SOFTWARE MANAGEMENT IN THE LHCb ONLINE SYSTEM

N. Neufeld*, E. Bonaccorsi, L. Brarda, J. Closier, G. Moine, CERN, Geneva, Switzerland
H. Degaudenzi, EPFL, Lausanne, Switzerland

Abstract

LHCb has a large online IT infrastructure with thousands of servers and embedded systems, network routers and switches, databases and storage appliances. These systems run a large number of different applications on various operating systems. The dominant operating systems are Linux and MS-Windows. This large heterogenous environment, operated by a small number of administrators, requires that new software or updates can be pushed quickly, reliably and as automated as possible. We present here the general design of LHCb's software management along with the main tools: LinuxFC / Quattor and Microsoft SMS, how they have been adapted and integrated and discuss experiences and problems.

INTRODUCTION

LHCb [1] is one of the four large experiments at the Large Hadron Collider at CERN. Like all these experiments a large IT infrastructure is at the core of the trigger and control system. In the case of LHCb it consists of more than 1000 servers, desktops and embedded computers running applications ranging from simple control of power-supplies to complex SCADA systems, which assure the overall control of the entire facility.

The wide variety of tasks brings with itself quite a variety of software and hardware which need to be managed and operated by a small team of system administrators. The hardware management is described elsewhere [2], but the large variety of the hardware increases the complexity of the software management as well.

REQUIREMENTS

The following requirements must be met by a useful software management system for LHCb. In the context of this paper the term *user* refers to *both* a human operator or developer and a software application or framework using a specific piece of software.

- Multi-platform: Linux and MS-Windows must be supported in several versions as well as in 32-bit and 64-bit.
- Versioning: it must be easy to change forth and back between different versions of a software.
- Scalability: software management is quite simple when every user gets the software from a single location. For a large system a more scalable solution is needed however.
- Speed: Software distribution should be very fast, lest there will be a strong temptation to bypass it as urgent changes are frequent in the commissioning phase of an experiment.
- Management capabilities: software management is more than then simply managing versions of packages. In particular operating system (OS) software needs specific configuration which can vary from individual machine to individual machine.
- Compatibility: ideally one system would allow the mangement of *all* software in the entire experiment on every single computer. However there is a lot of existing infrastructure and legacy tools, which in practice cannot be replaced. So a software management tool must be open and compatible with existing solutions.

The first and last point already high-light the major difficulty that in practice there can be no common tool for all. Accordingly a set of tools and the underlying rationale and policies will be described in the following.

LHCB'S SOFTWARE BASIS

In this section the software used in the LHCb Online system is categorized and existing "native" tools are described where they exist.

Operating Systems

OS software here is understood to comprise not only the core OS (which is quite small) but also all the software which make a general purpose OS useful: such as the shell, an ssh-client, word-processor, web-browser, to name but a few.

As operating systems Scientific Linux CERN [3] in version 4 and 5, both in 32 and 64-bit are used. Scientific Linux derives from the commercially very succesful RedHat distribution and consequently uses the Redhat Package Management (RPM) [4] system to manage packages. RPM provides efficient version management of packages because it is built around a database. Further automatisation is done using the YUM (for Yellowdog Update Management) [5] tool, which is widely used on RedHat and the freely available Fedora desktop systems. YUM has a powerful dependency management, which makes installing a software package very easy as YUM will automiatcally install all required packages in the required version. RPM and YUM by themselves provide little or no support for configuration management and they are biased towards *updating*, i.e. running the latest version available. Installing

* niko.neufeld@cern.ch

LHCb-CONF-2009-043
20/10/2009

several versions of the *same* package requires some special effort.

The second family of operating systems are Microsoft Windows XP and Windows 2003 server. Currently these are only used in their 32-bit versions. Microsoft has its own propriatary system for OS updates (Windows Update) which can be mirrored locally. Software packages are (ideally) distributed as MSIs, but this is not such a widely established standard as RPM in the RedHat world and indeed we have Windows applications which come as exectuables, various install-kits or even as bare ZIP-archives. Version management is also rather weak as it consists essentialy of free-form version numbers in the Windows registry, whose usage are not enforced by the OS. Configuration management is not done by either of these tools. In modern Windows installations ("domains") this is achived by the use of *group policies*. For package management we use Microsoft System Management Server (SMS) [6].

Application Software

The "physics" software in LHCb, that is all the code written to analyse and process physics data, is version-managed by a custom system called CMT [7] and distributed in the form of tar-archives. Configuration of the physics applications is done either by text files (so called option files) or via steering scripts written in Python. One of CMTs strong points is that it is very easy to have multiple versions of the same package installed and choose (in principle) freely between them at run-time. Configuration is limited to versions of sub- and dependent packages.

The Experiment Control System (ECS) is based on the JCOP framework [8]. JCOP has developed its own component management system which includes versioning support and (optionally) a centralised database with automated installation of components. These components are essentially scripts and data-structures for use of the PVSS SCADA software used throughout the ECS. JCOP provides a configuration and package management build using an Oracle database, which is currently not used by LHCb.

SOFTWARE MANAGEMENT IN LHCB

LHCb has an isolated local area network. Isolated means that (almost) no packets are directly routed between the CERN networks, let alone the Internet and the LHCb controls network [9]. Access from outside is only possible via dedicated gateway machines, which are dual-homed. This architecture allows LHCb to only update the very few exposed gateway machines whenever important security patches come out. These machines do not run any operationally important tasks, i.e. the experiment can run without these machines, albeit in this case it can be monitored only by on-site operators.

For machines inside the network, OS updates are infrequent and reserved for special maintenance windows. For Linux these updates are in form of RPM package-lists for Windows typically as service packs. Both RHEL4/5 and Windows 2003 are very mature operating systems. Excepting security there are not so many updates or feature additions.

Linux Software Management

Linux software which is *not* part of the physics software and hence managed using CMT is only accepted in form of RPMs. As it is quite easy to create RPMs, any software which we are asked to install is first packaged into an RPM, usually by an administrator. Good RPMs use *pristine* sources, that means that the packager takes the original software as it is and only applies a patch to change whatever is necessary to package the software properly. Unless there are strict requirements against this, all software packaged by the LHCb Online administrators is installed in `/usr/local`. As has been mentioned RPMs and YUM repositories are not very suitable for managing multiple versions. However even rigorous testing can not cover all potential problems with a new version of a package, and more often than not it becomes necessary to revert to a previous version on all or, more difficult, part of the cluster. A toolsuite has been developed for the LHC grid initiative, which is called Quattor [10]. An adaptation of Quattor for control systems developed at CERN is LinuxFC (Linux for Controls). LinuxFC/Quattor allow repositories containing all versions of a package and to define in a database which machine receives which version of a specific package. The descriptions of (groups of) machines are called *templates*. Installation of the RPMs is handeled by service processes running on each machine. Quattor components allow configuring almost every "standard" package on a Linux system, including for example privileges for users.

Quattor makes it easy to group machines and also automatizes the installation. Quattor has two disadvantages: firstly it does not provide a tool for dependency checking, so that missing dependencies are only discovered at package installation, secondly it has a distinct flavour of a Unix system administration tool, which results in a steep learning curve and consequently a tendency of people to try to bypass it. We are actively working on higher-level tools to simplify the utilization and increase the the user-friendliness.

For example we have developed a Nagios plugin [2] to verify that the actual installation of a package has worked on a node. An alarm and an email are generated, when the installation fails on a specific node. This is useful as conflicts can depend on the specific software environment of a node and by no means all nodes are equal as in general only required software is installed.

Windows Software Management

There are only about 100 Windows machines in the LHCb Online system, much fewer than Linux (well over 1000), nevertheless our experience is that it is crucial to have automated management as well.

Conceptually what we do is very similar to the Linux part by replacing RPMs with MSIs and Quattor with a combination of group policies and SMS. Again software which we get is re-packaged in MSIs for easy deployment. While the first impression SMS and the Microsoft group policy editor is very good and they seem easier to use than Quattor, in practice it takes a lot of experience and experimentation to decide which configuration task to achieve with which tool. Microsoft has released an updated version of SMS now called System Center Configuration Manager 2007, which we are currently putting production and which promises to greatly facilitate managment, in particular when the migration to Windows 2008 server and Windows 7 will have taken place.

CMT Managed Software

The physics application software used in the High Level Trigger are distributed as CMT packages in Unix tar-balls. Since they contain no meta-information in these tar-balls, all version management is done by external tools using version-numbers in the file-names of the tar-balls. While conceptually simple this approach has disadvantages, when something goes wrong. SMS and RPM both are based on a data-base approach and allow for easy roll-back, which is inherent in the tools, i.e. roll-back requires little or no support from the packager. On the other hand since CMT is also used to manage the physics software on remote site connected to the LHC computing grid, we can leverage the distribution facilities developed for this purpose and updates run automated pulling packages in from a central web-repository.

We install this software only once in a shared filesystem [11]. Replication to more places within the LHCb network, which might be desirable to improve scalability, will require additional work, because these nodes have no access to the central repository.

This software does not require much special configuration other than selecting the desired version and configuring load-paths accordingly. The configuration of internal parameters is part of the operation of the trigger itself and not managed by the system administrators.

Operational Aspects

Only administrators are allowed to perform updates. While this ensures a minimum level of checking, it poses a burden on a rather small team. Quattor and SMSC both have functionality for fine-grained privilege separation which would allow to manage sub-sets of software and nodes, however both these tools are complex and difficult to learn for physicist users. We are currently working on high-level tools to facilitate certain tasks in a robust way, which will then allow delegating them.

Updates must be tested on a test-platform first. We have test installations of all major systems, including an entire "dummy-detector" for this purpose. Afterwards updates are pushed, usually during a maintenance window. Reboots are rarely necessary, we try to maximise the uptime of the machines, normally only the applications are restarted. Many changes on Windows require that the users login a new session, as many group policies are typically applied at login.

Software installations, even major ones, are fairly quick. Typical times for a change on all Linux machines (more than 1000) are in the order of 5 minutes, even though there is only a single software server currently in operation. In the future this can be upgraded if scalability demands it. Software installations on the shared filesystem are, naturally even quicker, however there are scalability issues at run-time.

CONCLUSION

Strict version and configuration management is mandatory to successfullly run a large infrastructure, in particular with a very small team of system administrators. LHCb Online has four broad categories of software, each with their own installation tool. These come from long traditions and are well maintained outside LHCb so we have abstained from trying to integrate them into one common tool, which would be an enormous effort in repackaging. LinuxFC/Quattor and Microsoft SMS are at heart of software and configuration management for all the base services and all the non-LHCb specific software. While powerful these tools remain complex. We try to make the tools more user-friendly for non-system experts to ease the burden by wrapping them in high-level scripts and utilities, which hide the complexity and automatise most of the routine tasks.

REFERENCES

[1] A. Augusto Alves et al. The LHCb Detector at the LHC. *JINST*, 3:S08005, 2008.

[2] Enrico Bonaccorsi and Niko Neufeld. Monitoring the LHCb experiment computing infrastructure with NAGIOS. In *Proceedings of ICALEPCS 2009*, 2009.

[3] Scientific Linux CERN home-page.

[4] Redhat Package Manager home-page.

[5] Yellowdog Updater Modified home-page.

[6] Microsoft Systems Manager Server home-page.

[7] Configuration Management Tool home-page.

[8] The Joint Controls Project home-page.

[9] Guoming Liu and Niko Neufeld. Management of the LHCb network based on SCADA system. In *Proceedings of ICALEPCS09*, 2009.

[10] Quattor homepage.

[11] Rainer Schwemmer and Niko Neufeld. In *Proceedings of ICALEPCS09*, 2009.