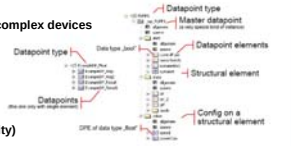
Controlling a large Data Acquisition System using an industrial SCADA system

Stefan Koestner (CERN)
on behalf of the LHCb Online Group

PVSS – an industrial SCADA toolkit:

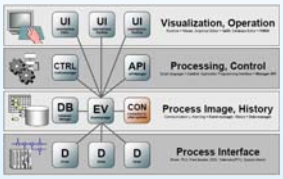
German abbreviation for 'Process visualisation and control system'.
Scalability is essential for big experiments!!!

- structured namespace (device oriented) - create and manipulate complex devices
- no built-in limit for number of devices and elements
- Data held in memory of event manager and real time DB – accessible from all managers
- attributes used for processing and alerts (configs)
- complex value archive arrangements and flexible alarm handling
- interface to record archive data in an Oracle database (accessibility)



- communicating processes:
  architecture can take advantage of multi-CPU systems
  load distribution is possible
- distributed systems:
  systems may be distributed across machines
  various autonomous systems may communicate with each other
  allows for hot-standby redundancy
- cross-platform:
  truly mixed systems possible (Windows GUI & Linux data server)

- highly modular design:
  specific managers (processes) for different tasks
- client-server architecture (provider-consumer relationship):
  communication between processes is event-driven
  standard TCP/IP message interface
  active only on demand (publish/subscribe)
- C-like scripting language (procedural higher level language)
- API interface/manager for extra functionality (C++ library)



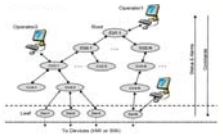
JCOP – CERN's joint controls framework:

JointCOntrols Project – a common toolkit designed for the use for LHC

- additional drivers for commonly used devices (especially high voltage power supplies and CAN bus)
- interface to an Oracle configuration database to upload sets of parameters (recipes) to hardware
- API managers for communication protocols e.g. DIM and OPC
- implementation of a hierarchical control structure using Finite State Machines
- coherent interface

Modelling the Experiment Control System (ECS) with Finite State Machines

- in order to model the experiment at an expert system level states and actions have to be defined
- from the top node (run control) commands can be propagated downwards a hierarchical tree
- status and alarms can be propagated upwards the hierarchical tree to the run control
- the final branch of a tree is always a device unit acting on the hardware (connected to datapoint!)
- additional control units can be implemented to give a better logical structure (domains)

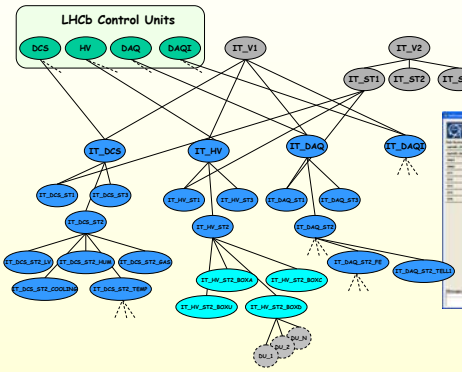


Implementation

- each domain has a defined transition from one state into the other with the possibility to autorecover (in case of error)
- change in hardware (datapoint) can trigger a state transition of a device unit
- rules defined for control unit if its children make a state transition
- commands can cause state transitions and act on datapoint of device unit, triggering itself a callback function

Partitioning

- to allow for more flexibility, subparts of the tree can be excluded (either its state is ignored or commands are not allowed)
- helps to commission subdetectors or to ignore faulty hardware



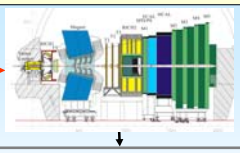
User Interface

- each device unit offers a set of panels to configure and interact with the hardware (datapoints)
- the control unit panel from where commands can be launched shows its own and the children's state

The LHCb experiment:

Single-arm forward spectrometer
dedicated to B-physics

- pp@14 TeV, luminosity=$2 \cdot 10^{32}$ cm^{-2}s^{-1}
- acceptance: 15-300(250)mrad
- 10^{12} bb/year, full B spectrum



L0 trigger
~ 1 MHz, 4 µs fixed latency
Time synchronized to FE-chips

L1 electronics:

- 1 MHz readout (~35 GB/s)
- collection and preprocessing
- push-protocol supposes enough buffer in next stage

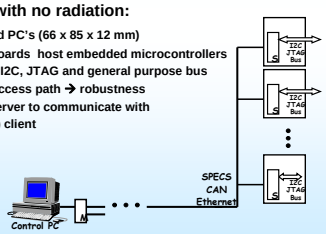
Electronics Interface:

In radiation areas:

- SPECS (Serial Protocol for ECS)
  custom protocol made for LHCb to work in radiation and B-field environment for remote electronics
  10 MB/s, slave is radiation tolerant
  PCI master card can talk with up to 32 slaves
  generic server to communicate with ECS (DIM) client

In areas with no radiation:

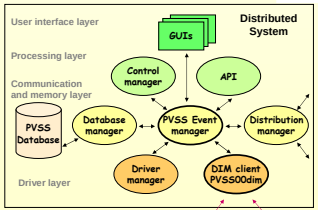
- Credit Card PC's (66 x 85 x 12 mm)
  readout boards host embedded microcontrollers
  access to I2C, JTAG and general purpose bus
  isolated access path → robustness
  generic server to communicate with ECS (DIM) client



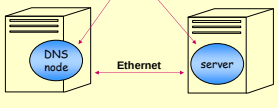
DIM – Distributed Information Management System

- portable lightweight communication layer to interface hardware from ECS
- if server crashes it can easily republish on DNS node (Robustness)
- clients can be installed on any machine just specifying DNS node (Portability) – no need to take care of connectivity

- DIM server running on CCPC or SPECS master PC publishes services to DIM Name Server (DNS) from where the client (ECS) can subscribe to it.
- Data exchange peer to peer from server to client
- Client sends commands (write/read) – server updates services (data/status)



- The PVSS software architecture comprises a number of managers which may run on independent machines and communicate with the event manager via a PVSS internal protocol on top of TCP/IP
- a DIM driver is implemented which allows to communicate with hardware over ethernet

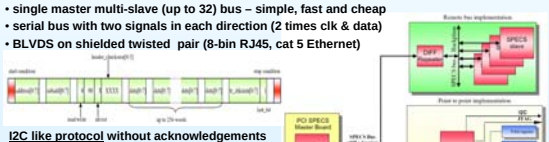


Support PC running DNS node

SPECS master PC running SPECS server or embedded controller running CCPC server

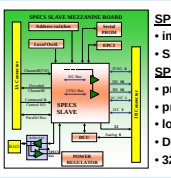
SPECS – Serial Protocol for the Experiment Control System

- 10 Mbit/s serial link for configuration of remote electronics
- single master multi-slave (up to 32) bus – simple, fast and cheap
- serial bus with two signals in each direction (2 times clk & data)
- BLVDS on shielded twisted pair (8-bin RJ45, cat 5 Ethernet)



I2C like protocol without acknowledgements

- slave can send interrupt and master repeats
- clock only active during transfer (10 MHz)
- variable number of words (<256) with 10 bits



SPECS slave designed as portable VERILOG code

- implemented in ACTEL APA150 FPGA (tested up to 40 krad)
- SEU and SEL immune due to triple voting and one-hot state used

SPECS mezzanine board houses the SPECS slave

- provides 16 long distance JTAG chains (4 signals each)
- provides 15 long distance I2C busses with selectable frequency
- local i2C bus and 16 bit parallel bus (8 bit address range)
- DCU chip with 6 ADC channels (12 bit) and 1 temp sensor
- 32 configurable I/O lines

SPECS multi-master board – 4 different busses

- standard 32-bit 33 MHz PCI board
- inserted on the same PC where SPECS server is running

CCPC – Credit Card sized PC

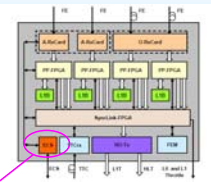
- 15 different types of DAQ and trigger boards (in total some 400)
- a few hundred registers each (monitor) and memory blocks (upload)
- common readout board (Tell1) based on FPGA technology to adopt specific needs

FPGA-based readout board

- ADC conversion or optical conv.
- synchronisation, reordering, pedestal subtraction, common mode sub., zero suppression (PP)
- MEP packing, IP framing (SL)
- Gigabit Ethernet Link

access to board via gluecard over a PLXPCI9030 bridge:

- Parallel bus (8/16/32), 3 fast JTAG chains (2 MHz) to load firmware, 4 I2C lines and 9 GPIO lines.
- low level libraries under SLC4



Separated ECS interface (CCPC)

- 4" – SM520PCX from Digitallogic
- i486 compatible microcontroller (AMD ELAN 520)
- Linux Kernel at 133 MHz
- reading from local bus 20 MB/s
- filesystem shared over server