E U R O P E A N S O U T H E R N O B S E R V A T O R Y

Organisation Européenne pour des Recherches Astronomiques dans l'Hémisphère Austral

Europäische Organisation für astronomische Forschung in der südlichen Hemisphäre

VERY LARGE TELESCOPE

VLT Common Software

Installation Manual

Doc.No. VLT-MAN-ESO-17200-0642

Issue 1.14

Date 12/03/01

	Name	Date	Signature
Prepared	P.Sivera	12/03/01	
Approved	E.Allaert		
Released	G.Raffi		

Change Record

Issue/Rev.	Date	Section/Page affected	Reason/Initiation/Document/Remarks
1.0	06/05/94	All	First Issue
1.1	16/06/94	All	CCS added. Installation verification improved
1.2	15/12/94	All	NOV94 version. Section 2, 3, 4 updated according to the following major changes: • CCS evt / ntt / tims / dbl added • HOS / Sequencer added • INS Common Software added • LCC and Drivers updated • porting to SUN-Solaris 2.3 Sections 5, 6, 7 added.
1.3	06/03/95	All	FEB95 version. Section 2, 3, 4 updated according to the following major changes: • CCS new modules: eccs, fnd, evh, evhEt, panel, samp. • LCC and Drivers updated Other minor changes on all sections.
1.4	17/08/95	All	JUL95 version. Section 2, 3, 4 updated according to the following major changes: • general information moved to the “Overview Manual” • CCS lite • all modules updated • new verification procedure
1.5	22/01/96	All	DEC95 version. • problem reporting moved to a separate document general • TCS added • new GNU installation procedure

Change Record

Issue/Rev.	Date	Section/Page affected	Reason/Initiation/Document/Remarks
1.6	07/06/96	All	JUN96 version. • new tcl/tk installation procedure • new VxWorks installation procedure
1.7	11/11/96	All	NOV96 version. • HP-UX 9.x phased out • CCD Control Software added
1.8	31/05/97	All	MAY97 version. • /usr/local not used any longer
1.9	22/11/97	All	NOV97 version. • HP-UX10.20 and Tornado1.0.1 • Fiera Control Software
1.10	02/11/98	All	OCT98 version.
1.14	12/03/01	All	OCT99 version. See Release Notes for more.
1.13	12/12/00	All	NOV2000 version
1.14	12/03/01	Installation, Verification	Upgraded to MAR2001, added Appendix A with instruction about PECS

TABLE OF CONTENTS

1 INTRODUCTION

1.1 PURPOSE

This manual provides the installation procedure of the VLT Common Software. A general overview of the VLT Common Software is given in [5].

This document assumes that you have an overview of the VLT Common Software, a good UNIX knowledge and some experience in installing public domain software. Unless you are the system manager, you may need his help to perform some of the installation steps.

1.2 SCOPE

This document describe the installation of the MAR2001 version of The VLT Common Software. The detailed scope is given is section 2.3.

The installation procedure allows selective installation (section 3).

1.3 APPLICABLE DOCUMENTS

N / A

1.4 REFERENCE DOCUMENTS

In addition to documents that are part of the VLT Software documentation kit (see section 2.3.2 for the complete list), the following documents are referenced in this document.

[1] VxWorks Documentation

[2] HP-RTAP Documentation

[3] VLT -PLA-ESO-00000-0006, 2.0 --- VLT Software Management Plan

[4] VLT -PRO-ESO-10000-0228, 1.0 --- VLT Software Programming Standard

[5] VLT -MAN-ESO-17200-0888, 1.0 --- VLT SOFTWARE Overview

[6] VLT -MAN-ESO-17200-0981, 1.0 --- VLT SOFTWARE Problem Report & Change Request User Manual

[7] MIDAS - Documentation (applicable version)

[8] VLT -MAN-ESO-17200-2238, 1.0 --- VLT Common Software Combined OS Installation Manual

[9] VLT -MAN-ESO-13640-1707, --- FIERA CCD Controller - Software Maintenance Manual

1.5 ABBREVIATIONS AND ACRONYMS

The following abbreviations and acronyms are used in this document:

CCS	Central Control Software
CCSlite	VCS providing a subset of functions (no RTAP needed)
DBMS	Database Management System
Full CCS	VCS providing all functions (needs HP / RTAP)
HOS	High Level Operations Software

HW	Hardware
ICS	Instrument Control Software
I/O	Input/Output
LAN	Local Area Network
LCC	LCU Common Software
LCU	Local Control Unit
N/A	Not Applicable
NFS	Network File System
NOCCS	VCS in a minimal configuration
OS	Operating System
OSB	Observation Software base
PCF	Point Config File
ROS	Remote Operation Software
RTAP	Real-Time Application Platform (from Hewlett-Packard)
SPR	Software Problem Report
SW	Software
TBC	To Be Confirmed
TBD	To Be Defined
TCS	Telescope Control Software
UIF	(Portable) User Interface (Toolkit)
VCS	VLT Common Software
VLTSW	synonym of VCS
VLT	Very Large Telescope
WAN	Wide Area Network
WS	Workstation

1.6 GLOSSARY

N/A

1.7 STYLISTIC CONVENTIONS

The following styles are used:

`teletype`
: in the text, names of programs, files, directory, etc.
in installation examples, your typing.

`<...>`
: for parts that have to be substituted with the real content before typing.

bold and *italic* are used to highlight words.

2 OVERVIEW

2.1 GENERAL

The VLT Common Software currently includes:

- VLT development utilities
- Central Common Software (CCS)
- Instrumentation Common Software (INS)
- Telescope Control Software (simulation)
- Sequencer
- WS-LCU communication software (Qserver)
- LCU Common Software (LCC)
- Drivers
- Motor Control
- LCU contributions
- CCD Control Software (formerly disstributed as a separate tape)
- FIERA Control Software
- Infrared Detector Control Software (IRD)
- installation and installation verification procedures
- simple examples

and all the relevant documentation in both printed and PostScript file format.

> ***The present version of the VLT Common Software replaces entirely any previous ESO distribution of VLT Software. No support will be given to previous versions.***

The distribution kit contains all the sources needed to regenerate the software on both HP and SUN systems. Distributing the sources also provides developers with:

- examples of applications (e.g., test programs)
- skeletons for the development of similar modules (e.g., drivers)

For a complete description of each module mentioned, please refer to the appropriate user manual, included in the shipment.

The major differences with the previous version as well as currently knows problems and work around are summarized in the RELEASE NOTES, that are published together with the documentation and also available on WEB.

Even if you are interested in only one part of the VLT Software, please read ALL of the following sections.

2.1.1 Copyright

The VLT Common Software is distributed outside ESO for the development of applications related to the VLT Project and ruled by the “General Conditions of ESO Contracts”. Any other use is not al-

lowed without prior authorization from ESO.

The rights of Third Parties (Free Software Foundation, VxWorks, etc.), whose software is for convenience copied on the VLT Common Software tape, are ruled by their copyright notice included in their software.

The file COPYING, included in the distribution kit, contains the full copyright notice.

2.2 SUPPORTED CONFIGURATION

2.2.1 Hardware

WS

- HP 9000 / 700 series
- SUN

LCU:

- Motorola MVME 167 CPU board - any version
- Motorola MVME 2604 CPU board - any version
- VMIVME-3111 Analog I / O board - version ID 0x0d
- ACROMAG AVME948X Digital I / O board - any version
- ESO Time Reference Board - ESO internal version
- VME4SA-X1 4-channel DC Servo Amplifier - any version
- VME4ST 4-channel STP Servo Amplifier - any version
- MACCON - MAC4- INC motion controller incremental encoder - version 4.1A or 4.2
- MACCON - MAC4- SSI motion controller SSI encoder - version 4.1A or 4.2
- MACCON - MAC4- STP motion controller STP encoder - version 2.0A or 2.1
- VMIVME-5576 Reflective Memory Board
- ESD ISER8 - 8 port RS232 serial interface
- NAT NET01 - Ethernet board
- BI016 - transputer interface
- Heidenhain IK320 Encoder, firmware: “246 118 06”

2.2.2 Software

UNIX Operating System:

SUN: Solaris 2.6 for CCSlite and NOCCS. No CCS(RTAP) version for Solaris.

HP: HP-UX B.10.20 for Full CCS (+ RTAP)

HP-UX B.11.00 for CCSlite and NOCCS. No CCS(RTAP) version for HP11.

Remark: we do not distribute by default both the HP-UX operative systems. By default HP10.20 will be delivered unless the consortium makes explicitly request for HP11 instead, or both.

Graphical Libraries: X11R5 and Motif1.2.

For both SUN and HP, these libraries are an add-on to the basic OS. You are required to install the appropriate extensions in order to be able to compile and link graphical applications:

SUN: Motif

HP: “C-Development bundle” (includes c89-compiler, X11, Motif, etc.)

RTAP: (required only if you intend to install Full CCS on HP10.20)

HP-UX 10: RTAP 6.7

VxWorks:

- TORNADO 1.0.1 (VxWorks 5.3) on HP-UX 10.20 and 11
- TORNADO 2.0 (VxWorks 5.4) on SunOS

MIDAS (only if INS/pco installation is done):

- 01FEB patch level 1.1

2.3 CONTENTS

The VLT Common Software is distributed as:

- one CD containing:
  a. the software. (For convenience, the CD also contains some public domain software used in the generation of the VLT Software)
  b. the documentation kit
- an additional CD containing VxWorks (5.3 or 5.4) is distributed on request

2.3.1 Software

1. PUBLIC DOMAIN SOFTWARE
   a. GNU C-compiler (2.95.2)
   b. GNU binutils (2.9.1)
   c. GNU Make (3.75)
   d. GNU awk (3.0.4)
   e. GNU gdb (4.17)
   f. GNU flex (2.5.4)
   g. GNU bison (1.27)
   h. GNU gzip (1.2.4)
   i. GNU texinfo (3.12)
   j. GNU emacs (20.6)
   k. GNU sed (3.02)
   l. GNU diffutils (2.7)
   m. GNU rcs (5.7)

n. GNU cvs (1.10)
o. GNU ddd (3.2)
p. GNU zip (2.3)
q. GNU unzip (5.32)
r. GNU tar (1.13)
s. Tcl 8.0.5
t. Tk 8.0.5
u. [incr Tcl] 3.0.1
v. [incr Tk] 3.0.1
w. iwidgets 3.0.1
x. tclX 8.0.4
y. TkX 8.0.4
z. BLT 2.4u
aa. msqltcl 1.99
ab. img 1.2.3
ac. Tktable 2.5
ad. snack 1.7.0
ae. tclCheck 1.1.13
af. expect 5.28
ag. rman 3.0.7
ah. groff 1.15
ai. tkman 2.1b2
aj. tkinspect 5.1.6
ak. Sybtcl (HP-UX only)
al. pgplot

2. VLT SOFTWARE
(the version number of each module is listed in <VLTSW>/INSTALL/buildFromArchive)

UNIX:

a. VLT Utilities
installation scripts: (vltsw)
Kit/VLT Makefile and Man-page browser (vlt)
document development support toolkit (doc)
software module handling utilities (mod)
utilities to check files for potential compatibility problems (compat)
code and document templates (templates)
standard environment (stdEnv)
compatibility checker (compat)
emacs customization kit (emacs)
code management module (cmm)

tool for automatic test (tat)
workpage time account (wp) (FOR ESO INSTALLATION ONLY)

b. Queue Server Emulator (qsemu)

c. Central Common Software:

CCS/Ccs
CCS/alrm
CCS/book
CCS/cai
CCS/ccs
CCS/ccsei
CCS/cmd
CCS/db
CCS/dbl
CCS/dblcb
CCS/eccs
CCS/envs
CCS/err
CCS/errch
CCS/evh
CCS/evhEt
CCS/evt
CCS/fftw
CCS/fnd
CCS/his
CCS/log
CCS/msg
CCS/ntp
CCS/plot
CCS/samp
CCS/scan
CCS/tims
CCS/vcc
CCS/vccmake
PANEL/fedit
PANEL/panel
PANEL/ptlib
PANEL/uif

d. HOS (High-level Operaton Software)

sequencer (seq)
brooker for observing blocks (bob)
access configuration and control (acc)

template server (tplsrv)

e. INS Common Software - File handling
   - SLX / miscellaneous utilities (misc)
   - SLX / setup file handling (slx)
   - SLX / setup file handling - C++ version (oslx)
   - SLX / setup file handling - tcl interface (slxtcl)
   - SLX / file handling tool - user interface (fht)
   - INS / data transfer module. (dxf)
   - INS / ins system tools (ist)
   - INS / protocol converter. (pco)
   - INS / common templates (insc)
   - INS / VLT OnLine Archive interface (volac)

f. ICB INS Common Base ICS (auto, ctoo, egen, ic0, ic0dev, ic0dig, ic0lcu, ic0mot, ic0sen, icb, lccdev, lcctoo)

g. CCD Control Software

h. OSB (boss, ibac, ixac, osb)

i. FIERA CCD Detector ontrol Software

j. IRD Infrared Detector Control Software - NEW -

k. TCS (Telescope Control Software)
   - TCS / Star Catalogue interface (catif)
   - TCS simulation kit (agws, catif, m2com, msw, prs, tcs, tcssim, tif, trkws)
   - TPOINT Telescope pointing model

l. Real-Time Display (DMD / rtd) and catalog lirary (DMD / catlib)

m. DICB FITS keyword Dictionaries (dicDPR, dicOBS, dicPAF, dicTPL) - NEW -

n. an example of a WS application (examples / wsapp)

VxWorks:

o. LCU Common Software
   - LCC library (lcc)
   - LCC engineering interface (lccei)
   - Command Interpreter templates (citmp)
   - Command Interpreter tools (too)

p. LCU QServer (lqs)

q. Drivers
   - driver log utility (lculog)
   - driver common utilities (lcudrv)
   - empty driver skeleton (xxx)
   - MACCON driver (mcon)
   - Analog I / O driver (aio)
   - ACROMAG Digital I / O driver (acro)
   - Servo Amplifier driver (ampl)

NET01 Ethernet I/F Board driver (net01)
Time Board driver (tim)
Heidenhein IK320 Encoder driver (ikon)
Serial I/F driver (iser)
Reflective Memory (rmn)
LCU configuration tool (lcuboot)
LCU status monitoring tool (lcustat, lcuwd)
driver engineering interface basic tool (inducer)
acro engineering interface (acrox)
aio engineering interface (aiox)
ikon engineering interface (ikonx)
Time Board engineering interface (timx)

r. Motion Control
   Motion Controller wrapper (mcm)
   Motion Controller SDL (mac4)
   Motor Control Module API (mot)
   Motor Command Interface (motci)
   Motor Engineering Interface (motei)
   SDL Common Interface (sdl)
   Servo Amplifier SDL (vme4sa)

s. an example of an LCU application (lexamples/cuapp)

2.3.2 Documentation

The documentation is organized in the following volumes

1a&b: CCS - workstation part
2a: LCU Common Software and Motor Library
2b: LCU Drivers
3: HOS & TCS
4: CCD
5: INS

See the table of contents, delivered as part of the kit with the name TOC, for the actual list of documents.

Documents are provided:

- as paper copy **only on motivated request**
  To people who have already received the previous versions, only new or changed documents are shipped as well as instructions on how to merge the new set into the existing one.
- as a tar compressed file DOC.tar.Z in the VLTSW CD that contains all the documents in ps format (most of them are also in pdf format)
- on the web:
  http://www.eso.org/projects/vlt/sw-dev/wwwdoc/MAR2001/dockit.html

2.4 BACKWARD COMPATIBILITY

The software included in the present delivery is backward compatible with the previous versions that have been already declared as baseline. For what is concerning software that appear for the first time with this release, please refere to the appropriate User Manual to see whether is released as a baseline version (i.e., backward compatible in future versions) or still in a preliminary form (i.e., it may change in future versions)

In any case, after the installation of the current release, any existing software must be regenerated.

The “Release notes” points out the differences between the current version and the previous one. In addition the “compat” utilities is provided to help in locating the lines of code to be changed (see 3.10 for more), if any.

2.5 NEW DEFAULT SHELL

Starting with the MAR2001 release, the default shell is bash. We changed to bash from tcsh for different technical reasons, the main one being that tcsh is not supported any longer on HP11.
We re-wrote also the standard environment, now called PECS (see section 3.6 and 6) in order to use bash. The general organisation of the environment, the installation procedure and setting the environment for a user have changed. The implications of this are:

- a new account is needed (called pecsmgr) that is the owner of some core PECS files;
- the users vltmgr and vlt must have the default shell set to /bin/bash;
- vx will remain to /bin/csh
- any other user must be defined with default shell /bin/bash
- the syntax in the bash shell is different (see 6).

2.6 PROBLEM REPORTING/CHANGE REQUEST

Please use the procedure explained in [6].

3 INSTALLATION

The purpose of this installation manual is to guide the process of installing or upgrading the VLT Software. Please read this document carefully and follow the instructions provided. This will help you save time and get a better understanding of the VLTSW. Please report errors, misprinting and suggestions to improve this document (see 4.7).

The installation procedure assumes that you start from scratch. If you have already installed a previous version of the VLT Software and required environment, you are familiar (and expert) enough to understand each installation pass and to act as appropriate to be compatible with your existing installation.

There are three possible ways of installing the VLTSW:

Full CCS
: is the default one. It uses the RTAP product as core engine for the Workstation application.

CCSlite
: uses a proprietary implementation of a subset of RTAP functionality.

NOCCS
: a minimum set of features, including panel editor, sequencer and *slx family.

According to your project, you have to select one of the three.

The installation has been divided into steps and allows selective installation according to your needs: The installation steps are:

1. preparing the Operating System.
2. downloading the tape.
3. verification / installation of pre-required software:
    a. X11 & Motif
    b. GNU utilities
    c. creation of VLTROOT / VLTDATA areas
    d. Tcl / Tk
    e. VxWorks.
    f. RTAP
    g. JAVA
    h. MIDAS
    i. verification of the environment.
4. installation of VLT Common Software:
    a. VLT development utilities.
    b. CCS
    c. LCC
    d. Panel Editor
    e. Sequencer
    f. installation of Motor Control

g. (optional) installation of INS Common Software
h. (optional) installation of ICS Common Software
i. (optional) installation of Telescope Control Software
j. (optional) installation of CCD Control Software
k. (optional) installation of FIERA Control Software
l. (optional) installation of Infrared Control Software

You can choose between an automatic procedure that, according to the current environment, installs everything that can be installed, or to go through each step of the installation procedure.

Each step requires that the appropriate previous ones have been completed successfully and has a simple verification test at the end. Section 4 provides an overall verification test. The verification test is part of the installation and provides a tutorial on how to configure both WS and LCU: you should execute it. Section 5 provides a trouble-shooting guide.

Actually, the installation procedure is a wrapper built around the installation procedure of each module that is part of the VLT Software. To have more details or to work selectively on a specific module, please refer to the section INSTALLATION GUIDE in the appropriate User Manual. The INSTALLATION section of each User Manual has been written to allow the installation of the module as a stand-alone action. For this reason, those INSTALLATION sections may contain redundant information with respect to the present section, where many modules are installed at the same time and by semi-automatic procedures. In addition, although correct as concerns the description of the behavior of the module, a User Manual may contain out-of-date information as concerns the installation because the installation philosophy has been recently improved (standardization). In the event of contradiction between the module User Manual and the present document, the second takes priority.

REMARKS:

- each installation procedure produces in the INSTALL directory two log files:

  `buildXxxxOutput.`*hostname* the detailed output produced by make,

  `buildXxxxError.`*hostname* possible errors occurring during the compilation.

  The utility `tail -f` can be used to monitor the contents of such files during installation.
- during installation, directories are added to the PATH list and files are installed in directories that are in the PATH. Be aware that:

  `hash -r` shall be issued to make new commands “known” to the system.

  `which` may be incorrect because it is unaware of any path changes that have occurred in the current shell session (see `which`(1)).

3.1 INSTALLATION REQUIREMENTS

3.1.1 Hardware Requirements

Required hardware as defined in 2.2.1. Additional DAT tape drive to load the tape.

Disk space (WS):

- to download tape and generate the VLT Software (including the public domain software): 600 MBytes. This value can increase up to 1000 MBytes during generation process.
- to use the VLT Software: minimum 350 MBytes (VLTROOT Full CCS; for CCSLite: 250 MBytes) and 600 MBytes for the utilities (gnu, tcltk, VxWorks etc).

We suggest you to keep on-line the VLTSW sources (300 MBytes), so you can access them any time as examples.

3.2 UPGRADING THE VLTSW FROM "NOV2000"

If you are installing from scratch on either an HP or a SUN or if you are upgrading a machine that is running a version earlier than NOV2000 skip this section.

If you have a correctly installed NOV2000 you can do an UPGRADE:

1. stop any VLT process that may be running (qsemu, RTAP / CCS environments, etc.).
2. backup the current installation, both code (`/vlt`) and data (`/vltdata`),
3. remove the current installation of the VLTSW.
4. execute all the remaining steps as for an installation from scratch. For each step is indicated under the label UPGRADE any additional information concerning an upgrade from NOV2000. If there is no special information, do the step. If you are doing an upgrade and in the section there is a comment UPDATE, please be sure to read it ***BEFORE*** doing the actions described in the section.

Section 3.10 explain how to rubuild the existing configuration after the installation of the new VLT Software.

REMARK: upgrading from OCT99 or FEB2000 requires a re-installation of the operative system. Upgrading from NOV2000 requires *on HP11 only* the installation of new patches. Please, refer to [8] for any further information about the operative system.

3.3 PREPARING THE OPERATING SYSTEM

REMARK: if you are installing a FIERA SLCU, refer first to the manual in [9].

3.3.1 Installing the OS

The WS must run one of the UNIX operating systems listed in section 2.2.2 and must be installed according to the provided installation procedure. Use [8] for anything concerning the Operative System installation (HP or SUN).

3.3.2 Add Users and Groups

All users shall belong to the same group, called "vlt" with group-id 300; so add this new group using as root the command "`groupadd`". The following entry:

```
vlt::300:
```

should be in the `/etc/group` file.

The following users, belonging to the same group as any other developer, shall be defined at WS level (if possible, please use the same UID and GID, at Paranal the one listed below will be used) :

`vltmgr`
used to download the tape and to perform the installation. Be sure that this user can have access to the amount of disk space as required by the installation (3.1). Such a user should be the only one authorized to make a new installation of any VLT software in the VLT root area. This user can be also used to perform the verification process. In alternative, any user of the same group of `vltmgr` can be used. If there are several machines, it is suggested to have `vltmgr` only in one machine, automounted from the others. Example:

```
vltmgr:<your_password>:450:300:VLT  Manager,,,:/home/vltmgr[1]:/bin/bash
```

`vlt`
used to run the default environment and the msqld daemon at boot time and the logging system clean-up procedure. This user must be local to each machine and having `/vlt/vlt` as `$HOME`. Example:

```
vlt:<your_password>:2684:300:VLT User:/vlt/vlt:/bin/bash
```

`vx`
used by LCU to access VxWorks kernel at boot time. It shall contain only a `.rhosts` file that allows access to LCU node(s). This user must be local to each machine and having `/vlt/vx` as `$HOME`. Example:

```
vx:<your_password>:138:300:Vx,,,:/vlt/vx:/bin/csh
```

`pecsmgr`
his home directory (that must be /etc/pecs) is the repository for everything conserning the standard environment (see 3.6). Example:

```
pecsmgr:<your_password>:3262:300:PECS Sw Mgr:/etc/pecs:/bin/bash
```

`rtap`
it is the RTAP/PLUS administrator. For backward compatibility, create it on a CCSLite machine too. Example:

```
rtap:<your_password>:300:100:RTAP/Plus Admin:/opt/rtap/A.06.70:/bin/csh
```

REMARK: To create these users, it is recommended to use as root the command "`useradd`". In this way, the home directory will be automatically created and all the necessary files (/etc/passwd and, on SUN, /etc/shadow) will be automatically and correctly upgraded.

UPGRADE: define only the pecsmgr user. You should already have the other users.

3.3.3 Define Disk Areas

The VLTSW is using three different areas:

`/vlt` to store binaries (products & VLT SW)

`/vltdata` to store ENVIRONMENTS related data

`/data` to store observation data

The three areas are implemented as separate logical volumes (on the same or on different physical

1. On FIERA machines, the convention is to create the home directory of vltmgr as /export/home/vltmgr

disks). The layout used for operational machines is standardised as follows:

```
/vlt      2500 MB
/vltdata  1000 MB
/data      500 MB (or bigger depending on the available disk(s))
```

Depending on the available disks, development computers may have different layouts. Directory names are mandatory. As an alternative to a separate logical volume, you can use a symbolic link to a directory somewhere, for instance:

```
$ mkdir <physicalDir>
$ ln -s <physicalDir> /vlt
```

Depending on your current configuration, in the following steps you may need to become “root” to add / change directories.

The ownership of the three areas shall be as follows:

```
/vlt      drwxrwxr-x  vltmgr  vlt
/vltdata  drwxrwxr-x  vltmgr  vlt
/data     drwxrwxr-x  vltmgr  vlt
```

REMARKS[1]*: the VLT directories will be NFS-mounted by the LCUs. Remember to declare* `/vlt, /vltdata, /data` *as exportable from the host. Exported file systems are controlled by:*

- `/etc/exports` *on HP (use SAM to change it)*
- `/etc/dfs/dfstab` *on SUN*

see your UNIX documentation and/or the man-pages for more. In general, after having changed any of those files, run as root:

exportfs -av *on HP*

or

`shareall` *on SUN*

REMEMBER: to use the physical name of the directories (for instance: /diskb/vlt) and not to mount directory that are already mounted from another system.

UPGRADE: you should have already these directories, but with different size. Please refer to [8] for the correct size for the machine you have. Use the standard HP-UX - SUN tools to change the size.

3.4 DOWNLOADING THE SOFTWARE FROM THE CD

The VLT Software is delivered as a compressed Unix tar file on a CD labeled VLTSW-MAR2001.
In what follows, we suppose that the cdrom is mounted under the directory /cdrom.
Where you have the required disk space to generate the VLT Software, untar the sources (in the rest of the document we assume that ~vltmgr is where the content of the tape is available).

1. login as `vltmgr`
2. $ mkdir VLTSW
3. `$ cd /home/vltmgr/VLTSW`

1. These remarks apply to Full-CCS and CCS-lite installations only.

4. `$ zcat /cdrom/VLTSW.tar.Z | tar xf -`

   You will get the following subdirectories:

`./CCD/`	CCD Control Software
`./CCS/`	WS Common Software
`./COPYING`	copyright notice
`./DICB/`	`Data Dictionaries`
`./DMD/`	`RTD & CATLIB`
`./Drivers/`	available drivers
`./FCD/`	FIERA CCD
`./HOS/`	HOS/Sequencer
`./ICB/`	`INS Base ICS`
`./INS/`	INS Common Software
`./IRD/`	Infrared Detector Software
`./Kit/`	VLT development utilities
`./LCC/`	LCU Common Software
`./Motor/`	Motor Control Module
./OSB/	Observation Software base modules
`./PANEL/`	Panel editor
`./PRODUCTS/`	GNU tools
`./Qserver/`	WS-LCU Communication Software
`./RTAPpatches/`	RTAP patches
`./SLX/`	Setup file tool
`./TCS/`	Telescope Control Software
`./TPOINT/`	Telescope Pointing Model
`./examples/`	examples of applications
`./tcltk/`	tcl/tk and extensions

IMPORTANT: GET THE LATEST NEWS ABOUT THE INSTALLATION!

Doing an installation manual absolutely error free and tested in all possible configurations is pratically impossible, but, when discovered, there is no point to repeat the same error in every site the VLT Software is going to be installed. To ease the installation process, from JUN96 onward, via the ESO WWW, the latest report about errors, installation problems, tips, etc is available on-line. Before starting the installation, please read the URL:

`http://www.eso.org/vlt/sw-dev/vltsw`

If you do not have access to it, you can still request the latest news by e-mailing/faxing/calling at the addresses given in [6] for reporting problems.

3.5 GET THE INSTALL PROCEDURE

Before starting, be sure that "/usr/ccs/bin" is in the PATH. For these preliminary steps, we are going to use the native make, that is normally under that directory.

Installation procedures are delivered as part of the vltsw module, but for convenience a temporary copy is done from these original files:

```
$ cd ~/VLTSW/Kit/vltsw/src
$ make -f Makefile.boot \
TAPE_INSTALL=$HOME/VLTSW/INSTALL prepare_installation
```

3.6 SET THE ENVIRONMENT

3.6.1 PECS

The standard environment consists of a set of environment variables (PATH, MANPATH, etc). To be sure that the environment is always properly defined, a standard set of files is provided.

To install it, we prepared a simple procedure that **must be run as root**:

```
# cd /home/vltmgr/VLTSW/PRODUCTS/pecs/install
# ./buildPECS
```

Now you have the basic standard environment with all the necessary variables.
This environment has to be customised. For this purpose a configuration file is supplied:

/etc/pecs/releases/000/etc/locality/apps-`hostname`.env

Now:

1. edit your configuration file according to the instruction given in the file itself
2. have a look also at the general configuration file

   /etc/pecs/releases/000/etc/locality/apps-all.env

   and edit it as appropriate; basically comment out the last two lines:

   #SOFTWARE_ROOTS=/usr/server

   #ACC_HOST=te13
3. for each of the users vltmgr and vlt:
   a. login as that user
   b. execute the following:

```
$ /etc/pecs/bin/pecssh mklinks -i
    PECS_ROOTDIR [/etc/pecs]: "return"
    PECS_RELEASE [000]: "return"
    [...]
    Do you wish to install VUE support files? [y]: "n"
```

   c. logout and login again (if you are using the CDE, exit and re-login)

PECS allows customization at user level, whenever the user wants to change any of the default settings.. For example, if you want to change the VLTROOT definition, edit the file

$HOME/.pecs/apps-all.env
The new VLTROOT will be valid for that user on all the machines he logs in (provided the home directory is exported on more than one machine). If you want to change the VLTROOT on a specified machine texx, put the new VLTROOT setting in the file:
$HOME/.pecs/apps-texx.env
More information can be obtained in Appendix A.

3.6.2 Define the VLT Root and VLT Data Areas

The VLT Root area is where binaries, includes, scripts, etc. of the VLT Common Software will be located. This area is normally indicated by the environment variable VLTROOT and it is located under the /vlt directory:

```
/vlt/<RELEASE>/CCS          for Full CSS installation
/vlt/<RELEASE>/CCSlite      for CSSlite  installation
/vlt/<RELEASE>/NOCCS        for NOCCS    installation
```

where <RELEASE> is a directory whose name should be e.g. FEB2000 if you are installing the FEB2000 Release and so on. This directory <RELEASE> is created under /vlt during the PECS installation.

The VLT data area is where the environments, configuration and temporary data are located. This area is normally indicated by the environment variable VLTDATA and it is located under:

```
/vltdata                    for all installation types
```

VLTROOT and VLTDATA shall be correctly defined in /etc/pecs/releases/000/etc/locality/apps-`'hostname'`.env.

Populate them, as vltmgr:

```
$ cd ~/VLTSW/INSTALL
$ buildVLTROOT
$ hash -r
```

3.7 INSTALL PRODUCTS

WARNING: (for existing installations) before any of the following installations, check that neither INTROOT is defined nor $INTROOT/bin is in the PATH

3.7.1 GNU tools

The VLT Software project has adopted the GNU tools as standard development tools (make, compiler, debugger, etc. See 2.3.1 for the complete list).

The installation of public domain software shall be done using the provided kits (in `VLTSW/PRODUCTS/gnu`).

The installation is performed as "`vltmgr`" and according to the standard scheme: all the required libraries and scripts in directories under a dedicated directory defined by the GNU_ROOT environ-

ment variable. /usr/local is not touched.

The standard environment provides the following definitions::

```
$ export GNU_ROOT=/vlt/<RELEASE>/gnu
$ echo $PATH        ->    ${GNU_ROOT}/bin:${PATH}
$ echo $MANPATH     ->    ${GNU_ROOT}/man:${MANPATH}
$ echo $LPATH       ->    ${GNU_ROOT}/lib:${LPATH}          (HP only)
$ echo $SHLIB_PATH  ->    ${GNU_ROOT}/lib:${SHLIB_PATH}     (HP only)
$ echo $LD_LIBRARY_PATH  ->    ${GNU_ROOT}/lib:${LD_LIBRARY_PATH}(SUN
only)
```

Before starting, be sure that "." is in the PATH and /usr/local exists with the ownerships:

drwxr-xr-x 8 root root 1024 Dec 22 1999 /usr/local/

To install:

```
$ cd ~/VLTSW/INSTALL/
$ buildGNU > buildGNU.log 2>&1
```

In case of errors or for more detailed information, please refer to the INSTALL file of each package. To save disk space, after installation you can remove the directory ~/VLTSW/PRODUCTS/gnu/<OS>.

A simple test is performed by the procedure itself at the end of the installation.
UPDATE: remove current /vlt/gnu and regenerate under /vlt/<RELEASE>/gnu.

3.7.2 Tcl/Tk

The basic Tcl/Tk and the extensions are required by several components of the VLT Software.

For your convenience, the original tar-files of the products listed in 2.3.1 have been included in the tape directory ./tcltk. We preferred to include the distribution kits in their original format so that you may change/adapt according to your needs.

The installation is performed as "vltmgr" and according to the standard scheme: all the required libraries and scripts in directories under a dedicated directory defined by the TCLTK_ROOT environment variable. /usr/local is not touched.

The standard environment provides the following definitions::

```
$ TCLTK_ROOT=/vlt/<RELEASE>/tcltk
$ echo $PATH             ->  ${TCLTK_ROOT}/bin:${PATH}
$ echo $MANPATH          ->  ${TCLTK_ROOT}/man:${MANPATH}
$ echo $LPATH            ->  ${TCLTK_ROOT}/lib:${LPATH}          (HP)
$ echo $SHLIB_PATH       ->  ${TCLTK_ROOT}/lib:${SHLIB_PATH}     (HP)
$     echo     $LD_LIBRARY_PATH          ->          ${TCLTK_ROOT}/
lib:${LD_LIBRARY_PATH}(SUN)
```

Before starting, be sure that "." is in the PATH , $DISPLAY is defined and the application can access the graphic server (xhost, authorize), if not verification will fail!

To install:

```
$ cd ~/VLTSW/INSTALL/
$ buildTcltk > buildTcltk.log 2>&1
```

REMARK: In case of errors or for more detailed information, please refer to the INSTALL file of each package.

A simple test is performed by the procedure itself at the end of the installation.

To save disk space, after installation you can remove the directory `~/VLTSW/tcltk/<OS>`.

UPDATE: remove current $TCLTK_ROOT directory and regenerate.

3.7.3 VxWorks-TORNADO

If you want to generate the LCU software, a fully installed VxWorks (also called TORNADO) environment is required.

The procedure currently used at ESO to install and configure VxWorks for the VLTSW is available in `~/VLTSW/doc/install_<Release>_Tornado.ascii`

TORNADO is a licensed product of WindRiver. During installation, you will be asked to provide some information for ESO to be able to supply the appropriate license. In doubt, please verify with your official ESO contact person.

Remember that the VxWorks directory shall be read-accessible by the LCU, i.e., the file downloaded at boot time is readable by the username (`vx`) used by the LCU to access the host system.

To use VxWorks, each user needs the environment variable setup as described in the installation procedure and provided by the standard environment.

UPDATE: remove current $WIND_BASE directory and regenerate.

3.7.4 RTAP

Full CCS requires RTAP[1]:

HP-10.20: version 6.7

RTAP is a licensed product of Hewlett-Packard. Before installing/using it, be sure you have the proper licenses. In doubt, please verify with your official ESO contact person.

The procedure currently used at ESO to install and configure RTAP for the VLTSW is available in `~/VLTSW/doc/install_<Release>_RTAP.ascii`

1. RTAP is available on HP10.20 only. HP11.00 and SUN use CCSLite and NOCCS only.

3.7.5 OS configuration for CCSlite

A CCSlite installation requires a customization of kernel parameters.

Install according to `~VLTSW/doc/install_<Release>_CCSlite.ascii`

3.7.6 MIDAS

The MIDAS distribution is included in the tape under `~/VLTSW/PRODUCTS/midas`. Before using MIDAS please check the license conditions: you may need to sign an official agreement with ESO.

The procedure currently used at ESO to install and configure MIDAS for the VLTSW is available in `~/VLTSW/doc/install_<Release>_MIDAS.ascii`

3.7.7 JAVA

A binary distribution of jdk 1.1.8 and SWING 1.1 is included in the tape under `~/VLTSW/PRODUCTS/java`.

The procedure currently used at ESO to install and configure JAVA for the VLTSW is available in `~/VLTSW/doc/install_<Release>_Java.ascii`

3.8 ENVIRONMENT VERIFICATION

The following steps assume that your environment is properly set. Unpredictable results may arise if not.

As said, you should have already recorded in your shell startup file (or, better, in a file you source at any login) all the environment definitions needed in the previous steps. To check this:

1. logout the current session

2. login again (if not done by `.cshrc`, source your file)

3. check that all environment variables you need (VLTROOT, VLTDATA, PATH, MANPATH, VxWorks environment variables, etc.) are properly set, i.e., they contain the path to access GNU, tcl/tk, VLT, etc.

4. check that the right version of the GNU tools are accessible:

   ```
   $ make -version
   ```
   GNU Make version 3.75, . . .

   ```
   $ gcc -v
   ```
   Reading specs from ....
   gcc version 2.95.2

5. (for existing installations) check that neither INTROOT is defined nor $INTROOT/bin is in the PATH

6. check that DISPLAY variable is defined and your server can display windows (e.g.: xhost +)

3.9 VLT COMMON SOFTWARE INSTALLATION

The whole installation can be performed by only one command. The prerequisite is that the environment has been properly configured and all the needed tools are available. The installation procedure uses NOCCS, WIND_BASE and RTAPROOT to decide what to to, according to the following table:

	NOCCS (1)	CCS-lite-WS	CCS-lite	CCS-WS	Full CCS
WIND_BASE	ignored	not defined	defined	not defined	defined
RTAPROOT	ignored	not defined	not defined	defined	defined
./buildVLTROOT	X	X	X	X	X
./buildKit	X	X	X	X	X
./buildDMD	X	X	X	X	X
./buildPanel	X	X	X	X	X
./buildCCSinclude		X	X	X	X
./buildDrivers			X		X
./buildQserver		X	X	X	X
./buildLCC		X	X	X	X
./buildCCS++		X	X	X	X
./buildSlx	X	X	X	X	X
./buildCCS		X	X	X	X
./buildHOS	X	X	X	X	X
./buildDriversEi		X	X	X	X
./buildTools	X	X	X	X	X
./buildMotor		X	X	X	X
./buildLSF		X	X	X	X
./buildTPOINT(4)		X	X	X	X
./buildTcssim(4)(8)		X	X	X	X
./buildCatif(4)(8)		X	X	X	X
./buildDICB(4)(8)		X	X	X	X
./buildICB(3)		X	X	X	X
./buildINS(2)		X	X	X	X
./buildOSB				X	X
./buildCCD(5)		X	X	X	X
./buildFCD(6)		X	X	X	X
./buildIRD(7)	X	X	X	X	X
./buildExamples	X	X	X	X	X

In alternative to the automatic procedure, you can execute each step independently. This may be convenient if you need to save space, by default the whole software is built, or if you want to repeat

only one specific step. Each build procedure is individually described in the next section.

The automatic build procedure can be forced to skip some of the software by defining (to any value) one or more of the following environment variables:

(1) NOCCS
to get a NOCCS installation this variable MUST be defined! For instance: `setenv NOCCS on`

(2) NOINS
the Instrumentation Common Software is not build

(3) NOICB
the Base ICS Software is not build (only if NOTCS also is defined)

(4) NOTCS
the Telescope Control Simulation Software is not build

(5) NOCCD
the CCD Detector Control Software is not build

(6) NOFCD
the FIERA Detector Control Software is not build

(7) NOIRD
the InfraRed Detector Software is not build

(8) this build is run on HP only

If needed, such variables must be defined as accessible to sub-shells (i.e., use setenv or export) before running the installation procedure. Any string is a valid value, for instance:

```
$ export NOICB=abc
```

forces `build` to skip `buildICB`.

REMARK: by default, the VLTSW LCU code is built for both types of supported CPU: Motorola 68k and Power PC. If you use only one type of CPU and want to save time and space, you can set accordingly the following variables:
NOPPC=on if you don't want to build the LCU code for Power PC LCUs;
NO68K=on if you don't want to build the LCU code for Motorola 68k LCUs.

3.9.1 Automatic Installation

To perform the automatic installation:

```
$ cd ~/VLTSW/INSTALL
$ ./build > build.log 2>&1
```

From another xterm you can spy the building process using:

```
$ tail -f ~/VLTSW/INSTALL/build.log
```

Unless you are doing a NOCCS installation, the build procedure will ask you to execute some "change" command as root. Execute as required.

For a Full CCS installation:

```
# chown root /vlt/<RELEASE>/CCS/bin/msgServer1
# chmod u+s  /vlt/<RELEASE>/CCS/bin/msgServer1
```

For a CCSLite installation:

```
# chown root /vlt/<RELEASE>/CCS/bin/msgServer1
# chmod u+s  /vlt/<RELEASE>/CCS/bin/msgServer1
# chown root /vlt/<RELEASE>/CCS/bin/ccsScheduler
# chmod u+s  /vlt/<RELEASE>/CCS/bin/ccsScheduler
```

To save time, the build[Xxxxx] procedures do not execute the make clean. An additional procedure (buildClean) is provided to:

- clean up after a successful installation in order to save space
- clean up after a failure or a change of configuration (for instance defining/undefining RTAPROOT)

To execute the clean-up:

```
$ cd ~/VLTSW/INSTALL
$ ./buildClean
```

For a preliminary verification, execute some of the verification test according to the steps that have been executed (see table), as described in the next section. A more complete verification is described in Section 4.

If you had already a VLT Software installation do not forget to restore the previous configuration (see 3.10).

3.9.2 Step-by-step Installation

In alternative to the automatic installation, each installation step can be individually executed using:

```
$ cd ~/VLTSW/INSTALL
$ ./build<xxxxx>
```

The table in 3.9 provides the list of build<xxxxx> scripts that must be executed. Follow the order of the table, because there could be dependencies among the different builds. In particular, if you want to run the buildCCS, run always before that the buildCCSinclude.

3.9.3 Quick Verification

In this section, few simple utilities are executed to have a first check on the installed code. A complete verification and validation procedure is described in chapter 4.

To check installation and the environment setup (VLTROOT, PATH, etc.) try the man-page browser:

```
$ vltMan
```

On the new window (TkMan):

click “OK” to dismiss message about new features
set “Mono” ON (located on the lower-right corner)
click “Volumes” (upper-right)
click “(5) File Formats”
single-click `vltDirectoryStructure` to display the man-page
click “Quit” (lower right)

In the same way you can access the man-pages of all the installed software.

Sequencer is available on all type of installation:

- start up the Sequencer shell:

```
$ seqWish
   ... a new window (Main Console TkCon) appears showing
   MainConsole display active
   (<username>)1 %
```

- get the tcl version and the list of [incr Tcl] commands:

```
seqWish> info tclversion
8.0
seqWish> info command itcl*
itcl_info itcl_class
```

- except get the list of available “sequencer” commands:

```
seqWish> info command seq*
```

  - NOCCS:
  - `seq_logData seq_findFile seqIcon seq_fitsDate seq_errCloseStack seq_errResetStack seq_isoTimeToClock seq_isoTime seq_errAdd seq_relToAbsPath seq_timeOfDay`
  - CCSlite:

    `seq_ccsExit seq_ccsExitOrig seq_logData seq_evtSingleDisable seq_findFile seq_dbListPut seq_errGetStackSize seq_msgSendReply seq_dbGetAttrNames seq_ccsInit seq_errGetFromStack seq_dbDirAddrToName seq_dbListDestroy seq_dbMultiWrite seq_msgDispatch seq_waitTillActive seq_dbGetAlias seq_evtAttach seq_dbGetFamilyNames seq_dbListList seq_msgRecvReply seq_dbGetDirAddr seq_dbMultiRead seqIcon seq_fitsDate seq_errCloseStack seq_errPrint seq_evtSingleEnable seq_evtList seq_msgDispatchBreak seq_dbListCreate seq_dbGetFieldNames seq_dbListRemove seq_dbLockPoint seq_errResetStack seq_dbUnlockPoint seq_msgCheck seq_deleteHandle seq_dbWriteSymbolic seq_dbListAdd seq_isoTimeToClock seq_isoTime seq_errAdd seq_dbSetCwp seq_dbReadSymbolic seq_ccsAsyncInput seq_dbGetCwp seq_dbGetAttrInfo seq_msgSendCommand seq_dbListExtract seq_waitTillDead seq_relToAbsPath seq_timeOfDay seq_evtDetach seq_msgList`
  - CCS:

```
seq_ccsExit  seq_ccsExitOrig  seq_logData  seq_evtSingleDisable
seq_findFile seq_dbListPut seq_errGetStackSize seq_msgSendReply
seq_dbGetAttrNames        seq_ccsInit        seq_errGetFromStack
seq_dbDirAddrToName      seq_dbListDestroy      seq_dbMultiWrite
seq_msgDispatch seq_waitTillActive seq_dbGetAlias seq_evtAttach
seq_dbGetFamilyNames       seq_dbListList       seq_msgRecvReply
seq_dbGetDirAddr     seq_dbMultiRead     seqIcon     seq_fitsDate
seq_errCloseStack seq_errPrint seq_evtSingleEnable seq_evtList
seq_msgDispatchBreak     seq_dbListCreate     seq_dbGetFieldNames
seq_dbListRemove        seq_dbLockPoint        seq_errResetStack
seq_dbUnlockPoint          seq_msgCheck          seq_deleteHandle
seq_dbWriteSymbolic seq_dbListAdd seq_isoTimeToClock seq_isoTime
seq_errAdd  seq_dbSetCwp  seq_dbReadSymbolic  seq_ccsAsyncInput
seq_dbGetCwp        seq_dbGetAttrInfo        seq_msgSendCommand
seq_dbListExtract seq_waitTillDead seq_relToAbsPath seq_timeOfDay
seq_evtDetach seq_msgList
```

- exit the shell:

```
seqWish> exit
```

The panel editor and the UIF library are also present in all installation:

```
$ panel
```

The “PanelEditor” window should appear. Ignore warnings concerning ccsInit failing because the environment is not yet active. Click “File” “Quit” “Yes” to close it

This complete this first quick test of your installation. The complete procedure is described in chapter 4.

3.9.4 Whatis database creation

After the installation (from scratch or upgrade), run as vltmgr:

On HP-UX:

```
$ su root -c \
    “MANPATH=$VLTROOT/man:$TCLTK_ROOT/man:$GNU_ROOT/man:$MANPATH \
     /usr/sbin/catman -w”
```

On SUN:

```
$ su root -c \
    “MANPATH=$VLTROOT/man:$TCLTK_ROOT/man:$GNU_ROOT/man:$MANPATH \
     /bin/catman -w”
```

3.10 RESTORING THE PREVIOUS CONFIGURATION

If you were already using any previous VLT software version, please:

1. remove any existing INTROOTs and recreate their content from scratch. DO NOT MIX OLD AND NEW CODE!!!!

2. Check the existing application before regenerating them. The “Release notes” points out the differences between this version and the previous one.

   The `compat` utility can help in locating the lines of code that may need to be changed. At the present time, there is not a compatibily check for NOV2000. The latest available keywords are OCT1999 and Y2000. It is also still valid the Tcltk check, with the keyword TclTk8. To use compat:

   ```
   $ cd <yourModule>
   $ compat <KEYWORD> ‘find . -print‘
   ```

   See `compat`(7) man-page for more.

3. regenerate your code.

4. regenerate the existing environments both WS(s) and LCU(s) using the configuration tools provided by the new version. If any, remember to add your database definition to the basic one.

At this point your existing applications should be able to run using the new version of the VLT Common Software.

4 VERIFICATION

If you are already familiar with the VLT Common Software, simply browse section 4.4 to get familiar with any new tool introduced by this version.

This section describes how to setup a simple application environment in order to exercise some, but the most important, features of the VLT Common Software. The purpose of this action is twofold:

- providing a way to verify the correctness of the installation you have just completed
- providing a simple tutorial on the use of the VLT Common Software.

The detailed information on how to configure each module is provided by the appropriate User Manual.

The scope of this section is limited to a WS and a LCU in a one-environment-per-node configuration. For the configuration and verification of Drivers and Motor Control and for more environments on a WS, please refer to the User Manuals.

The following sections assume that:

- you are familiar with configuring UNIX and, as appropriate, RTAP tools and/or VxWorks utilities and that node names, IP addresses, etc. are already correctly configured on your WS.
- you have one WS with Full CCS or CCSLite and one LCU and you set up an environment on each. In this chapter `<wsenv>` and `<lcuenv>` will be used to represent the name of the two environments.

Generally speaking, in order to verify the correct file setup, configuration activities should be done by a username that is not the one used for installation. The database is already created having a user called “ccsuser”: add it to your system and use it for all configuration activities described by the present chapter. When you are more familiar with CCS, LCC and RTAP or CCSLite, other users can be defined.

4.1 UNDERSTANDING ENVIRONMENTS

The VLTSW is based on the concept of environments where processes can run and exchange messages with other processes, either in the same environment or in a remote environment, on the same machine or in other machines.

Environment can be:

CCS environment (on WS)
: based on an RTAP environment, providing database and communication facilities (q-server), and used to build WS applications. There can be one or more such environments per WS.

CCS-Lite (or QSEMU) environment (on WS)
: a limited environment used when no RTAP is installed (CCS-Lite).

LCC environment (on LCU)
: provides database and communication facilities (lcu-Qserver) to build real-time applications. Maximum one environment per LCU.

An environment is uniquely identified by the environment name. Remember that environment names are limited to 7 chars and the first letter is mandatory “w” for WS and “l” for LCU.

REMARK: Real VLT environment names must follow the conventions defined by the applicable version of "VLT LAN's Specification". In the following example generic names built using the machine node name are used.

From the communication point of view, each environment is identified by the node on which is running and a TCP/IP port number. The same number can be used on different nodes for the same type of environment. Currently we use:

2160 for LCC environments,

one number in the range 2001-2999 for each CCS and QSEMU environment.

4.1.1 Configuring Environments

To be able to communicate, environments need to know about each other.

1. there is a TCP/IP channel (environment name and port number) for each environment. Actually, the couple node-number shall be unique in the system. So the same number can be used for all the LCC environments. TCP/IP channels are defined in the file `/etc/services`.
2. (for CCS-Lite only) each CCS-Lite (QSEMU) environment needs to know where other CCS, QSEMU, LCC environments are located, they can be both local or remote to the WS. `$VLTDATA/config/CcsEnvList` provides such a mapping
3. each LCU needs to know from which host it has to boot and which are the other environments: CCSs and QSEMUs (an LCU should not talk directly to other LCUs). This is done by the `$VLTDATA/ENVIRONMENTS/<lcuenv>/bootScript` file.
   If LCU(s) need to communicate with more WS environments than the assigned boot environment, then the file "`$VLTDATA/config/lqs.boot`" must be created and edited. The standard file in "`$VLTROOT/vw/bin/$CPU/lqs.boot`" can be taken as template.
4. each CCS (RTAP) environment needs to know where other CCS, QSEMU, LCC environments are located, they can be both local or remote to the WS. `/etc/$RTAPROOT/RtapEnvList` provides such a mapping.

In addition:

each CCS environment shall have a `$VLTDATA/ENVIRONMENTS/<wsenv>` for the database files (`dbl/ and DB/`), snapshots and database image (`Rtap...`) and process configuration table (`RtapEnvTable`)

each QSEMU environment shall have a `$VLTDATA/ENVIRONMENTS/<wsenv>` for the database files (`dbl/ and DB/`), snapshots and database image and process configuration table (`CcsEnvTable`)

each LCC environment shall have a `$VLTDATA/ENVIRONMENTS/<lcuenv>` for the database files (`dbl/ DB/`) and boot files (`bootScript`, `devicesFile`, ...)

In addition to the basic environment configuration, other configuration tables are required by applications like CCS Log System and CCS Scan System.

The verification process will drive you to a first set up of these files.

REMARKS:

the VLT Software Environments Common Configuration User Manual provides a detailed

documentation about environments and configuration tools. The panels layout as well as several examples of expected output are also given.

4.2 THE APPLICATION EXAMPLE

To have the possibility to work-out the CCS or CCSLite and LCC software, a simple example, consisting of an LCU application and of a WS application, is provided.

Each application is implemented as a VLT software module and also provides an example of use of the VLT Programming Standards. Please note:

- the standard subdirectory structure,
- the use of naming conventions,
- the standard Makefile,
- the documentation in form of man-pages,
- the command definition table (LCU only),
- the error files.

4.2.1 The LCU Application

The LCU application consists of one process (`lcuapp`) able to treat the following incoming commands:

SETVAL with one integer parameter (ASCII format):

IF 0 < value < 100

THEN

- the value is written in the `:PARAMS:SCALARS.scalar_int32` point of the LCU local database
- the message "`Value received = <value>`" is logged
- an empty reply is given back

ELSE

- the message "`Value received = <value> (out of range)`" is logged
- an error message "`Value out of range`" reply is given back

EXIT:

- an empty reply is given back
- the process quits

The implementation of the LCU application is in `~/VLTSW/example/lcuapp` and is generated during the installation.If needed, can be generated as follows:

```
$ cd ~/VLTSW/examples/lcuapp/src
$ make clean
$ make all
$ make man
$ make install
```

4.2.2 The WS Application

The WS application consists of two programs:

`wsappShowValue` that continuously displays an item from the local database until a “q” character is typed in.

`wsappSetValue` that works with a companion partner, named `lcuapp`, on another environment. It prompts the user for input:

IF an integer number is typed in:

- the number is formatted into a message (SETVAL) and sent to the partner
- the reply message from the partner closes the loop and prompts for a new input.

IF a “q” character is typed in

- an EXIT message is sent to the partner
- as the reply message from the partner comes back the process terminates as well.

The implementation of the WS application is in `~/VLTSW/example/wsapp` and is generated during the installation.If needed, can be generated as follows:

```
$ cd ~/VLTSW/examples/wsapp/src
$ make clean
$ make all
$ make man
$ make install
```

4.3 INSTALLING AND CONFIGURING ACC DATABASE

The ACC database contains all the information (IP address, machine type, node name, etc.) used by the configuration tools and it uses `mSQL` as database engine (see the ~/VLTSW/tcltk/`msql-<version>/doc/License` for copyright).

Except for the differences explicitly indicated, it is the same procedure for both HP and SUN. This installation is done as “`vltmgr`”. By default, “`vltmgr`” is also the username authorized to write the database. For more information, please refer to the ACC and `msql` documentation.

You need only one database running for all machines. The environment variable ACC_HOST tells the VLT Software the host where the database daemon is running. For every machine, including the one where the daemon runs, define:

```
$ export ACC_HOST=<host>
```

in the /etc/pecs/releases/000/etc/locality/apps-‘hostname‘.env and install the mSQL software on that `<host>` only.

The mSQL code is generated and installed in `$TCLTK_ROOT` during the `buildTcltk` step. Such a generation is enough to correctly generate the Sequencer. The following actions are needed to create and to start the database and to load it with the configuration data:

1. create user access list

```
$ cp $TCLTK_ROOT/include/msql.acl $VLTDATA/msql/msql.acl
```

(you should not need to edit this file. If you do it while the daemon is running, see below, use the utility `msqladmin reload` to make the changes effective)

2. Fix /etc/services (HP ONLY):

   HP-UX provides a default value for `msql` in `/etc/services`. This interferes with our installation.

   Please comment in `/etc/services` as follows:

```
.....
#msql        1111/tcp                # Mini SQL database server
.....
```

3. start the mSQL daemon:

```
$  msqld&
mSQL Server 1.0.16 starting ...
```

4. create acc database:

```
$ msqladmin create acc
Database “acc” created.
```

5. load the definition of the basic tables used by the VLT Configuration tools:

```
$ msql acc < $VLTROOT/config/accCreateVCCTables.sql
Welcome to the miniSQL monitor.  Type \h for help.
mSQL >     ->     ->     ->
Query OK.

.......

Bye!
```

6. check database structure:

```
$ msqlrelshow acc
Database = acc

  +---------------------+
  |       Table         |
  +---------------------+
  | prog_environment    |
  | station             |
  | subnet              |
```

```
  | lcu_progenv         |
  +---------------------+
$ msqlrelshow acc station

Database = acc

Table    = station

 +-----------------+----------+--------+----------+-----+
 |     Field       |   Type   | Length | Not Null | Key |
 +-----------------+----------+--------+----------+-----+
 | station_name    | char     | 30     | Y        | Y   |
 | station_type    | char     | 30     | N        | N   |
 | subnet_name     | char     | 30     | N        | N   |
 | ip_address      | char     | 20     | N        | N   |
 | active_status   | char     | 15     | N        | N   |
 | location        | char     | 80     | N        | N   |
 | responsible     | char     | 80     | N        | N   |
 | telephone       | char     | 80     | N        | N   |
 | email           | char     | 40     | N        | N   |
 | station_arch    | char     | 16     | N        | N   |
 | station_cpu     | char     | 16     | N        | N   |
 | station_bsp     | char     | 16     | N        | N   |
 | attic           | int      | 4      | N        | N   |
 +-----------------+----------+--------+----------+-----+

.......
```

7. create a plain ASCII data file. First get the templates

   ```
   $ cp $VLTROOT/config/accData.sql $VLTDATA/msql/
   ```

   then edit the file `$VLTDATA/msql/accData.sql` as explained by the comments. When ready load it:

   ```
   $ accLoadData
   ```

8. check the database content:

   ```
   $ msql acc
   Welcome to the miniSQL monitor.  Type \h for help.

   mSQL > select * from station\g
   Query OK.
   ```

```
.....

mSQL > \q
Bye!
```

Test the installation by quering the database from a VLT utility:

```
$ vccShow
--- VCC: VLT Common Configuration ---
... Hosts defined: ..........
... Environments defined: .....
........
 Env     Host   TCPport  IP address      Arch    CPU     BSP
------- ------- ------- --------------  ------- ------- -------
l...
```

Remember to include the definition of ACC_HOST among the environment variables defined at the login.

4.3.1 Automatic Startup of Database Daemon at boot Time

As “root” do the following steps to have the database daemon always active after a reboot of the machine:

```
- HP:
  # cp /home/vltmgr/VLTSW/PRODUCTS/pecs/templates/msql /sbin/init.d/msql
  # chmod 555 /sbin/init.d/msql
  # chown bin /sbin/init.d/msql
  # chgrp bin /sbin/init.d/msql

  # cd /sbin/rc3.d
  # ln -s /sbin/init.d/msql S910msql

  # cd /sbin/rc2.d
  # ln -s /sbin/init.d/msql K089msql

- SUN:
  # cp /home/vltmgr/VLTSW/PRODUCTS/pecs/templates/msql /etc/init.d/msql
  # chmod 555 /etc/init.d/msql
  # chown bin /etc/init.d/msql
  # chgrp bin /etc/init.d/msql

  # cd /etc/rc3.d
  # ln -s /etc/init.d/msql S91msql

  # cd /etc/rc0.d
  # ln -s /etc/init.d/msql K08msql
```

4.4 CONFIGURE AND VERIFY

This section guides you through the configuration of one CCS(CCSLite) environment and one LCU. It is assumed that both VxWorks and RTAP or CCSlite are installed.

If VxWorks is not installed, please skip everything concerning LCU.

4.4.1 Configure the LCU TCP/IP Channels

In order to have communication between LCU and WS (Engineering User Interface), a TCP/IP channels (environment name and port number) is required for each environment. TCP/IP Channels are defined in the file `/etc/services` (you need to be `root` to edit it). Add to such a file one line for each LCU and for each WS environment. E.g.:

```
/etc/services
        .
        .
    #------ begin VLT configuration
    lted         2160/tcp        # lqs on ted
    wte19        2301/tcp        # ws environment on te19
    #------ end VLT configuration
```

4.4.2 Configure the LOGGING System.

The Logging System is not automatically started with the environment because it is independent to it. The Logging system is installed as follows:

1. log file location:

```
export VLT_LOG_FILES=/vltdata/tmp
export VLT_LOG_ROOT=/vltdata/config
```

These variables are normally set by the standard environment (alias PECS).

2. as vltmgr, create the empty files:

```
$ cd $VLTDATA/tmp
$ touch logFile
$ touch logAuto
$ chmod 666 logFile logAuto
```

3. as `root`:

   a. configure the `syslog`:

   ```
   vi /etc/syslog.conf
           . . . . .
   ```

   insert

   ```
   *.info;mail,local1,local2.none
                          <TAB>/var/adm/syslog/syslog.log (HP)
                          <TAB>/var/adm/messages          (SUN)
   ```

```
# The following two lines configure the VLT logging system
#
local1.warning<TAB>/vltdata/tmp/logFile
local2.warning<TAB>/vltdata/tmp/logAuto
```

b. force `syslogd` to read the new configuration file

```
$ kill -HUP ‘cat /etc/syslog.pid‘
```

c. enable user vlt to use crontab by adding “vlt” to:
HP: `/var/adm/cron/cron.allow`
SUN: `/etc/cron.d/cron.allow`

4. as `vlt`:
   a. add the automatic clean-up of the log files to “vlt” crontab:

```
$ export EDITOR=vi   (for the tcsh shell)
$ crontab -e
```

Add the following line (or substitute any previous line containing the `logVLTBackup` command):

```
0,5,10,15,20,25,30,35,40,45,50,55 * * * * bash -c \
’. $HOME/.bashrc; logVLTBackup vlt vlt 12 100000’
```

5. as vlt, verify the syslog:
   a. in one window: `$ logger -p local1.warning “this is my text”`
   b. in another window: `$ tail -10 $VLT_LOG_FILES/logFile`
   .... the second window should show your message.

4.4.3 Configure the WS Environment

Each CCS or CCSlite environment is defined by a name that shall be unique in the network. Hereafter `<wsenv>` is used to name the CCS or CCSlite (or QSEMU) environment you are configuring. Substitute each occurency of <wsenv> with the real name.

1. as vltmgr, define the environment as the “local” environment, i.e., the one to access data from when environment name is not specified:

```
$ export RTAPENV=<wsenv>
```

Do it for all windows you may have opened. This definition is needed from now on and it should be added to the other ones defining the environment (e.g., added to `/etc/pecs/releases/000/etc/locality/apps-‘hostname‘.env`)[1]

1. The standard environment provides the default definition: RTAPENV=w${HOST}

2. make this environment known to the system:

   RTAP
   Add a line as follows to `/etc/$RTAPROOT/RtapEnvList`

   ```
   <wsenv>  <VLTDATA>/ENVIRONMENTS/<wsenv>   # environment on <ws>
   ```

   CCSlite
   Add a line as follows to `$VLTDATA/config/CcsEnvList`

   ```
   <wsenv>  <VLTDATA>/ENVIRONMENTS/<wsenv>   # environment on <ws>
   ```

3. using the configuration tool:

   ```
   $ vccEnv &
   ```

   - enter the actual environment name and press Return (or, depending on the keyboard, Enter). Fields are filled with defaults values, normally appropriate.
   - click “Create” button. The window will show the log of ongoing operation.
   - click “Init” button. say “OK” to confirmation window.
   - click “Start” button. The logfiles in `$VLTDATA/ENVIRONMENTS/<wsenv>/*.log`
   - or `$VLTDATA/ENVIRONMENTS/<wsenv>/.*.log` (note the initial “dot”) should not report any errors.

The default database is already configured to support the verification tests. The `RtapEnvTable` (or CcsEnvTable) file defines the processes that shall automatically start at database startup.

For full CCS, you can use the RTAP tools to inspect the database structure (`RtapDbConfig`, `RtapPtSelector`, `RtapPtDisplay`) or to monitor the running processes (`RtapPerfMon`).

4.4.4 Verify the WS Environment

The easiest way is to use `ccsei` to interact with the WS environment. `Ccsei` is the interactive utility to exchage messages with local and remote applications, to read / write in databases, to monitor the logs, to call other development and debugging tools. The same functions are available for both LCU and WS environments.

If the WS environment has been properly started, all ccsei utilities should work. Invoke the main menu first:

```
$ ccsei
```

... the list of available tools is presented.

REMARK: from now on, the main menu of `ccsei` is supposed to be always displayed on the user screen.

Try some simple operations:

See a log:

1. from the `ccsei` main menu, click “CCS Log Monitor”. The “VLT Log Monitor” tool is started. Enable the monitoring by clicking the “MONITOR” option. On another xterm, create a log:

```
$ logger -p local1.warning “this text should be displayed”
```

It should be displayed.

Send a Message

2. click “CCS Command Window”. The “ccsei Message Command Window” is presented
3. write “cmdManager” in the ”Process” field, click “Help” and select “On Commands”. The “Help On Commands” panel is presented.
4. select the “VERSION” command: the command format is displayed and the command name also appears after the prompt of the Command Window (`1<wsenv> cmdManager VERSION (Par)>`).
5. click after the `VERSION (Par)>` and press the <Enter> key to send such a command to the cmdManager process. The “Replies” field should give the version number of the currently running `cmdManager`.
6. close the “Help On Commands” window;
7. click “File” and then “Quit” to close the “ccsei Message Command Window”

Read / Write into the database:

8. click “CCS Database Monitor” in the main menu. The “ccsei Database Monitor” window is presented. Click “CCS Database Monitor” again to have a second window. The first will be used for a continuos monitor of a variable, the second to change the variable content.
9. in both windows: click “Browse”, then select “PARAMS”, “SCALARS”, “scalar_int32”, “Accept” to move the point name to the main menu, “Dismiss” to close the point browser. Both should display the same value.
10. In the first window, click “Move to list” and “Activate Monitor”
11. In the second window, click on the “DB Value” field, type another value and press <Enter>. The first window should show the new value in the field labeled “Data Quality and Values:” (a little delay is normal).
12. close all the panels, but the main menu.

More about the `ccsei` can be found in the “CCS User Manual”.

4.4.5 Configure the LCU Files on the WS

To work the LCC requires that some files are set on the WS acting as boot node.

Each LCU environment is defined by a name that shall be unique in the network. Hereafter:

`<lcuenv>` is used to name the LCU environment you are configuring

`<lcunode>` is the LCU node name.

Substitute each occurency of `<lcuenv>` and `<lcunode>` as appropriate.

For each LCU environment:

1. be sure that:

- the LCU is booted and the remote login from the WS to the LCU is possible (no one is locking the LCU shell).
- the “vx” username is defined and the LCU can execute a remote shell to it (either `~vx/Xauthority` or `~vx/.rhosts` correctly set)

2. using the configuration tool:

```
$ vccEnv &
```

- enter the actual LCU environment name (<lcuenv>) and press Return (or, depending on the keyboard, Enter). LCU-host (<lcunode>), the Boot-env (<<wsenv> or <qsemuenv>) are taken from the database.
- click “Maximum” and then click on “Create”.
- click “Init”.
- the “Config...” button invokes the “vccConfigLCU” panel to change configuration options. The default values are enough for the validation test, but you can set what needed by your applications. Click “WriteFiles” when ready to regenerate target files. Click “Continue” to override the current files. Then go back to the vccEnv panel.
- click “Start” to reboot the LCU. The log of the reboot process allows you to follow it. At the end of the boot, the console should display the message:

```
LCC INITIALISATION SUCCESSFUL.
```

REMARK: to be able to display the boot log, the LCU Configuration tool locks the connection to the LCU for about 120 seconds. After this time the connection is released (`vccConfigLcu: connection closed`). If the boot process takes longer, part of the log may not be displayed. In such a case, check the LCU console.

3. check that among running LCC processes there are:

```
$ rlogin <lcunode>

<lcuenv>->i
    (the processes may be listed in defferent order!)
  NAME        ENTRY       TID    PRI   STATUS   ......
---------- ------------ -------- --- ---------- ......
  ......      ....         .....
  ......      ....         .....
lccServer  _cmdInit       61d964  80 PEND+T
rdbServer  _cmdInit       6005cc  80 PEND+T
msgServer  731da6         6864f8 100 PEND
lccTime    737fa0         681548 100 DELAY
lccEvent   71848a         652438 100 READY
lccPeriodic733a92         64d488 100 READY
lccDevice  711d1e         622914 100 DELAY
lccWatchdog73d86e         6c7e50 150 READY
tLqsClk    784410         6ce1bc 250 DELAY
lccLogger  728d18         68b4a8 250 READY
```

```
value = 0 = 0x0
<lcuenv>->
```

and that the LCU environment table has been correctly loaded

```
lted-> lqsPrintEnvTbl
Environment   Host name   TCP port   .....  .....
-----------   ---------   --------
  wte19         te19         2226
  lted          ted          2160
value = 59 = 0x3b = ';'
lted->
```

In case the LCU does not boot correctly check that: the boot directory is NFS mounted:

```
<lcuenv>-> nfsDevShow
device name         file system
-----------         -----------
/vltdata           wtatcam:/vltdata
/VLTROOT           wtatcam:/vlt/OCT99
value = 0 = 0x0
<lcuenv>->
```

4.4.6 Make the LCU known to the CCS environment

1. make the LCU known to RTAP: add a line as:

   ```
   <lcuenv>  <lcunode>   # LCC environment on <lcu>
   ```

   to:

   `RTAP: /etc/$RTAPROOT/RtapEnvList` (HP 10.20 only)

   `CCSlite: $VLTDATA/config/CcsEnvList (HP11.00 and SUN)`

2. configure reporting node for LCU log activity

   ```
   $ cp $VLTROOT/include/logLCU.config.template \
                          $VLTDATA/config/logLCU.config
   ```

   and modify it according to your configuration.

   ```
   $ cat $VLTDATA/config/logLCU.config
   <lcuenv> <wsenv>   (*)
   ```

   (*) remember to terminate the last line with EOL

3. check the connection
   a. by a simple message exchange:

```
$ msgSend <lcuenv> lccServer LCCGVER ""
MESSAGEBUFFER:
"@(#) LCC ....<version>... ....<date>....."
```

4. force the logManager to read the new configuration:

```
$ logConfig
```

5. from the `ccsei` main menu, start a "CCS Log Monitor", click on "MONITOR", then create a log from the LCU

```
$ rlogin <lcu>

<lcuenv>-> logData "test", 101, "this is a log from an LCU"
value = 2 = 0x2
```

check that the log is displayed by the monitor tool.

REMARK: a delay of few seconds is normal because, in order not to overload the network, logs are transmitted by the LCU to the upper level only periodically.

4.4.7 Verify the LCU environment

`Ccsei` can also be used to verify the just created LCU environment.

Send a Message

1. click "CCS Command Window". The "ccsei Message Command" window is presented
2. from the [+] menus, select `<lcuenv>` as environment and then `lccServer` as process.
3. send the command `LCCGVER` (either by typing it after the prompt or by selecting from the menu); type 'enter". The "Replies" field should display the version number:

```
REPLY: "@(#) LCC ....<version>... ...<date>..."
```

In the same way, other commands can be used (see LCC User Manual)

Read / Write into the LCU database:

4. click "CCS Database Monitor" in the main menu. The "ccsei Database Monitor" window is presented. Click "CCS Database Monitor" again to have a second window. The first will be used for a continuos monitor of a variable, the second to change the variable content.
5. continue as in 4.4.4, but from the [+] menus, select `<lcuenv>` as environment . Notice that the selection of the database point using the "Browse" facility is slower because the system queries the database structure from the running LCU.

4.4.8 Interact with an LCU Application

The lcuapp is used to test and demonstrate the way of working of `ccsei`.

1. from the ccsei main menu, click “Log Monitor”. The “VLT Log Monitor” tool is started. Enable the monitoring by clicking the “MONITOR” option.
2. make the `lcuapp` known to `ccsei`. Add one line containing `lcuapp` to the file `$VLTDATA/ENVIRONMENTS/<lcuenv>/PROCESSES`
3. load `lcuapp` into the LCU and start it: open an rlogin on the LCU. Give the following commands

```
$ rlogin <lcunode>

<lcuenv>-> cd getenv(“VLTROOT”)
<lcuenv>-> cd “vw/bin/MC68040” (if you use 68k LCUs)
or
<lcuenv>-> cd “vw/bin/PPC604”  (if you use PPC LCUs)
<lcuenv>-> ld < lcuapp
<lcuenv>-> sp lcuapp
```

   Two logs should appear in the LCC Log / Inspect Window telling that `lcuapp` has started and is waiting for commands.
4. start a “CCS Database Monitor” window in monitoring mode (check “Move to list” and “Activate monitor”) on the <lcuenv>`:PARAMS:SCALARS.scalar_int32` variable.
5. from a “CCS Command Window”, select `<lcuenv>` as environment and then `lcuapp` as process, then send a `SETVAL <nn>` message to `lcuapp` to change the value in the LCC database. <nn> is an integer; the valid range is 0-100.
   - the Database Monitor should show the new value (`<nn>`)
   - the Log Monitor should report on the LCU activities.
   - the Command Window should not display error replies

   Repeat for different values. If `<nn>` is out of range, an error and a different log should be received.
6. from a “Command Window”, send `EXIT` to `lcuapp` to stop the application.

4.4.9 Verify Cooperation between WS and LCU Applications

The scenario we are going to use now is the same as the one used to test the `lccei`, but a WS appl cation is used to send commands to the LCU companion.

At this point you are already an expert in configuring both WS and LCU so set things up as follows:

- the WS environment is running (see 4.4.3)
- the Logging system is active (`logManager`) and a `logMonitor` window is displayed (see 4.4.2 4.4.6)
- th LCU is properly configured, i.e., it can communicate with the WS environment (see 4.4.6)
- `lcuapp` is running on the LCU (see 4.4.8)
- `ccsei` is monitoring the LCC database position `:PARAMS:SCALARS.scalar_int32` (see 4.4.8)

At this point you can run the WS application that sends the number you type to `lcuapp` in order to

be stored, if in the range, into the LCC database.

```
$ wsappSetValue <lcuenv>
wsappSetValue - WS application example - Version 1.1, June 94

Enter a number to send to the LCU ....
Enter 'q' to quit
<nn>
<nn>
....
q
```

If in range, the number you type will be displayed by the Monitor window. Repeating for different values of <nn> you can experiment with several cases, including logs, errors, replies, etc. Type a "q" to stop both `lcuapp` and `wsappSetValue`.

These actions complete the verification of CCS/CCSLite and LCC installation. If you have CCSlite, please go to 4.4.12.

4.4.10 Configure the CCS/Scan System

This sections applies only to Full CCS installations

The Scan System allows you to mirror an LCC database variable into a CCS database variable.

To configure the Scan system follow what is described in the CCS User manual. Briefly:

1. configure the Scan system on the LCU:
   a. create in the LCC database the device point corresponding to <wsenv>. Edit the LCC database description on the WS:
      ```
      $ cd $VLTDATA/ENVIRONMENTS/<lcuenv>/dbl
      $ vi USER.db
        ...edit according to the instructions given in the file itself
      $ make db
      ```
   b. reboot the LCU (vccEnv, select <lcuenv>, "Start")
2. configure the Scan system on the WS.
   a. stop the environment (vccEnv, select <wsenv>, "Stop")
   b. create in the CCS database the device point corresponding to <lcuenv>
      ```
      $ cd $VLTDATA/ENVIRONMENTS/<wsenv>/dbl
      $ vi USER.db
      ```
      ... edit the USER.db file, and rename (name and alias) the device point according to your <lcuenv>.
      ```
      $ make db
      ```

c. re-Initialize the environment (vccEnv, select <wsenv>, “Init”, “Start”)

3. enable scanning from LCU:

```
$ scanDevOn <lcuenv>
```

4.4.11 Verification of the Scan System

This sections applies only to Full CCS installations

The scenario we are going to use now is the same as the one used to verify the cooperation between WS and LCU applications (see 4.4.9), but the Scan System is used to mirror the LCC database value into the WS and a WS application is used to display it.

Set things up as follows:

- the CCS environment, including Scan System, is running
- the CCS/Logging system is active (`logManager`) and a `logMonitor` window is displayed.
- the LCU is properly running, including `lcuapp`.

Now configure the Scan system so that the content of the variable accessed by lcuapp is also mirrored in a CCS database variable having the same name:

```
$ scanConfig ":PARAMS:SCALARS.scalar_int32"
             "@<lcuenv>:PARAMS:SCALARS.scalar_int32" POLL 1
```

(both variables are already defined in the database configuration shipped in the tape, so you do not need to create the variables, but only to establish the link.)

Then start the WS application that displays the CCS environment position:

```
$ wsappShowValue
wsappShowValue - WS application example - Version 1.1, June 94

Display DataBase variable (press 'q' to quit).
===============================================

Reading from: @ccs:PARAMS:SCALARS.scalar_int32


     Time:  05:33:54      Value:            88
```

Change the value in the LCU database

```
$ dbWrite "@<lcuenv>:PARAMS:SCALARS.scalar_int32" <nn>
```

<nn> should be displayed.

The last test involves nearly all the CCS and LCC modules. Run again the WS application that sends the number you type to `lcuapp`.

```
$ wsappSetValue <lcuenv>
wsappSetValue - WS application example - Version 1.1, June 94

Enter a number to send to the LCU ....
Enter 'q' to quit
<nn>
<nn>
....
q
```

If in range, the number you type will be displayed by `wsappShowValue`. As before, you can experiment with several cases, including logs, errors, replies, etc.

4.4.12 WS Environment Shutdown

To terminate the verification, close the environment properly using the vccEnv "Stop" or:

```
$ vccEnvStop -e<wsenv>
```

These actions complete the verification of CCS / CCSLite and LCC installation.

4.5 OTHER TESTS

The verification procedure has the purpose to check that basic features work correctly and to get you acquainted with the VLT Software.

During installation you may have decided to install many more modules than the one the verification procedure has tested. The User Manual of each software module contains additional tests. Please refer to such documentation to configure and test drivers, INS software, etc as appropriate to your installation.

4.6 CONFIGURING WORKSTATION STARTUP

As appropriate, the startup of the standard environment, qsemu, logManager, etc. can be part of the WS startup process. The automatic startup of the ACC database server has been described in 4.3.1, the automatic start of the WS environment is described in the RTAP / CCSlite installation procedure (see 3.7.4, 3.7.5).

4.7 REPORT TO ESO

You are kindly request to provide by mail, fax or e-mail (see 2.6 for addresses) the list of the products you have installed and the computer configuration (type, OS, etc).

Please contribute to improve the quality of this manual, especially the trouble-shooting list (see 5):

add to your report any suggestions you may have to improve the installation procedure or any mistakes you may have made.

Problems or errors in the installation procedure should be reported using the SPR form (see 2.6).

5 TROUBLE SHOOTING GUIDE

This sections is intended to provide a first level debugging help. More can be found in the specific documentation. The following list is, of course, not exhaustive, and is made up with cases which occurred during previous installations. Please fell free to contribute to it (see 4.7). A case you have encountered and solved and that is not described hereafter, can occur again to somebody else. Let's help each other to save time!

5.1 Problems with NFS files

- NFS does not allow to mount a disk already mounted with NFS. Check that each NFS-mount is to the machine where the files are really located.

5.2 Missing setuid on MSQL deamon

SYMPTOM:

vltmgr:~ 41 > vccEnvStart -e <environment>
vccEnvStart@<host>: Warning: VccInfo: no valid result from query: GetByEnv <environment> envType
vccEnvStart@<host>: Warning: VccInfo: no valid result from query: GetByEnv <environment> hostName
....

RESPONSE:

ls -ld `which msqld`
-r-sr-x--- 1 vltmgr vlt 268607 Oct 6 22:05 msqld

If the permissions, owner and group don't match as above
then fix them.

REMARKS: if the permissions are correct and the message:
... Warning: VccInfo: no alid result from query...
is still there, it is probably due to some missing records in the accData.sql file. Check that all the entry in each table is appropriate and no information regarding your environment(s) are missing.

5.3 "couldn't write PID" after starting the mSQL server

Check the right ownerships of the file $VLTDATA / msql / msql.pid.
It should appear as follows:

-rw-r--r-- 1 vltmgr vlt 4 Dec 11 08:23 msqld.pid

5.4 The LCU does not boot

- check that the LCU is properly connected to the network
- on the LCU check the boot parameters (see 4.4.5): the host name, where to look for the VW boot file, the username
- on the WS check that: the “vx” username is defined and that the VxWorks file is readable by such a user.
- check that “vx” doesn’t have a .cshrc that produces output!

5.5 The LCU boots but does not find the bootScript file

- on the LCU check that the bootScript file is correctly specified. Remember that NFS does not manage links: you shall specify the physical absolute path.
- on the WS check that:
  - VLTROOT can be NFS mounted (see 3.3.3)
  - the file is readable by the “vx” user (see 4.4.5)

5.6 LCC does not start successfully

- on the WS check that:
  - VLTROOT can be NFS mounted
  - bootScript is correct and contains the absolute path to VLTROOT and BOOTROOT
  - the user specified in the bootScript exists on your UNIX system. To make it simple, use “vx” for both booting the LCU and in the bootScript file.
  - logfile and rebootFile are available in $BOOTROOT/ENVIRONMENTS/<lcuenv>, even as empty files, and readable/writable by the “vx” user.
  - devicesFile is available in the $BOOTROOT/ENVIRONMENTS/<lcuenv>. Remember that even if you do not have any devices, the file must exist and with a minimal content, see devicesFile(5) and being readable by the “vx” user.

5.7 VxWorks code is not compiled

- check VxWorks environment (see 3.7.3)

5.8 RTAP environment does not start

- check RTAP environment (see 3.4.3)
- check that operating system parameters have been changed according to RTAP/CCSlite installation.
- if the logfile: $VLTDATA/ENVIRONMENTS/<env_name>/.ccsScheduler.log reports the following

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
ccsScheduler: S-BIT not set for ccsScheduler
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

run the following commands:

```
chown root /vlt/<RELEASE>/CCSLite/bin/ccsScheduler
chmod u+s /vlt/<RELEASE>/CCSLite/bin/ccsScheduler
```

5.9 Lccei does not start (properly)

- check that the PATH contains $VLTROOT/bin
- check that $VLTROOT/config/Lccei has been loaded by xrdb
- check that $VLTROOT/CDT contains valid files (the format can be checked using lcccdt), one for every process defined in $VLTDATA/ENVIRONMENTS/<lcuenv>/PROCESSES file
- check that qsemu is running

5.10 Lccei cannot communicate with an LCU

- check that qsemu is up and running
- check $VLTDATA/config/qsemu.config
- check that the environments defined on WS (qsemu -env) and LCU (lqsPrintEnvTbl) are consistent.

5.11 RtapUnlockExe fails

- check that the development license is available.
- on SUN: check that you are using the GNU-gcc compiler has produced by the buildGNU.

5.12 RTAP does not shutdown

- if RtapShutdown fails, use

  ```
  $ kill <RtapSheduler-processId>
  ```

  If still does not work, try to kill RtapMQDBM

  ```
  $ kill <RtapMQDBM-processId>
  ```

  (on both cases, do not use kill -9 to stop! This is documented in Rtap manuals.)

5.13 LOG system does not work

There are several facts that may affect the syslog, either at system level or due to bad VLT Software configuration. The following may help to find some of the possible causes:

- check that syslog is working:

  ```
  $ ps -efa | grep syslog
  ```

- create a log in one of the system files (the commands used in the following example are for documentation only and may not work in your configuration. Please check them with your system manager):
  - have a look in /etc/syslog.conf and chose a “selector”
  - use logger to create an entry in the corresponding log file (the following is from an HP):

    ```
    $ logger -p mail.debug “aaa”
    ```

  - check that the file associated with the selector

    ```
    $ tail /usr/spool/mqueue/syslog
    Mar 06 17:14:44 te13 vltsccm: aaa
    ```

- check that the VLT_LOG_... variables are defined and that the content of /etc/syslog.conf is consistent with such a definition. Restart the syslogd to be sure that it is using the right files:

```
$ kill -HUP ‘cat /etc/syslog.pid‘
```

Be aware of the different behaviour of HP and SUN:

on HP: if not existing, the files are created.

on SUN: if not existing, the syslog DOES NOT create them! Create empty files as described in 4.4.2.

- put a log in each of the log files:

```
$ logger -p local1.warning  “this should go in logFile”
$ tail $VLTDATA/tmp/logFile
.................
$ logger -p local2.warning  “this should go in logAuto”
$ tail $VLTDATA/tmp/logAuto
.................
```

- if you have CCS(RTAP), create a log from the WS :

```
$ logUserData aa aa aa
$ tail $VLTDATA/tmp/logFile
.................
```

- if you have an LCU, create a log from the LCU:
  - check that `$VLTDATA/config/logLCU.config` exists as well as the pair `<lcuenv><wsenv>`. If needed edit the file.
  - check that `logManager` is active. If not start it: `$ logManager &`
  - force logManager to read the configuration file: `$ logConfig`
  - on the LCU, give the following command to create a log

```
$ rlogin <lcu>
<lcuenv>-> logData “test”, 101, “this is a log from an LCU”
```

Remember that `logMonitor` can be used to display the current VLT logs.

5.14 lccei does not accept CDT

CDT in NOV95 format or older are not valid any longer. Please edit them according to the new format as described in the LCC User Manual.

5.15 Scan system does not start on the LCU

LCU Database in FEB95 format or older are not valid any longer. Please change the LCU DB according to the new format.

5.16 Declarative conflicts during installation

The include files of this release cannot be mixed with previous ones. May be you have forgotten to save the old installation before starting this installation (see 3.6.2).

5.17 Force an error log from an LCU

The following procedure allows you to test the complete chain LCU-log system:

- Start “lccei”
- Select your environment and the process lccServer from the pull-down menus “Environment” and “Process”.
- Send the commands “LOGSTRT” and “ERRSTRT” to your LCU.
- Select the process rdbServer.
- Send the command “DBGAINF :PARAMS:TABLES.full_table(-1)” or to any other table or vector in the database. This will produce the error log:

```
(mapMapValue) “lccERR_INV_RECEL : invalid number of records/
   fields (number: number: -1  expected: )” (Warning 63) Help: -
```

- Now select the log window from the pull-down menue “File”. The Log Window will pop up.
- Select “Show All Messages” or “Show Matching Messages” and “Logging” and “Error”.This will display the error log in the log window.

5.18 Security error messages from Tk

Q.C4- X server insecure (must use xauth-style authorization) Tk requires you to have a secure X server before you can use the send command.

See the question 2.A.7 “How can I get Tk3.3 to even start - I get security error messages.” in Thomas Accardo’s Tk Toolkit Usage FAQ as well as `http://ce-toolkit.crd.ge.com/tkxauth/` for instructions on how to make your server secure.

5.19 LCU boot problem: INTROOT contains old code

You may experience problem at boot time if you have INTROOT defined and old software is loaded there.

`lcuboot’ will load modules automatically from INTROOT if they are there. If, for any reasons, old versions of LCC of of some other modules loaded during the boot are stored there, the boot process may be not complete or, if severe errors - like unresolved symbols - are found it may also abort.

After installation of any new version, current INTROOTs, if any, should be cleaned and the software regenerated using the newest version in VLTROOT.

5.20 LCU boot problem: tim board not installed

When LCC tries to read the time-board. If no board is installed, an error message is produced but the booting process continues regularly. If you do not have such a card, please ignore the message.

5.21 “unbalanced” Parentheses Warning from tclCheck

The “warnings” arise from the utility `tclCheck’, which is automatically called by vltMake for every TCL file. Such “unbalanced” parentheses warning in correct code can be removed by using “\” as in the following examples:

```
> > File ccseiDb.tcl:
> > Inside a string: unmatched ( ending line 294 char 27
> >
> 294:            set i [string first “(“ $att]
E.g.: replace by: set i [string first “\(“ $att]

> > File ccseiDbBrowser.tcl:
> > Expecting } got ] : line 309 char 31
> >
> 309:    # -attrGeometry $attrGeometry]
also:     # -attrGeometry $attrGeometry\]

Line 385:     set line “ (“
replace with: set line “ \(“

Line 415:     append line “ 1==1)”
replace with: append line “ 1==1\)”
```

5.22 Man pages not correctly formatted

A problem can arise if a slash (“/”) is used in writing a man page, as in the example below:

```
/*****************************************************
  * NAME
  * evhFILE - contains methods for the evhFILEIO class
  *
  *
  * SYNOPSIS/DESCRIPTION
  * The file contains definitions of methods for the
  evhFILEIO class. The
  * following methods are defined:
  *
...
```

In this example, the “/” between SYNOPSIS and DESCRIPTION causes a completely messed up output. Till now, no investigation has been done to solve the problem. Waiting for a proper resolution, simply, avoid the use of such a slash.

5.23 Different LCU environments on the same nodeDifferent LCU environments on the same node

If there are several environments assigned to the same node and, by mistake, one writes two or more of them in the logLCU.config file, sometime errors occur (no reply from LCU, message “environment not active” etc.).

NEVER PLACE IN logLCU.config two or more LCU environments associated to the same node !!!

5.24 LCU slow speed.

During ASM commissioning, we pointed out a dramatic slow speed of the operations on a LCU: A process supposed to complete in 16 seconds was running more than 5 minutes. After investigation, we found out that a flag was set on the LCU, that enables run-time printouts on the LCU console, while no rlogin <lcu> had been issued. The effet is then that the serial line (9600 bps) is used to issue the approx. 300000 characters output during the sequence. A rough calculation of the output time is:

300000 char * 10 bits / char / 9600 bps > 300 seconds (= 5 minutes)

Therefore, take care of this kind of setting (the flag was set in the userScript) that may terribly bias the performance of a LCU if no rlogin is used.

____oOo____

6 PECS (Pluggable Environment Contribution System)

6.1 USER ENVIRONMENT SETTINGS

PECS supports personal settings for the following types of settings:

- environment variables
- X resources
- GUI root window menu items
- shell aliases

All personal settings belong in files in the ~/.pecs directory, which should exist and contain (comments-only) example files. The big problem with putting a setting in one of these files, is knowing which file to put it in! This section explains the steps to determine where to put a setting:

1. Decide the 'type' of the setting

Four 'types' of personal settings are supported by PECS:

a. environment variables (type=env)
b. X resources (type=xrdb)
c. GUI root menu items (type=wmrc)
d. shell aliases (type=ali)

Note down what 'type' it is.

2. Decide the 'application-scope' of the setting

Settings either:

a. relate to a PECS-aware application (application-scope=apps)
   (e.g. VLTROOT, INTROOT, GNU_ROOT, NOCCS, PRINTER, RTAP_EXISTS)
b. don't relate to a PECS-aware application (application-scope=misc)
   (e.g. DFLOW_ROOT because it relates to older versions of VLTSW only, NNTPSERVER, LESS, MAIL, PATH=$HOME/bin:$PATH).

3. Decide the 'host-scope' of the setting

Settings either:

a. are applicable only on certain hosts (host-scope=<name-of-host>)
b. are applicable on all hosts (host-scope=all)

4) Put the setting in the right file

Now that you've worked out the three attributes 'type', 'application-scope' and 'host-scope', it's easy; the setting belongs in the file:

~/.pecs/<application-scope>-<host-scope>.<type>

e.g., to change your prompt set 'PS1' in ~/.pecs/apps-all.env, and
to change your VLTROOT set VLTROOT in the same file.

5) Activating the setting

To activate the change in setting first run:

```
make_xdefaults
```

and then log out and back in.

Let's look at a couple of examples; consider each of the following settings, look at the values it has for each attribute, and see the name of the file it goes in.

Example #1: Changing VLTROOT

Suppose you're working on a machine where VLTROOT is set to /vlt/MAR2001/CCS, and you need to point it at FEB2000 instead just for a quick test. Well,

it's an environment variable (type=env)
it relates to VLTSW, which is a PECS-aware application (application-scope=apps)
you only need it for a quick test on host 'te13' (host-scope=te13)

So, having worked that out, the setting belongs in the file:

~/.pecs/apps-te13.env

So, in the file you just put:

VLTROOT=/vlt/FEB2000/CCS

Note that environment variables supported by a PECS-aware application are automatically made available to sub-processes ('exported') so you do not need to export it yourself (though it does no harm to do so).

Example #2: the 'which' alias

Yes, the 'which' command under Bourne-like shells does not work quite the same as it did under Tc-sh. Under Bash, the nearest equivalent to 'which' is 'type' or 'type -all'. So to make 'which' an alias for 'type -all':

it's a shell alias (type=ali)
it's not related to a PECS-aware application, it's just personal setting (application-scope=misc)
you want it everywhere (host-scope=all)

So, having worked that out, the setting belongs in the file:

~/.pecs/misc-all.ali

So, in the file you just put:

```
alias which='type -all'
```

Aliases are read per-shell-process, so it is not necessary to log out; your next new xterm will see the alias.

Note that Bash aliases have an '=' between the alias and its value, unlike Tcsh.

Example #3: you want to add 'top' and 'xv' to your 'User Menu'

PECS provides a GUI root window sub-menu entitled 'User Menu'. Suppose you want to create the menu to contain the command
'top'

it's a GUI root window menu item (type=wmrc)
it's not related to a PECS-aware application (the user menu never is) (application-scope=misc)
you want to see this menu everywhere (host-scope=all)

So, having worked that out, the setting belongs in the file:

```
~/.pecs/misc-all.wmrc
```

So, in the file you put:

```
Menu user_menu
{
   "My Menu"   f.title
   "Top"       f.exec "xterm -T top -n top -e top"
   "Xv"        f.exec "xv"
}
```

Note that 'top' isn't on all systems, so simply calling 'top' is a bit naive.

To activate the settings select 'Restart' from the GUI root window menu.

Example #4: you don't like the Xclock updating every second

PECS allows you to specify new values for X resources, but it also allows you to 'unset' existing ones (this is not a standard behaviour of X).

You will probably find your Xclocks showing a second hand, because of the X resource:

```
XClock*update: 1
```

To cancel this setting, just provide a blank value for it. So,

it's an X resource (type=xrdb)
it's related to a PECS-aware application (yes, this resource is defined by VLTSW)
(application-scope=apps)
you want to see this menu everywhere (host-scope=all)

So, having worked that out, the setting belongs in the file:

~/.pecs/apps-all.xrdb

So, in the file you put:

XClock*update:

To activate the change you need to log out and log in again.

Example #5: you don't like to use
"export VARIABLE=VALUE"
and you want to keep using
"setenv VARIABLE VALUE"
Very bad idea, but if you are so obstinate, you should write a function, because aliases which require embedded parameters must be written as functions, but should still be put in the ".ali" file. Here's an example:

```
export_function()
{
   export $1=$2
}
alias setenv=export_function
```

6.2 A CRASH-COURSE IN BOURNE SHELL

What follows is a little cookbook of common things you may wish to put in your ~/.pecs/*.env files:

To set a variable do this:

```
VARIABLE1=value
VARIABLE2="value with spaces"
export VARIABLE1 VARIABLE2
```

(Notice there are no spaces around the '='; bourne shell is sensitive about this.)

To safely test if a variable is set do this:

```
if [ "X$VARIABLE" != X ]; then

   code to do if VARIABLE is set

else

   code to do if VARIABLE is not set

fi
```

To set an alias, put something like this in your ~/.pecs/misc-all.ali file:

```
alias ALIASNAME="alias value"
(e.g: alias ls="/bin/ls -a")
```

A switch statement looks like this:

```
case "$VARIABLE" in

   value1) do this
         and then this
         and this is last ;;

   value2) do this
         and something else ;;

   *)    this is the default action ;;

esac
```

If you want to do a tcsh rehash, you should do:

```
hash -r
```

____oOo____