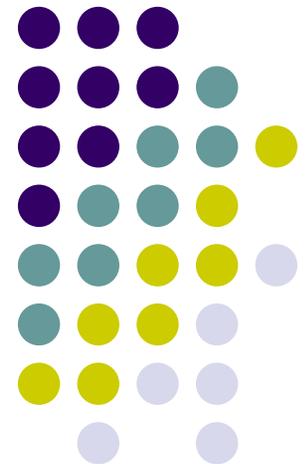


A rule-based Control and Verification framework in ATLAS Trigger-DAQ

2006 Conference for
Computing in High Energy and Nuclear Physics
13-17 Feb. 2006 Mumbai, India



Presented by Andrei Kazarov
CERN-ATD/PNPI Petersburg

Presentation contents



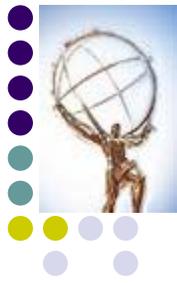
- Part one: Expert system-based architecture of Run Control system
 - Goals
 - Design and Architecture
 - Implementation
- Part two: DVS: diagnostics and verification framework:
 - DVS overview
 - Recent developments
 - Use for ATLAS commissioning

A challenge for Control system: the scale of ATLAS Trigger-DAQ



- ATLAS T/DAQ is composed of a huge number of hardware and software components:
 - 1800 read-out VME boards
 - 1800 fiber links
 - 150 ROS PCs each hosting 4 ROB-IN cards
 - 500 LVL2 PCs
 - 90 SFI PCs
 - ~2000 EF PCs
 - ~30 SFO PCs
 - ~50 infrastructure PCs (file servers)
 - ~200 Ethernet switches
 - And $O(10000)$ applications running

Run Control: Design goals



With the given system size, h/w and s/w failures are very probable, and it is very important to have testing and diagnostics facilities embedded in the Control System in order to:

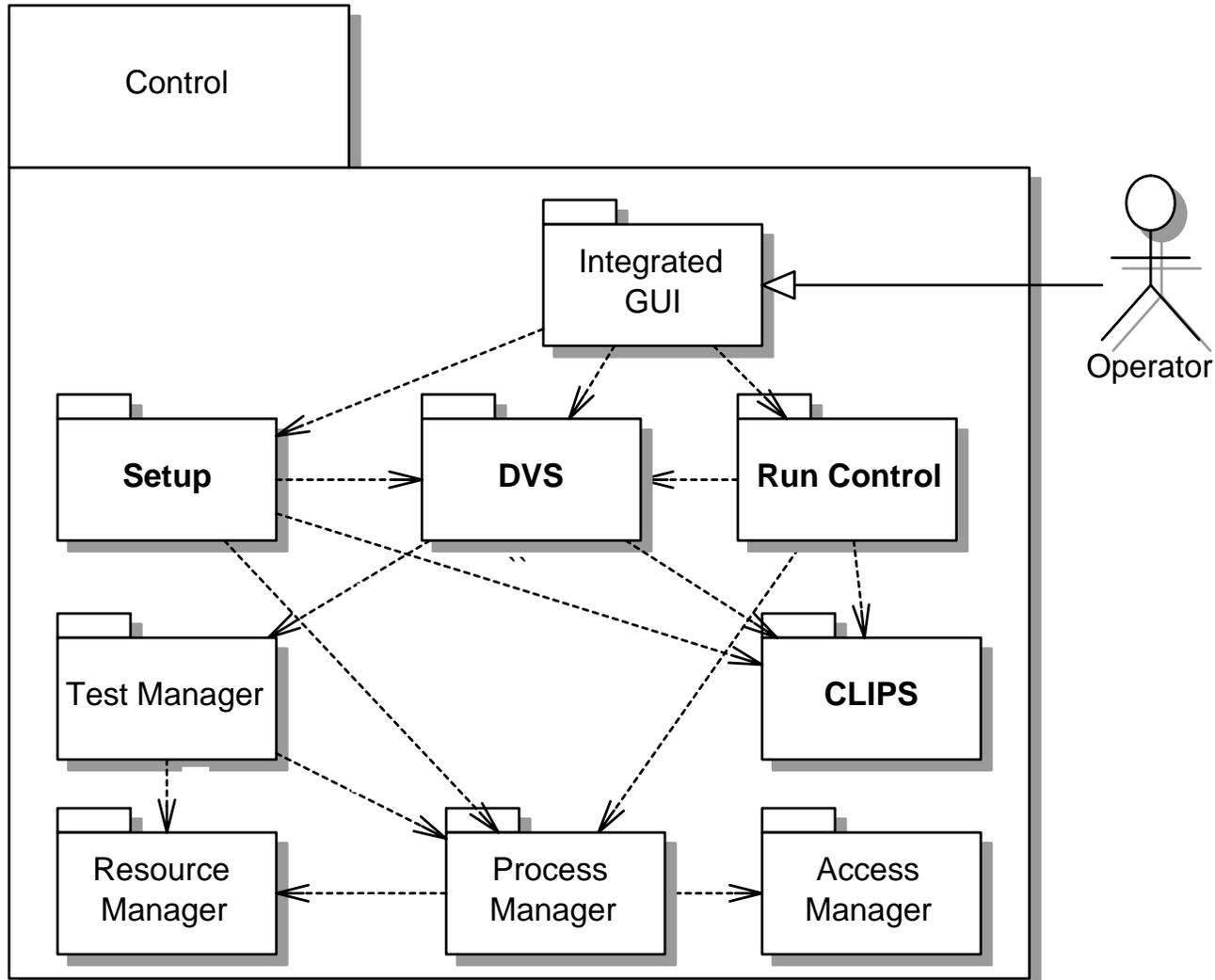
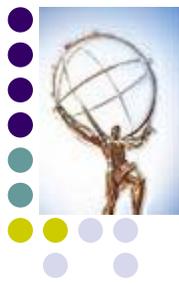
- Detect problems as early as possible by means of probing the system
- Make use of system's developers expertise (knowledge)
- Automate verification of a large system
- Minimize system down-time, using recovery procedures based on problem diagnosis

Design principles

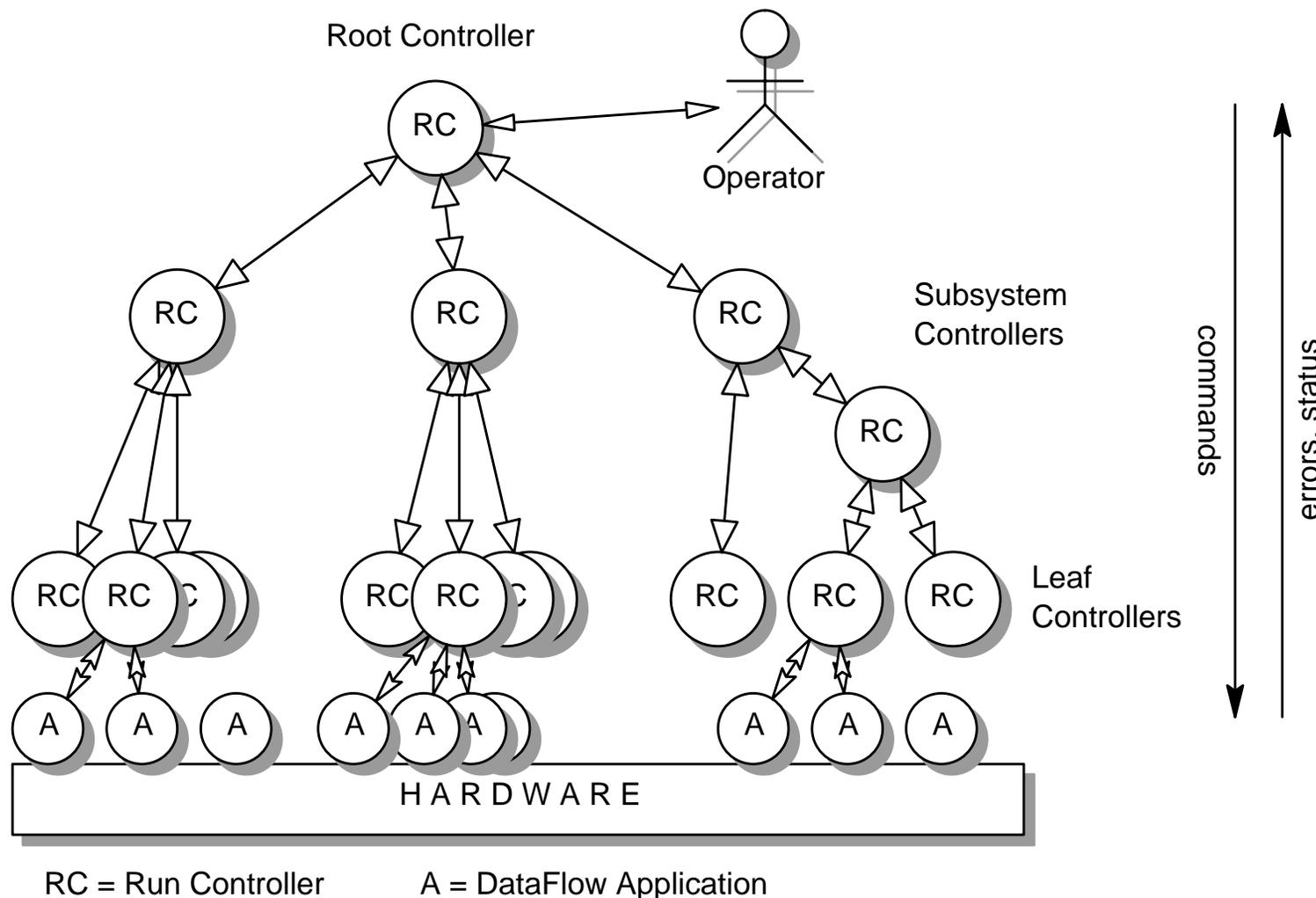
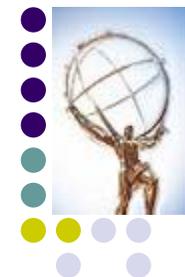


- **Framework** approach: system shall be configurable and extensible by experts and users, also during the experiment lifetime
- **Expert system** approach: system's behavior is described in rule-based language, allowing accumulation of expert's knowledge and easy adaptation in changing conditions
- **Hierarchical distributed** architecture of the Run Control system, reflecting the structure and the scale of the experiment

Control Subsystem High-Level Design



Run Control: a tree of controllers



Controller's behavior



- Each Run Controller is an implementation of a Finite State Machine and a small Expert System (i.e. engine + some rules)
- Each controller has a state, determined by states of children by the rules
- A simple rule is just 'if all my children are in state A, change state to A'
- *More complex **recovery rules** should analyze errors and make some decisions (disabling a sub-tree, executing recovery actions, reporting to parent)*

DVS (more details in part II)

Diagnostics and Verification System



A *framework* which allows to:

- Configure a test for any component in the system
- Have a testable view on the particular configuration of a system in a user-friendly GUI
- Automate testing of the system
- Make diagnostics conclusion in case of a problem detected during testing (provided some knowledge put in the Knowledge Base)

Setup component: infrastructure supervision



- Setup component is a ‘boot-strap controller’ for the initial infrastructure of TDAQ
- It brings the system to a state where it can accept RC commands
- It uses DVS to verify in depth system’s h/w in order to detect potential problems ASAP and confirm the system’s integrity *before* launching any process
- It contains additional rules to start, restart and verify applications and diagnose related problems
- Functionality of applications are also confirmed by the execution of tests

CLIPS: expert system shell



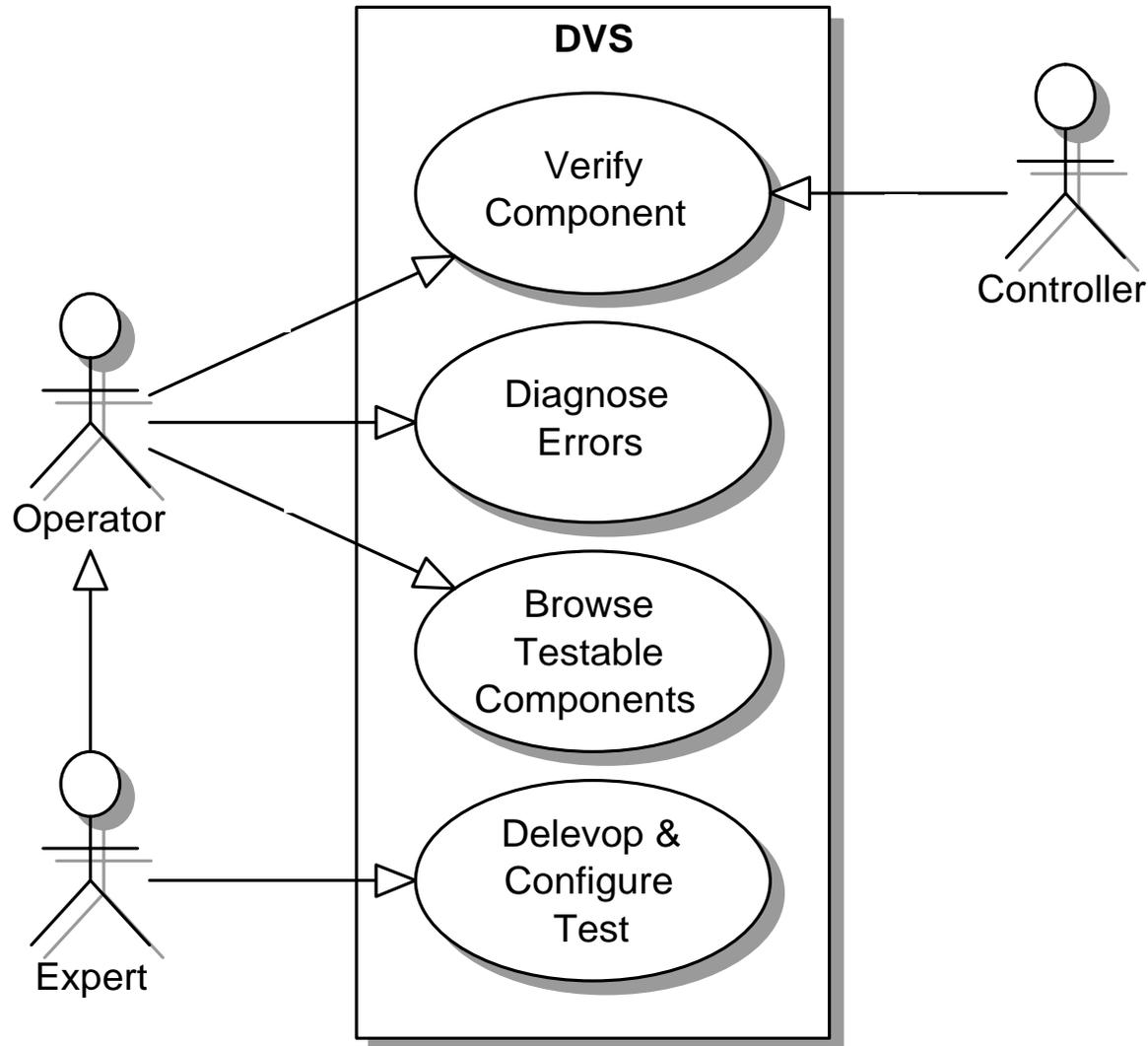
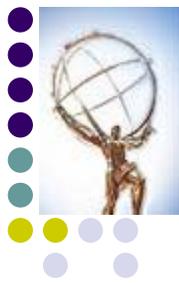
- ‘C’-Language Integrated Production System
- Produced by NASA
- Free, open (written in ‘C’) and well-documented
- Embeddable in other s/w products as a library
- Features: rule-base programming paradigm (rules and facts), OO language (classes and objects), conventional procedural constructs

Part II: DVS, diagnostics and verification framework

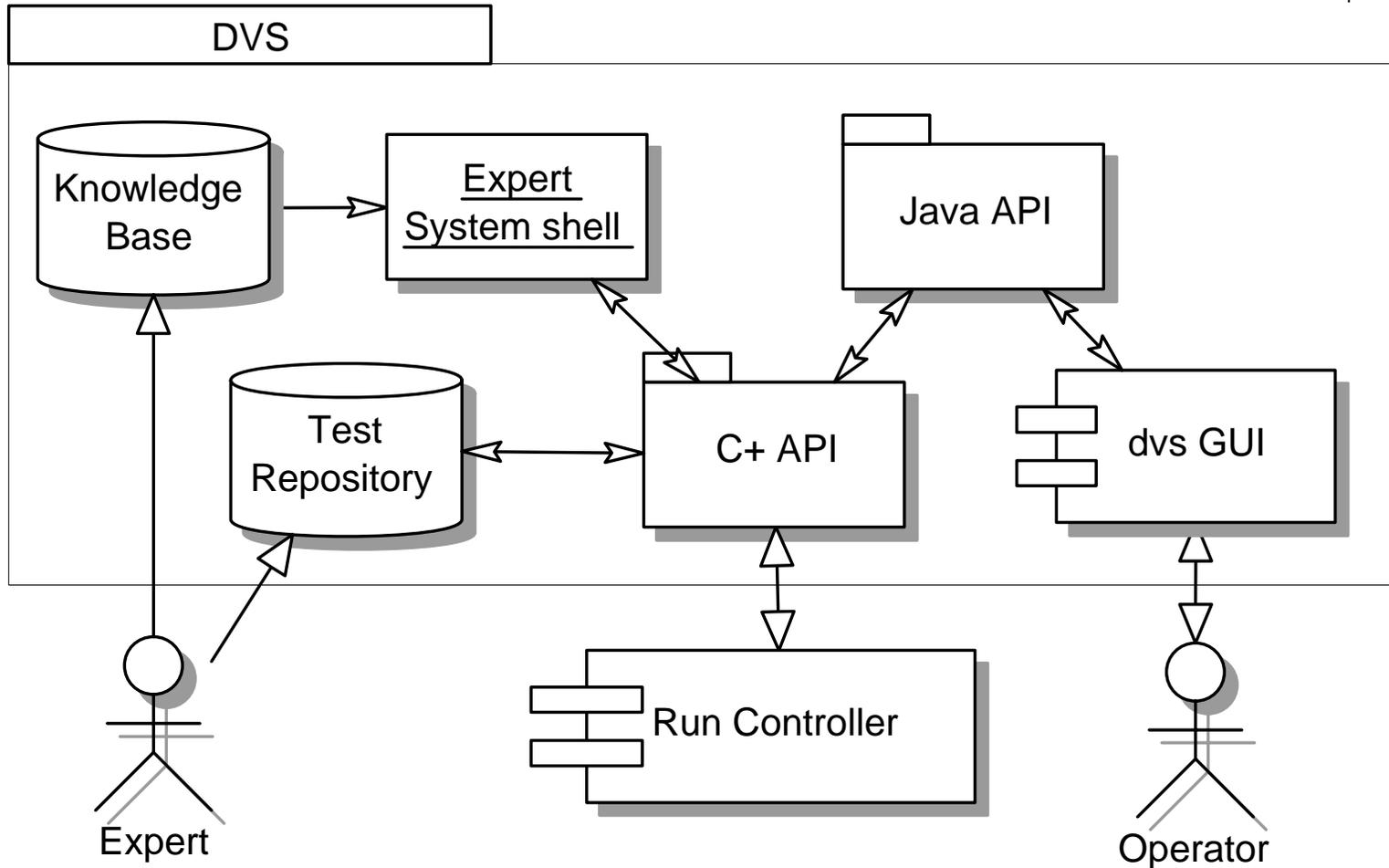
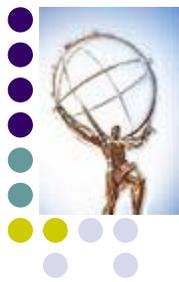


- Overview
- New features, added on request by users, basing on the experience of its use in the real environment
- Usage of DVS for ATLAS commissioning

Use Cases for DVS



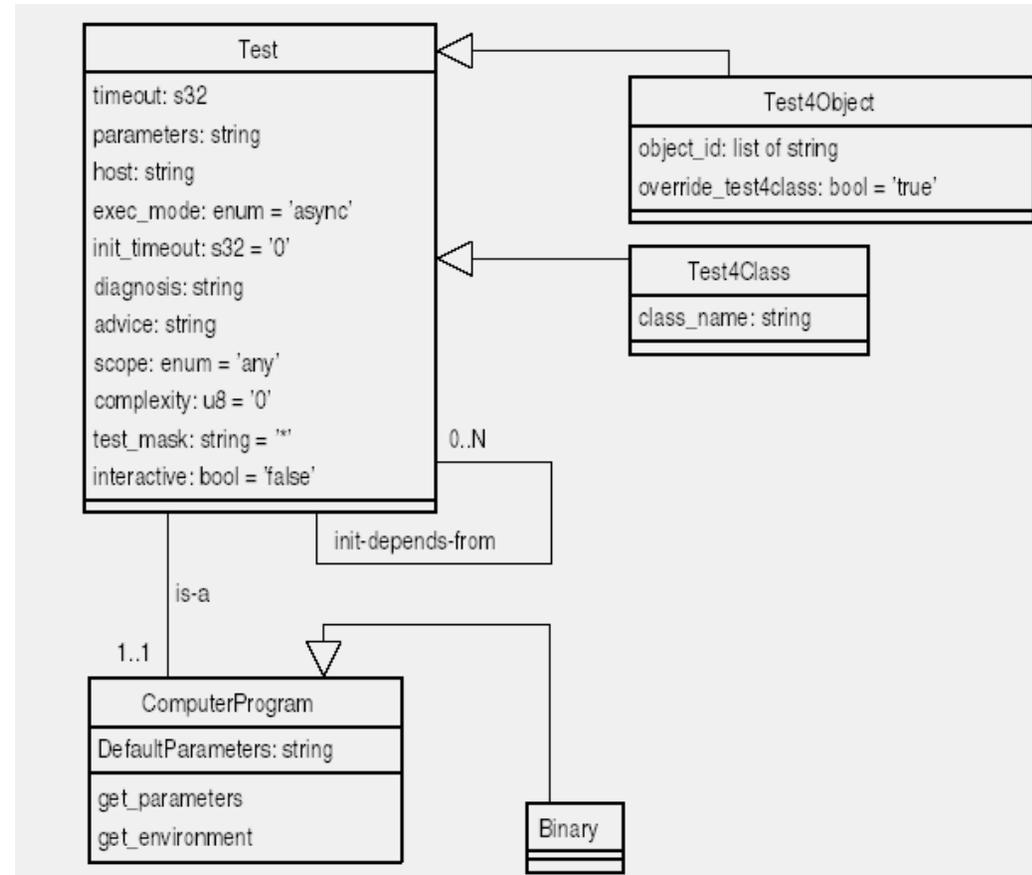
DVS architecture



What is a test



- Test is a binary, running on a particular host in a system
- Test verifies a particular functionality of a TDAQ component
- For a single component, a number of tests can be associated
- Test returns a value: PASSED, FAILED, UNRESOLVED, TIMEOUT
- Tests can be organized in sequences, executed synchronously or asynchronously
- Tests and their relationships are fully described in a database



DVS for end-users

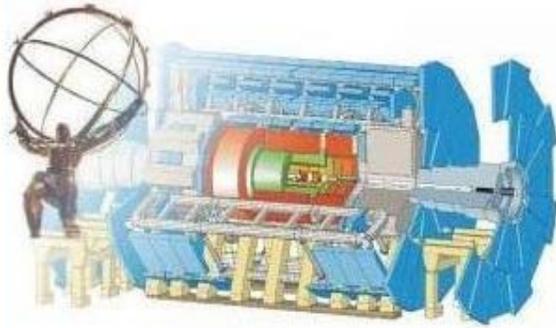


The screenshot shows the Diagnostics Verification Framework (DVS) software interface. The window title is "Diagnostics Verification Framework: Configuration MobiDAQ". The interface is divided into several sections:

- Left Panel (DAQ Components):** A tree view showing the configuration hierarchy. It includes:
 - Sector 13
 - TileDrawerModule LBC45
 - TileDrawerModule LBC46
 - TileDrawerModule LBC47
 - TileDrawerModule LBC48
 - TileDrawerModule LBC49
 - TileDrawerModule LBC50
 - TileDrawerModule LBC51
 - TileDrawerModule LBC52
 - Pedestal 1KHz
 - TriggerSyncPedestal TrigPedestal
 - RODEMUChanPedestal Drawer-01
 - RODEMUChanPedestal Drawer-02
 - RODEMUChanPedestal Drawer-11
 - RODEMUChanPedestal Drawer-12
 - RODEMUChanPedestal Drawer-21
 - RODEMUChanPedestal Drawer-22
 - RODEMUChanPedestal Drawer-31
 - RODEMUChanPedestal Drawer-32
 - Pedestal 100KHz
 - TriggerSync Trig1000
 - RODEMUChan Drawer-01
 - RODEMUChan Drawer-02
 - RODEMUChan Drawer-11
 - RODEMUChan Drawer-12
 - RODEMUChan Drawer-21
 - RODEMUChan Drawer-22
 - RODEMUChan Drawer-31
 - RODEMUChan Drawer-32
 - CIS Low Gain
 - TriggerSyncCISLG TrigCISLG
 - RODEMUChanCISLG Drawer-01
 - RODEMUChanCISLG Drawer-02
 - RODEMUChanCISLG Drawer-11
 - RODEMUChanCISLG Drawer-12
 - RODEMUChanCISLG Drawer-21
 - RODEMUChanCISLG Drawer-22
 - RODEMUChanCISLG Drawer-31
 - RODEMUChanCISLG Drawer-32
 - CIS High Gain
 - Trigger Readout
 - LED
 - Network
- Right Panel (Test Log):** A text area displaying the test execution log. The log shows the start and completion of a test for the "tile_drawer_CISLG" module. The test parameters are: `-v -m 1 -t 10 -c 1 -d42 -s2 -f 100 -a 700 -b`. The test output includes: "The module is barrel", "RCD: VmeSlaveMap: Dump: id = 0, pci = 003e6000, vme = 04000000", "Reading 1 events per trigger from SSPCI-2", "Mode 1", "TileDigiFragment::m_drawerid LBC52_20060117_1029", "Starting acquisition", "Events remaining 10", "Events remaining 5", "DEBUG: TGraph created", "Events 10 2194740us". The test concludes with "Test for RODEMUChanCISLG Drawer-32 PASSED" and "Testing completed".
- Bottom Panel:** A status bar with the text: "Welcome to the OnlineSW Diagnostics and Verification Framework", "initializing... OK", "Partition MobiDAQ loaded", "Test verbosity switched ON".

The Windows taskbar at the bottom shows the system tray with the date "Tue Jan 17 10:30 AM" and several open applications, including "Terminal", "ATLAS TDAQ Software", and "Diagnostics Verification".

Use of tests from Setup



A T L A S
T/DAQ
Software

Release nightly

Infrastructure Status

Hardware

Segment Infrastructure

setup hosts

- lxplus001.cern.ch
- lxplus002.cern.ch
- lxplus003.cern.ch
- lxplus004.cern.ch
- lxplus005.cern.ch
- lxplus006.cern.ch
- lxplus007.cern.ch
- lxplus008.cern.ch
- lxplus009.cern.ch
- lxplus010.cern.ch
- lxplus104.cern.ch

Retry

Ignore and Continue

Shutdown

Abort

19:18:14	ERROR	INTERNAL	Failed to verify partition h/w. Analyze results and make a decision: retry, continue or exit
19:18:12	INFORMATION	INTERNAL	Starting verification of the h/w used in the partition
19:18:11	INFORMATION	INTERNAL	Connected to setup manager. Getting status of infrastructure...
19:18:01	INFORMATION	INTERNAL	Connecting to Setup Manager in partition onlsw_test_lxplus
19:17:59	INFORMATION	INTERNAL	Initializing panels and connecting to infrastructure, please wait

New features:



- Tests *levels* and *masks* for more precise test selection, which allows to promptly configure test repository without editing the database
- Asynchronous and synchronous mode for execution of tests for complex objects
- Test *scope* to prevent conflicting tests from being executed when system is taking data
- Tests *verbosity* can be defined globally at runtime
- Test's *runtime output* for long-running tests
- Test report combined and saved in a file (and then to production DB)

New features: interactive tests

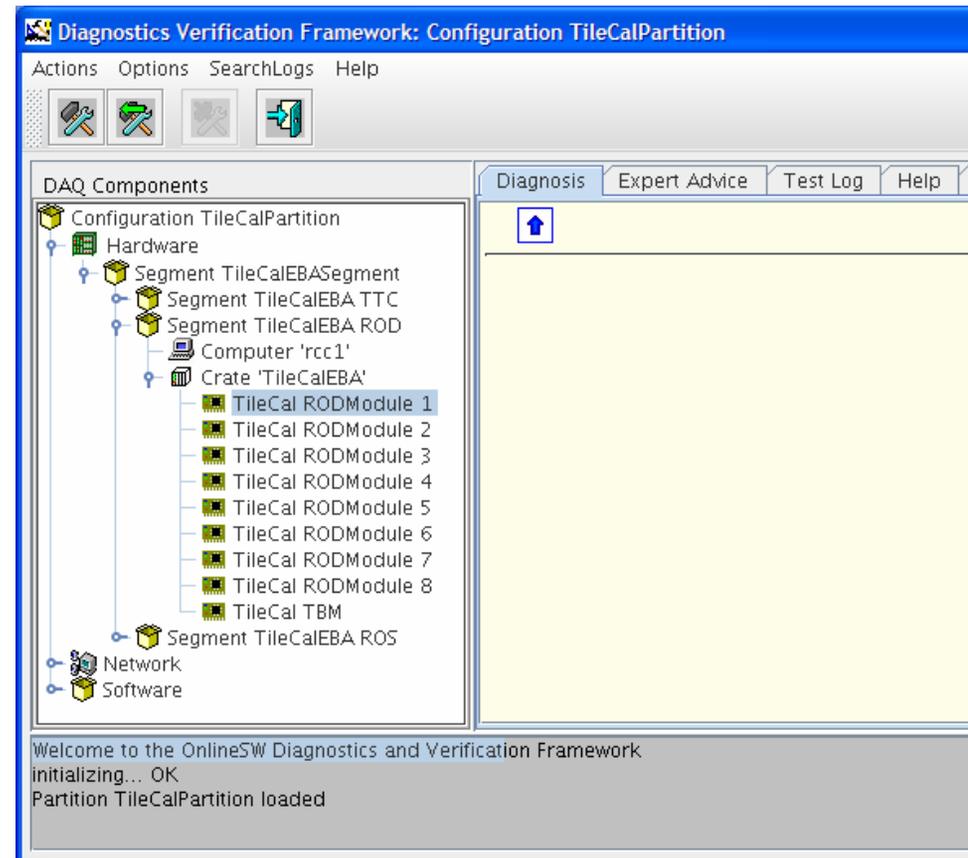


- Normal tests are non-interactive, no input is accepted and an exit code is returned
- New type of interactive tests, called **'actions'**, were introduced to:
 - allow users execute more complex test scenarios, requiring some user's input
 - use already existing console utilities
- Action is configured as a test, but it is launched in a terminal window

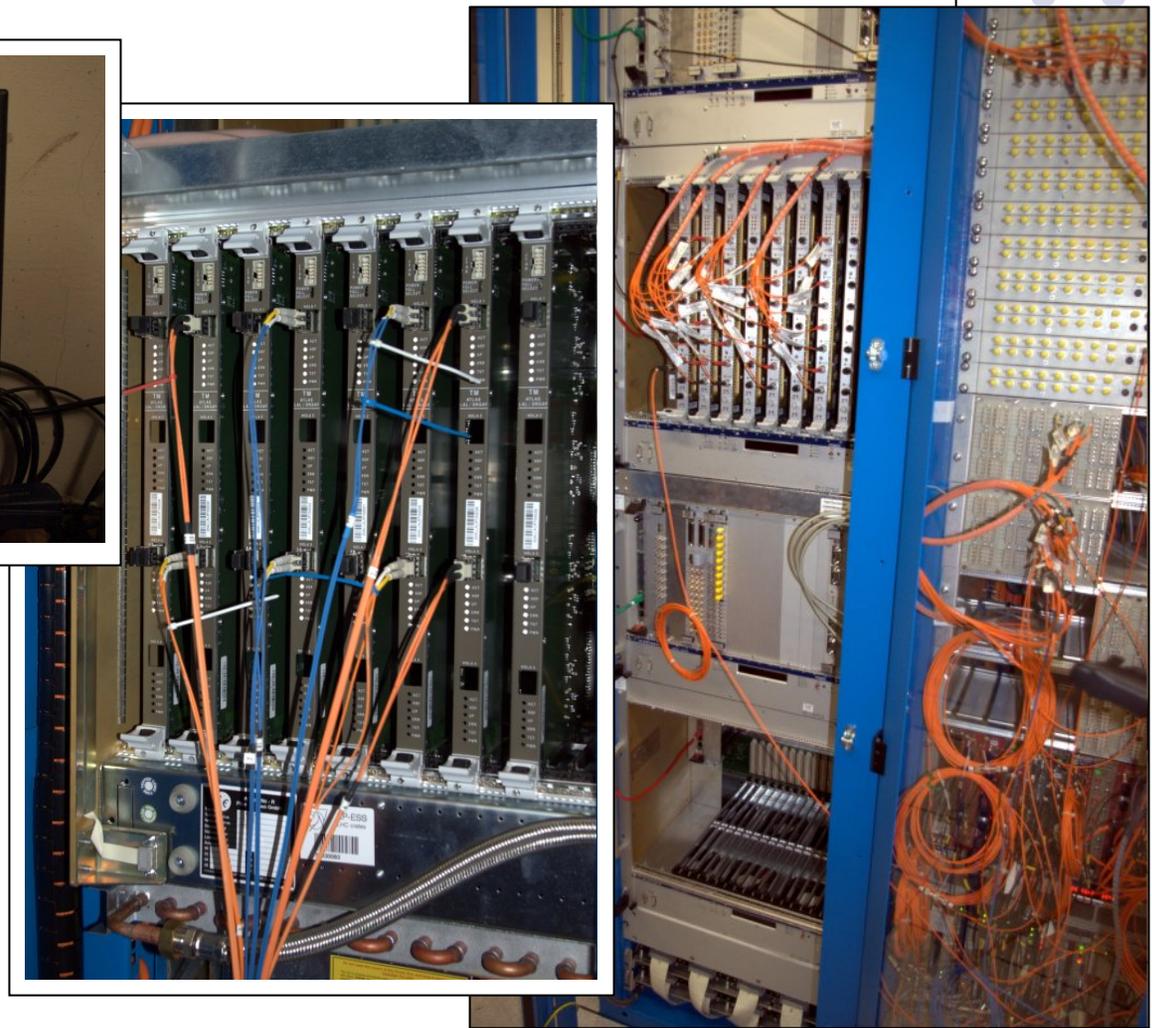
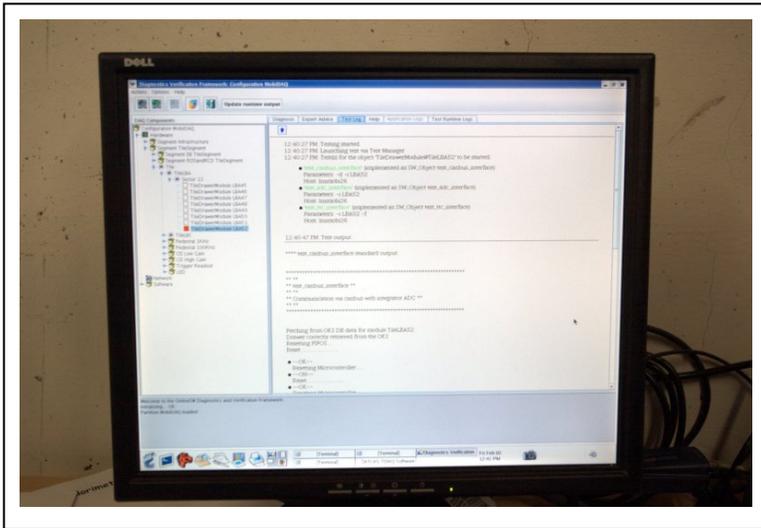
DVS usage for subdetector commissioning



- Developed tests for Tile ROD modules:
 - test_rod_allrregisters: test all ROD components
 - test_rod_local: test Local and Busy components
 - test_rod_oc: test each of the 4 OC FPGAs
 - test_rod_pu: test each of the 4 PUs (Dummy or DSP)
 - test_rod_staging: test each of the 4 Staging FPGAs
 - test_rod_ttc: test the TTC FPGA



'MobiDAQ': DVS-base testing setup for Tile subdetector



http://atlas.web.cern.ch/Atlas/SUB_DETECTORS/TILE/Commissioning/mobidaq/HowTo.htm

MobiDAQ test suit



mobidaq's Home

Start Here
Trash
Terminal

Diagnostics Verification Framework: Configuration MobiDAQ

Actions Options Help

Update runtime output

DAQ Components

- TileDrawerModule LBC48
- TileDrawerModule LBC49
- TileDrawerModule LBC50
- TileDrawerModule LBC51
- TileDrawerModule LBC52
- Pedestal 1KHz
 - TriggerSyncPedestal TrigPedestal
 - RODEMUChanPedestal Drawer-01
 - RODEMUChanPedestal Drawer-02
 - RODEMUChanPedestal Drawer-11
 - RODEMUChanPedestal Drawer-12
 - RODEMUChanPedestal Drawer-21
 - RODEMUChanPedestal Drawer-22
 - RODEMUChanPedestal Drawer-31
 - RODEMUChanPedestal Drawer-32
- Pedestal 100KHz
- CIS Low Gain
 - TriggerSyncCISLG TrigCISLG
 - RODEMUChanCISLG Drawer-01
 - RODEMUChanCISLG Drawer-02
 - RODEMUChanCISLG Drawer-11
 - RODEMUChanCISLG Drawer-12
 - RODEMUChanCISLG Drawer-21
 - RODEMUChanCISLG Drawer-22
 - RODEMUChanCISLG Drawer-31
 - RODEMUChanCISLG Drawer-32
- CIS High Gain
- Trigger Readout
- LED
- Network

Diagnosis Expert Advice Test Log Help

Events remaining 90
Events remaining 85
Events remaining 80
Events remaining 75
Events remaining 70
Events remaining 65
Events remaining 60
Events remaining 55
Events remaining 50
Events remaining 45
Events remaining 40
Events remaining 35
Events remaining 30
Events remaining 25
Events remaining 20
Events remaining 15
Events remaining 10
Events remaining 5
Data taking finished
: created default TCanvas with name
Mean RMS 0.622035
Mean RMS 1.31658
Events 1000 31980us
**** tile_drawer_pedestal error out
Info in : eps file /data/test_results/
been created

Created by "tiledaq" on "pctb-tdaq-on102" at "13/5/04 19:02:21"
Last modified by "mobidaq" on "lnxmobi26" at "13/1/06 19:52:20"

Logical Name Type

Include files

- dal/schema/core.schema.xml
- DAQRelease/sw/tags.data.xml
- tngr/schema/test-repository.schema.xml

Objects

- integ_peds@Test4Class
- test_adc_interface@Test4Class
- test_canbus_interface@Test4Class
- test_detector_dummy@Test4Class
- test_drawer_register@Test4Class
- test_dummy_env@Test4Class
- test_hv_channels@
- test_hv_interface@
- test_int_ped@Test
- test_integrator_interface@Test4Class
- test_root_interfa
- test_trigger_inter
- test_ttc_interfac
- tile_drawer_CISHG
- tile_drawer_CISLG
- tile_drawer_LED@T
- tile_drawer_pedest
- tile_drawer_reado
- tile_drawer_trigg
- tile_drawer_trigg
- tile_drawer_trigg
- tile_drawer_trigg

Ok's Object : Test4Class@test_integrator

Object ID: test_integrator_interface

Class: Test4Class

Data File: /home/mobidaq/public/databases

Data:

Attributes:

timeout 1000

parameters -q -g0 -i #this.Name -f

host lnxmobi26

exec_mode sync

init_timeout 0

diagnosis

advice

scope any

complexity 2

test_mask integrator

interactive false

OK Apply Cancel

Welcome to the OnlineSW Diagnostics and Verification Framework
Initializing... OK
Partition MobiDAQ loaded
Test verbosity switched ON
You have set the global complexity level to: 2
You have set the global complexity level to: 0
Starting ConfDB GUI: 'oks_data_editor partition/MobiDAQ.data.xml'

22

CHEP 2006 Mumbai India 13-17 February 2006

A.Kazarov 'A rule-based control and verification framework for ATLAS T/DAQ'

MobiDAQ in action

Summary



- ATLAS T/DAQ Control system is a distributed framework, based on expert system technology
- Behavior of the system is described in rules
- It includes configurable framework for test description and execution (DVS)
- It is widely used in ATLAS commissioning