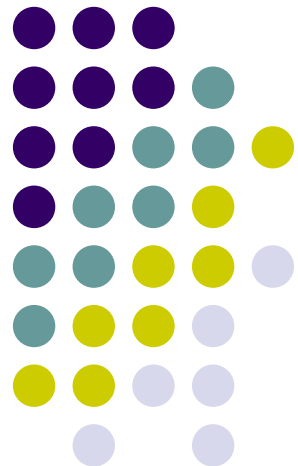
A rule-based Control and Verification framework in ATLAS Trigger-DAQ

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

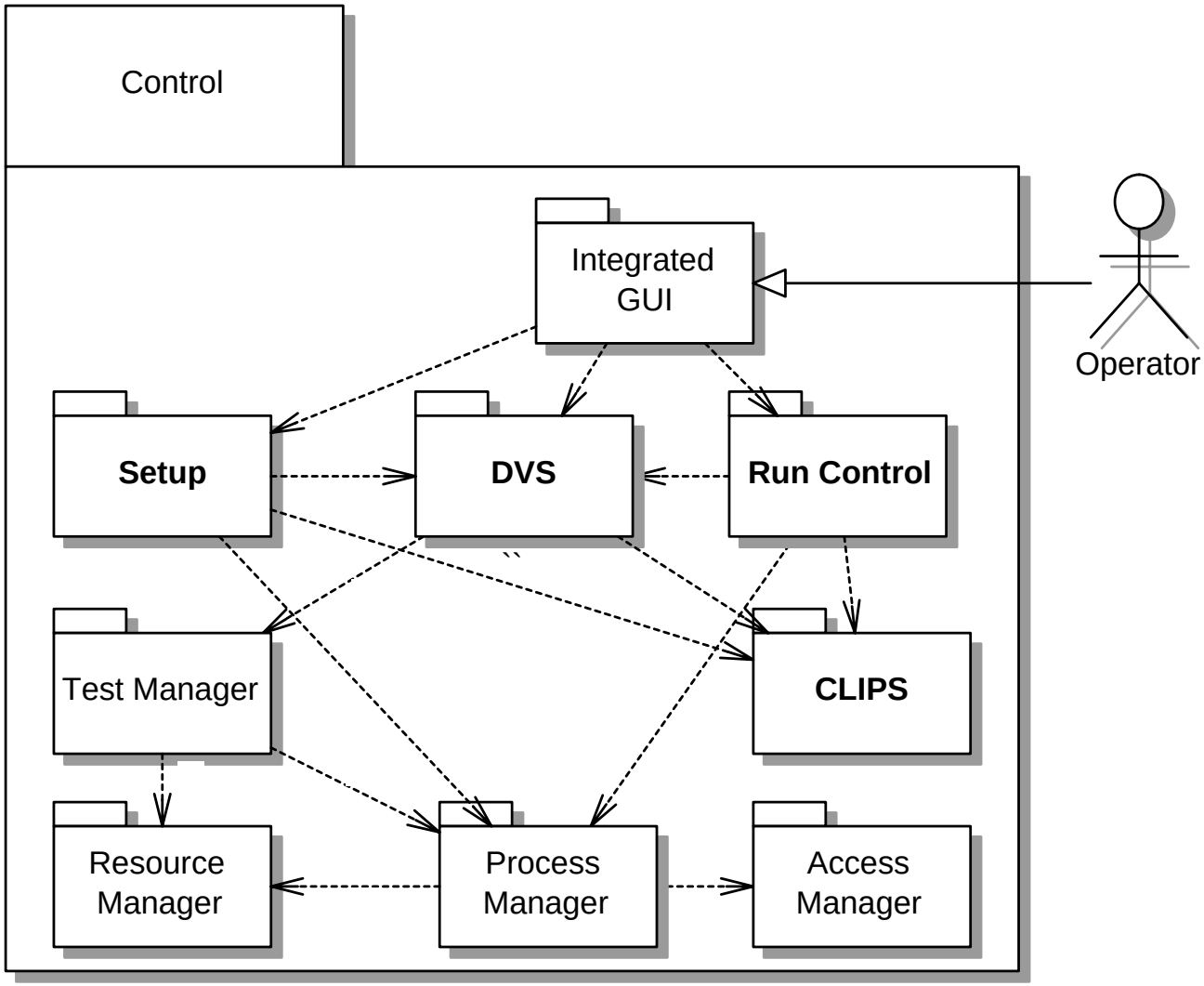
With the given system size, h/w and s/w failures are very probable, and it is very important to have testing and diagnostics facilities embedded in the Control System in order to:

- 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

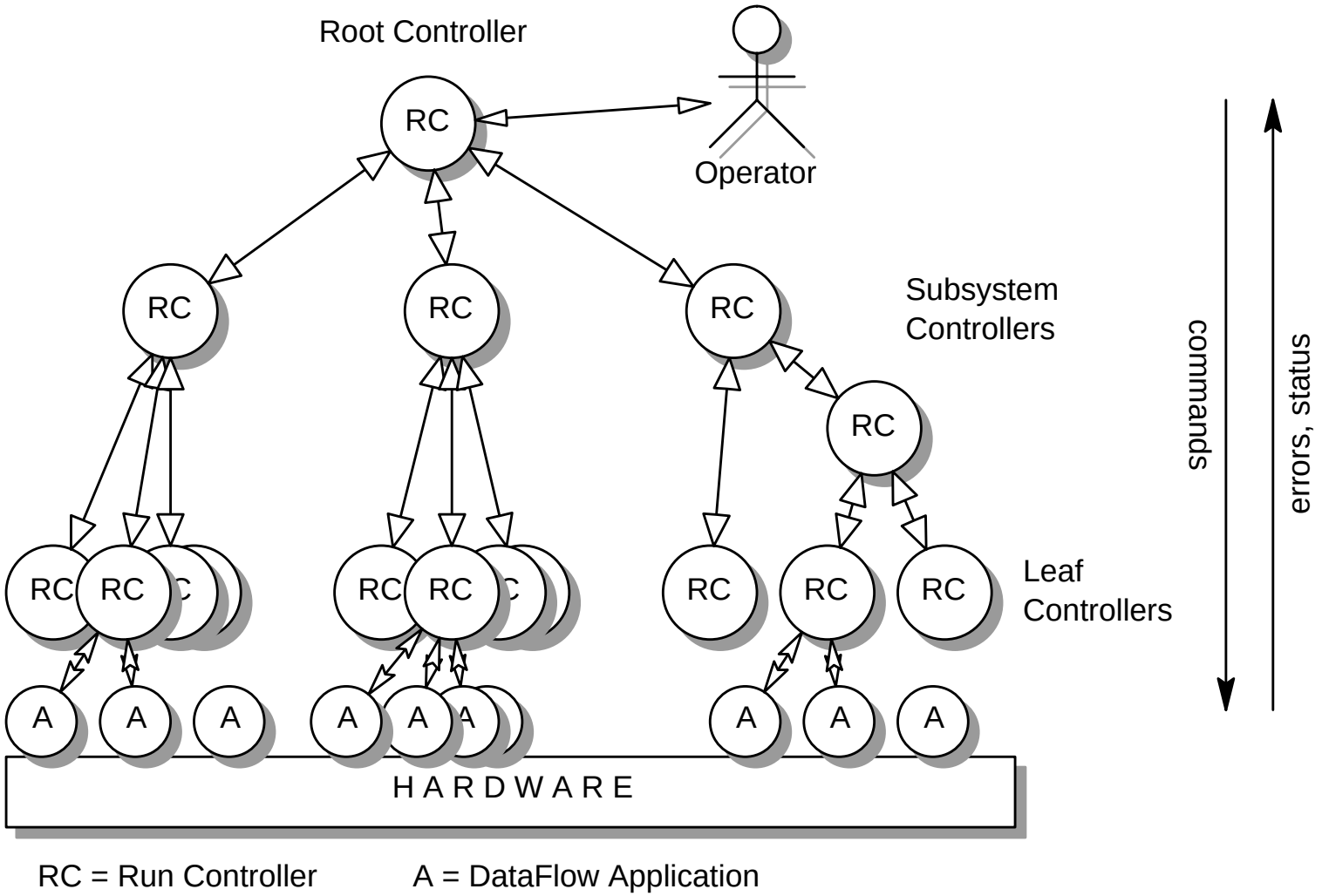
Design principles

- **Framework** approach: system shall be configurable and extensible by experts and users, also during the experiment life-time
- **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



RC = Run Controller A = DataFlow Application

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

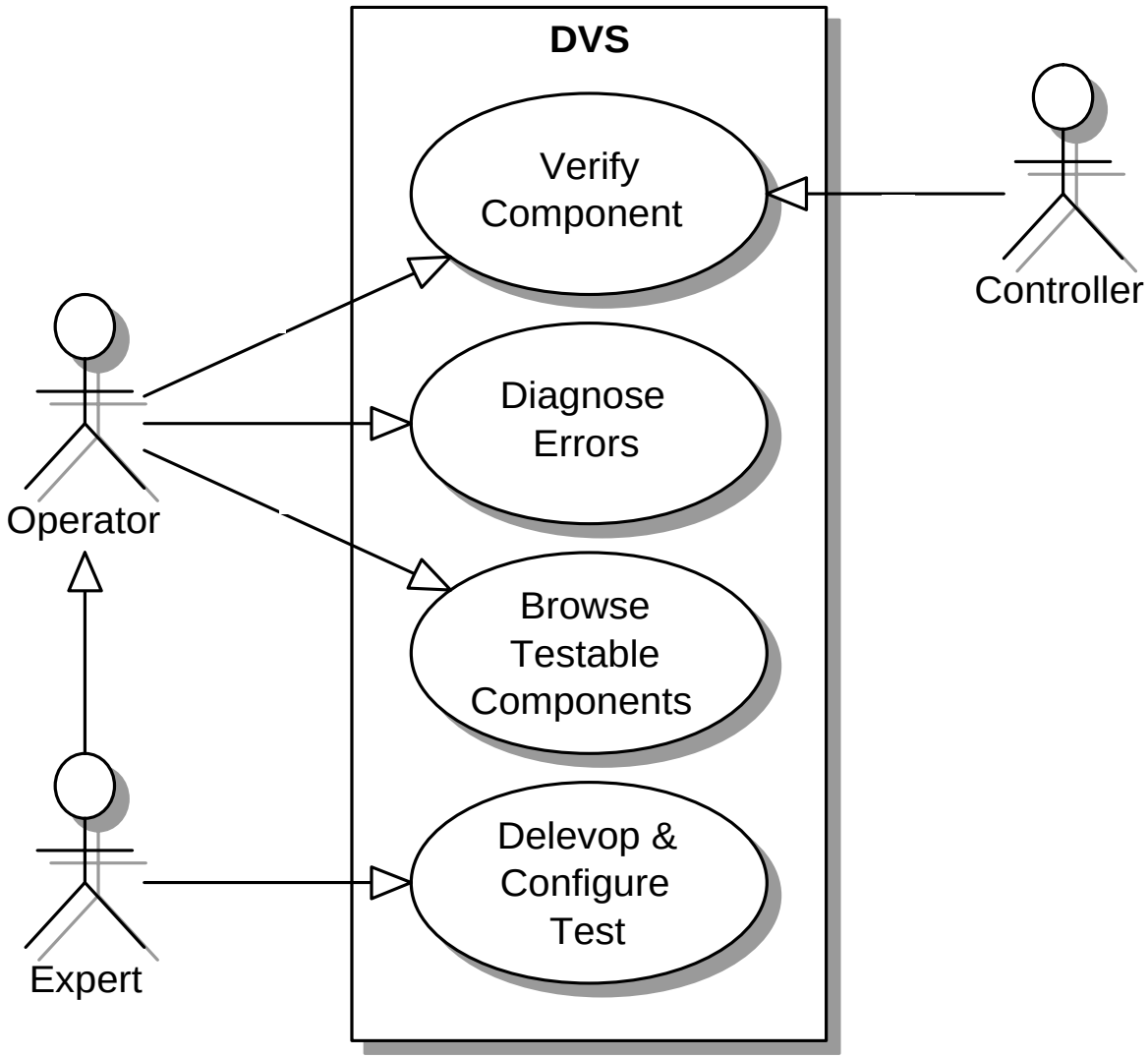
CLIPS: expert system shell

- '**C**'-**L**anguage **I**ntegrated **P**roduction **S**ystem
- 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

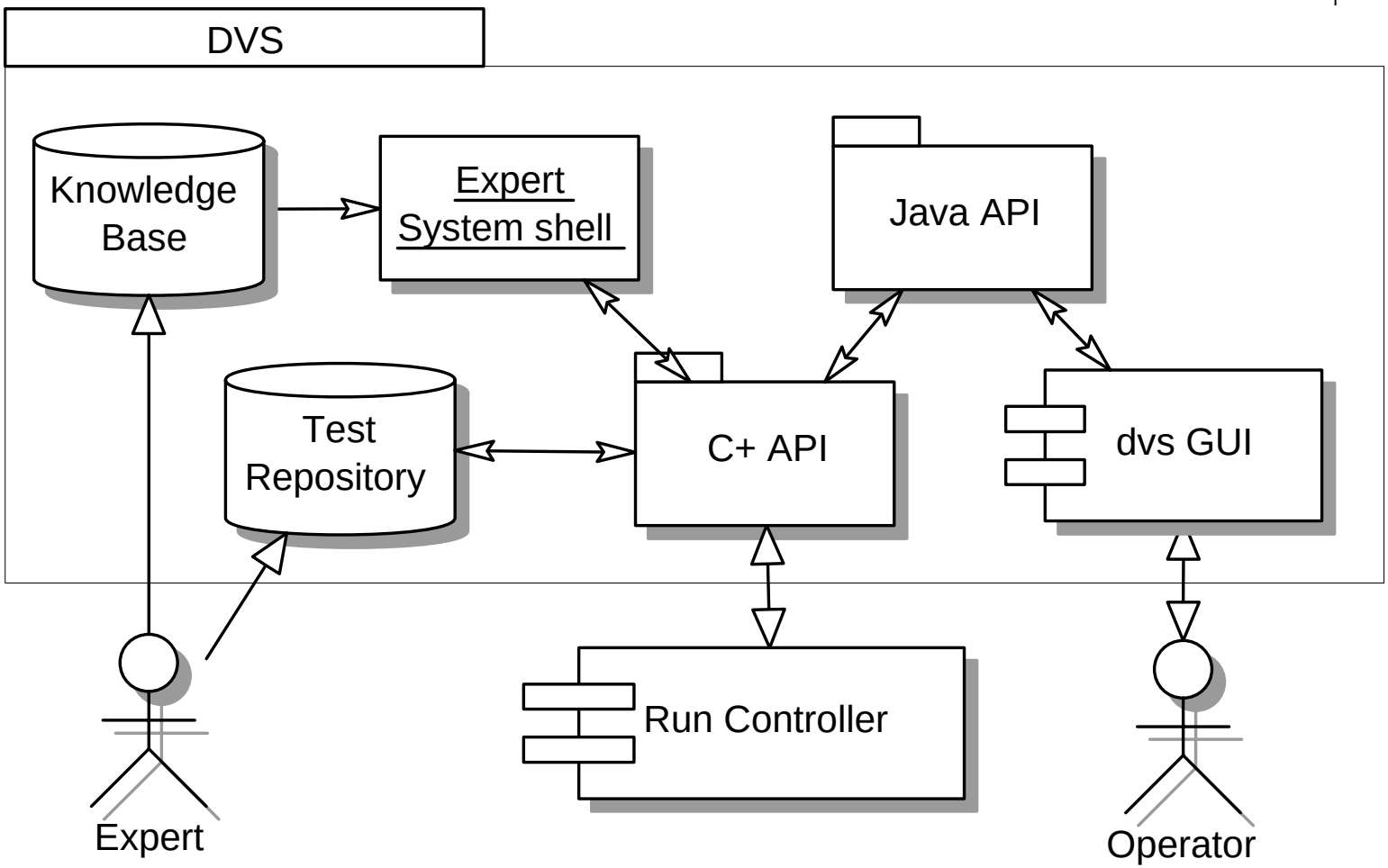
Part II: DVS, diagnostics and verification framework

- 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

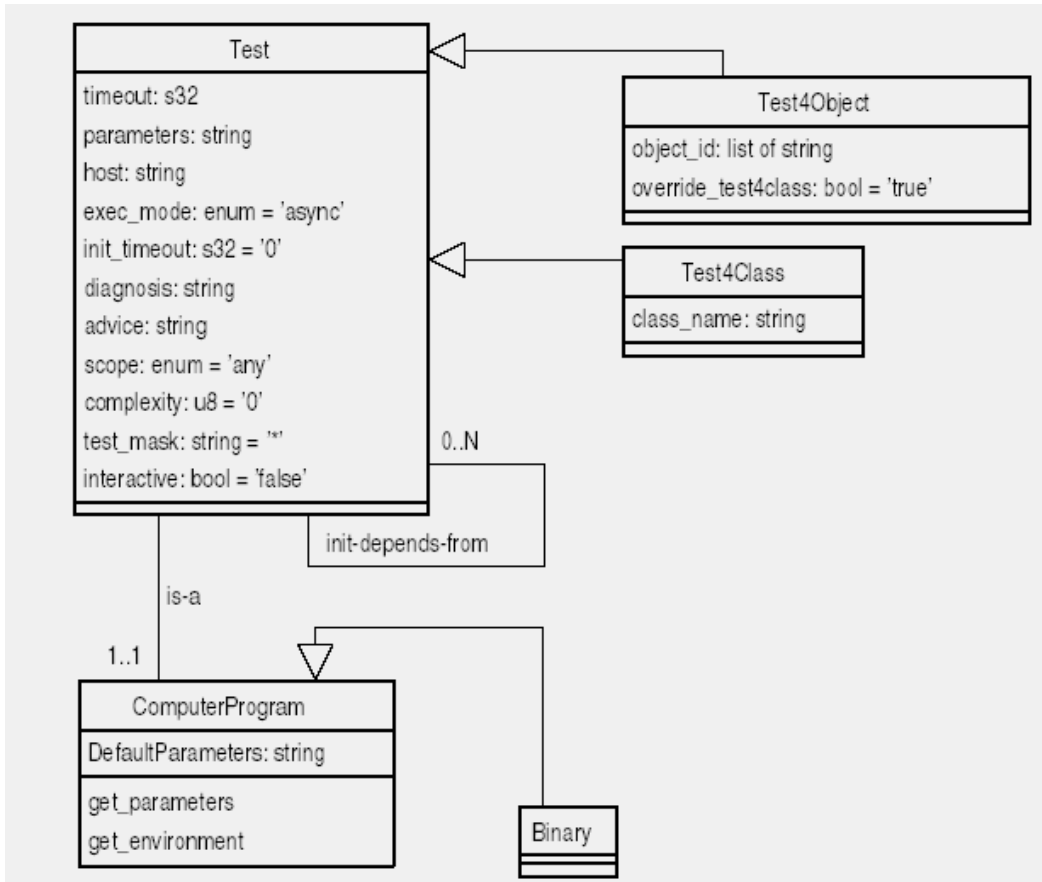


DVS architecture

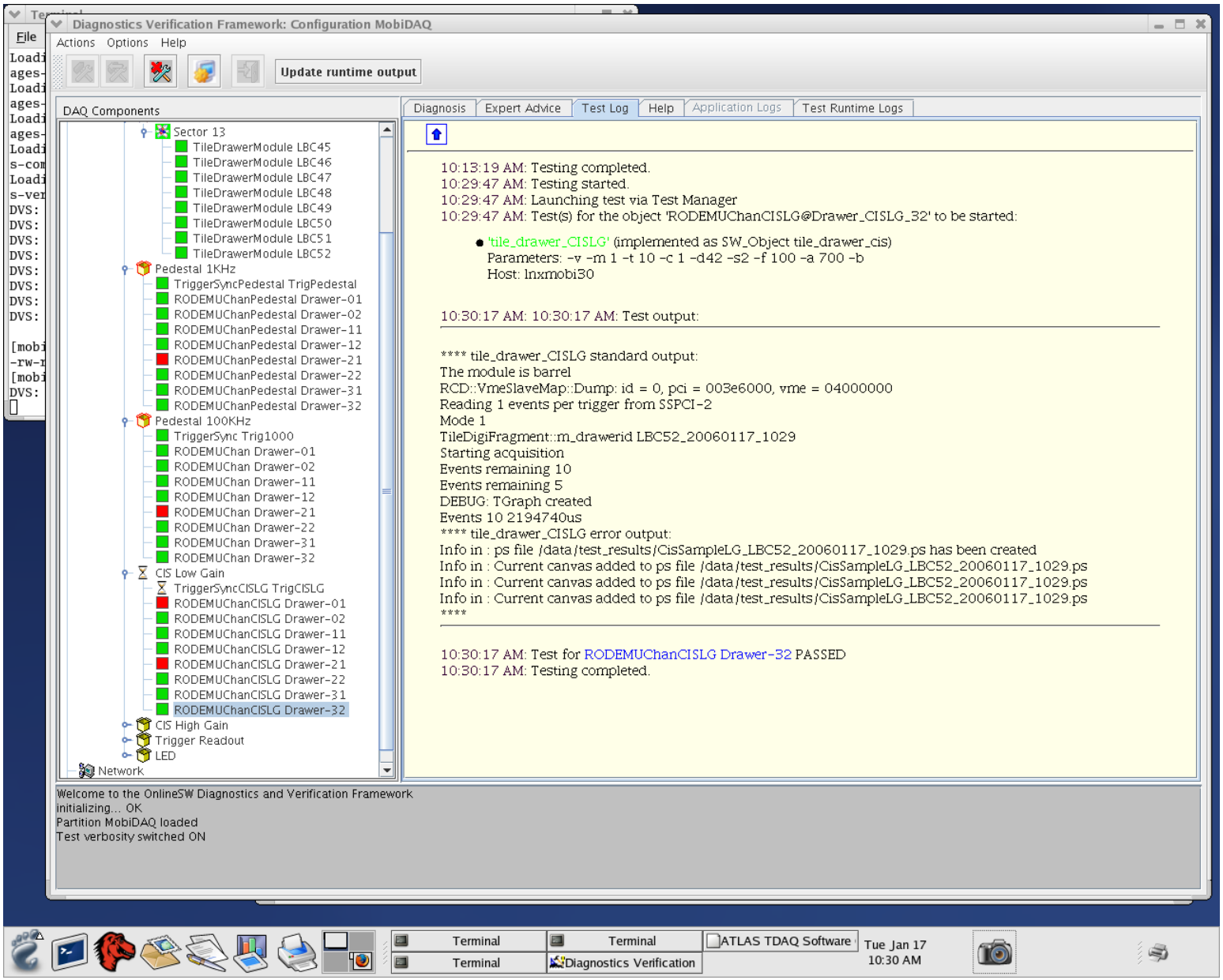


What is a test

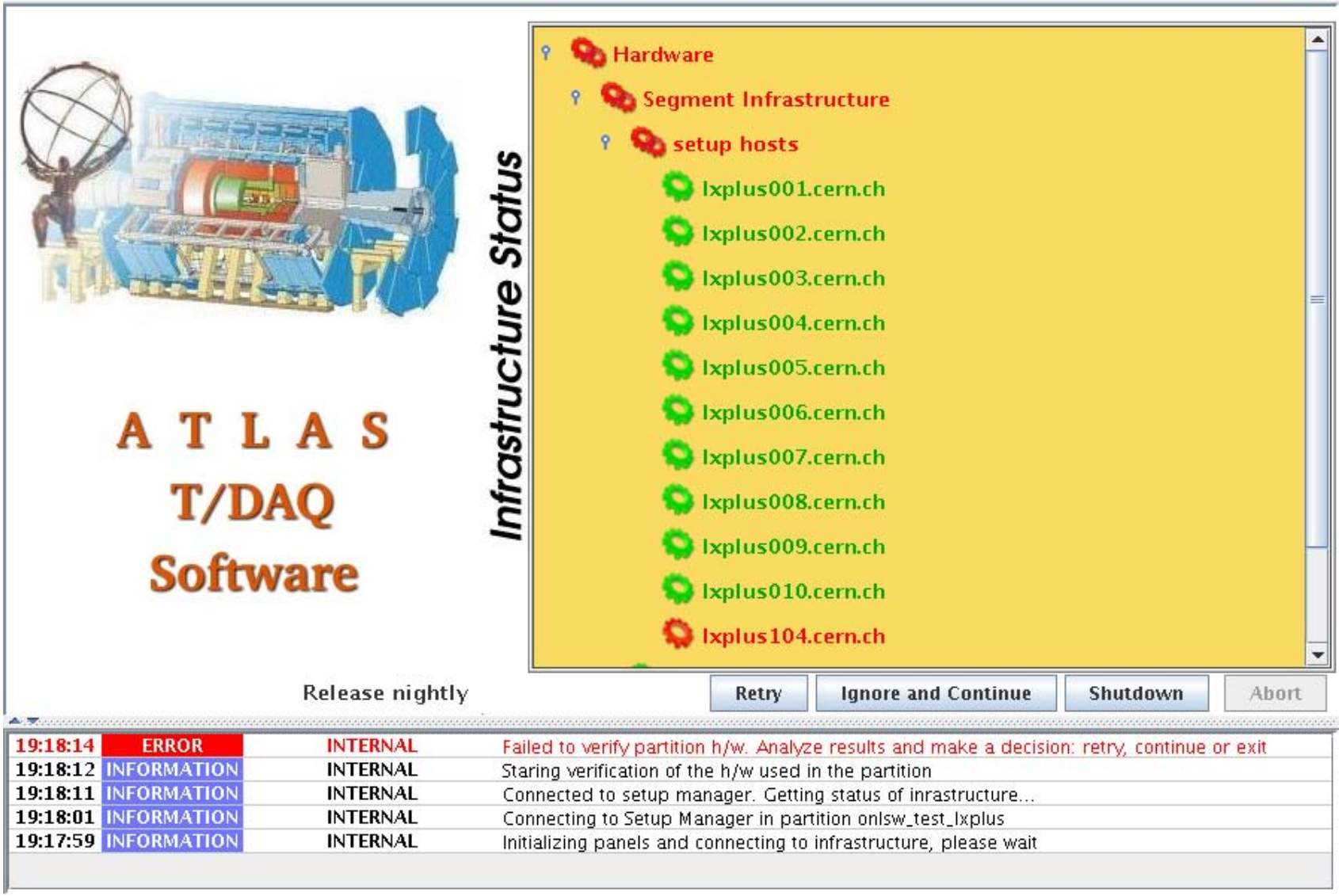
- Test is a binary, running on a particular host in a system
- Test verifies a particular functionality of a TDAQ component
- For a single component, a number of tests can be associated
- Test returns a value: PASSED, FAILED, UNRESOLVED, TIMEOUT
- Tests can be organized in sequences, executed synchronously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



DVS for end-users



Use of tests from Setup



New features:

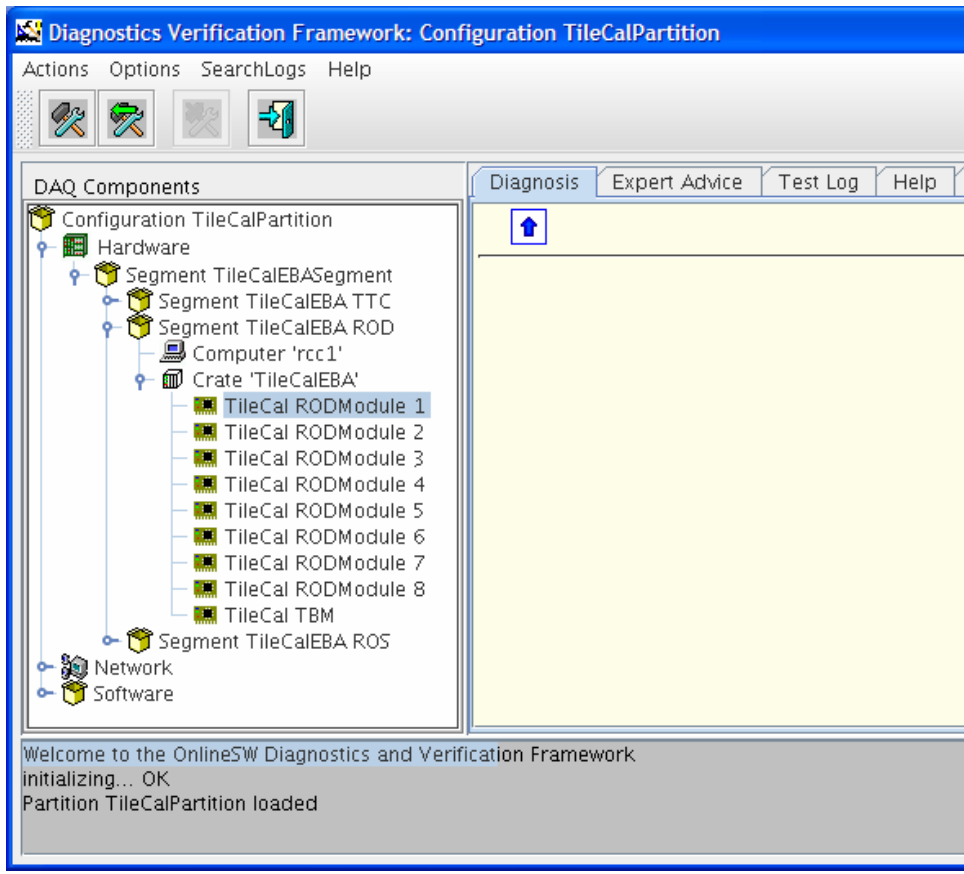
- Tests *levels* and *masks* for more precise test selection, which allows to promptly configure test repository without editing the database
- Asynchronous and synchronous mode for execution of tests for complex objects
- Test *scope* to prevent conflicting tests from being executed when system is taking 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

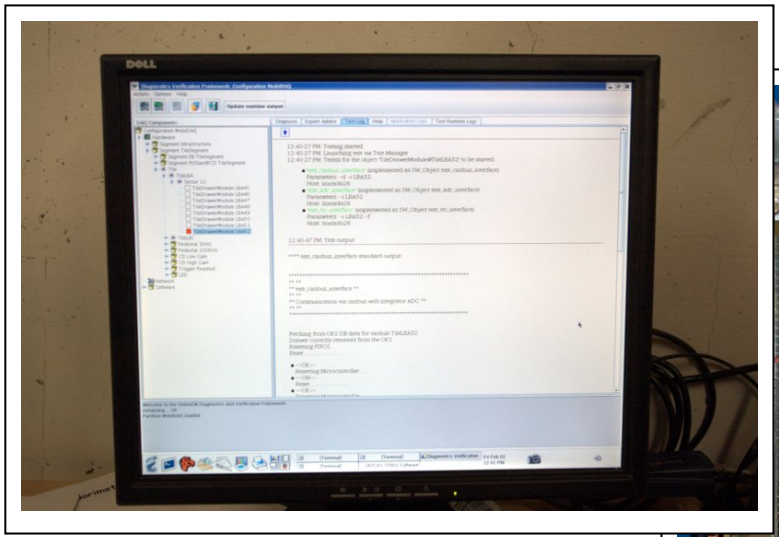
- Normal tests are non-interactive, no input is accepted and am exit code is returned
- New type of interactive tests, called '**actions**', were introduced to:
  - allow users execute more complex test scenarios, requiring some user's input
  - use already existing console utilities
- Action is configured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

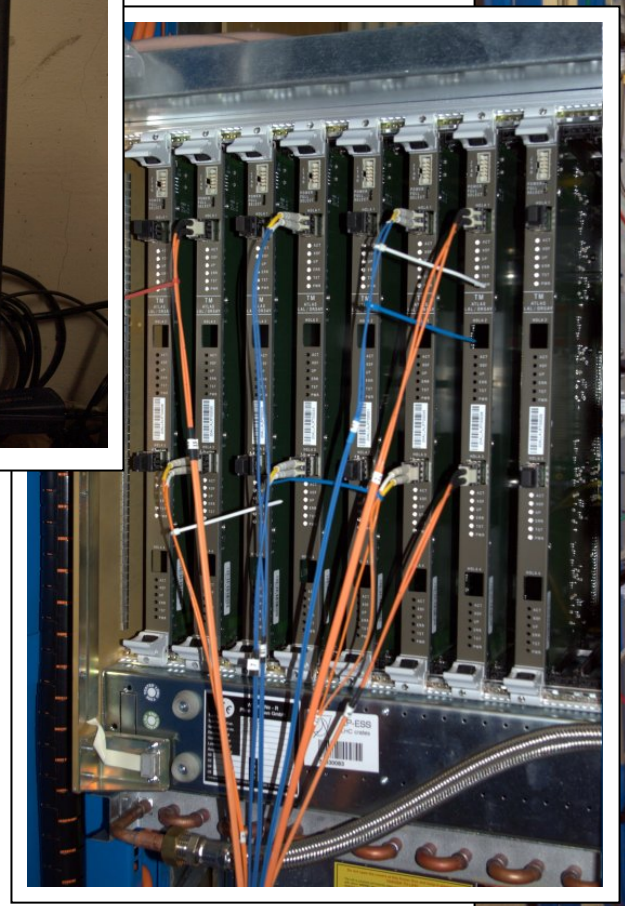
DVS usage for subdetector commissioning

- Developed tests for Tile ROD modules:
  - test_rod_allrwregisters: 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



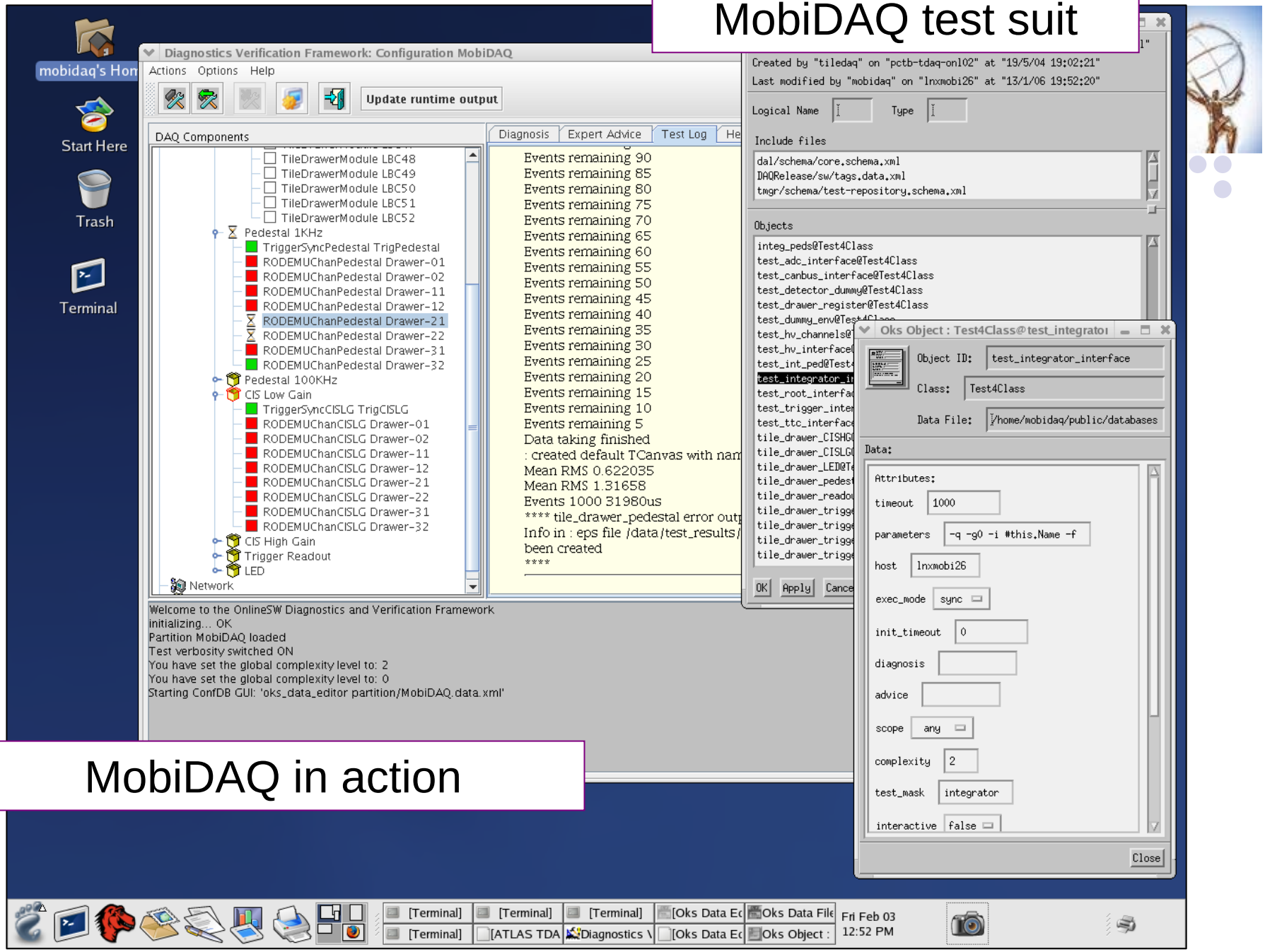
‘MobiDAQ’: DVS-base testing setup for Tile subdetector





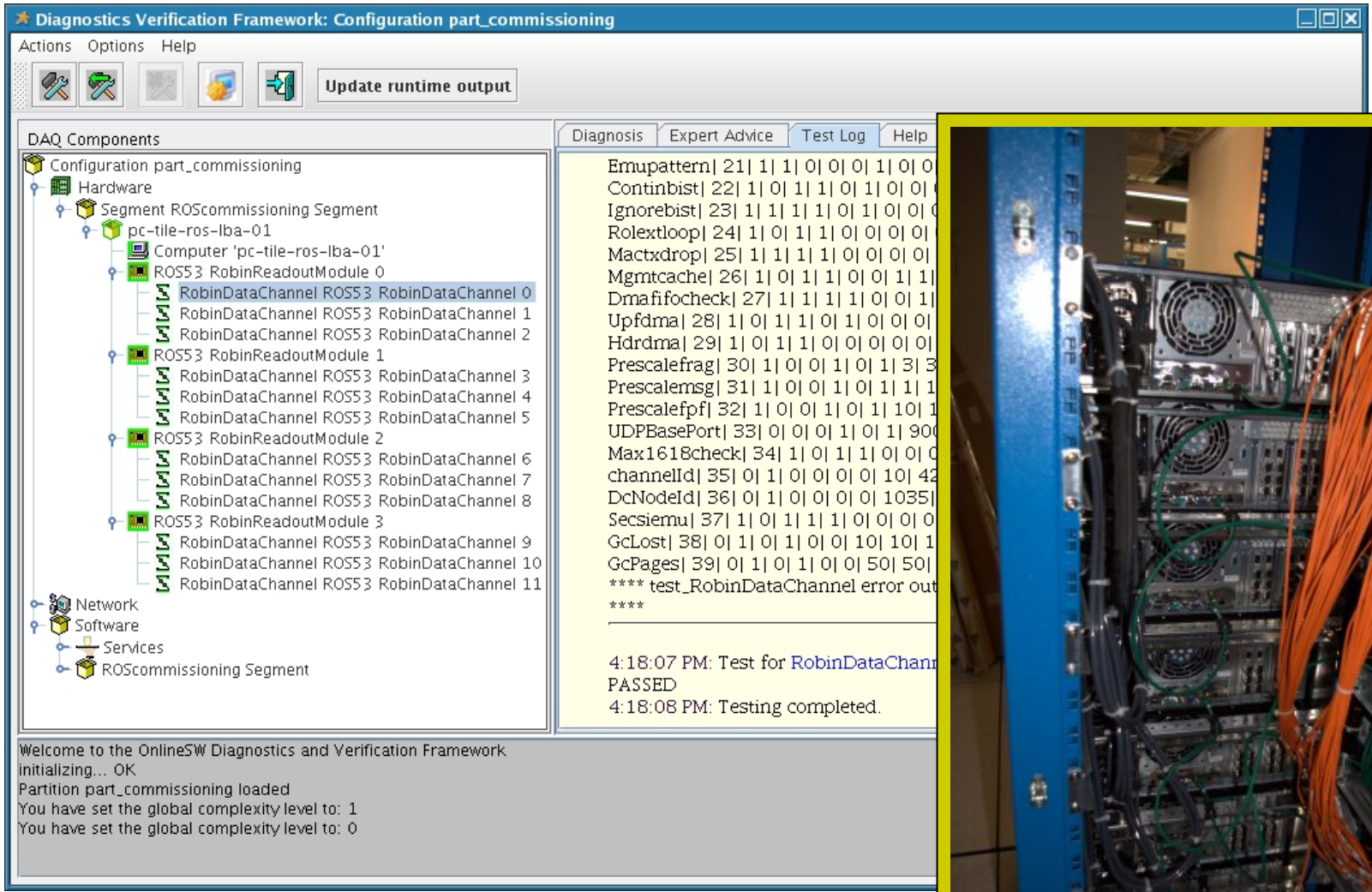
http://atlas.web.cern.ch/Atlas/SUB_DETECTORS/TILE/Commissioning/mobidaq/HowTo.htm

MobiDAQ test suit



MobiDAQ in action

DVS for ROS commissioning



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