tmp and concatenate it to the existing file ~/.ssh/authorized_keys:

At archeso@pastage1-2:

```
cat /tmp/id_rsa.pub >> ~/.ssh/authorized_keys
```

NOTE: The DNS may not be able to resolve hostnames at Paranal (it does for La Silla ones), therefore it would be better to replace them with their IPs. In such case the .ssh/authorized_keys file should look like this:

ssh-rsa <key> archeso@134.171.224.14.	DataTransferWS for VLT-VLTI at Paranal
ssh-rsa <key> archeso@134.171.237.14	DataTransferWS for VISTA at Paranal
ssh-rsa <key> archeso@134.171.101.10	DataTransferWS for NTT,3P6,2P2 at La Silla
ssh-rsa <key> archeso@acmusb1	UsbWS at Garching

9.3.4 The OlasWS enviroment

This environment defines the different OlasWSs to subscribe to.
If DataTransferWS for VLT(all UTs) + VLTI want to subscribe to each of their OlasWSs, the environment OlasWS should be defined as:

```
export OlasWS=“wu1dhs wu2dhs wu3dhs wu4dhs wvgdhs”
```

The DataTransferWS for VISTA should have it already defined as:

```
export OlasWS=wvcdhs
```

And the DataTransferWS for La Silla should have it already defined as:

```
export OlasWS=”wg5dhs w2p2dhs w3p6dhs”
```

DataTransferWS for Garching should have OlasWS empty or undefined.

Changes to the OlasWS environment should only be done after stopping all previous dhsSubscribe processes and before starting new ones. The sequence should be:

```
dhsSubscribeControl stop datatransfer
export OlasWS=”wg5dhs w2p2dhs w3p6dhs”
dhsSubscribeControl start datatransfer
```

The default value for OlasWS contains only one hostname. For permanent multiple subscriptions edit the value of OlasWS at the end of the ~archeso/.pecs/misc-all.env. Example for wgsdts:

```
if [ -f  ~flowmgr/${DFS_RELEASE}/admin/dataflowUser.sh ]; then
  . ~flowmgr/${DFS_RELEASE}/admin/dataflowUser.sh
fi
export OlasWS="wu1dhs wu2dhs wu3dhs wu4dhs wvgdhs"
```

9.3.5 The DataTransfer ENABLED environments

These are environment variables used to indicate which processes should be launched with the execution of the datatransferControl script (e.g. at boot time):

```
DFS_DHSSUBSCRIBE_DATATRANSFER_ENABLED
DFS_ADO_ENABLED
DFS_NTS_HEADER_ENABLED
DFS_NTS_RAW_ENABLED
DFS_DTS_ENABLED
DFS_AR_ENABLED
```

The lists of respective scripts launched when the environments are set to “1” are:

```
dhsSubscribeControl start datatransfer
adoControl start
ntsControl start header
ntsControl start raw
dtsControl start
```

```
arControl start
```

The current settings for these environments are located at the end of ~archeso/.pecs/misc-all.env file:
At UT1-4 and VLTI (wgsdts):

```
export DFS_DHSSUBSCRIBE_DATATRANSFER_ENABLED=1
export DFS_ADO_ENABLED=1
export DFS_NTS_HEADER_ENABLED=1
export DFS_NTS_RAW_ENABLED=1
export DFS_DTS_ENABLED=1
```

At VISTA (wvcdts):

```
export DFS_DHSSUBSCRIBE_DATATRANSFER_ENABLED=1
export DFS_ADO_ENABLED=1
export DFS_NTS_HEADER_ENABLED=1
export DFS_DTS_ENABLED=1
```

At NTT, 2P2 and 3P6 (wlsdts):

```
export DFS_DHSSUBSCRIBE_DATATRANSFER_ENABLED=1
export DFS_ADO_ENABLED=1
export DFS_NTS_HEADER_ENABLED=1
export DFS_DTS_ENABLED=1
```

At Garching (pastage1 and pastage2): none

9.3.6 The datatransfer directory

Contains configuration files and scripts for different DataTransfer processes. It is not the purpose of this manual to describe them, but just to mention that the content of this directory is under configuration control, and the proper installation would be:

```
export CVSROOT= :pserver:archeso@cvssrv.hq.eso.org:/project1/CVS
or
export CVSROOT= :pserver:archeso@134,171.42.7:/project1/CVS
touch .cvspass
cvs login
cvs co DTS/datatransfer-$HOSTNAME
ln –s DTS/datatransfer-$HOSTNAME datatransfer
```

Typically, any file modified under directory datatransfer should be checked in the repository. Example, if for operation needs, the file ~/datatransfer/config/bbcp.conf is modified, then commit the changes with the command:

```
cvs ci  ~/datatransfer/config/bbcp.conf
(Type: name, and the reason for the change)
```

Other useful CVS commands are:

```
cvs log <file>       # to see all the releases committed.
cvs status <file>    # to see the current status of the file: up-to-date, locally modified
```

9.3.7 Other DataTransfer environments

For the execution of local Shell and Python scripts under the ~/datatransfer directory, add the following entries to the ~archeso/.pecs/misc-all.env:

```
export PYTHONPATH=$HOME/datatransfer/lib:$HOME/datatransfer/config
export PATH=$PATH:$HOME/datatransfer/bin:/opt/cfitstools/bin:/opt/bbcp/bin
```

Many other environment variables are defined to their default values in file ~flowmgr/dfs/admin/dataflowUser.sh.
However and in particular cases they may need to be over-written, or removed. This is the case of the environment DFS_FITS2HDR=”dfits –x 0” on DataTransferWS wlsdts (VISTA) that should be removed by adding the following entry to the ~archeso/.pecs/misc-all.env file:

```
unset DFS_FITS2HDR
```

9.3.8 The /data structure

It is created automatically with the DFS installation scripts. This is just a presentation of the expected structure:

```
|-- header
|-- raw
|-- msg
|-- bad
|-- reduced
|-- backlog
|-- lists
    |-- archivedataorganizer
    |   |-- incoming
    |   |-- processed
    |   |-- processing
    |   `-- rejected
    |-- archivereplicator
    |   |-- incoming
    |   |-- processed
    |   |-- processing
    |   |-- rejected
    |   `-- unpack
    |-- disktransferscheduler
    |   |-- incoming
    |   |-- processed
    |   |-- processing
    |   `-- rejected
    |-- networktransferscheduler
```

```
|   |-- incoming
|   |-- packing
|   |-- processed
|   |-- processing
|   `-- rejected
`-- networktransferscheduler4hdr
    |-- incoming
    |-- packing
    |-- processed
    |-- processing
    `-- rejected
```

NOTE: Some directories might not be necessary at all, as they depend on the configuration of each DataTransferWS. For simplicity of the installation they were all created though.

9.3.9 Final check with dfscheck.sh

Run the script

```
% dfscheck.sh
```

And correct all pending issues