

SYNCHRONIZATION OF MOTION AND DETECTORS AND CONTINUOUS SCANS AS THE STANDARD DATA ACQUISITION TECHNIQUE

D. Fernández, F.Becheri, T.Coutinho [on leave], G.Cuní, R.Homs, G.Jover, J. Klorá [on leave], O.Matilla, J.Moldes, C.Pascual, Z.Reszela, D.Roldan, S.Rubio, X.Serra, and the Alba Team.



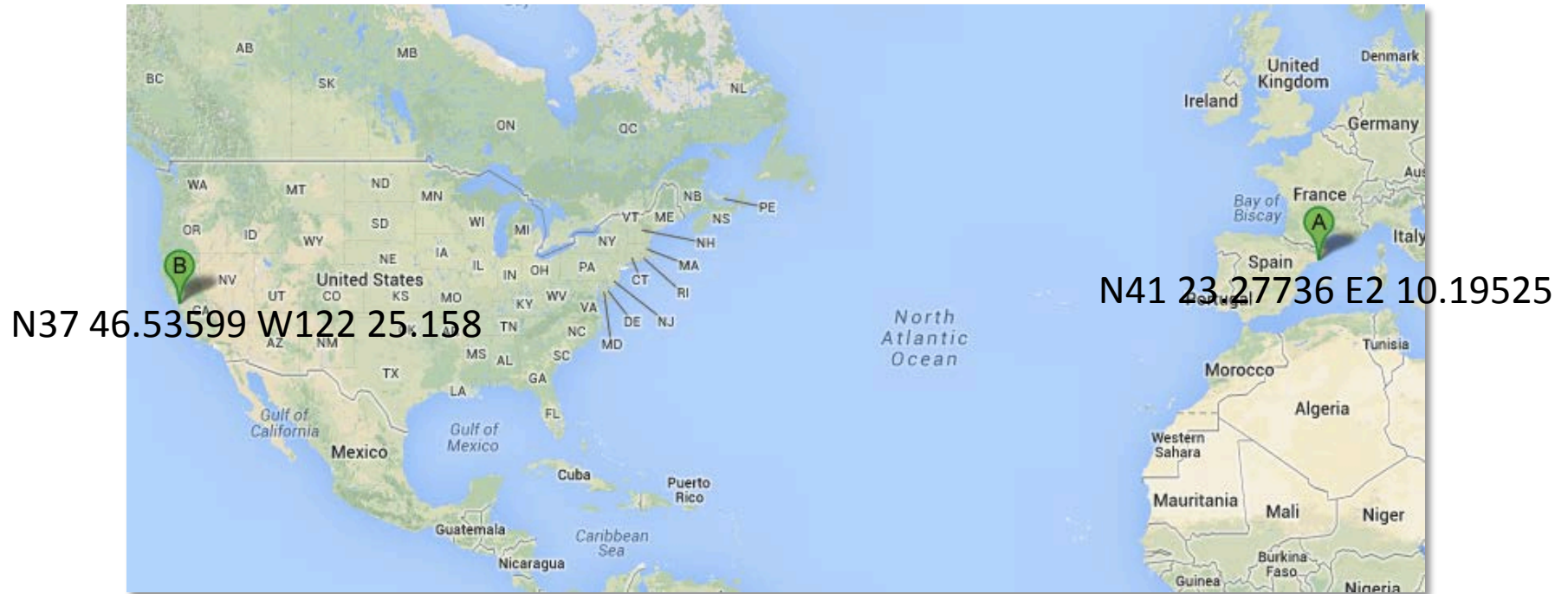
5949 miles / 9573.99 km / 5169.54 nautical miles



N37 46.53599 W122 25.158

N41 23.27736 E2 10.19525

5949 miles / 9573.99 km / 5169.54 nautical miles



N37 46.53599 W122 25.158

N41 23.27736 E2 10.19525



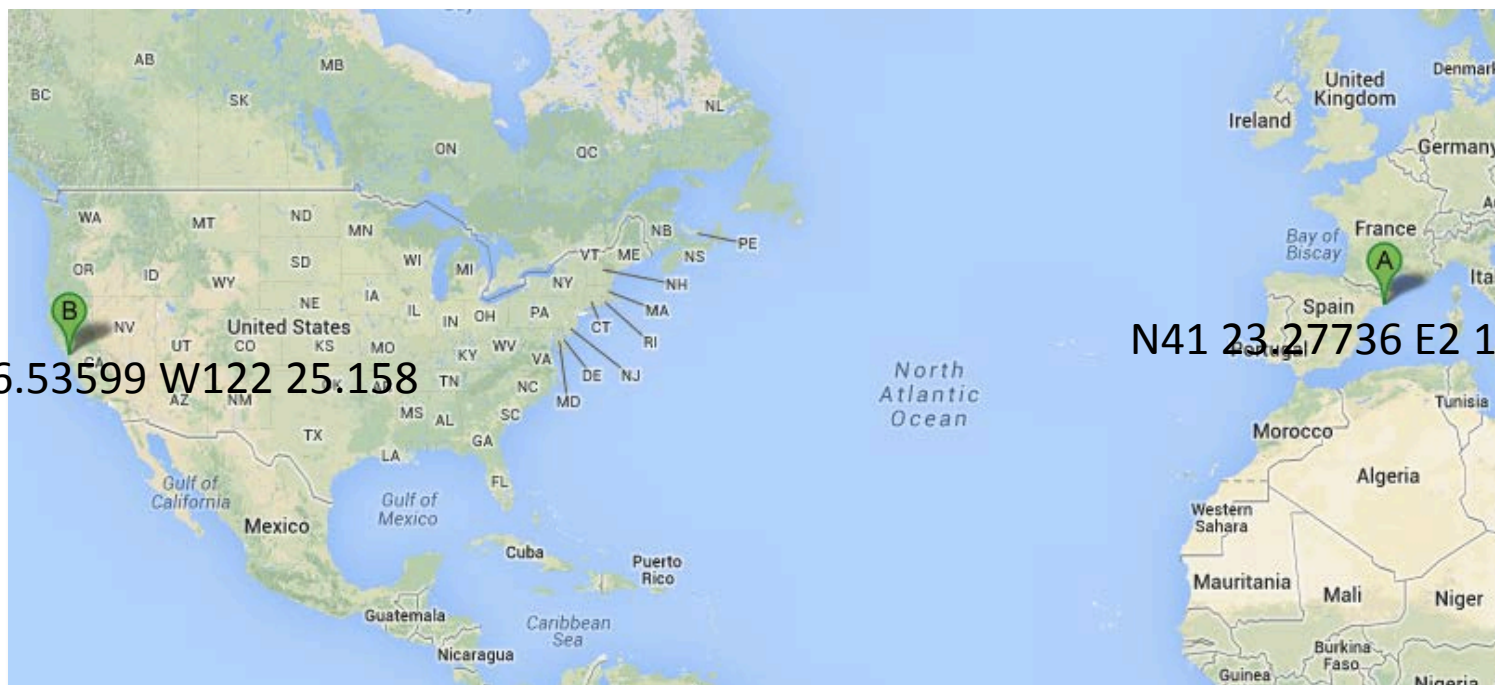
N37 46.53599 W122 25.158

N41 23.27736 E2 10.19525



5949 miles / 9573.99 km / 5169.54 nautical miles

N37 46.53599 W122 25.158



N41 23.27736 E2 10.19525



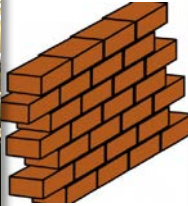
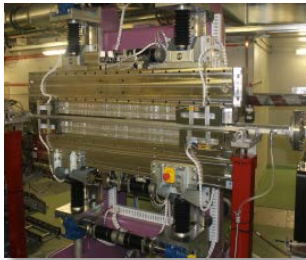
- **3GeV Accelerators Commissioned in 2011**
 - Ethernet as a fieldbus. Linux Diskless cPCI IOCs B&R PLCs
 - Firsts Tests of TOP-UP and FOFB in progress (2013)

- **7 Beamlines commissioned in 2012: Official Users since then.**
 - Electrometers, VTF Counters, CCD cameras (Rayonix, PCO, Princeton...)
 - Pixel detectors: Pilatus, Mythen, ImXPAD1400...

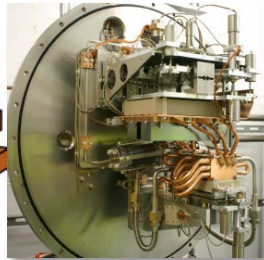
...preparing for time resolved experiments...



ID



DCM



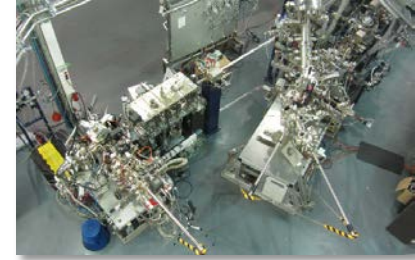
MAD



XMCD



PEEM, KNAPP

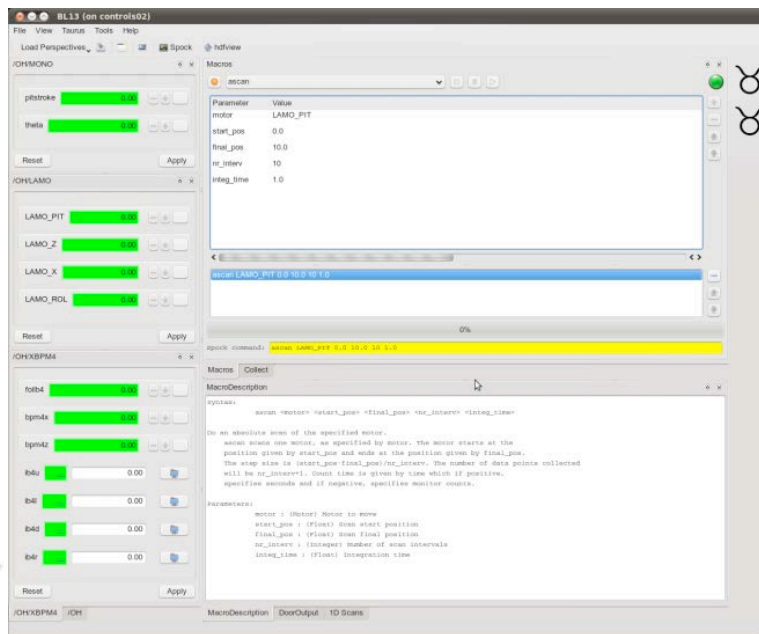
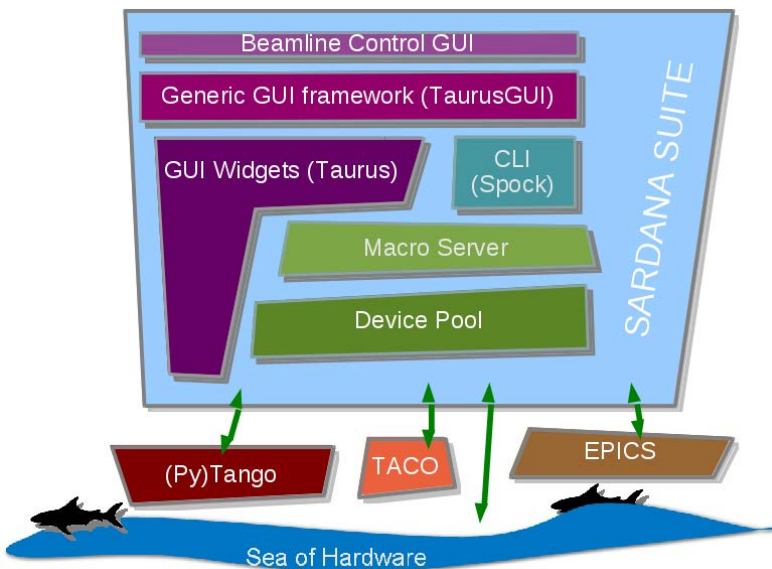


(...)

- **Insertion devices:** e.g. helical undulator (6 motors) and 6 pseudomotors
- **Monochromators:** e.g. Double crystal monochromator (direct drive 4 DEG/s)
- **Experimental station:** scalars, counters, 0D, 1D, 2D,
 - Counters, Electrometers, Cameras, etc.
 - Slow channels and fast channels

Need a “scanning machine” having **movable channels** and **experimental channels**.
:Configure any combination of motors and counters + detectors in a step scan.
:Write my own sequences “macros” and scans,
:have pseudo-motors and pseudo-counters as a combination of several channels

Sardana: towards the scientific SCADA



- Generic, configurable Graphical User Interfaces. Trends, plots, forms, channels, synoptics, ...
- Interfaces with PLCs, databases, and different fieldbuses.
- Historical archives for supervision data.
- Configuration management. Configuration tool and state snapshots capabilities.
- Alarm handling, states, notification, acknowledgement and archiving.
- Self diagnostics and management tools
- ...

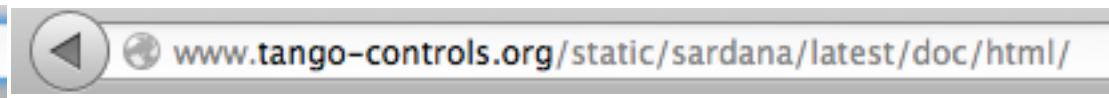
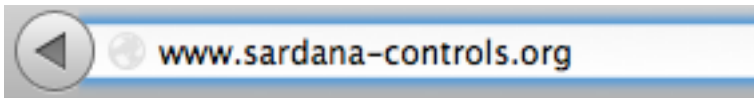


Sardana: towards the scientific SCADA



FLEXIBILITY

- Simple to download, install and startup.
- . and above all: well documented.

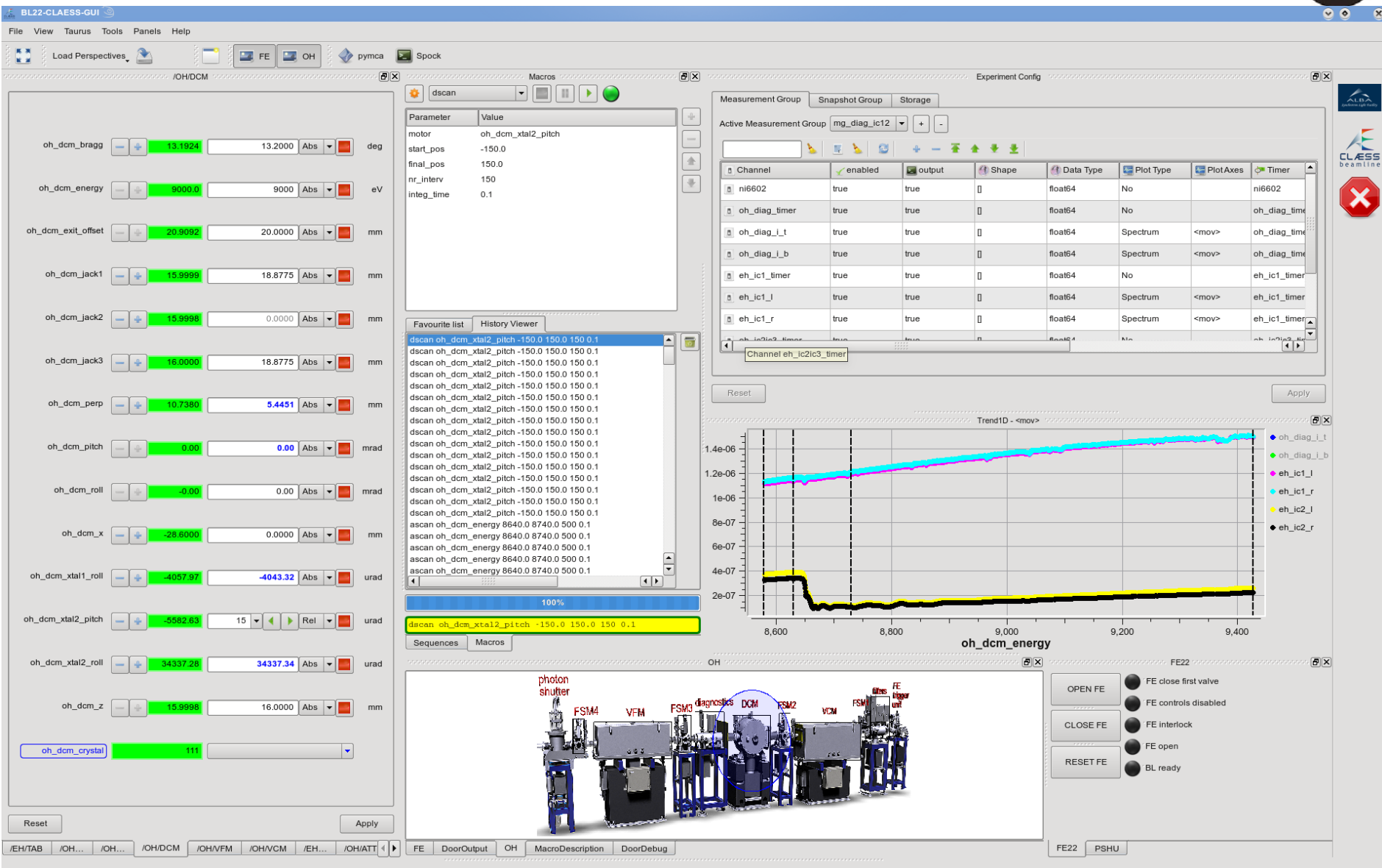


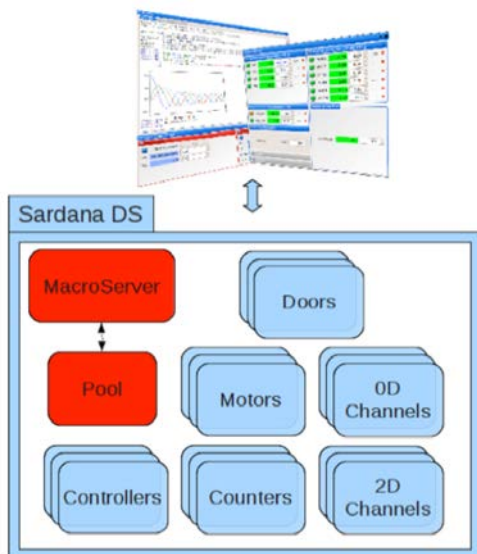
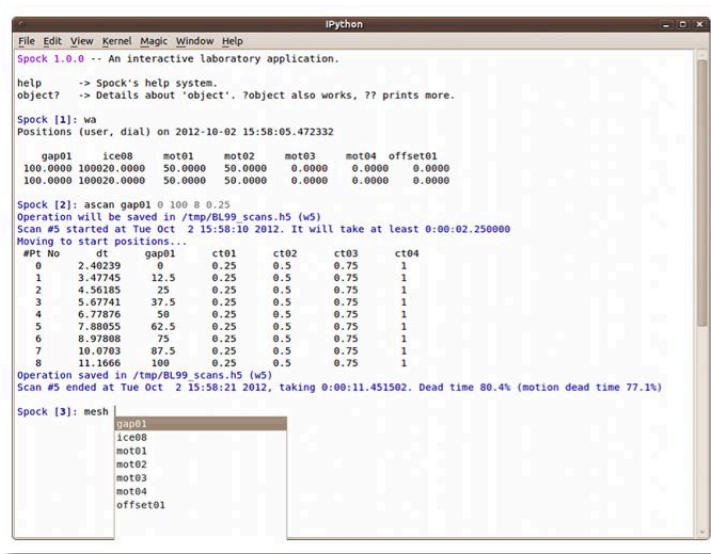
The screenshot displays the Sardana control interface, which is a complex software environment for managing scientific experiments. The interface is divided into several main sections:

- Parameter List:** A table on the left showing various parameters and their values. For example, 'can Energy' is set to 685.0 745.0 301 1.0.
- Macros:** A section for managing macros, including a 'Load Perspectives' button and a list of macros like 'mistral-3D', 'hdfview', 'pymca', and 'Spock'.
- Experiment Config:** A central panel for configuring the experiment. It includes a 'Measurement Group' and a 'Snapshot Group'.
- Measurement Group:** A panel for selecting and configuring measurement elements. It shows a list of elements like 'MS_BL09' and 'Motors', and a table of parameters for each element.
- Snapshot Group:** A panel for selecting and configuring snapshot elements. It shows a list of elements like 'machine_current', 'fe_xbpm_v', 'fe_xbpm_h', 'fe_z', 'fe_x', and 'fe_v_offset'.
- Storage:** A panel for selecting and configuring storage elements. It shows a list of elements like 'machine_current', 'fe_xbpm_v', 'fe_xbpm_h', 'fe_z', 'fe_x', and 'fe_v_offset'.
- 3D Model:** A 3D model of the experimental setup, showing the various components and their relative positions.

The interface also includes a 'Trend' window for monitoring data over time, and a 'Limits' window for setting and monitoring limits for various parameters.

WECOAB03. Continuous scans as the standard data acquisition technique



```
File Edit View Kernel Magic Window Help
Spock 1.0.0 -- An interactive laboratory application.

help      -> Spock's help system.
object?   -> Details about 'object'. ?object also works, ?? prints more.

Spock [1]: wa
Positions (user, dial) on 2012-10-02 15:58:05.472332
gap01    ice08    mot01    mot02    mot03    mot04    offset01
100.0000 100020.0000 50.0000 50.0000 0.0000 0.0000 0.0000
100.0000 100020.0000 50.0000 50.0000 0.0000 0.0000 0.0000

Spock [2]: ascan gap01 0 100 8 0.25
Operation will be saved in /tmp/BL99 scans.h5 (w5)
Scan #5 started at Tue Oct 2 15:58:10 2012. It will take at least 0:00:02.250000
Moving to start positions...
#Pt No   dt   gap01    ct01    ct02    ct03    ct04
0       2.40239 0      0.25    0.5    0.75    1
1       3.47745 12.5   0.25    0.5    0.75    1
2       4.56185 25     0.25    0.5    0.75    1
3       5.67741 37.5   0.25    0.5    0.75    1
4       6.77876 50     0.25    0.5    0.75    1
5       7.88055 62.5   0.25    0.5    0.75    1
6       8.97808 75     0.25    0.5    0.75    1
7       10.0703 87.5   0.25    0.5    0.75    1
8       11.1666 100    0.25    0.5    0.75    1
Operation saved in /tmp/BL99 scans.h5 (w5)
Scan #5 ended at Tue Oct 2 15:58:21 2012, taking 0:00:11.451502. Dead time 80.4% (motion dead time 77.1%)

Spock [3]: mesh
gap01
ice08
mot01
mot02
mot03
mot04
offset01
```



- Flexible, present in beamlines in “virtually” all synchrotrons. Extensively used in the day-to-day: Alignment, data acquisition, day-to-day operation in the Beamline.
- They are at the center of the control system:
 - Require motion, counter-timers, often a scripting-macro language,
 - Integrated with data formats, detectors, experimental channels, sample environment...

...very robust ...although ...slow ...

- A **step scan**, depending on the experiment, exposure times and number of points can be **time consuming**.
- In some cases (**time resolved**), the experiment itself needs to be done on certain time constraints.
- The longer the scan takes, the more vulnerable is to changes in the environment (thermal drifts, machine current, orbit, etc...)
- Occasionally, it is more convenient (or mandatory) **taking data during the motion**:
 - **The data is taken at given intervals while the motor(s) are moving.**
- Acquisitions can be synchronized in different ways:
 - Time (pulses given at certain time) usually by a time frame generator or a timer
 - Position (reading encoder positions and producing pulses at the given points)
- ...what if non linear trajectories or non equal intervals?

- Running in most Beamlines synchrotrons
- Overcome the “slow” issue of step scans

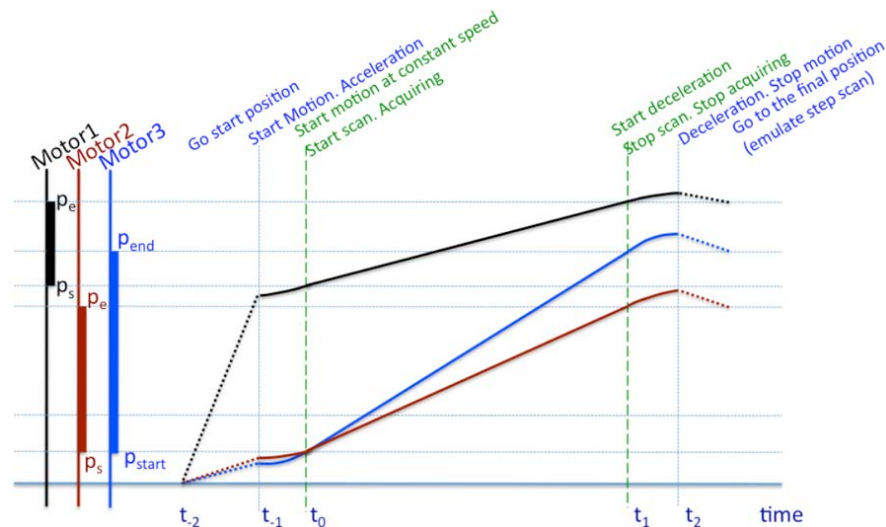
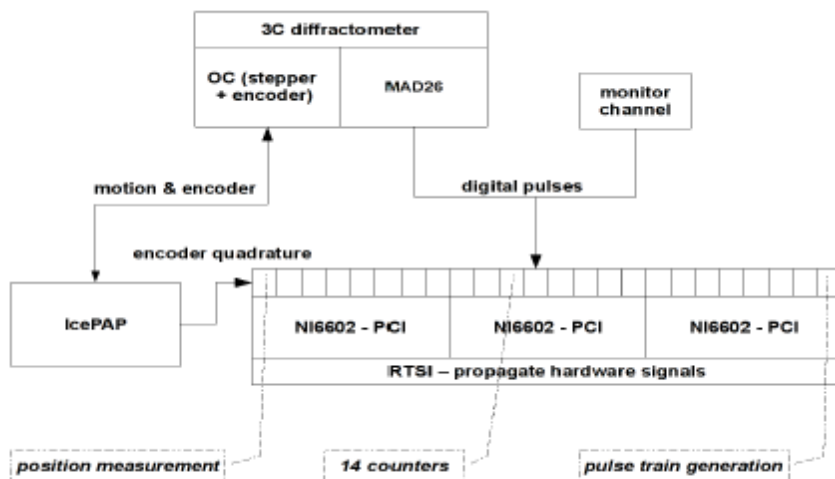
Need dedicated hardware with dedicated cabling for a particular purpose.

: Solve the problem for a particular type of experiment.

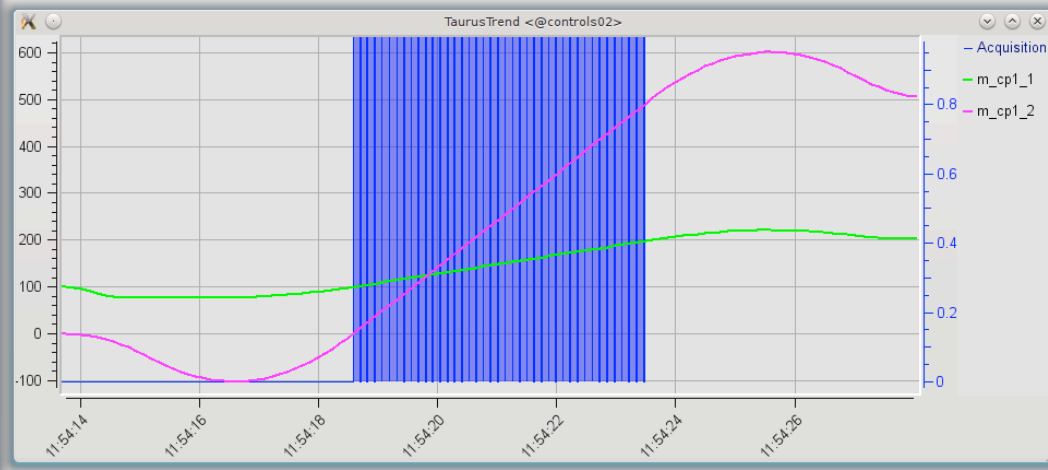
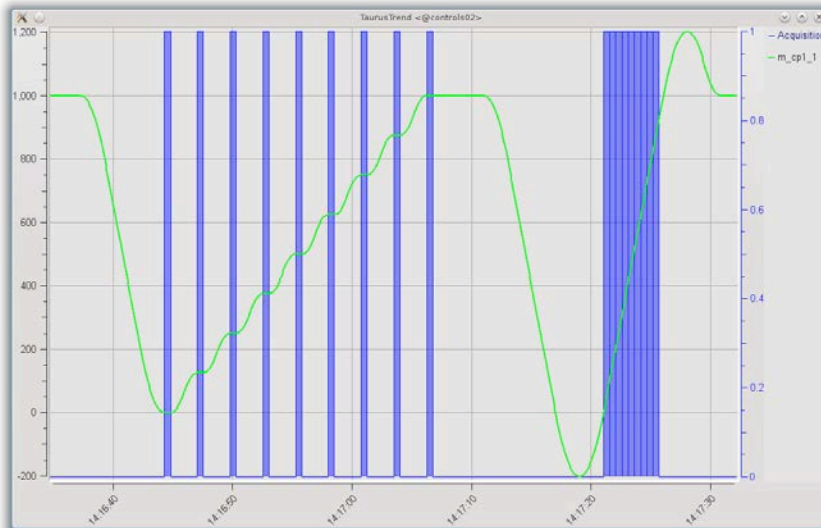
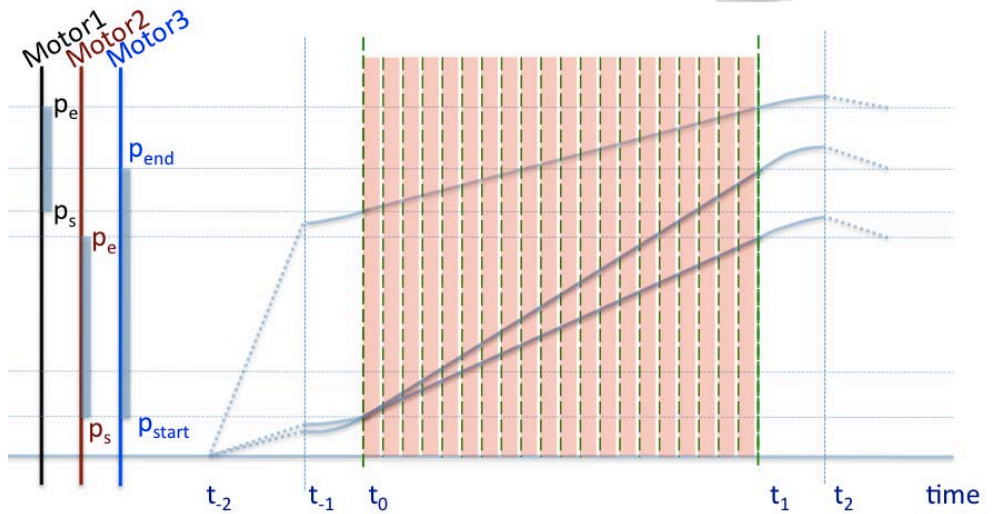
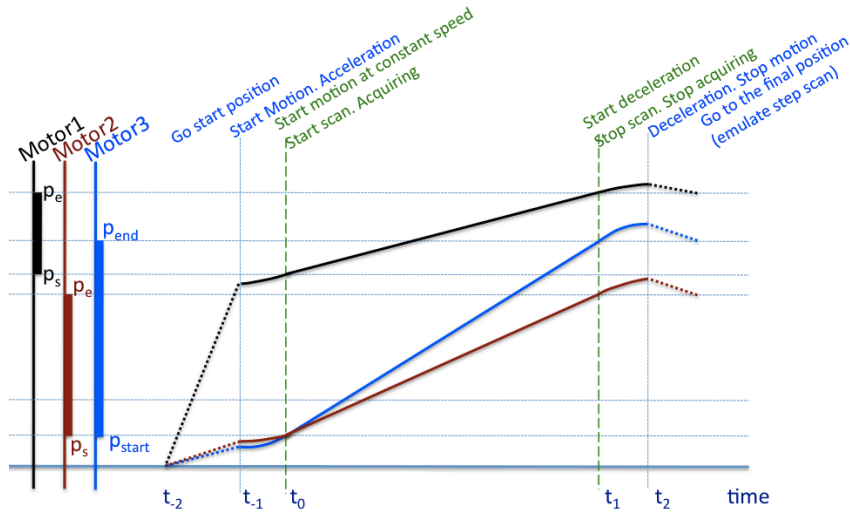
: Adapted to the purpose.

: Fast, optimized.

They do not solve all problems, being necessary to reprogram the beamline for other experiments using another hardware.



Continuous scans: Generic approach



- *Do **continuous scans** in a **generic way** as if they were step scans*
 - *Share the **same syntax**, have the same flexibility. Have the same motors, pseudomotors, channels and detectors available*
- Trigger objects
 - Produce triggers (a timer or any particular axis at the given positions)
 - All motors shall be prepared to act as input for synchronization.
 - This means having encoder signals or indexers routed multiplexed and feed to the triggering hardware
- Buffering
 - Different data rates need different buffer capabilities.
 - Slow data from sample environment, accelerators, etc. do not need/can not achieve the same data rate. Interpolated
- Timestamps:
 - Having a accurate enough timestamp would make eventually unnecessary hardware triggers in several applications.
 - In the case of a spectroscopy moving the energy with a motorized Bragg angle (4 degrees per second, taking 4000 points in one second), we would need a precision of about 13 microseconds (not achievable by NTP)

The control system. Block diagram

```

terminal
File Edit View Terminal Tabs Help

$ spock
tcourinho@PC131:~$ spock
Setting spock environment... [DONE]
Setting global environment... [DONE]
Connecting to user...
70 new macro(s) available

Spock 0.1.0 -- An interactive Macro Server client.
Running on top of Python 2.5.2 and Python 2.6.4
Using Door BL90/Door/001 to access Macro Server BL90/MacroServer/001.

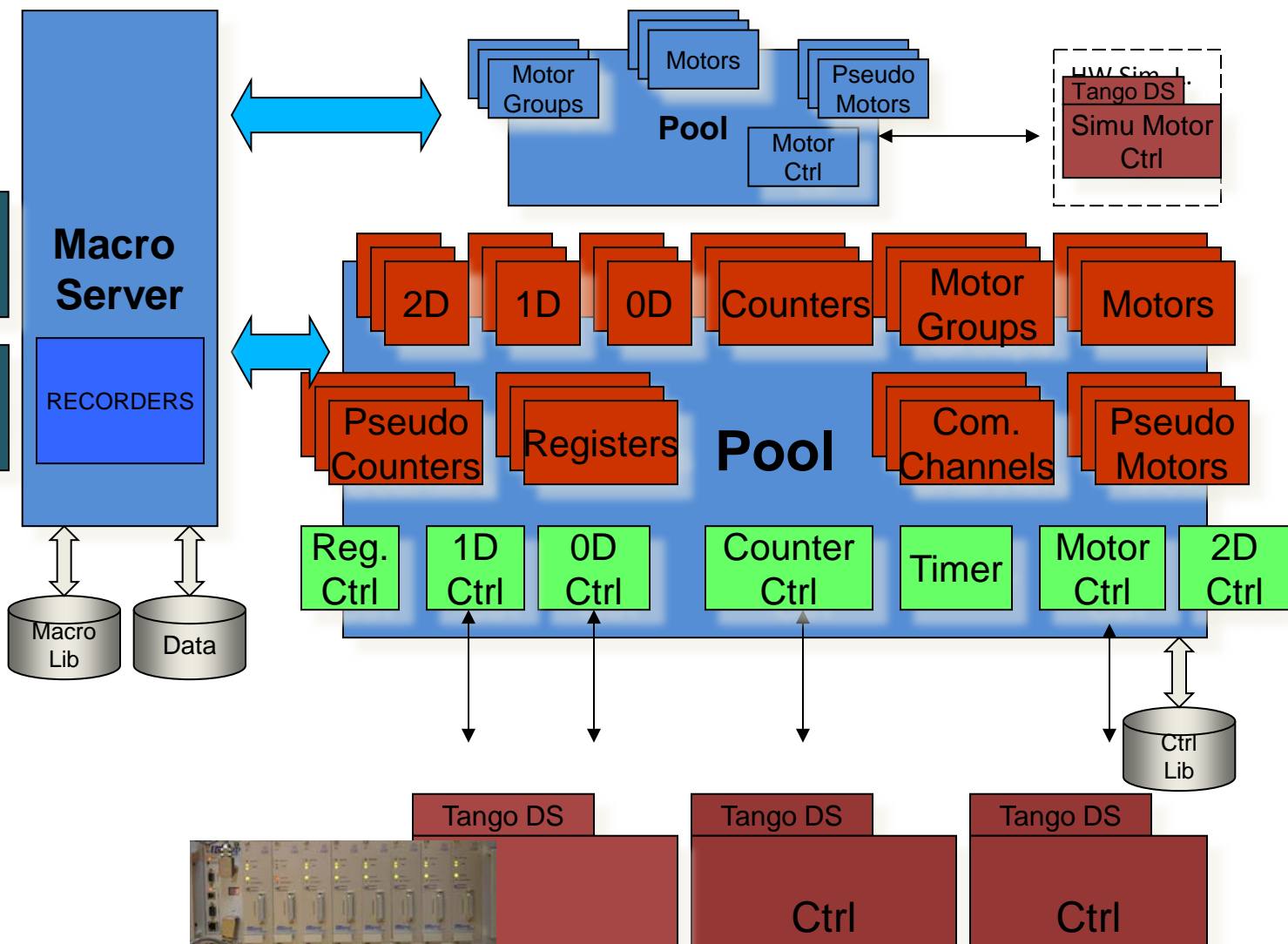
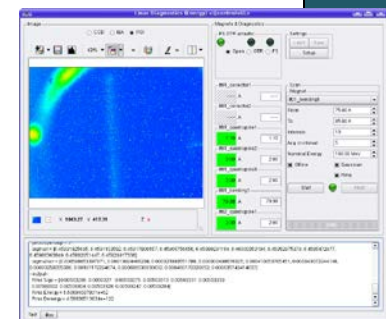
$ spock: wa
Current Positions (user, dial)

BL90 gap1 BL90 offset1 BL90 SimMot1 BL90 SimMot2 BL90 SimMot3
200.00000000 74.50000000 174.50000000 25.50000000 0.00000000
200.00000000 74.50000000 174.50000000 25.50000000 0.00000000

BL90 SimMot4
200.00000000
200.00000000
200.00000000
    
```

Door

Door



The control system. Block diagram

```

terminal
File Edit View Terminal Tabs Help

$ SP00K:
tcourinho@PC131:~$
tcourinho@PC131:~$ spock
Setting spock environment... [DONE]
Setting global environment... [DONE]
Connecting to dier...
70 new macro(s) available

Spock 0.1.0 -- An interactive Macro Server client.
Running on top of Python 2.5.2 and Python 2.6.4
Using Door BL90/Door/001 to access Macro Server BL90/MacroServer/001.

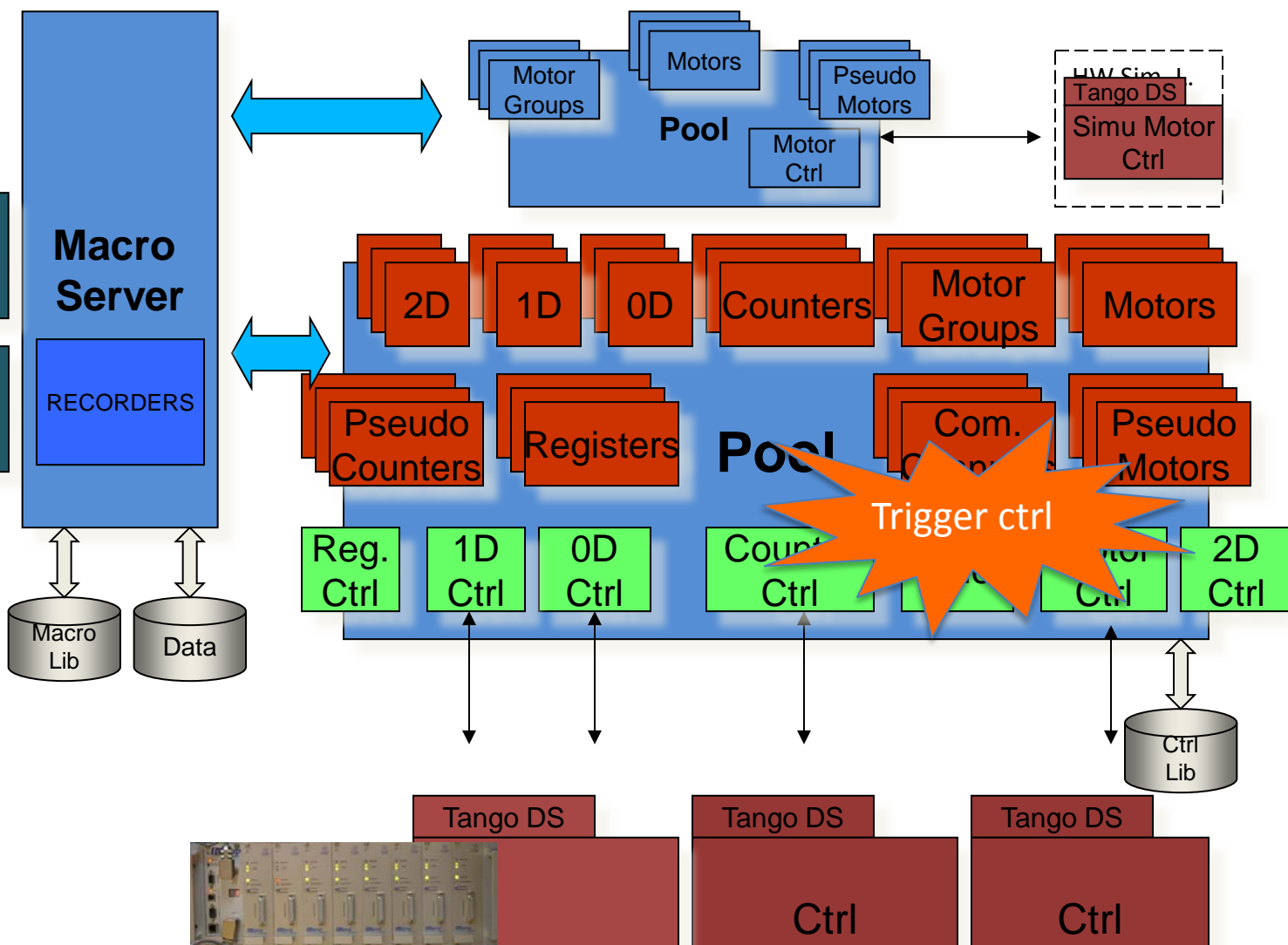
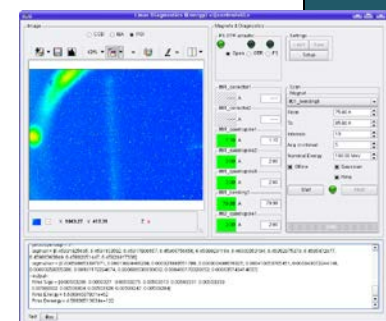
$ SP00K: wa
Current Positions (user, dial)

BL90 gap1 BL90 offset1 BL90 SimMot1 BL90 SimMot2 BL90 SimMot3
200.00000000 74.50000000 174.50000000 25.50000000 0.00000000
200.00000000 74.50000000 174.50000000 25.50000000 0.00000000

BL90 SimMot4
200.00000000
200.00000000
200.00000000
  
```

Door

Door



The control system. Block diagram

```

terminal
File Edit View Terminal Tabs Help

$ SP00K:
tcourlinhoPC131:~$
tcourlinhoPC131:~$ spock
Setting spock environment... [DONE]
Setting global environment... [DONE]
Connecting to user...
70 new macro(s) available

Spock 0.1.0 -- An interactive Macro Server client.
Running on top of Python 2.5.2 and Python 2.6.4
Using Door BL90/Door/001 to access Macro Server BL90/MacroServer/001.

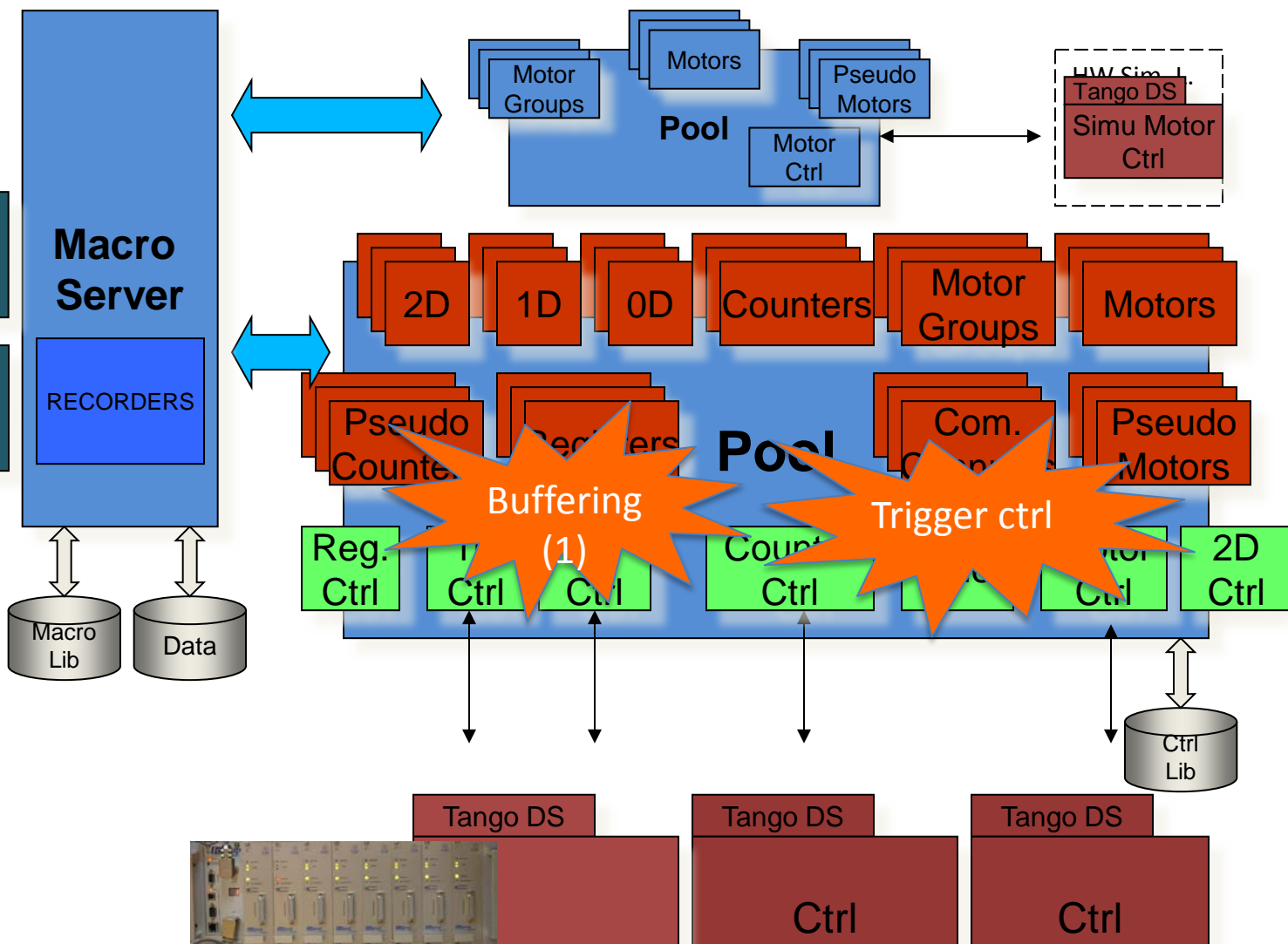
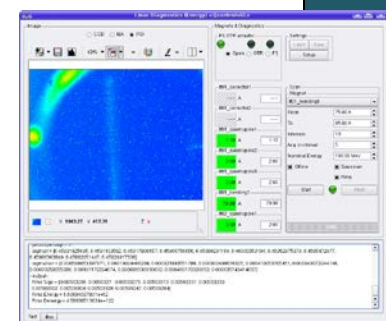
$ SP00K: wa
Current Positions (user, dial)

BL90 gap1 BL90 offset1 BL90 SimMot1 BL90 SimMot2 BL90 SimMot3
200.00000000 74.50000000 174.50000000 25.50000000 0.00000000
200.00000000 74.50000000 174.50000000 25.50000000 0.00000000

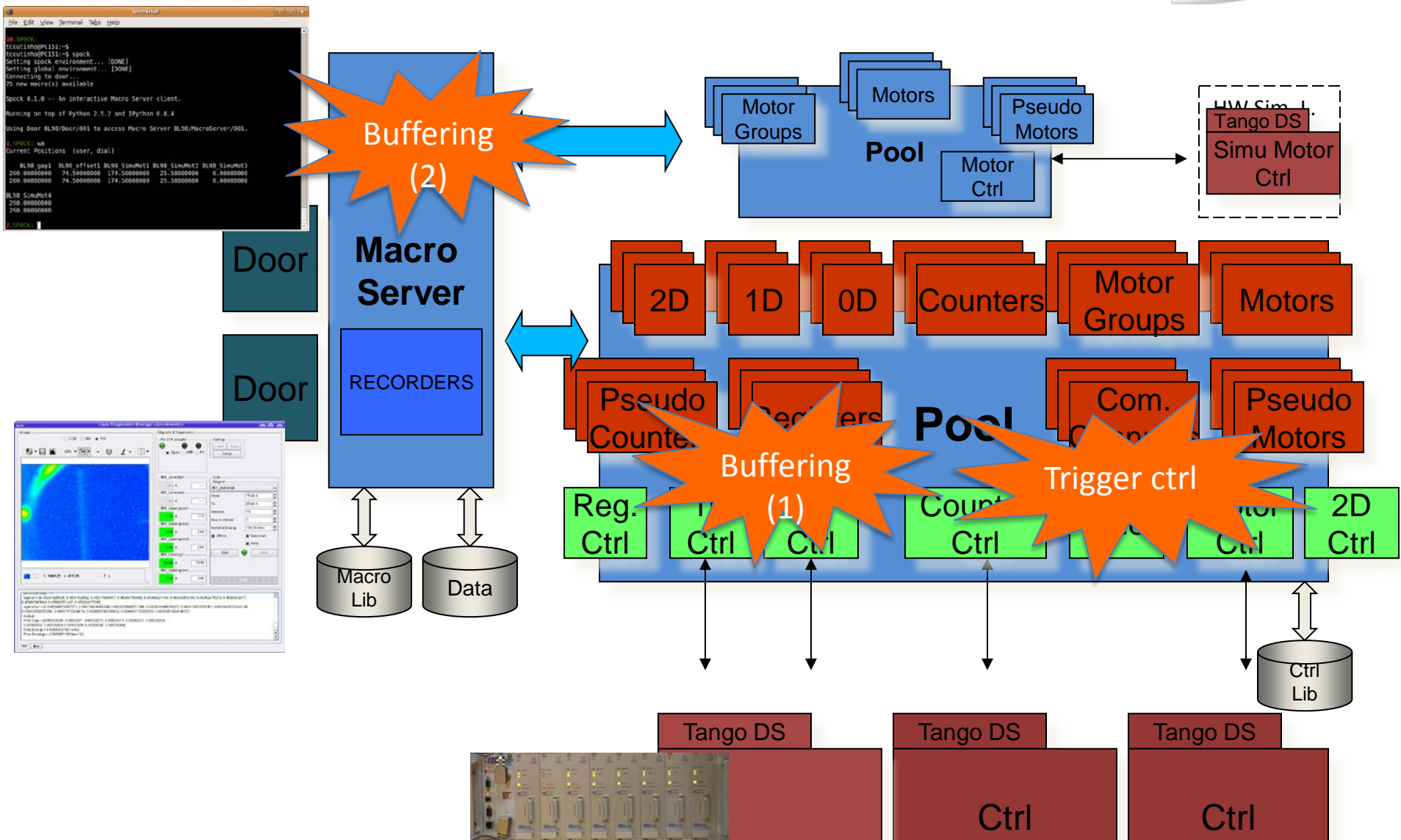
BL90 SimMot4
200.00000000
200.00000000
200.00000000
  
```

Door

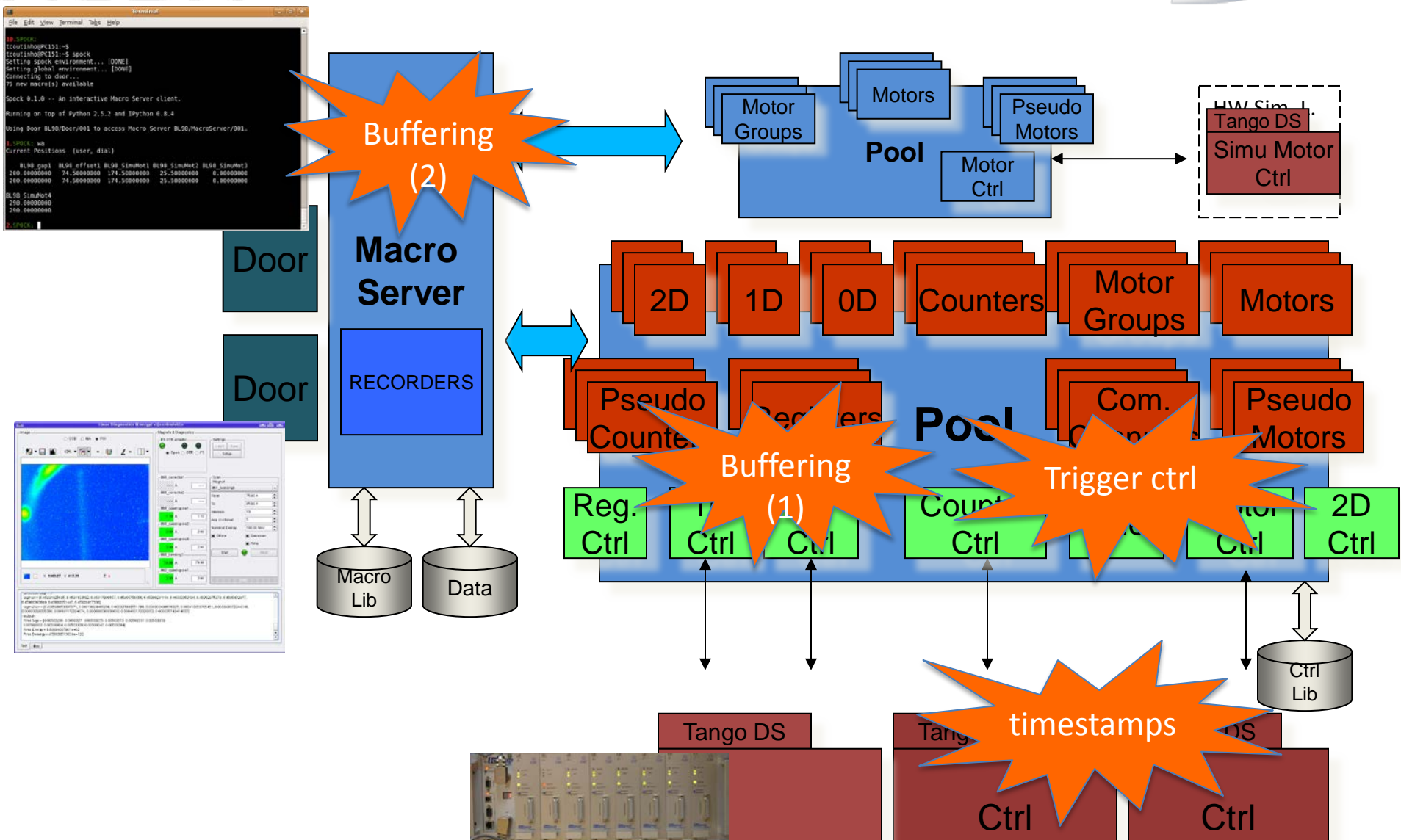
Door



The control system. Block diagram



The control system. Block diagram



- **Triggering at time** intervals occasionally is simpler and more convenient than getting the motor position. Motor encoders treated as experimental channels.

- **Buffers** manage fast acquisition combined with local buffers and slow acquisitions managed directly.



- **Scalers, 0D,**

- **1D, 2D ... LIMA (ESRF)**



- **Timestamps**

- Timing system (MRF) from the Accelerators distributed to the Beamlines.
 - IOCs synchronized by NTP (10 ms offsets)
 - PTP to be considered.
 - Data is timestamped when acquired, the closest to the hardware.

- **Generic scientific SCADAS require a tighter integration of software components,**
 - Sequencer (friendly (powerful) (scripting) language (python))
 - Users write their own sequences/macros. Standard macro library.
- **Hardware easy to install and plug in. Flexible software**
 - Motion control, and fast detectors/channels triggered/configured with no hardware/cabling re-installation.
- **Triggering and buffering**
 - Central trigger object and device managing multiple motor inputs (potentially all in a beamline) and detectors.
 - Slow channels and fast channels, archived and interpolated
- **Timestamps**
 - Close to hardware. Detectors must have an accurate timestamp which attach to the data.
 - Accurate, but feasible with standard technologies, (microsecond range).
 - Triggers are the first option when available:
 - Timestamps do not avoid triggers.

TUPPC094 X. Serra et al.



Thank you for your attention.

