REMBRANDT – THE REMOTE BEAM INSTRUMENTATION AND NETWORK DIAGNOSIS TOOL

T. Hoffmann, H. Bräuning, GSI, Darmstadt, Germany

Abstract

As with any large accelerator complex in operation today, the beam instrumentation devices and associated data acquisition components for the coming FAIR accelerators will be distributed and installed over a large and partially inaccessible radiation exposed area. Besides operation of the device itself, like acquisition of data, it is mandatory to control also the supporting LAN based components like VME/µTCA crates, front-end controllers (FEC), middle ware servers and more. Fortunately, many COTS (commercial off-the-shelf) systems provide means for remote control and monitoring using a variety of standardized protocols like SNMP, IPMI or iAMT. REMBRANDT, the REMote Beam instRumentation And Network Diagnosis Tool, is a Java framework, which allows the authorized user to monitor and control remote systems while hiding the underlying protocols and connection information such as IP addresses, user-ids and passwords. In addition to monitoring the device state (like voltage and current load), the main features are the remote power switching of the systems and the reverse telnet boot process observation of FECs. REMBRANDT is designed to be easily extensible with new protocols and features. The software concept, including the client-server part and the database integration, will be presented.

INTRODUCTION

At FAIR [1], the Facility for Antiproton and Ion Research, presently under construction at GSI, Darmstadt, Germany, several new accelerators and storage rings such as the superconducting heavy ion synchrotron SIS 100, the high energy storage ring HESR, the collector ring CR, the inter-connecting high energy beam transfer lines HEBT, the super fragment separator S-FRS and experiments will be built (see Fig. 1).

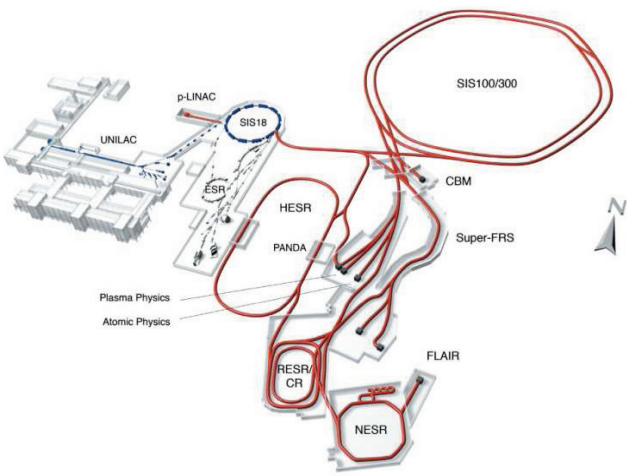

Figure 1: Scheme of the existing GSI and the new FAIR accelerators. Circumference of SIS100: 1100m

These installations are equipped with roughly 24 different beam instrumentation (BI) device types combining approximately 1000 readout channels, which are connected to many diverse data acquisition (DAQ) systems. These DAQ systems are well distributed all over the spacious FAIR campus in 17 dedicated electronic rooms, several niches and experimental caves, some of them inaccessible at beam time due to radiation safety requirements. To reduce trouble-shooting efforts and reaction times on DAQ system failures, the BI data acquisition concept for FAIR includes an out-of-band (OOB) remote control and monitoring system for all network based DAQ hardware components. The surveillance of general IT infrastructure and services such as switches, DHCP, Sendmail, etc. which could be handled by existing tools [2] like OpenNMS or Nagios, are not part of REMBRANDT and belong to the accelerator network IT department.

The following supported BI DAQ hardware was specified [3]: VME64X for multi-channel applications, 'µTCA for Physics (MTCA.4)' for high data rate applications and Industry PC (IPC) systems, based on PICMG 1.3 backplanes, as low cost solution with minor amount of readout channels. As a main requirement, all these diskless systems are able to execute a full hardware power cycle either by OOB LAN access or indirectly via a remote controlled power cycle of the hosting crate. In addition, other device specific hardware properties, such as fan speed, voltages, temperatures etc. can be monitored.

ARCHITECTURE

REMBRANDT is based on a client-server architecture, which is shown in Fig. 2. The clients as well as the server are fully implemented in Java. For the clients, Java was an obvious choice for several reasons. First, all graphical user interface (GUI) applications in the control room will be implemented in Java. Second, with Java the clients will automatically run on any operating system supported by Java. Thus, the same application can be started in the main control room (Linux X-terminals) and/or on the office PC (typically Microsoft Windows). For a monitoring application, running on office PCs is a viable scenario. In addition, the web-start feature of Java makes remote deployment of the GUI clients very simple. Implementing the server in Java may not be as obvious. However, a monitoring framework is not expected to have a high data rate and thus does not require the utmost in processing speed. Java on a modern server provides more

ISBN 978-3-95450-139-7

than enough computing power for this purpose. With the clients and the server implemented in Java, communication between both is very much simplified by using the inherent Java tools. In addition, a survey showed, that for the major protocols to be implemented (IPMI, iAMT, SNMP, Telnet, SSH), open source libraries already exist for Java.

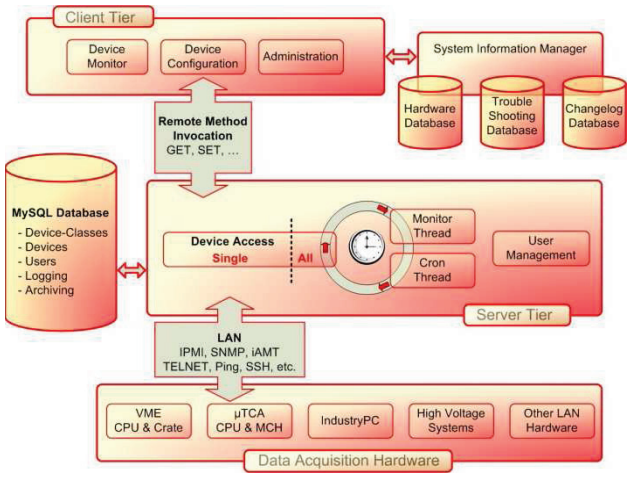

Figure 2: The REMBRANDT software architecture

Server

The REMBRANDT server tier is hosted on a single storage server. The use of multiple redundant servers is not foreseen. The storage server also hosts the MySQL database which contains all relevant data used by the REMBRANDT system.

The server obtains the list of devices to be monitored from the database. Each device belongs to exactly one device class. A device class describes the access methods and the properties to be obtained from the device. The properties are for example the power state or sensor information like fan speed, bus voltages etc. Access methods describe how the properties are retrieved from the device and how they can be set on the device. They specify the protocol to use and protocol specific access codes, like OID's for SNMP. Currently REMBRANDT supports the following protocols:

- SNMP (Simple Network Management Protocol) via the snmp4j library [4]
- iAMT (Intel Active Management Technology) via Java web-services
- iAMT SOL (Serial-Over-LAN) by porting AMTTERM [5] to Java
- IPMI (Intelligent Platform Management Interface) via the Verax IPMI library [6]
- Rudimentary Ping which checks if the host is reachable but does not really implement the ping protocol
- telnet via the apache commons net library [7]
- ssh via the ssh2 library from JCraft [8]

In addition, simple expressions or more complex Groovy [9] scripts can be specified to extract properties from the returned data (i.e. extract specific bits from integer values). Multiple protocols are allowed per device class. The device class also describes more interactive access methods like telnet or ssh, which can be used to obtain an interactive shell with the component or monitor its boot process.

Besides monitoring, REMBRANDT is also expected to support a certain level of remote maintenance. This can range from simply switching the device on or off, to configuring thresholds for over/under current or fan speeds. To prevent cluttering up the user interface and to implement a simple access control, device classes specify operating modes. Currently, only two modes are realized: 'control' and 'configuration'. More may be implemented easily. Each mode contains its own list of properties, protocols and access methods. The standard mode is the 'control' mode, which allows monitoring the device, resetting and switching it on or off. The 'configuration' mode is provided for further expert settings like fan speed or current limits, which may negatively affect the device function.

A device finally is an instance of a device class which provides the actual network addresses. For each protocol, an IP address must be specified. The address may be different for different protocols of a single device. This allows for example to ping the network port of a FEC and obtain the console of the same FEC via a terminal server with a given IP address and port. Besides the IP address, other relevant information like network ports, user names and passwords are part of a device definition. All data is stored in the database with the passwords being encrypted to provide to some degree a level of security.

Devices can be grouped to represent systems. In this case a system is assigned a summary status based on the status of each device of the system.

The REMBRANDT server periodically queries all devices in the 'control' mode. The query is performed in a separate monitor thread typically every 10 seconds. The devices are queried sequentially and the results are evaluated after data from all devices has been received. This evaluation allows to assign error states to the devices. If a VME FEC is not reachable and the corresponding VME crate is 'off', this is a normal 'power off' state. On the other hand, an unreachable FEC, while the crate is 'on', is certainly an error condition. Changes in the device state (i.e. power on / off, over current etc.) are logged into the database together with a time stamp and severity level.

While being implemented very easily, the sequential query of all devices in a single monitor thread has one major draw back. If many devices are not reachable, the monitor thread may be blocked for a significant time while waiting for timeouts. REMBRANDT measures the access time to the individual devices and provides tools to optimize the timeout settings (see Fig. 3). However, too short timeouts are also to be avoided as they lead to

ISBN 978-3-95450-139-7

intermittent incorrect error messages when the network response was slow. One solution will be to parallelize the query over several threads while incuring a higher complexity in synchronization. This has been planned but not yet realized.

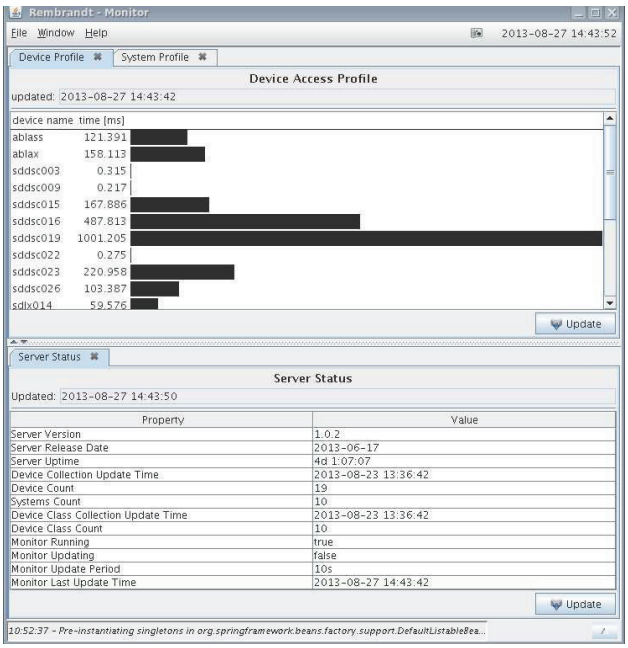

Figure 3: The REMBRANDT Server Monitor. Top: Online access time measurements for timeout optimization. Bottom: General server status overview.

In general, clients will periodically retrieve the device data collected by the server's monitoring thread. However, clients may also access single devices at any time via the server directly. This is always the case when data is sent to a device.

REMBRANDT controls access via internal usernames and passwords. Each client requests the username and password at startup and verifies it with the server. Any request to the server also includes the username and password. The server validates the information before executing the request. Users are also assigned to roles which define the level of access granted. In general, any user can monitor the system (read access). Specific roles include the ability to power on/off devices (i.e. access them within the 'control' operation mode), configure devices (i.e. access within the 'configure' operation mode), create new devices and device classes, and create new users (write access). Currently, these roles are not device specific. In the future, a more detailed role-based access system may be implemented to restrict users to specific devices.

The server also executes maintenance tasks. These are always executed at midnight although a more flexible scheduling could be implemented. Currently, this is used to periodically purge old log entries and other periodically acquired data from the database to keep the table size reasonable.

Clients

The REMBRANDT system currently provides several clients:

- a monitoring client, which is used to monitor the devices and possibly reset them or turn them on or off. It uses the 'control' operation mode to access devices.
- a remote configuration client, which is used to configure devices. It uses the 'configure' operation mode to access devices.
- a system information manager, which manages static information about the devices.
- a database tool, which allows to create new device classes and devices.
- a server monitor, which displays information about the server as well as latency times in the device access.
- an administration tool to create new users for the REMBRANDT system.

All clients interact with the server exclusively by Remote Method Invocation (RMI), which is a proven standard mechanism within JAVA for procedure calls within a network. Thus, complex subscribe/publish procedures are avoided.

The monitoring and configuration clients obtain not only the acquired device data from the server but also the list of all devices handled by the server. This ensures consistency between server and client independent of the state of the database. Other information can be obtained directly from the database.

The most significant REMBRANDT client is the monitor client, which presents the status and device information of all devices obtained by the server. A typical screenshot is shown in Fig. 4.

The panel on the left side is the device explorer which is used to select devices. The grouping of devices into systems is clearly shown. The client provides several panels (i.e. in the two panels on the right of the device explorer) to show the system information. A simple tabular panel can be used to show the status of selected devices, which can be added to the form by drag and drop from the explorer. Two other panels provide more detailed information of systems and devices. The latter is shown in the screenshot in the top right panel. It provides the most detailed information including everything obtained from the device like fan speeds, currents etc.

The client also provides a console panel for those devices which have a corresponding protocol (telnet, ssh or iAMT SOL) specified. The console panel provides easy access to login consoles and terminals directly or via intermediary components, which route the serial console output to Ethernet like terminal servers (i.e. MOXA CN2510) or certain VME crates. One major usage is the observation of the device boot process for problem diagnostics. The console panel also provides more advanced features like automatic login with the device

ISBN 978-3-95450-139-7

specific username and password or a configurable menu to send commands to the device via the console.

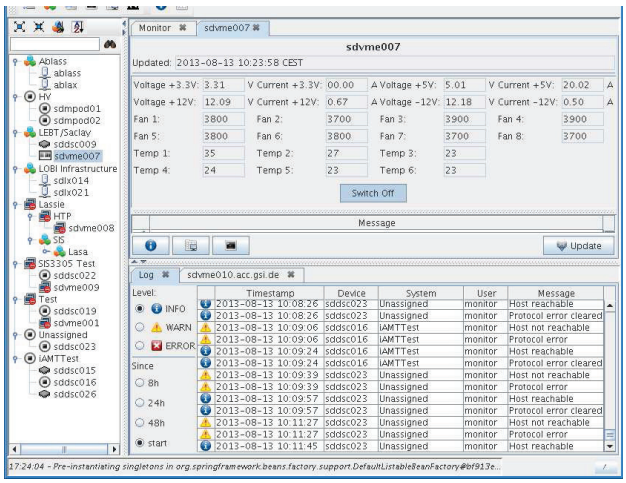

Figure 4: The REMBRANDT Device Monitor. Left: device tree sorted by different categories. Top: SNMP data set for a VME crate. Bottom: general message overview.

Further panels show the status log messages (lower right panel in the screenshot) or static device information from the system information manager. The information manager is an extensible system to provide static and semi-static device information. It currently contains three modules.

The first module accesses the hardware database of the BI department. This database collects information on all DAQ modules in the system, their models, location, usage, contact person and more. Using the system information manager, this information is available read-only to the REMBRANDT clients.

The second module implements a kind of changelog for devices. It is intended as a log for changes in the device configuration applied by the expert, as for example, changes in the current threshold settings. Entries in the changelog occur not automatically, but must be made by the user. All entries contain a short message describing the change, the device name, timestamp and name of the user.

The third module attempts to be a rudimentary expert system. In case of protocol errors reported by the server after device access, this module will scan the 'knowledge database' for problems matching the reported error. It will then display the corresponding information/solutions found in the database. The knowledge database is populated using the system information manager client and is currently in the build up phase.

OUTLOOK

REMBRANDT will significantly help to keep a status overview over the huge amount of expected BI DAQ and infrastructure systems at FAIR. It is currently in the test phase and already covers over 90% of the foreseen devices types. But development has not finished. One major point to investigate now will be the scalability with a much larger number of devices. In addition further protocols are foreseen like

- IPMI Serial-Over-LAN
- Evaluation of logfiles like syslog
- Monitoring of FESA [10] based DAQ front-end software via RDA [11]

Future improvements may also include active messaging, for example by e-mail in case of problems with critical devices.

Furthermore, REMBRANDT should be considered as only one, albeit major, pillar in the global remote monitor and control concept for beam diagnostic devices. It is complemented by IP based KVM switches for VGA/USB equipped systems and PLC controlled power supply units of beam line installed DAQ components, such as digital cameras, to provide permanent access and full remote reset capability.

ACKNOWLEDGEMENT

The authors would like to thank Nicolas de Metz-Noblat from CERN (BE-CO) for many fruitful discussions and his generous support over the last years.

REFERENCES

[1] FAIR - Baseline Technical Report, GSI (2006).
[2] http://www.opennms.org/ and http://www.nagios.org
[3] Detailed Specification of the FAIR Beam Diagnostic Data Acquisition system, https://edms.cern.ch/document/1179740/4
[4] SNMP4J: http://www.snmp4j.org
[5] amtterm: https://www.kraxel.org/cgit/amtterm
[6] Verax IPMI Library: Verax Systems http://www.veraxsystems.com
[7] Apache Commons Net: http://commons.apache.org/proper/commons-net
[8] JSch (Java Secure Channel) SSH2 implementation http://www.jcraft.com/jsch
[9] Groovy: http://groovy.codehaus.org
[10] T. Hoffmann, M. Schwickert, G. Jansa, "FESA at FAIR", PAC09, Vancouver, BC, Canada, p. 4794.
[11] N. Trofimov et al., "Remote Device Access in the New CERN Accelerator Controls Middleware", ICALEPCS'01 San Jose, California, USA, p. 496.

ISBN 978-3-95450-139-7