ROSALI: AN APPLICATION ALLOWING ONLINE/OFFLINE ALGORITHM IMPLEMENTATION TO ASSESS BEAM INSTRUMENTATION PERFORMANCE

M. Moles
CERN, Geneva, Switzerland

Abstract

A software tool called "Rapid Online Software ALgorithm Implementation (ROSALI)" has been developed at CERN. This application is intended to provide instrumentation experts and accelerator physicists with a tool, which allows monitoring and storage of beam measurements and rapid algorithm implementation via specialized actions on the embedded Mathematica kernel. The users are able to build or modify online a sequence of actions implementing their algorithms. Those sequences can subscribe to ongoing measurements from several beam instruments or retrieve data from previous recordings, merge these measurements to obtain correlation diagrams or perform dedicated calculations. This document presents the current state and the foreseen extensions of this application. The application has been tested last year on the SPS and it will be used this year on the new LEIR machine at CERN.

INTRODUCTION

The AB/BDI software section at CERN is responsible for providing all the software necessary to develop, test, diagnose and maintain the different instruments produced by the group. This control software for front-end computers is implemented using the FESA framework [1].

The *__R__apid __O__nline __S__oftware __AL__gorithm __I__mplementation* application; known as ROSALI [2] is an application for the instrumentation experts and accelerator physicists to monitor, store and analyse the beam measurements made by our instruments. As Figure 1 shows, ROSALI can retrieve these measurements directly from the acquisition front-ends or from previously stored data files. The storage of these measurements can be made through a C library following the ROSALI file format (explained later on) directly in the front end software or using the ROSALI application itself. This file format ensures that all data saved from the front-ends or ROSALI will be consistent and readable from the ROSALI application.

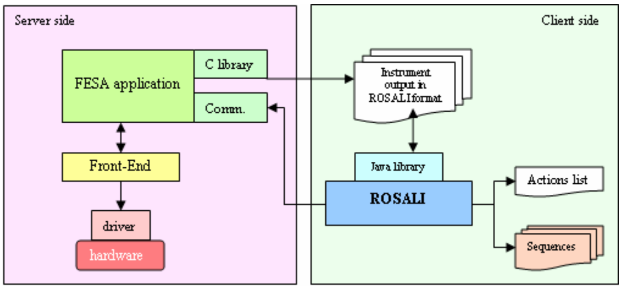


Figure 1: System architecture

ROSALI also provides a rapid and easy way to build and play sequences of actions for testing and assessing the instruments. These actions can subscribe to, store or display ongoing measurements from several beam instruments, retrieve offline data and perform dedicated calculations in the Mathematica kernel [3]...

Another characteristic of ROSALI is to provide harmless access to the instruments. This is achieved by giving full access to the acquisitions and status but restricting the access to the settings to authorized users.

ROSALI FILE FORMAT

To store and share between actions all data coming from the instruments a new file format has been designed based on the following requirements:

- The data files should be human readable.
- This format should be able to handle the main types of data coming from beam instrumentation, i.e lines, XY scatters and surfaces.
- This format should allow the storage in a single file of heterogeneous types of data related to the same acquisition such as 2 profiles, 1 image and the corresponding setting's values.
- This format should be easy to handle in Excel, Mathematica and in-house C/C++/Java programs.

It is composed of tab-separated variables implemented in an XML light format allowing direct use in Excel. Dedicated Mathematica, C and Java libraries [2] have been provided to cover other usages.

ROSALI APPLICATION

As shown in Figure 2, the graphical interface of the application is divided into three main areas together with the toolbar.

- The *Sequence* area contains the list of actions to execute.
- The *Parameters* area contains the parameters of the selected action. These parameters can be used as input (equipment name, target storage directory…) and/or output (result file name,) by the action. The value of an input parameter can also be set as a reference to the value of an output parameter of another action.
- The *Viewer* area shows the result after the execution of the selected action. Each action can have its own viewer (simple table, standard data viewer or custom panel dedicated to beam profile measurements for instance)

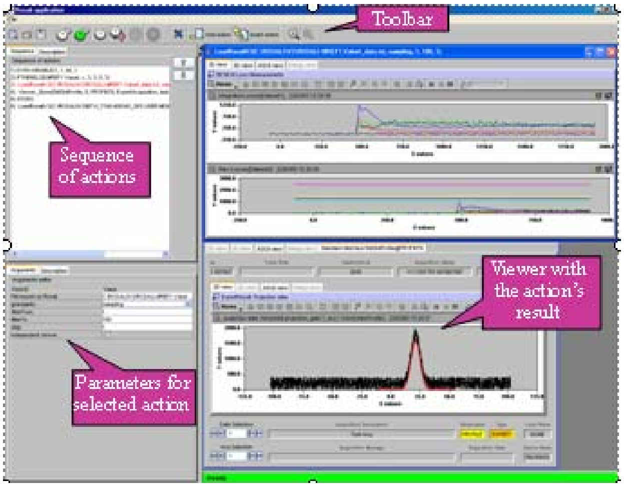

Toolbar
Sequence of actions
Viewer with the action's result
Parameters for selected action

Figure 2: ROSALI user graphical interface

To create a sequence in the ROSALI context, the user has to select the actions from a menu in the *Toolbar*, setting the values for their input parameters in the *Parameters* area; then the sequence will be ready for execution. The result obtained from the execution of a single action will be showed in the *Viewer* at each step of the loop. The user can execute the whole sequence, just one action or a set of actions from the selected one until the end. ROSALI provides users with control statement actions like *if-then*, *repeat*, *goto* or *stop* to create dynamic and/or never ending sequences.

The sequence can then be saved and reloaded for subsequent uses. The sequence saved will contain the list of actions and corresponding parameters together with the creator's contact details, the date of last modification and a brief description of the sequence and how to use it.

ROSALI ACTIONS

An action is a single instruction to execute in the ROSALI environment. Every action has a set of parameters that define its behaviour and the contact information of the developer who gives support to that action. To ensure the harmless objective of the application, every action has a list of users who can see and use the action.

The ROSALI actions can be classified in two groups: the *ready-for-use* actions and the *user-defined* actions that need some development from the user in order to be executed.

Ready-For-Use Actions

For these, the user only needs to set the values for the input parameters. These actions can:

- Subscribe to ongoing measurements from several beam instruments and save them in a file. If selected, the measurements will be displayed in a dedicated viewer depending on the instrument family, being position, profile, intensity or loss monitors.
- Retrieve and display data from previous recordings.
- Merge these measurements to obtain correlation diagrams.
- Execute the different control statement actions.
- Synchronize the sequence execution to machine events.
- …

The action builder tool (see next chapter) allow BDI software expert to upgrade ROSALI with new actions whenever necessary.

User-Defined Actions

The ready-for-use actions can provide standard and common services useful for every user but not linked to a specific evaluation or experiment. To allow users to implement and modify online their data treatment and algorithms, ROSALI also provides *user-defined* actions. The main one currently available is the RosToMath action that gives access to the Mathematica kernel from ROSALI via Mathlink. This action basically includes in its viewer all the power of Mathematica including graphic output. The user can in the viewer edit Mathematica cells, execute them and see the result, being text or graphics. The Mathematica code written will be executed each time the RosToMath action is executed in the sequence.

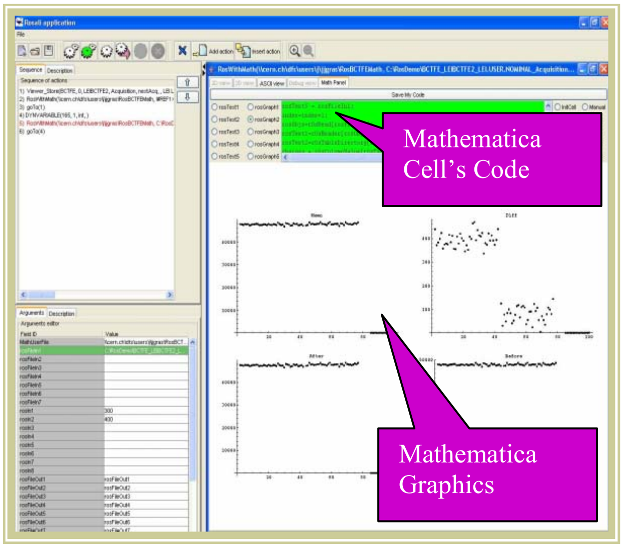


Figure 3: ROSALI / Mathematica Action

Fig. 3 shows the result of a simple use case where a user is testing a scraper efficiency and wants to record every cycle the current evolution and see the evolution of the beam losses before and after scraping of many cycles.

The user built the following sequence of actions:

1. store next BCT acquisition (standard ready-for-use action)
2. add the point corresponding to last acquisition in the loss history graphs via the RosToMath action with dedicated Mathematica code written by the User. This Mathematica code has a direct access on the RosToMath action's parameters (to give the scraping window time in our case) and on

data stored in ROSALI files via a dedicated Mathematica function libray[*] [2].

3. Goto action 1 (standard ready-for-use action)

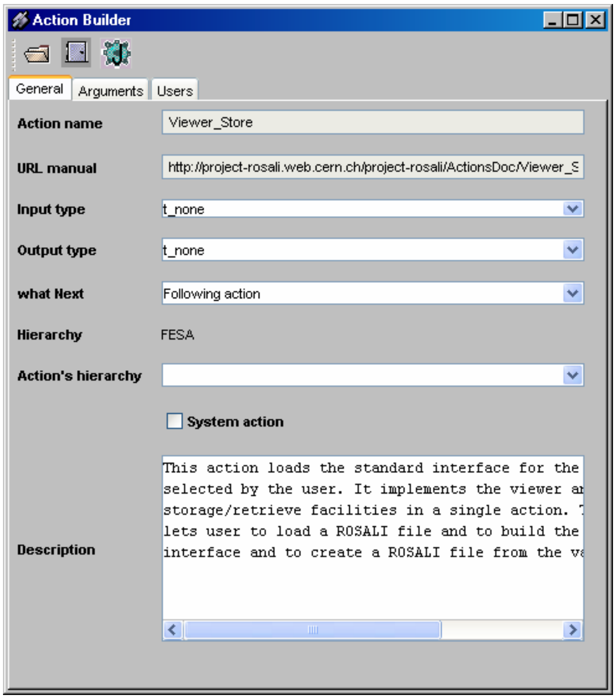
The user can build this action in a few minutes without any external interventions, test it immediately and store it for future usage. It will give him in real time the evolution of the scraping efficiency. The user can then add the following actions to the same sequence:

4. increment acquisition index (standard ready-for-use action)
5. add the point corresponding to acquisition number 'index' in the loss history graphs via the RosToMath action with dedicated Mathematica code written by the User.
6. Goto 4 (standard ready-for-use action)

It will give him the same information offline from the files stored during previous tests and also provide the opportunity to implement a new algorithm on these data. The user just has to select action 4 and start the sequence from there.

ACTION BUILDER TOOL

To make easier the ROSALI developer's task of creating a new ROSALI action (being ready-for-use or user-defined), the **ActionBuilder** tool has been provided (see Figure 4).



Action name: Viewer_Store

URL manual: http://project-rosali.web.cern.ch/project-rosali/ActionsDoc/Viewer_S

Input type: t_none

Output type: t_none

what Next: Following action

Hierarchy: FESA

Action's hierarchy:

□ System action

Description:

```
This action loads the standard interface for the
selected by the user. It implements the viewer a
storage/retrieve facilities in a single action.
lets user to load a ROSALI file and to build the
interface and to create a ROSALI file from the v
```

Figure 4: ActionBuilder tool

This application lets developers to specify the action's general information and their input parameters in a visual way. The developer will set the action's name, the URL to the online documentation and a brief description as general information.

For every parameter, he/she will need to specify a name, the order of appearance in the list of parameters and an optional initial value. Once all the information is set, the developer will press the "Generate code" button that will generate the Java code for the action. This code is static, so the developer will need to supply the code to execute when the action is executed inside the ROSALI tool. The developer supplies an *execute()* method in which he/she has a direct access to the input parameters values via the methods *getArg(parameterName)* and *setArg(parameterName, value)*.

PROJECT STATUS

The application described in this paper has been tested last year on the SPS and it will be used this June for the commissioning of the new LEIR [4] machine which is all based in FESA instruments and from where we can take advantage of all ROSALI's features.

The software tools have been developed in Java and are thus platform independent. They are currently used under Linux and Windows 2000/XP.

CONCLUSIONS

The tests carried out on the SPS during the year 2004, show that ROSALI is a very useful application for obtaining and analysing online and offline data from the beam instruments. The non-dependent format of data allows physicists to treat them with different applications such as Excel, Mathematica or ROSALI itself. The dedicated and upgradable actions ensure that the physicists' needs will be always covered, developing new specialized actions if needed.

ACKNOWLEDGMENTS

The development of the ROSALI application was achieved in a short time thanks to the collaboration of many colleagues. In particular, Jean-Jacques Gras helped me in the design of the application, giving to me the practical view of it; BDI Software section who tested the application and supported me in the implementation. Finally, I would like to thank Jean-Pierre Koutchouk who implemented the Mathematica library and supported us with his advice in the use of Mathematica and its features.

REFERENCES

[1] A. Guerrero and al., "CERN Front-End Software Architecture for Accelerator Controls", ICALEPCS'03, Korea, October 2003.

[2] ROSALI web site: http://project-rosali.web.cern.ch/project-rosali

[3] Mathematica web site: http://www.wolfram.com/products/mathematica/index.html

[4] http://project-i-lhc.web.cern.ch/project-i-lhc

[*] This library also allows direct access to the ROSALI files through pure Mathematica notebooks