CERN
CH-1211 Geneva 23
Switzerland

the
Large
Hadron
Collider
project

EN Project Document No.
EN-UNICOS

CERN Div./Group or Supplier/Contractor Document No.
EN / ICE

EDMS Document No.

Date: 2010-01-19

Documentation

UNICOS FRAMEWORK

Abstract

This document describes the framework…

Prepared by:	***Checked by:***	***Approved by:***
UNICOS-PVSS core team EN/ICE		

History of Changes

Rev. No.	***Date***	***Pages***	***Description of Changes***
1.1 Draft	19.01.2010		First version

Table of Contents

1. INTRODUCTION

1.1 PURPOSE OF THE DOCUMENT

UNICOS stands for **UN**ified **I**ndustrial **CO**ntrol **S**ystem. This document describes the concept of the UNICOS framework and lists all the related documentations.

1.2 DEFINITIONS, ACRONYMS AND ABBREVIATIONS

From this point onwards, the following acronyms will be used:

- PVSS: SCADA used at CERN
- PLC: Programmable Logic Controller

1.3 CONTACT AT CERN

Any problem during installation or development can be reported to the following email address: UNICOS.Support@cern.ch.

2. UNICOS CONCEPT

UNICOS is a CERN framework developed to produce control applications for three-layer control systems. UNICOS was primary done for cryogenics control and following the needs for other types of control/monitoring system, the UNICOS framework was formalized. UNICOS provides developers with the means to rapidly develop full control or monitoring applications and provides operators with ways to interact with all the items of the process from the most simple (e.g. I/O channels) to the high level compound devices with little effort. In addition UNICOS offers tools to diagnose problems in the process, the control system and to access and operate the devices without specific development.

The framework (Figure 1) proposes a reusable environment composed of a set of components for the supervision and the front-end layers:

- UNICORE where packages can be added. It is based on PVSS and the JCOP framework.
- A basic package called UNICOS Continuous Process Control (CPC) is offered to develop process control applications.

A package is a set of generic software components combined together to extend UNICOS to a specific domain and configured to produce a control and/or monitoring application. Tools to create new packages are also provided.

An application is a set of packages combined and configured together to produce a control or monitoring application (Figure 2).

In addition to the CPC package, other packages have been developed to cope with new applications. Such packages cover mainly the development in the supervision layer.

UNICOS was designed keeping in mind:

- The application developer is not a SCADA expert; he has little or no SCADA knowledge.
- Control system is seen as a whole: abstraction of SCADA, Front-end, etc. Even if the final application is ditribued in many PVSS system, the application is seen and presented as a unique one.

- Less work as possible for the application developer.
- The logs files are not used, the errors are presented to the user via pop-up message, message list, etc.
- All the relevant information on the quality of the data is presented in one shot.

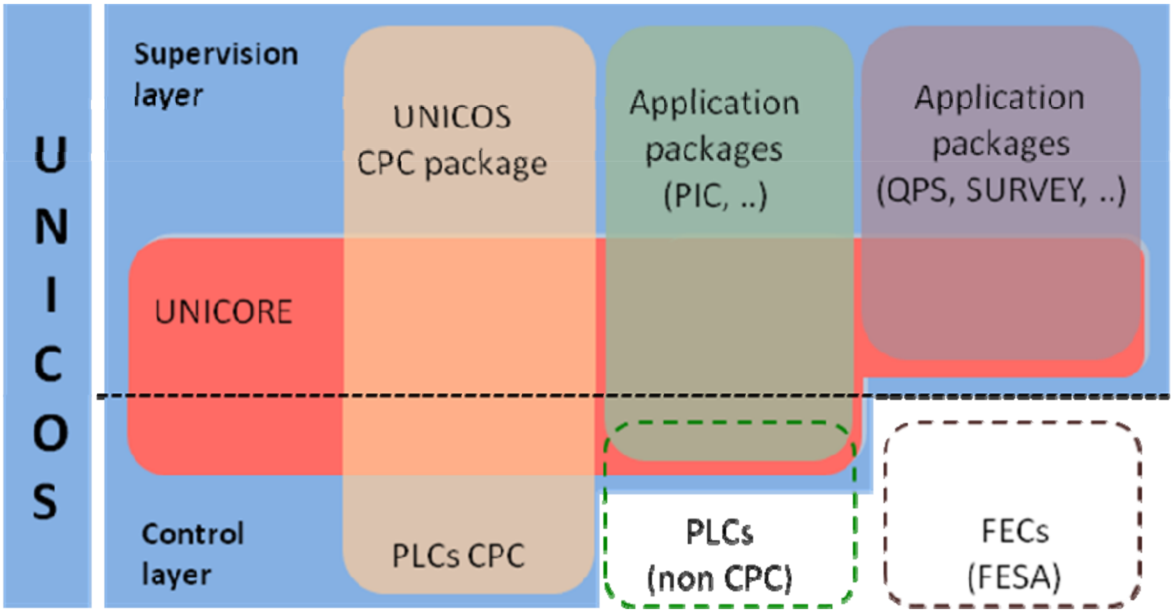


Figure 1: UNICOS Framework and application package.

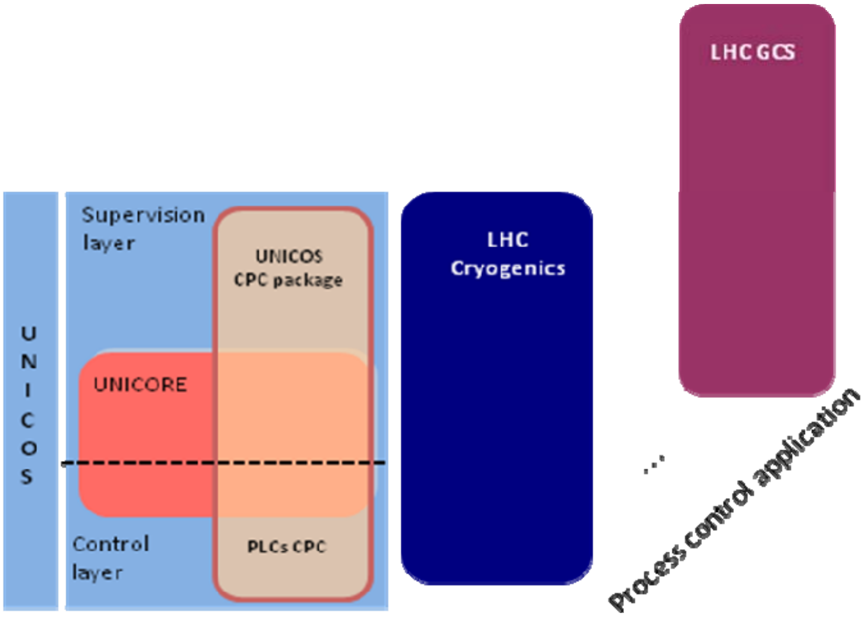


Figure 2: UNICOS Framework: application.

3. UNICORE

3.1 DESCRIPTION

UNICORE is deployed in two layers: the supervision and the process control layers. It is based on PVSS II® and reuses some of the **J**oint **CO**ntrol **P**roject (JCOP) framework components. At the supervision level, in both Linux and Windows operating systems, the following functionality is offered:

- Application distribution in many SCADA data servers and handling of the connection state.
- Monitoring the integrity of the application (e.g. status of the front-end).
- Interface to the LHC Software suite (LHCLogging, LASER Alarm System and Post-Mortem Data Analysis System).
- Client and Server Controls MiddleWare (CMW) interface.
- Interface to include additional packages.
- Device and file access control, based on the JCOP access control with as many domains as needed and four privileges per domain and with LDAP (Lightweight Directory Access Protocol) feature.
- Mass configuration: creation of the device with the corresponding configuration (link to the hardware, alarm, archive, etc.)
- Trending tool.

UNICORE was also developed to offer UNICOS application developers a configurable user interface which does not require neither the knowledge of the PVSS scripting language and nor the use of basic PVSS interfaces. In addition it offers operators a homogenous user interface that is entirely customizable with features such as navigation capabilities between panels and trends (WWW browser like, contextual buttons, pop-up navigation), access to the devices without creating a panel (tree device overview), process alarm and event list.

The supervision device hierarchy is based on a front-end device containing process devices (time stamping in the front-end is allowed). A front-end device is an entiy representing a piece of hardware or software entity holding devices. A front-end is also the entity through which the devices are accessible. Process device or device represents a piece of hardware sensor or software entity. A device is attached to one and only one front-end. Typical examples of front-end devices are PLCs, Front-End Computers (FEC), OPC servers, ELMB; while process devices are Analog Input, Digital Input, ELMB channel, etc. Front-end device and process device can be with or without hardware link.

At the front-end level the UNICORE implements the communication protocol with time stamping at the source and optionally an event publishing mechanism (TSPP).

3.2 CONCEPT

At the supervision level UNICOS contains:

- The unCore component.
- The unLHCServices component: interface to the LHC Software suite (LHCLogging, LASER Alarm System, CMW Server/Client and Post-Mortem Data Analysis System).

3.2.1 THE UNCORE COMPONENT

The unCore component is the core of the supervision part of UNICORE. It provides an interface to the device and front-end device for:

- Import (mass configuration): to import the device and front-end configuration
- Export: to export the device and front-end configuration in the same format as the import
- Widget (Figure 3): summarized view of the device or front-end device. There can be many widgets for a device type, but just one for the front-end type. The quality of the data must be shown in the widget.
- Faceplate (Figure 4): detailed view of a device or front-end device. There is one faceplate for a device type and and one for a front-end type. . The quality of the data must be shown in the faceplate.
- Trend configuration: more than one trend configuration is allowed for each device type and front-end type.
- Device action interface (Figure 5): action on the device. Device access control is allowed with 4 priviledges per access control domain and as many access control as needed.
- File access control: configurable access control for panel files, with 4 priviledges per access control domain and as many access control domain as needed.

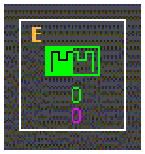
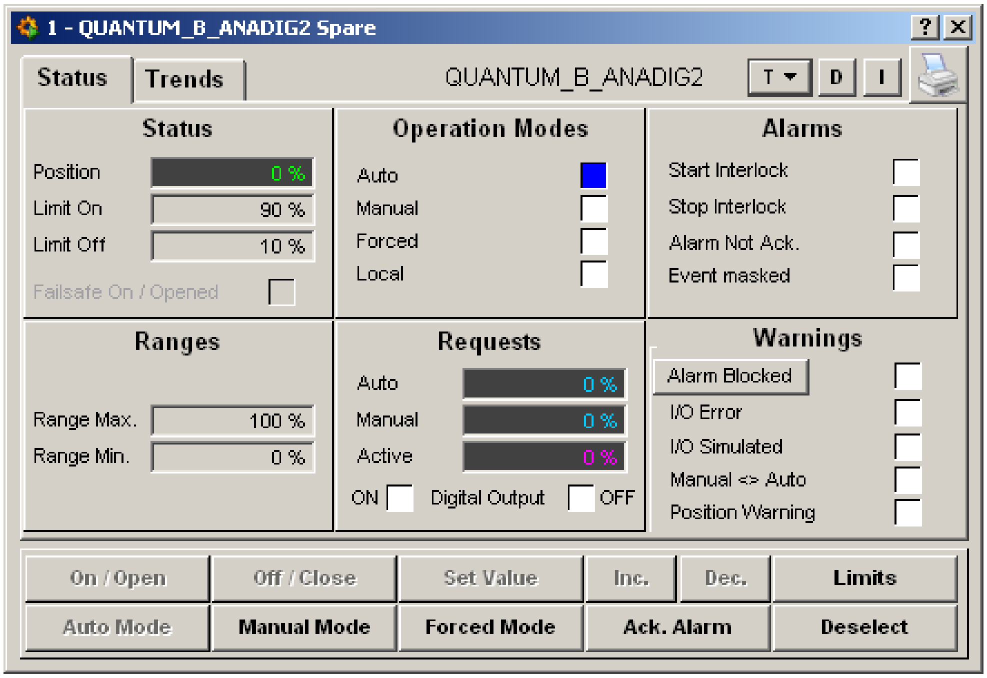


Figure 3: device widget.



QUANTUM_B_ANADIG2

Status

Position 0 %
Limit On 90 %
Limit Off 10 %
Failsafe On / Opened □

Operation Modes

Auto ■
Manual □
Forced □
Local □

Alarms

Start Interlock □
Stop Interlock □
Alarm Not Ack. □
Event masked □

Ranges

Range Max. 100 %
Range Min. 0 %

Requests

Auto 0 %
Manual 0 %
Active 0 %
ON □ Digital Output □ OFF

Warnings

Alarm Blocked □
I/O Error □
I/O Simulated □
Manual <> Auto □
Position Warning □

Figure 4: device faceplate.

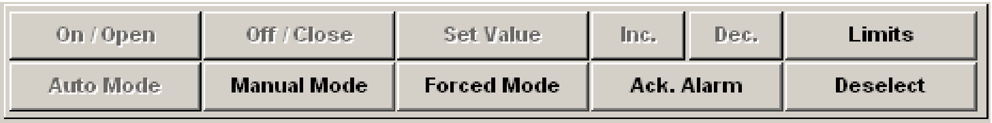


Figure 5: device action interface.

The unCore triggers the device and front-end device functions and panels. The unCore uses intensively:

- DistributedControl component (use to check the connection and disconnection of a remote system). Callback functions are called to handle the device and front-end device behaviour. Global variables to the panels or to the libs can be used even without the ‘global’ keyword.
- evalScript and execScript PVSS functions. Care must be taken with the use global variables in panels and libs and with $-parameters.

3.2.2 THE UNLHCSERVICES COMPONENT

This component contains all the interfaces to the LHCSoftware suite:

- LASER: LHC Alarm System
- LHCLogging: LHC Long term archiving
- CMWClient: CMW (RDA) client interface with RBAC
- CMWServer: CMW (RDA) server interface with RBAC

3.3 DEVICE NAMING & HIERARCHIES

Each device has two names:

- The logical name stored in the PVSS alias. This is usually the name used to identify the device. The PVSS system name separator ‘:’ is not allowed in the name. The syntax is [PVSSsystemName:]PVSSAlias, if the PVSSsystemName is not specified the local PVSSsystemName is used instead. For instance: DFBA_CV981, MB.A12R3, P8_82:DFBA_CV981, QPS_34:MB.A12R3
- The hardware name of the device, this is the PVSS DP name. This is the name of the device in the hardware hierarchy. The syntax is

  [PVSSsystemName:]prefix-FrontEnd-FrontEndApplication-DeviceType-xxxxx

  For instance: un-CFP_SHC8_LHC8-QSCB-Analog-00123,

  qps-CFC_SR3_IP3_DR3B-A34-DQAMCMB-10320,

  P8_82:un-CFP_SHC8_LHC8-QSCB-Analog-00123

  QPS_34:qps-CFC_SR3_IP3_DR3B-A34-DQAMCMB-10320

Each front-end device has also two names:

- The logical name stored in the PVSS alias. This is the name used to identify the front-end. The PVSS system name separator ‘:’ is not allowed in the name, the name must be only letters a..z, A..Z, numbers 0..9 and ‘_’. The syntax is [PVSSsystemName:]PVSSAlias, if the PVSSsystemName is not specified the local PVSSsystemName is used instead. For instance: CFC_SR4_DR4BC_IP4_DR4B, QPS_45:CFC_SR4_DR4BC_IP4_DR4B.
- The hardware name of the device, this is the PVSS DP name. This is the name of the front-end. The syntax is [PVSSsystemName:]FrontEndType_PVSSAlias For instance: DQGTW_CFC_SR4_DR4BC_IP4_DR4B, QPS_45:DQGTW_CFC_SR4_DR4BC_IP4_DR4B

3.4 LINK & PROXY

There are two ways of associating devices: link and proxy.

- Link: each UNICOS device can be linked together. This is a relationship between devices. This link can unidirectional or bi-directional. There is no limitation in the number of linked devices. The device can have full access to the linked devices
- Proxy: device and front-end device can be a proxy to UNICOS and non UNICOS device (device that does not have the UNICOS DPE structure in order to handle non UNICOS devices within the UNICOS utilities: import/export, trending, widget, faceplate, device actions). There is no copy of the proxy data into the UNICOS device. A typical exemple is the integration of JCOP devices into UNICOS. The proxy link is unidirectional: a UNICOS device or front-end is linked via a proxy to another device. A device or front-end device can be a proxy of only the configured types with as many proxies as needed. Proxy device and UNICOS device must be from the same PVSS system.

3.5 SYSTEM INTEGRITY

The systemIntegrity is a UNICOS utility to check the correct functioning (integrity) of a component. An interface to add new systemIntegrity check is provided. The systemIntegrity is composed of PVSS libs, a configuration panel and an operational panel. The systemIntegrity produces _UnSystemAlarm devices to show the problems on the integrity of the component. The systemIntegrity is used to check the correct functioning of the front-end device usually based on the check of a counter value periodically sent to the supervision. Having a correct communication link to the front-end usally guarantees that the devices are accessible.

3.6 GROUPING DEVICES

There 3 default possibilities to group devices:

- Hardware hierarchy: device PVSS DP name. The device is grouped by device type, one application and one front-end
- Subsystem1 (domain in the cryogenic vocabulary). A device can be in one or many subsystem1 group or in none of them.
- Subsystem2 (nature in the cryogenic vocabulary), subsystem2 is a sub group of subsystem1. A device can be in one or many subsystem2 group or in none of them.

A device being in a subsystem2 group, that is a sub-group of subsystem1, is in two groups: subsystem1 and susystem2.

3.7 FRONT-END DEVICE

A front-end device is an entiy representing a piece of hardware or software entity holding devices. Typical examples of front-end devices are PLCs, Front-End Computers (FEC), OPC servers, ELMB, etc. Front-end device can be with or without hardware link. A front-end can have many devices grouped by:

- Application: control application. The application is used to grouped devices that are part of the same control system or sub-system. Ex. QSCB cryogenic system, powering subsector, etc.
- Device type: the type of the device
- Device number: from 00000 to 99999

A front-end is also the entity through which the devices are accessible; the quality of the communication link with the front-end is checked via the systemIntegrity. A front-end device has:

- One widget
- One faceplate
- Device user access control, implemented with the file access control. This access control has limited functionalities and is only configurable on the front-end type.
- Many trending configuration: faceplate and trending plot
- Proxy interface

3.8 DEVICE (PROCESS DEVICE)

A process device or device represents a piece of hardware sensor or software entity. A device is attached to one and only one front-end in one and only one application. Typical examples of process devices are Analog Input, Digital Input, ELMB channel, etc. Device can be with or without hardware link. A device has:

- Widget: as many as needed
- One faceplate
- Device action interface
- Device user access control entirely customizable per device and per device type
- Selection with automatic de-selection: exclusive lock on the device to have access to the possible device actions.
- Many trending configuration: faceplate and trending plot
- Link interface: as many links as needed
- Proxy interface.

3.9 SOME RULES OF DEVELOPMENT

Here is a list of some rules of development:

- The PARA module is never used to configure the devices and the front-end devices. One has to use instead the UNICOS import and device configuration facility.
- The GEDI is used to build the synoptic (panel used to operate the system) with the drag & drop facility of the device widgets. Scripting in panels is allowed but it must be done with care.
- Widget, faceplate, any device and front-end utility must be easy to use and understandle to users with or without PVSS knowledge.
- Diagnostic on front-end, device, device and front-end action must be easily accessible and understanble.
- The errors in operating the device, the front-end device, or using them with the UNICOS facility must not be only written in the PVSS log files. If they are of interest to the user, developer, etc. they must be poped up in exception window (fwException JCOP facility), UNICOS message text component.

4. UNICOS DEVELOPMENT

There are different types of development:

- On a device of a package:

  - Add new widget to the list of widget of a device: refer to the documentation unicos-pvss-adding-front-end_and_device.doc.pdf
  - Import/export the device with a new front-end type: refer to the documentation unicos-pvss-adding-front-end_and_device.doc.pdf
  - Any other modifications of device behavior must be discussed and agreed with the responsible of the package.
- On a front-end of a package:
  - Import/export device of another package with the front-end: refer to the documentation unicos-pvss-adding-front-end_and_device.doc.pdf
  - Any other modifications of the front-end behavior must be discussed and agreed with the responsible of the package.
- Add a new front-end: refer to the documentation unicos-pvss-adding-front-end_and_device.doc.pdf
- Add a new device: refer to the documentation unicos-pvss-adding-front-end_and_device.doc.pdf.

LIST OF FIGURES

REFERENCE

1. unicos-pvss-device.pdf: adding a device, modifying a device behavior, etc.
2. unicos-pvss-front-end.pdf: adding a front-end, modifying a front-end behavior, etc.
3. unicos-systemIntegrity-device-type.pdf: systemIntegrity concept and development.
4. unicos-pvss-unCore-data-flow.pdf: description of the unicos facility data flow.