IMPROVING SOFTWARE SERVICES THROUGH DIAGNOSTIC AND MONITORING CAPABILITIES

P.Charrue, M.Buttner, F.Ehm, P.Jurcso, CERN, Geneva, Switzerland

Abstract

CERN's Accelerator Controls System is built upon a large set of software services which are vital for daily operations. It is important to instrument these services with sufficient diagnostic and monitoring capabilities to reduce the time to locate a problem and to enable pre-failure detection by surveillance of process internal information. The main challenges here are the diversity of programs (C/C++ and Java), real-time constraints, the distributed environment and diskless systems. This paper describes which building blocks have been developed to collect process metrics and logs, software deployment and release information and how equipment/software experts today have simple and time-saving access to them using the DIAMON console. This includes the possibility to remotely inspect the process (build-time, version, start time, counters,..) and change its log levels for more detailed information.

INTRODUCTION

Operating the CERN accelerators relies on a Controls Infrastructure in a perfect state. The CERN Beams Controls Infrastructure is deployed all around the CERN site, spanning from the Operator console in the control room to the device connected to a distant fieldbus close to the beam, via hundreds of servers, FrontEnds, networks and of course software applications.

The Beams Controls Group has studied, developed and deployed a complete diagnostic and monitoring infrastructure (DIAMON)[1] capable of detecting and repairing faults in the whole Controls infrastructure.

DIAMON

DIAMON is following the classic multi-layer system approach allowing high level of flexibility. As illustrated in Figure 1, specific DAQ modules, based on the C²MON[2] DAQ Core assure the communication between a variety of data providers and the DIAMON server. The communication can be uni- or bi-directional (support for settings and commands).

With a set of DAQ core plugins we establish communication to all required types of devices. The DIAMON server handles the messages received from the DAQ layer, computes the state of the rules (business logic) and provides the results to the client applications through C²MON Client API. The communication between components of the system layers is based on JMS messages and the scope of the monitored information is configured through a set of web applications operating on DIAMON configuration database. All configured points (further called *metrics*), alarms and rules (so-called *limits*) are handled inside the server's internal cache and the states are periodically persisted into the server's cache-persistency database. Updated metrics and limits are also periodically written into the Short-Term-Log (STL) database, which holds thirty days snapshots and provides the data on demand to the client applications. This allows our users to replay the chain of events and precisely analyse the problems retroactively. For all GUI clients and web-based client applications an instance of the Tomcat servlet container has been set-up, hosting a Spring MVC-based Java module which renders a set of web pages with the data received at runtime from the server as well as the offline data from the STL database.

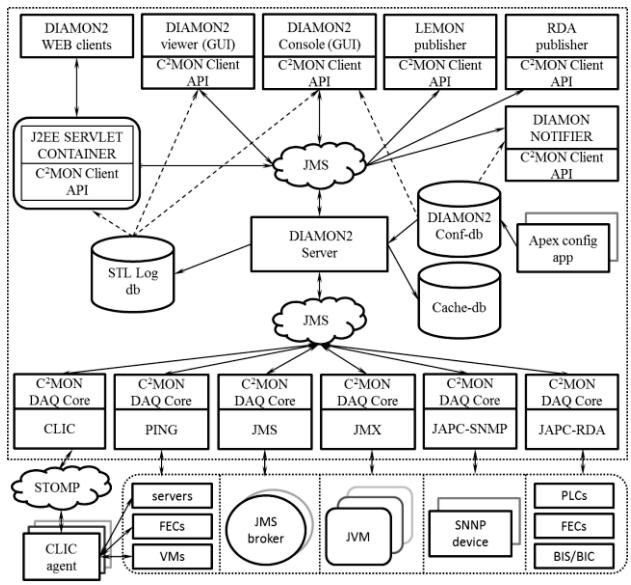

Figure 1: Overview of the DIAMON architecture.

THE DIAMON CONSOLE

The DIAMON console is the main tool that is used to retrieve and display monitoring data in a simple and homogenous way.

Specific views can be configured to target the family of devices one need to monitor, being all computers involved in the control of a specific accelerator, devices providing a specific service (such as Timing, Middleware, …), devices belonging to a specific group (such as Beam Instrumentation, RF, Power Converters, …) or the complete list of all monitored devices (~3000 at the time of writing).

Standard views are made available directly from the DIAMON console to cover the most used views, such as

ISBN 978-3-95450-148-9

one view per CERN accelerator (LINAC, BOOSTER, PS, SPS, LHC, …), one per most used equipment group (BI, RF, …), one per main framework (CMW, OASIS, …).

Figure 2 below gives a screenshot of the DIAMON console.

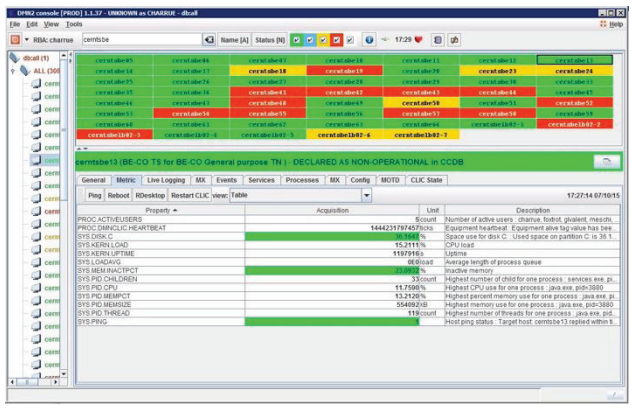

Figure 2: Diamon GUI.

The application window contains three separate panels:

- The tree on the left-hand side provides a hierarchical view on the monitored equipment, for instance with 3 levels: the accelerator, the controlling computer, the connected equipment. Users can configure the levels and content of the tree.
- The top right panel displays the full list of equipment corresponding to the element selected in the tree. The background color clearly displays the status of each item (green: OK; yellow: warning, red: error).
- The bottom right panel (the "Detail panel") displays all details for the element selected in the left hand-side panel (if the selected element is a leaf node) or in the top right panel.

DATA SOURCES

As shown in the bottom of Figure 1, the DIAMON infrastructure is fed with data from many different sources:

The first source of data is the CLIC agent running in every computer deployed in the controls infrastructure. It runs in fixed intervals of time and collects data about the system measurable such as CPU load, memory or disc usage, network throughput, presence of specific system and user software as well as data about connected hardware (timing boards, World FIP bus).

Then a few centralized agents running on a dedicated central server poll lists of devices to obtain the information to be monitored. This approach is used to monitor controls infrastructure equipment, where software can not easily be installed, like PLCs, video converters, power supplies etc.

In addition, DIAMON verifies in regular intervals network availability of all monitored computers using the PING DAQ.

DIAMON is also capable of exploiting information gathered by the Tracing initiative [3] from several SYSLOG and tracing files produced by the Controls computers. This information gives real time information from the running Operating System but also from Controls Frameworks such as FESA [4] (our FrontEnd realtime framework), CMW [5] (our Controls Middleware communication infrastructure) or our framework deployed to remotely restart specific application (wreboot)

GATHERING DATA FROM APPLICATIONS

The Controls Infrastructure is also composed of a huge number of JAVA and C++ applications used to control and drive the beams through the several machines in our accelerator chain. The presence and correct behaviour of these software applications are essential for the operation of our accelerators.

Therefore, DIAMON needs to be informed about any problems in the software applications deployed in the controls infrastructure. For this purpose, a specific framework for Java and C++ application has been deployed, JMX and CMX [6], allowing specific metrics to be published as shown in figure 3 below.

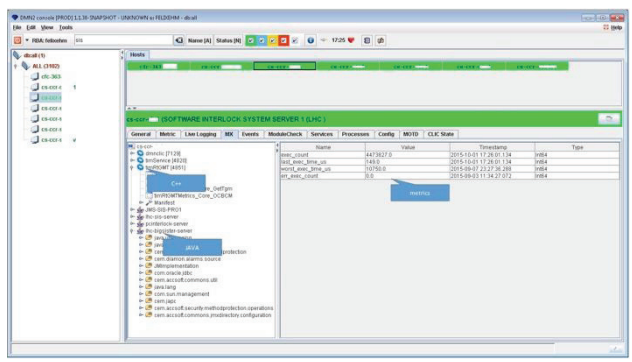

Figure 3: Metrics sent by C++ or JAVA controls application

Moreover, this framework can display the sometimes complex dependency graph allowing the detection of mismatch in the versions used to build specific application, as shown in Figure 4 below.

ISBN 978-3-95450-148-9

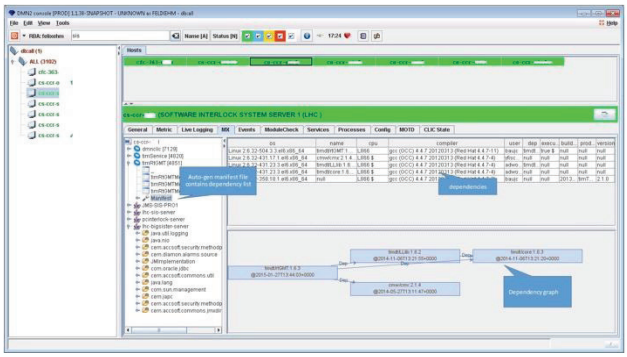

Figure 4: Display of a dependency graph for a C++ controls application

APPLICABILITY

This initiative to gather extremely different data coming from many different source into the same DIAMON tool has been very much appreciated both by the Controls experts and by the Beams Operators.

As shown on Figure 2, the Detail Panel offers a wide range of tools to access to diagnostic data, automatically configured according to the device being selected. Static data, such as the location of the device, the person responsible for its maintenance, the hardware and software configuration, etc, are available with one click of the mouse. Up to 30 days of history stored in the DIAMON server allow to easily visualise the last events on the device, such as reboot, restart of specific process, logins, …

Live logs, with on-the-fly tuning of the log level, can be visualised in real time, allowing a visualisation of the current behaviour of the selected device.

Users identified via our Role Based Access Control (RBAC) and having the correct authorisation rights can, still using the same standard DIAMON console, perform some operation on the selected device, such as triggering additional data acquisition, restarting selected processes or rebooting a machine.

The initial target of reducing the diagnostic time and of correlating data from heterogeneous sources has been reached today.

The CERN Controls Infrastructure contains more than 8'000 intelligent computers capable of producing more than 150'000 metrics. Thanks to the standard DIAMON console, to the JMX/CMX standardisation and to the concentration of the tracing initiative, it is now much easier to diagnose and make the correct action to solve a controls issue.

In addition, controls experts as well as equipment specialists are running local configuration of the DIAMON tool, with a specific target on the devices and infrastructure under their direct responsibility. By using all the DIAMON facilities (such as information mails or SMS), by correctly tuning the thresholds on the metrics under scrutiny, appropriate pro-active measures can be prepared and applied, thus avoiding possible problems to appear.

CONCLUSION

DIAMON infrastructure is a very powerful tool to gather diagnostic information on our distributed Controls Infrastructure. The investment into standardising and acquiring data from many different sources allows for a faster and better diagnostic of blocking situation for the Beams Operation, hence reducing our diagnostic time and therefore limiting to the maximum the time lost due to issues in the controls infrastructure.

In addition, this valuable collected data is also used by the controls experts to make pro-active actions that prevent an emerging issue to perturb Operation.

REFERENCES

[1] Diagnostic and Monitoring the CERN Controls Infrastructure: The DIAMON Project: First Deployment in Operation, ICALEPCS'09, Kobe, Japan, (2009)

[2] A Suwalska, M. Buttner, M. Brager, W. Buczak "C^{2}MON CERN control and monitoring platform", CERN, (2011)

[3] ACET – Accelerator Controls Exploitation Tools - https://wikis.cern.ch/display/ACET/Tracing

[4] FESA – Front End Software Architecture, https://wikis.cern.ch/display/FESA3

[5] CMW – Controls Middleware infrastructure, https://wikis.cern.ch/display/MW/

[6] F. Ehm et al. "CMX - A Generic In-Process Monitoring Solution for C and C++ Applications" Proceedings of ICALEPCS'13, San Francisco, USA, (2013)

ISBN 978-3-95450-148-9