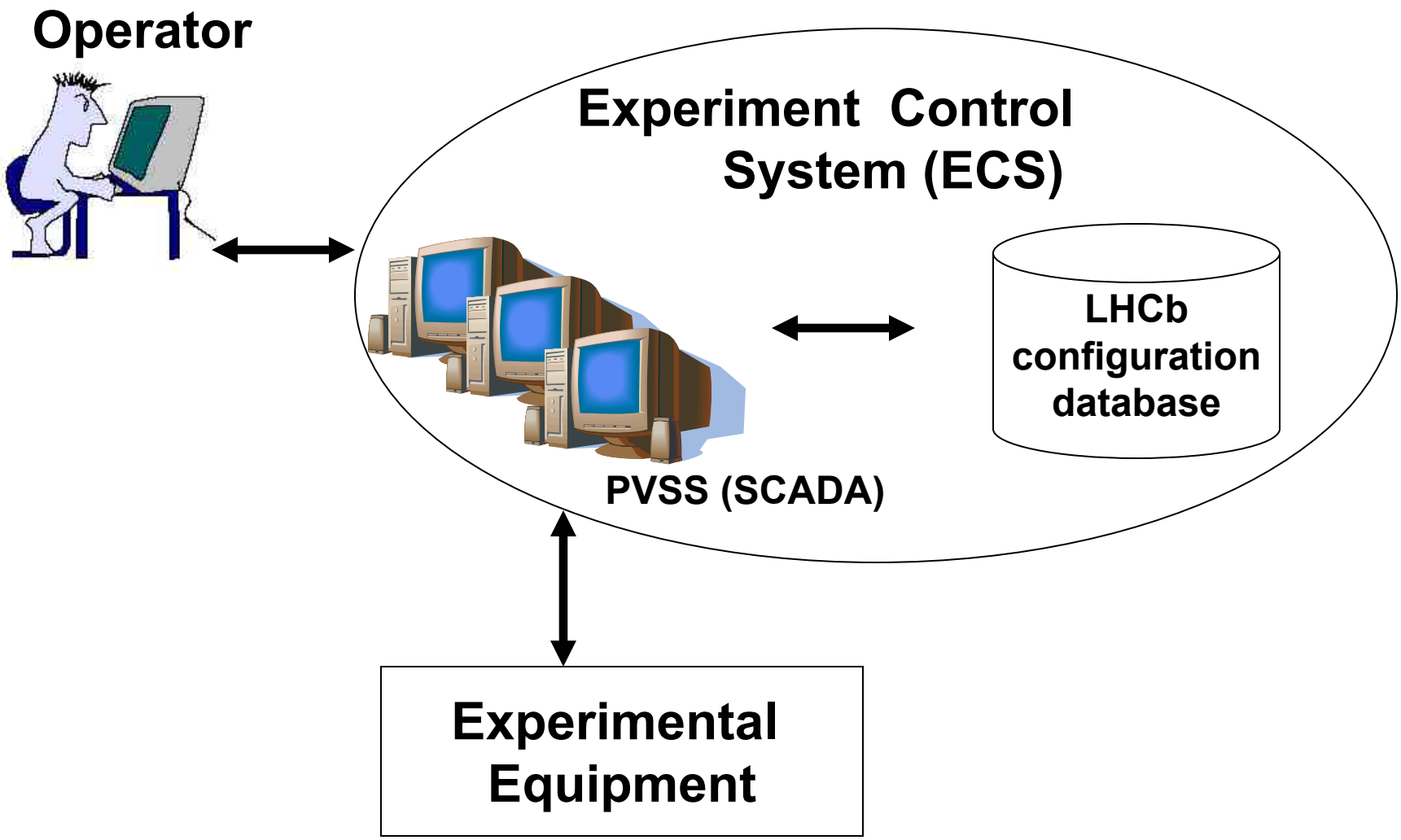
LHCb Online Configuration Database

Lana Abadie, CERN PhD student from University Pierre & Marie Curie (Paris VI) , Laboratoire SAMOVAR

CHEP04, Interlaken

System overview



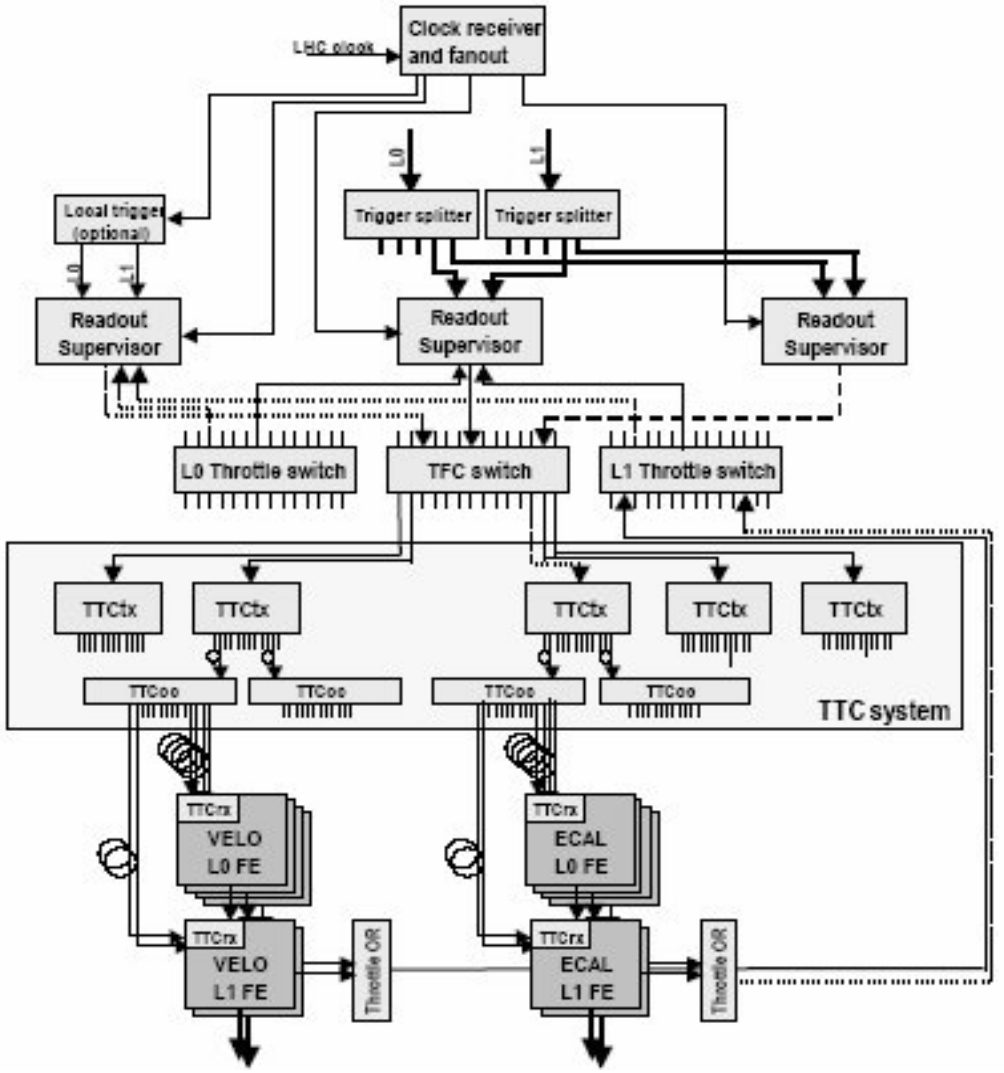
Objectives & requirements

- **Store information about all controllable devices**
  - their properties
  - the links between them
  - their hierarchy
- **Keep necessary information for the ECS**
  - to configure equipment
  - to operate the experiment
  - to monitor the system
- **Database design key issues**
  - schema
  - completeness
  - performance
  - maintenance

DATABASE SCHEMA

Timing & Fast Control (TFC) dataflow

Information from the schema
List of devices
Connectivities between devices



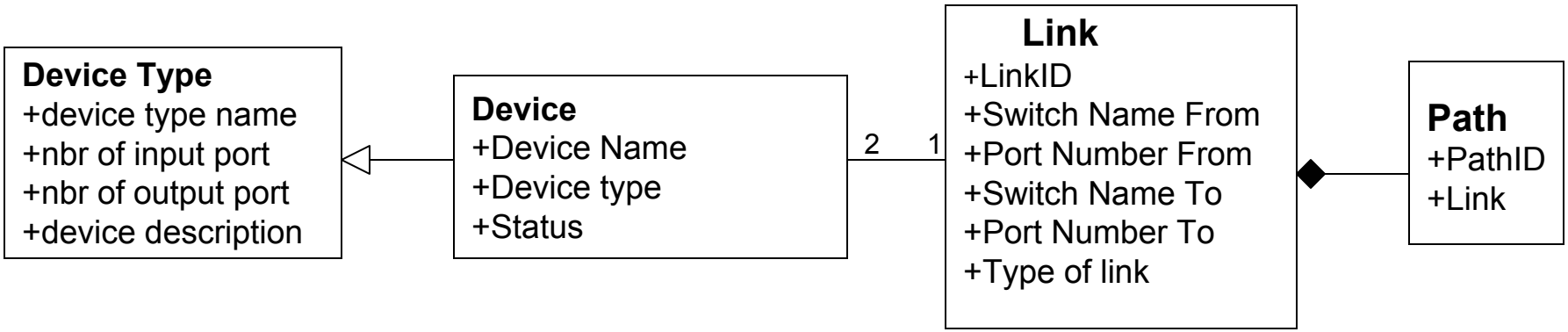
Use cases

- Collect use cases :

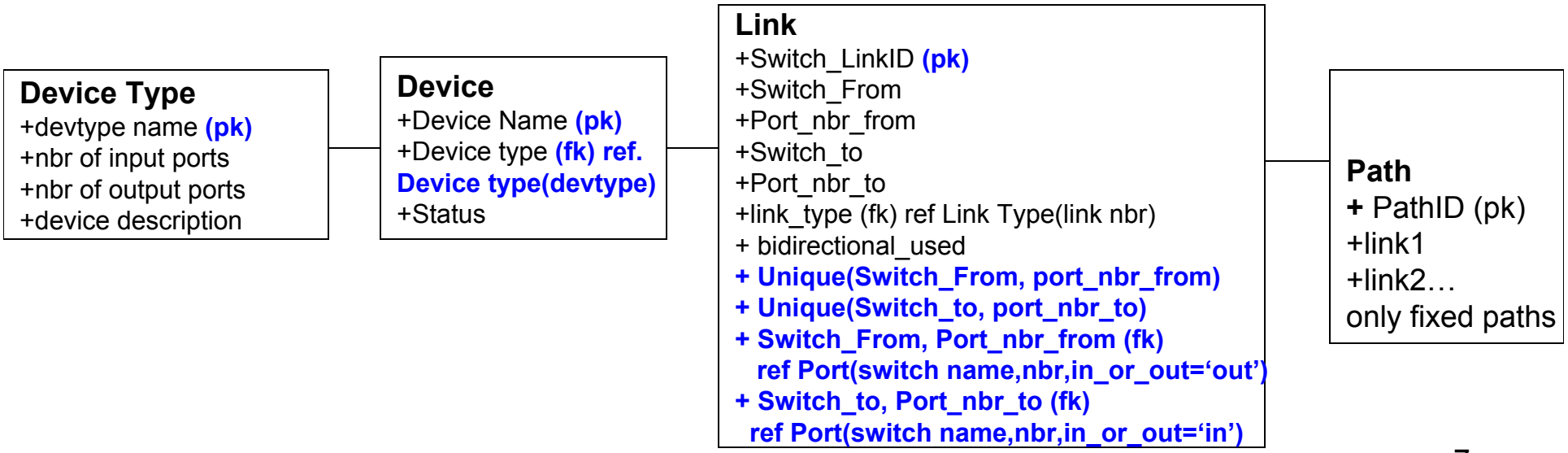
  Given a VELO card*, find a free readout supervisor and determine the routing table of the TFC switch*
- List the keywords : *device type, device, link, path...*
- Define them: *a link is a cable between a device output number and a device input number*
- Find connections between keywords: *a path is a sequence of links*

Entity relationship model

- Find the type of relation between tables



- Convert them into constraints



DATABASE IMPLEMENTATION

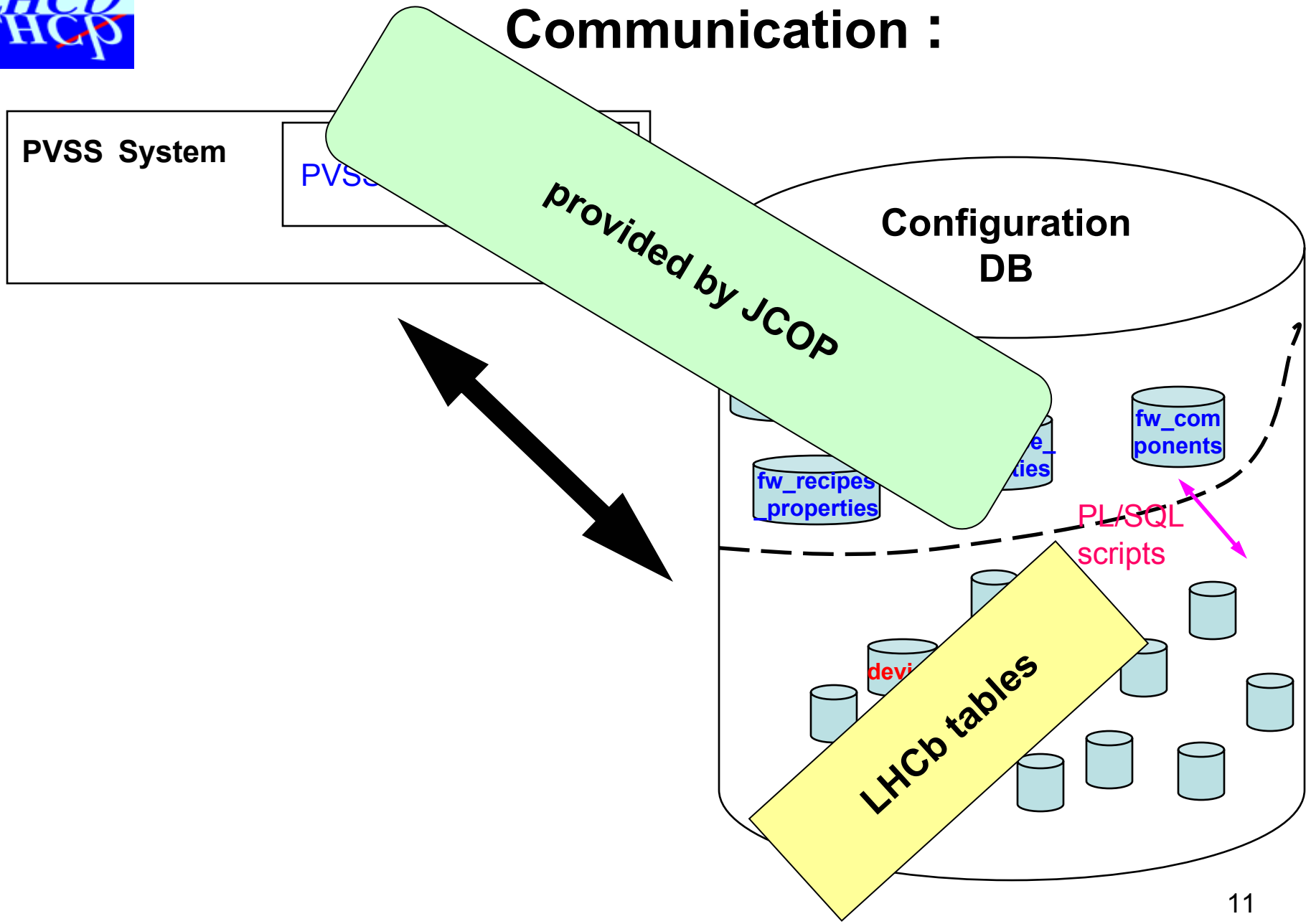
Implementation features

- Use of Oracle technology
- Use of ProC/C++ to access the database and C/C++ to encapsulate the SQL and PL/SQL statements to communicate with PVSS
- Use of JCOP configuration database tool
- Implementation of a tool (cdbVis) to edit and navigate through the database in Python
- Use of CVS to keep versions of projects and softwares

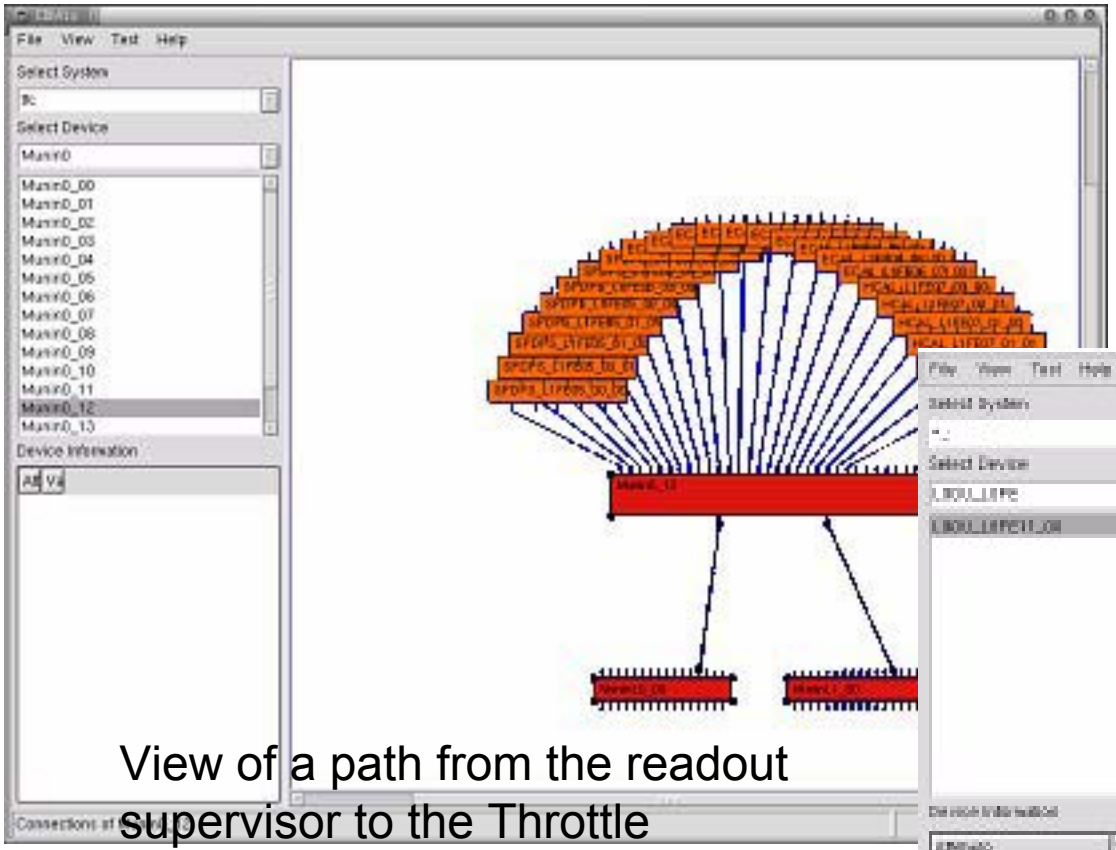
Integration of the JCOP configuration database tool

- **J**oint **Co**ntrol **P**roject: offers common tools and framework for PVSS
- Ensure compatibility between JCOP tables and LHCb tables
- Avoiding redundancy in the tables:
  - JCOP tables contain device properties
  - LHCb tables store connectivity and hierarchy information
- Adaptation of JCOP configuration database panels

Communication :

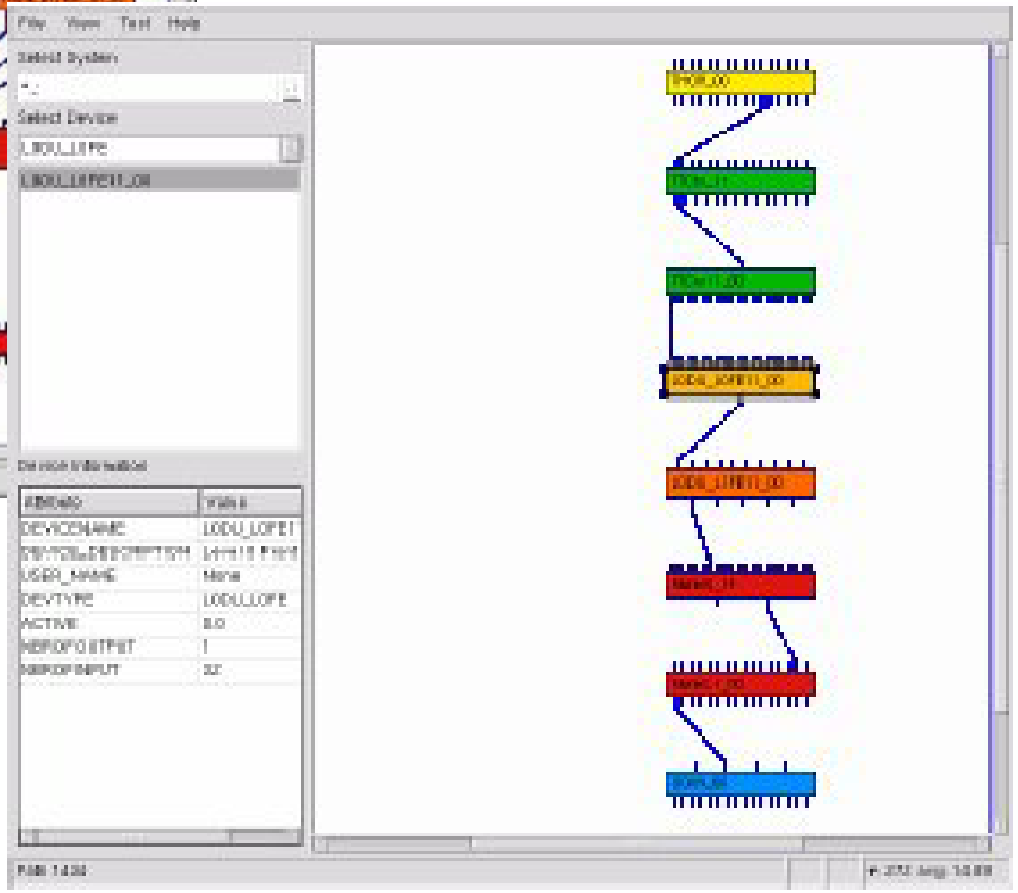


Navigator editor tool : cdbVis



View of a path from the readout supervisor to the Throttle

Display the connectivities of a selected device on its inputs and outputs



Ex. of concrete implementation :
TFC system

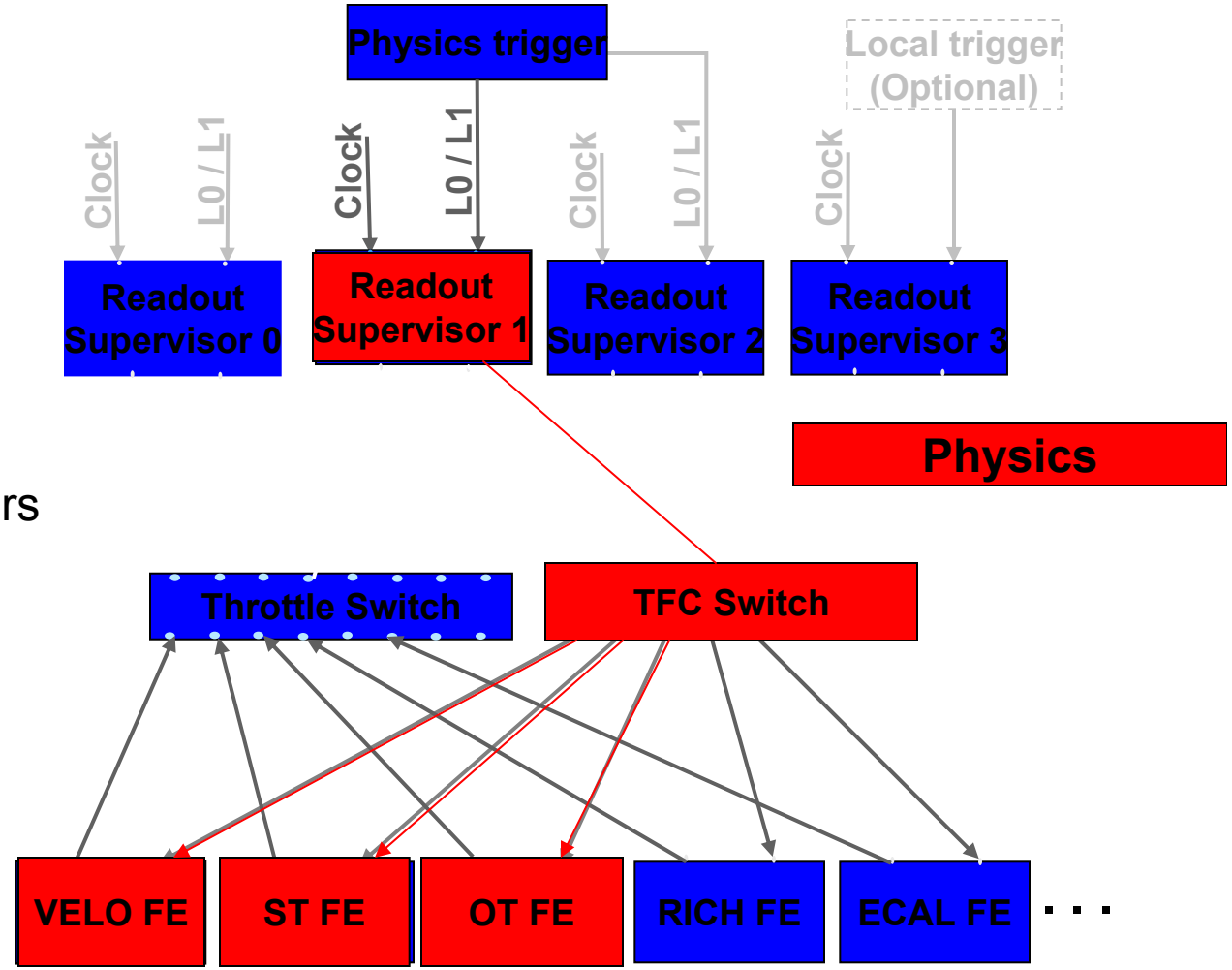
TFC requirements

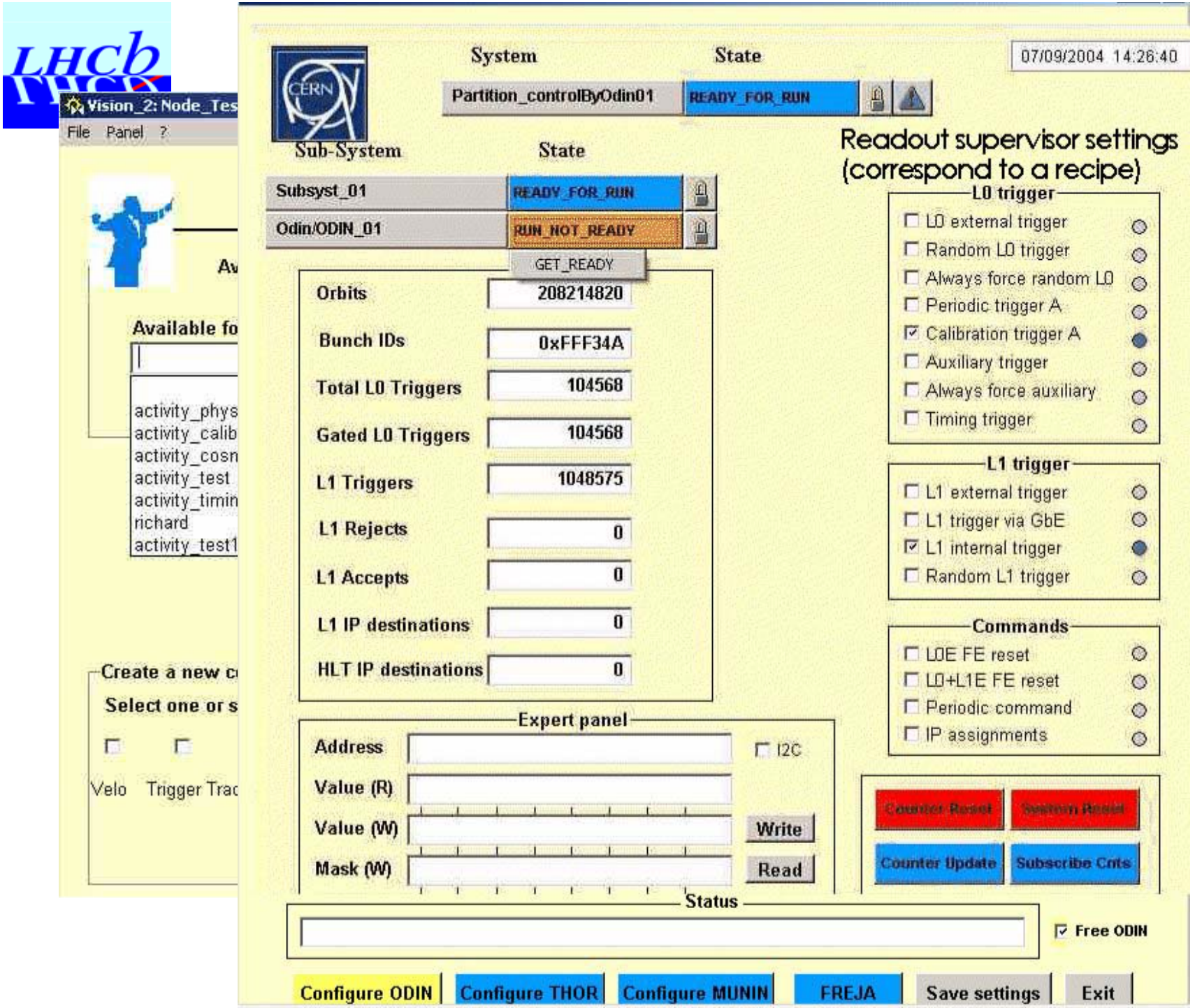
•select subdetectors and an activity

•get the **connectivities** between subdetectors and TFC switch

•List of free readout supervisors and allocate one

•**save/load** activities into/from the conf. DB




Vision_2: Node_Tes
File Panel ?
Av
Available fo
activity_phys
activity_calib
activity_cosn
activity_test
activity_timin
richard
activity_test1
Create a new c
Select one or s
Velo Trigger Trac
System
State
07/09/2004 14:26:40
Partition_controlByOdin01
READY_FOR_RUN
Sub-System
State
Subsyst_01
READY_FOR_RUN
Odin/ODIN_01
RUN_NOT_READY
GET_READY
Orbits 208214820
Bunch IDs 0xFFF34A
Total L0 Triggers 104568
Gated L0 Triggers 104568
L1 Triggers 1048575
L1 Rejects 0
L1 Accepts 0
L1 IP destinations 0
HLT IP destinations 0
Readout supervisor settings
(correspond to a recipe)
L0 trigger
L0 external trigger
Random L0 trigger
Always force random L0
Periodic trigger A
Calibration trigger A
Auxiliary trigger
Always force auxiliary
Timing trigger
L1 trigger
L1 external trigger
L1 trigger via GbE
L1 internal trigger
Random L1 trigger
Commands
L0E FE reset
L0+L1E FE reset
Periodic command
IP assignments
Expert panel
Address
I2C
Value (R)
Value (W)
Write
Mask (W)
Read
Counter Reset
System Reset
Counter Update
Subscribe Cnts
Status
Free ODIN
Configure ODIN
Configure THOR
Configure MUNIN
FREJA
Save settings
Exit

Conclusion

- Design schema for TFC and DAQ tables completed
- Production TFC control system (PVSS) now uses the configuration database

Future work

- Table Design for LHCb other subdetectors
- Extension of the cdbVis functionalities
- Design an API to enable clients to interact with the database