# SPIRAL2 CONTROL COMMAND: FIRST HIGH-LEVEL JAVA APPLICATIONS BASED ON THE OPEN-XAL LIBRARY

P. Gillette, E. Lemaître, G.Normand, L. Philippe, GANIL, Caen, France

### *Abstract*

The Radioactive Ions Beam SPIRAL2 facility will be based on a supra-conducting driver providing deuterons or heavy ions beams at different energies and intensities. Using then the fragmentation and ISOLD method, exotic nuclei beams will be sent either to new physics facilities or to the existing GANIL experimental areas. To tune this large range of beams, high-level applications will be mainly developed in Java language. The choice of the OPEN-XAL application framework, developed at the Spallation Neutron Source (SNS), has proven to be very efficient and greatly helps us to design our first software pieces to tune the accelerator. The first part of this paper presents some new applications: "Minimisation" which aims at optimizing a section of the accelerator; a general purpose software named "Hook" for interacting with equipment of any kind; and an application called "Profils" to visualize and control the Spiral2 beam wire harps. As tuning operation has to deal with configuration and archiving issues, databases are an effective way to manage data. Therefore, two databases are being developed to address these problems for the SPIRAL2 command control: one is in charge of device configuration upstream the Epics databases while another one is in charge of accelerator configuration (lattice, optics and set of values). The last part of this paper aims at describing these databases and how java applications will interact with them.

## THE SPIRAL2 FACILITY

### *Overview*

The SPIRAL2 facility (Fig. 1) is based on a high power, superconducting linac driver, which will deliver a high intensity, 40 MeV deuteron beams as well as a variety of heavy-ion beams with mass-to-charge ratio of 3 and energy upon to 14.5 MeV/nucleon. A possibility of construction of a second injector for heavy-ions with a mass-to-charge ration of 6 is incorporated in the design. The main RIB production scheme of SPIRAL2 is based on the fast-neutron induced fission of uranium target. Using a carbon converter, a 5 mA deuteron beam and a high-density (up to 11g/cm$^3$) 2.3 kg uranium carbide target, the fission is expected to reach a rate up to 5*10$^{13}$/s. A direct irradiation of the UC$_2$ target with beams of protons or 3,4He could also be used. The extracted 1+radioactives ions will be injected in the 1+/n+ charge breeder (ECR ion source) and post-accelerated by the existing CIME cyclotron.

The SPIRAL2 project is divided in two phases (Fig 1): the first one (LINAC buildings and associated experimental AEL) and the second phase (RIB production building and DESIR facility).

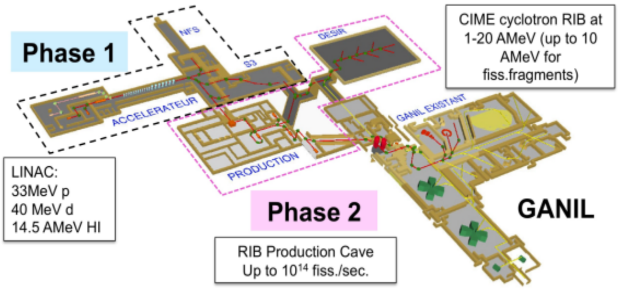


Figure 1: Accelerator layout.

## SOFTWARE ARCHITECTURE

### *Control System Main Choices*

The choice of EPICS [1] as a common framework was early decided to ease pieces of software development and integration efficiency. Mainly remote terminal units are VME VxWorks crates with MVME 5500 CPUs and Red Hat Enterprise Linux PCs hosting EPICS Input Output Controllers (IOCs). On the other side, Siemens S7 programmable logic controllers (PLC) are mainly dedicated to slow material protection systems needed for radio frequency, cryogenic, vacuum, or interlock systems.

### *Software Development*

To ease software sharing, a unique EPICS SPIRAL2 working environment, an equipment naming convention, and operational rules for interfaces have been specified. To survey a large facility as SPIRAL2, supervision screens developed with EPICS Extensible Display Manager (EDM) are needed[2]. An archive engine using the one provided by Control System Studio (CSS) integrated development environment has been set up this year. For the tuning of the accelerator, more sophisticated high level applications are involved. XAL [3],[4],[5] Java based library and framework originally developed at the Spallation Neutron Source (Oak Ridge National Laboratory), and dedicated to accelerator tuning, have been adopted. The accelerator hierarchy and model design, fulfilling the Spiral2 specific needs, have proved to be very effective.

Among the numerous tools included in XAL, the high level API for wrapping EPICS Channel Access protocol are widely used to read, write and monitor values on devices.

Some news applications have been written or adapted from the ones already provided by the SNS.

## THE MINIMIZATION APPLICATION

This software acts on a set of equipments to achieve the minimization of objectives on a set of diagnostics.

Some algorithms (Simplex, Powell, Coordinated steps, One dimension, Aleatory, Aleatory by map, Gradient) have been implemented, improved (when needed) and tested in order to face the situations that could occur during the tuning of the beam. It is based on standard XAL Framework and library tools, using node selector and PV selector, Scorer and AlgorithmSolver classes [6].

The application is divided in four parts: the user chose which active and passive elements he wants (up left), enters the start and the end desired with some options (up right), and then chooses and sets up the algorithm (bottom left). The results are drawn in real time in the same frame. The numerical results appear in the bottom right.

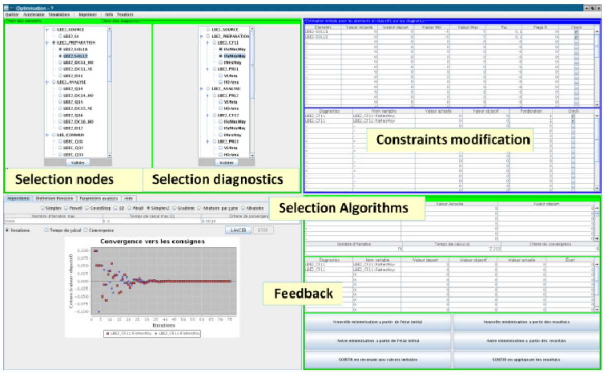


Figure 2: The minimisation software.

## THE HOOK APPLICATION

A general purpose software named "Hook" has been developed for directly interacting with equipment of any kind (for the moment power and high voltage supplies, harp wire monitors, motorisations, current diagnostics, slits). For each kind of equipment, a generic component is used to assign elements on a graphic grid and provide complete access to the equipment. It was designed to reflect the same look and feel as the one (written in Ada) used in the legacy control system of GANIL.

Using the tools of the XAL library, the user selects elements in a hierarchical presentation of the accelerator. Once the elements are selected, each one can be assign to a sizable cell of a grid which the user can interact with. The grid component is in charge of the realisation and destruction of the items views.

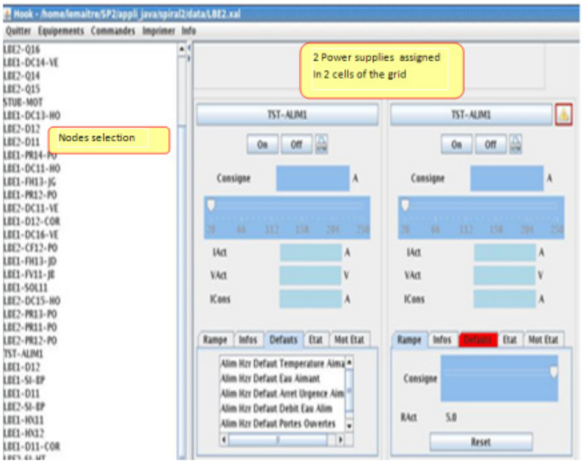


Figure 3: The hook software.

## THE PROFILS APPLICATION

To visualize and control the SPIRAL2 beam wire harps the application Profils has been developed. It offers a complete control over the electronic devices and specific calculations on the data read from IOC Profils. It allows the user to select several sections of a profile to calculate areas, half widths, centers etc. The results of calculation are available to Epics clients via virtual process variables generated by a Java server.

As for the hook application, a set of harps can be assigned on a grid. General purposes commands are then available for all the harps assigned (scaling, printing, saving, insertion into the beam line, integration time or high voltage setting, mean and offset set-up etc)

Individual commands are also available in the window of each element: electronic device set-up, graphical comparison between live, theoretical, saved or snapshot data, selection of the wires to take in account for drawing and calculation, fit method for reconstruction of broken wires etc.

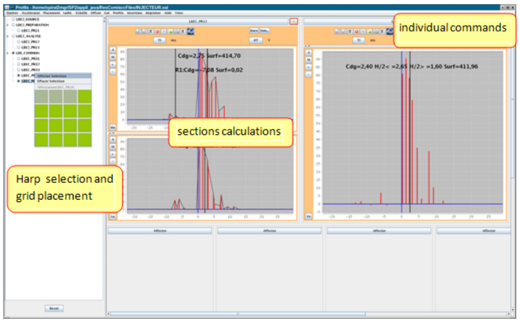


Figure 4: The Profils software.

## DATABASES AND DATA MANAGEMENT

The command control group of GANIL is accustomed to databases usage on order to manage data and configuration linked to the system. The first Ingres database has been introduced in 1990.

For Spiral2 project, four topics rely on databases:

- Devices configuration
- Accelerator configuration
- Alarms handling
- Archiving

## *Device Configuration*

To describe and exploit the 2500 equipments which are envisaged for the control of the SPIRAL2 facility, a relational database and associate management tools are developed at GANIL.

This database will be used offline and describes "simple" equipments such as slits, motors or power supplies. It enables generation of IOC configuration file (connections, substitutions and sequencers definition) as shown in Figure 5.

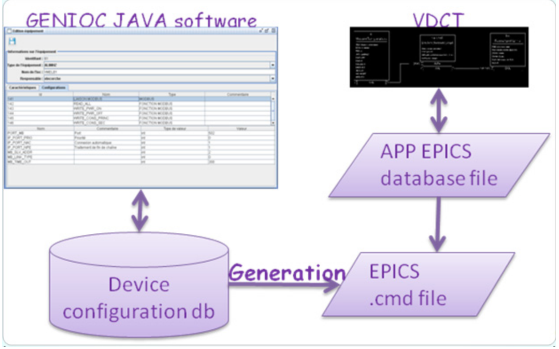


Figure 5: Device configuration process.

The main concept of this database is to describe a device type dynamically into the database's tables and instantiate this type to create a new device.

A first mock-up has been developed and tested with slits and power supplies devices.

## *Accelerator Configuration*

To manage Accelerator configuration, a relational database and associate management tools are developed at GANIL

This database will be in charge of accelerator configuration. It describes accelerator hierarchy, optics and enables generation of XAL file defining the accelerator .xdxf.

The hierarchy is made up of 4 levels shown in Figure 7:

- Level: setting unit for operation. A level is made up of Paths.
- Path: path followed by the beam. A path belongs to a unique Level. A path is made up of sequences.
- Sequence: Grouping by tuning function. A sequence belongs to one or many paths. A sequence is made up of parts.
- Part: It's a unit introduced to facilitate computing management. A part belongs to one or many sequences. A part is made up of nodes (devices that affect or survey the beam).

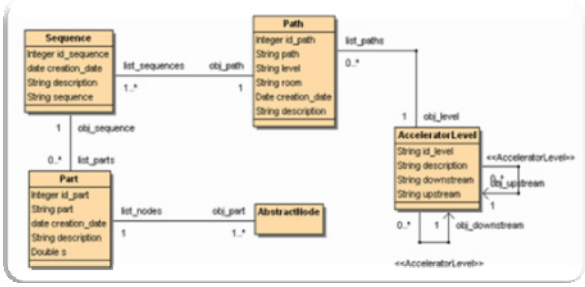
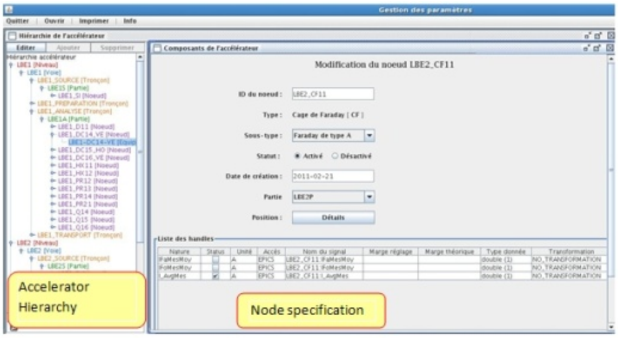


Figure 6: Accelerator hierarchy.

This database can be read and update off-line by a new dedicated software which enables generation of XAL file defining the accelerator optics files.


Gestion des paramètres
Quitter | Ouvrir | Imprimer | Info
Hiérarchie de l'accélérateur
Editer | Ajouter | Supprimer
Composants de l'accélérateur

Hiérarchie accélérateur
- LBE1 (Niveau)
  - LBE1 (Voie)
    - LBE1_SOURCE (Tronçon)
      - LBE1S (Partie)
        - LBE1_SI (Noeud)
    - LBE1_PREPARATION (Tronçon)
    - LBE1_ANALYSE (Tronçon)
      - LBE1A (Partie)
        - LBE1_D11 (Noeud)
        - LBE1_DC14_VE (Noeud)
          - LBE1-DC14-VE (Equip
        - LBE1_DC15_HO (Noeud)
        - LBE1_DC16_VE (Noeud)
        - LBE1_HX11 (Noeud)
        - LBE1_HX12 (Noeud)
        - LBE1_PR12 (Noeud)
        - LBE1_PR13 (Noeud)
        - LBE1_PR14 (Noeud)
        - LBE1_PR21 (Noeud)
        - LBE1_Q14 (Noeud)
        - LBE1_Q15 (Noeud)
        - LBE1_Q16 (Noeud)
    - LBE1_TRANSPORT (Tronçon)
- LBE2 (Niveau)
  - LBE2 (Voie)
    - LBE2_SOURCE (Tronçon)
      - LBE2S (Partie)

Modification du noeud LBE2_CF11

ID du noeud : LBE2_CF11

Type : Cage de Faraday [ CF ]

Sous-type : Faraday de type A

Statut : ● Activé ○ Désactivé

Date de création : 2011-02-21

Partie : LBE2P

Position : Détails

Liste des handles

| Nature | Status | Unité | Accès | Nom du signal | Marge réglage | Marge théorique | Type donnée | Transformation |
|---|---|---|---|---|---|---|---|---|
| FaMesMoy | □ | A | EPICS | LBE2_CF11:FaMesMoy | | | double (1) | NO_TRANSFORMATION |
| FoMesMoy | □ | A | EPICS | LBE2_CF11:FoMesMoy | | | double (1) | NO_TRANSFORMATION |
| I_AvgMes | ☑ | A | EPICS | LBE2_CF11:I_AvgMes | | | double (1) | NO_TRANSFORMATION |

Accelerator Hierarchy

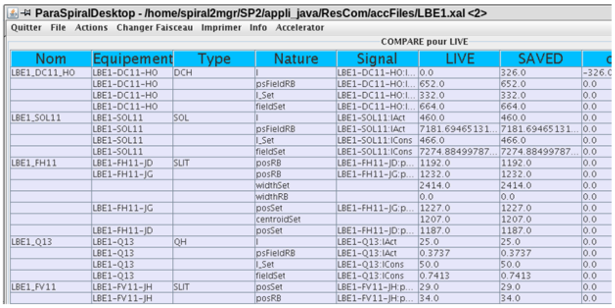
Node specification

Figure 7: Accelerator configuration database tool.

This database will be used off-line to assign values for a new beam and to schedule operation. The set of values of a new beam is either theoretical or extrapolated from beams realized previously. Theoretical values for a beam to be produced are exported from Tracewin software developed by D.URIOT (CEA) [7].

This database will be used online to apply or save sets of values.

An application dedicated to the setting of the accelerator allows the user to choose the injector to use, the beam to accelerate and the set of values to apply. Once these settings have been applied, numerous operations (setting, reading, comparing values, computing data for new energies or magnetic rigidities) are available. 50-60 percent of functions are implemented


ParaSpiralDesktop - /home/spiral2mgr/SP2/appli_java/ResCom/accFiles/LBE1.xal <2>
Quitter File Actions Changer Faisceau Imprimer Info Accelerator

COMPARE pour LIVE

| Nom | Equipement | Type | Nature | Signal | LIVE | SAVED | c |
|---|---|---|---|---|---|---|---|
| LBE1_DC11_HO | LBE1-DC11-HO | DCH | I | LBE1-DC11-HO:I... | 0.0 | 326.0 | -326.0 |
| | LBE1-DC11-HO | | psFieldRB | LBE1-DC11-HO:I... | 652.0 | 652.0 | 0.0 |
| | LBE1-DC11-HO | | I_Set | LBE1-DC11-HO:I... | 332.0 | 332.0 | 0.0 |
| | LBE1-DC11-HO | | fieldSet | LBE1-DC11-HO:I... | 664.0 | 664.0 | 0.0 |
| LBE1_SOL11 | LBE1-SOL11 | SOL | I | LBE1-SOL11:IAct | 460.0 | 460.0 | 0.0 |
| | LBE1-SOL11 | | psFieldRB | LBE1-SOL11:IAct | 7181.69465131... | 7181.69465131... | 0.0 |
| | LBE1-SOL11 | | I_Set | LBE1-SOL11:ICons | 466.0 | 466.0 | 0.0 |
| | LBE1-SOL11 | | fieldSet | LBE1-SOL11:ICons | 7274.88499787... | 7274.88499787... | 0.0 |
| LBE1_FH11 | LBE1-FH11-JD | SLIT | posRB | LBE1-FH11-JD:p... | 1192.0 | 1192.0 | 0.0 |
| | LBE1-FH11-JG | | posRB | LBE1-FH11-JG:p... | 1232.0 | 1232.0 | 0.0 |
| | | | widthSet | | 2414.0 | 2414.0 | 0.0 |
| | | | widthRB | | 0.0 | 0.0 | 0.0 |
| | LBE1-FH11-JG | | posSet | LBE1-FH11-JG:p... | 1227.0 | 1227.0 | 0.0 |
| | | | centroidSet | | 1207.0 | 1207.0 | 0.0 |
| | LBE1-FH11-JD | | posSet | LBE1-FH11-JD:p... | 1187.0 | 1187.0 | 0.0 |
| LBE1_Q13 | LBE1-Q13 | QH | I | LBE1-Q13:IAct | 25.0 | 25.0 | 0.0 |
| | LBE1-Q13 | | psFieldRB | LBE1-Q13:IAct | 0.3737 | 0.3737 | 0.0 |
| | LBE1-Q13 | | I_Set | LBE1-Q13:ICons | 50.0 | 50.0 | 0.0 |
| | LBE1-Q13 | | fieldSet | LBE1-Q13:ICons | 0.7413 | 0.7413 | 0.0 |
| LBE1_FV11 | LBE1-FV11-JH | SLIT | posSet | LBE1-FV11-JH:p... | 29.0 | 29.0 | 0.0 |
| | LBE1-FV11-JH | | posRB | LBE1-FV11-JH:p... | 34.0 | 34.0 | 0.0 |

Figure 8: ParaSpiral draw a comparison between online setting values and setting values saved.

### *Alarms Handling*

For reasons of compatibility and consistency between GANIL alarm handling process and that of SPIRAL2, we decide not to use EPICS standard Alarm Handler.

The alarms handling process, illustrated in Figure, rely on an Ingres database. This database assumes two main functions:

- Configuration of the process : how to build alarms messages and which alarms should be caught
- Storage of alarms issued.

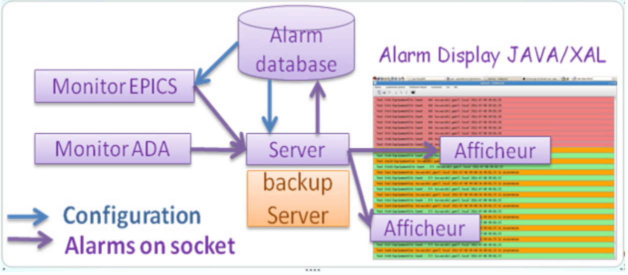


Figure 9: Alarm handling process.

The alarms process is made up of four software:

- Monitor EPICS: JAVA based development. It relies on channel access monitor. It is in charge of catching and formatting alarm message.
- Monitor ADA: ADA Based development. It assumes the same function as Monitor EPICS for GANIL alarms.
- Server: JAVA Based development. It receives alarms from monitor and dispatches them to displays. A failover server takes over if the main server failed.
- Display: JAVAL/XAL based development. It displays alarms in the form of button list.

This process has been tested and supports a throughput of 330 alarms per second.

### *Archiving*

The archiving system is planned to be implemented witch archive engines upstream a MySQL database, jointly the CSS Client operator interface.

The software architecture used for tests, illustrated in figure 6, shows that the load is balanced between two servers: one for insertions and one for reads.

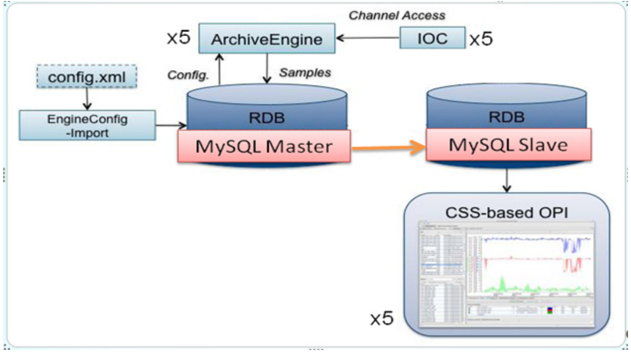


Figure 10: Test architecture for archiving.

The writing tests, presented in Table 1, were performed in September 2011.

The first results showed that we needed to update the MySQL JDBC driver and settle the **rewriteBatchedStatement** property to true to enable bulk insert and a huge performance gain.

Table 1 : Archiving Test

| Number of IOC | Number of PV/IOC | Number of Archive Engine | Number of PV/ Archive Engine | f | PV/s |
|---|---|---|---|---|---|
| **5** | 80 | 5 | 80 | 10Hz | 4000 |

The reading (CSS client) tests using MySQL partitioning started recently.

## CONCLUSION & NEXT STEPS

The use of the XAL library has been very efficient. While some of the SNS applications have been reused without modification: scanning applications, virtual accelerator, launcher, etc., others have had to be adapted to fit our needs: optimisation, magnet cycling, radio-frequency cavities tuning. Specific applications were designed from scratch using the XAL library and framework: admittance limitation, alignment, profiles monitoring and equipment control. New developments are foreseen: ion source spectrometry, beam power monitoring, longitudinal emittance measurement, etc.

Most of the applications require data from the database for their configuration especially through xdxf file. Furthermore, databases are at the heart of central service like alarms or archiver.

## ACKNOWLEDGEMENTS

Other contributors are: from the Ganil control group, D. Touchard (Epics environment and tools), C. Haquin (real time), P. Lermine (databases); from IRFU, D. Uriot (Tracewin), JF. Denis and F. Gougnaud (Tests at Saclay).

The basic hardware and software infrastructure and the network architecture are managed and provided by the Ganil Computing Infrastructure group.

**A special thanks to the OPEN XAL team for the software and support provided.**

## REFERENCES

[1] E. Lécorché et al. “Overview of the Spiral2 Control System progess” Icalepcs 2011.
[2] F. Gougnaud et al, “The Implementation of the Spiral2 Injector Control System”. This conference.
[3] J. Galambos, et al, “XAL the SNS application programming infrastructure” Epac 2004
[4] T. Pelaia II et al. “XAL Status” Icalepcs 2007.
[5] Open Xal : http://xaldev.sourceforge.net/.
[6] J.Galambos, et al.”Xal - U.S. Particle Accelerator School 2008” https://wiki.ornl.gov/sites/xaldocs/PHYS798X Course Material/Forms/AllItems.aspx.
[7] D. Uriot, N. Pichoff “New implement in TraceWin/Partran codes: integration in external field map”, PAC2003