EVIDENCE – A control system for laboratory applications and small-scale experiments

Oliver Grimm – *ETH Zürich, Institute for Particle Physics, Schafmattstrasse 20, 8093 Zürich, Switzerland – oliver.grimm@phys.ethz.ch*

Overview

Experiment control systems (ECS) typically comprise several individual programs that

- require configuration information,
- produce data,
- need to exchange information,
- visualize data of other programs and monitor system health.

EVIDENCE

ECS functions with minimum extra coding – uses established tools – DIM communication layer – lean and easy to adapt – centralized

EVIDENCE is designed for small-scale experiments for which established and comprehensive control systems (e.g. EPICS, DOOCS or PVS-II) are too complex.

The centralized approach defines the range of applications: for large systems, decentralization might be better.

EVIDENCE Functionality

A small number of server programs (the ECS backbone) contain the basic functionality.

They use a standard C++ class suggested also for application programs: allows easy central system surveillance (message logging and error severity monitoring) and an efficient way to access configuration information.

DIM Name Server

Central instance of DIM for communication between processes

IP address needs to be known to all DIM servers and clients

Configuration Server

Supplies configuration information to servers

Configuration read from single text file (format `.INI` style)
Clients informed automatically on changes to the file

```
[SQM]   # Sky Quality Monitor
address =       sky.quality.com
port    =       4711
Period  =       30
```

Data Logger

Central data, message and error logging

Subscribes dynamically to all DIM services (except if excluded) and writes all updates to text file

Alarm Monitor

Monitors control system health

Checks severity of server message services and for existence of servers
Generates master alarm (requiring explicit acknowledge to reset)
Can send email if server has error or is down

History Server

Provides recent data (histories) of all DIM services

Data kept in memory ring buffers for quick access
Used mainly for user interface

Bridge

Transmits DIM data, commands and procedure calls between 2 DIM networks

Can limit network load on main servers (e.g. from multiple GUIs)
Allows blocking of commands and procedure calls according to IP address

Functions of Evidence class

Starts DIM server

Provides textual status service SERVER_NAME/Status
Severity encoding (INFO, WARN, ERROR, FATAL)

Provides method for configuration request.
If data not available and no default, terminates with FATAL message
Allocated memory freed upon class destruction
Requests over network only send in case configuration file changed

Provides method for translating DIM service safely into text

Provides DIM exit and error handler

Installs signal handler
SIGQUIT (Ctrl-Backspace),, SIGTERM, SIGINT (Ctrl-C), and SIGHUP (terminal closed)

Catches un-handled C++ exceptions
gcc-style termination handler

Allows browsing of data returned by History server

Server programming example

```
#include "Evidence.h"

// Class declaration
class MyServer: public EvidenceServer { ... }

// Constructor
MyServer::MyServer(): EvidenceServer(SERVER_NAME) { ... }

// Request configuration data
char *DataDir = GetConfig("datadir");

// Create service
int SizekB = 0;
Service = new DimService(SERVER_NAME "/SizekB", SizekB);

// Update all clients
Service.updateService();

// Set warning message
Message(WARN, “SizekB is %d”, SizekB);

// Write something to the central log file
SendToLog(“Did you know SizekB was so large?”);
```

CERN's DIM

Distributed Information Management

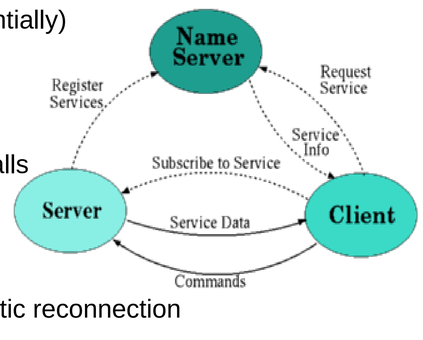
Communication system for distributed/mixed environments
Provides network transparent inter-process communication layer

C++, C, Fortran, Java and Python interfaces

Little extra coding required (work handled by DIM library)

Thread-based (data handlers internally executed sequentially)

Access control with standard tools (e.g. IPtables)



Server Provide data, commands, procedure calls

Clients
Subscribe to data
Send commands
Make remote procedure calls
Notification if a server dies and automatic reconnection
Individual TCP/IP connection to server

Name server
Provides service information and addresses
Involved transparently by DIM library
Not involved once connection between server and client established

References: C. Gaspar, M. Dönszelmann and Ph. Charpentier, Computer Physics Communications **140**, 102 (2001)
Web address http://dim.web.cern.ch/dim/

Graphical User Interface

Evidence Data Display

uses Qt and Qwt for visualization – display DIM Services and histories – sending of generic DIM commands – subscribes to DIM service only once

Qt and Qwt tool-kits available free of charge (LGPL)

Single subscription to DIM service limits server load if multiple widgets use same data

Easily portable: no operating-system specifics used, only standard C++ code, Qt/Qwt capabilities and DIM

GUI building follows standard Qt programming procedures.

References: Qt home page: http://qt.nokia.com/ Qwt home page: http://qwt.sourceforge.net/

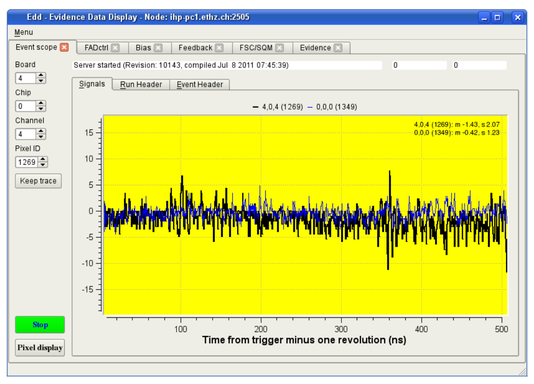
GUI examples from FACT project

First G-APD Cherenkov Telescope

Evidence used for the first test module and initial camera commissioning

Main window

Access to all subsystems via detachable tabs
2 signal traces with statistics (mean and sigma) shown



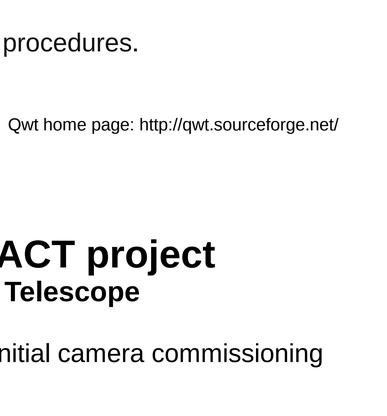
Service history window

Initial history obtained from History server (thus available on all GUI instances), subsequent accumulation by GUI

Stripchart mode keeps constant time span

Add traces by drag&drop or copy/paste

Extensive plot manipulations
- zooming along all axes, panning
- saving and printing

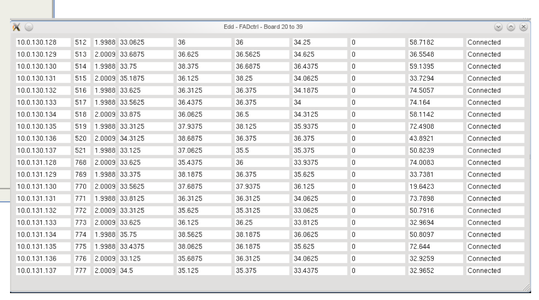


Monitoring main *Evidence* servers

Alarm status or unavailability of alarm server announced by pop-up window, optional acoustic alarm

Overview of 20 digitizer boards

Clicking on widget brings up history
Details of DIM service shown in bottom status bar when mouse over widget



Acknowledgment: The help of Clara Gaspar (CERN) was indispensable for understanding the DIM system and making good use of it.
EVIDENCE was developed within the FACT project (First G-APD Cherenkov Telescope)