LHCb 2005-057

Distributed Analysis Using DaVinci in the gLite Framework

Stuart K. Paterson[1]
University of Glasgow
August 2005

Abstract

The EGEE gLite framework/prototype was used to perform a distributed analysis on the Grid using DaVinci. Topics covered are ease of use, job splitting and the gLite package manager. DaVinci and its dependent package tarballs were inserted into the file catalogue from the release area to be managed by the gLite package manager. An analysis was carried out on the $B_S \to J/\Psi\Phi$ channel and is discussed here in a generic sense. 100K DaVinci events were successfully analysed using the gLite prototype, however reliability issues lead to the conclusion that the system is not yet a reliable tool for LHCb physicists at this early stage in its development.

[1]also Marie Curie Fellow at CPPM, Marseille, France

Contents

1 Introduction

LHCb[1] will start taking an unprecedented amount of B-physics data in 2007. To ensure sufficient resources are available to cope with this, distributed, heterogeneous compute systems are being pooled together to form the Grid. For this reason, it is paramount that B physics analyses in LHCb should be able to be carried out using the Grid. This note describes the first realistic physics analysis using the gLite Grid framework, its implementation and the results derived from this first attempt.

This document starts with a brief overview of DaVinci[2], the analysis software used in LHCb, in Section 2, followed by a description of the gLite framework [3] in Section 3, and concludes with a description of the analysis of $B_S \to J/\Psi\Phi$ carried out with gLite and DaVinci in Sections 4 to 6.

The gLite prototype is a reduced version of the EGEE Grid Middleware[4][5]. This follows a Service Oriented Architecture and utilises the AliEn[6] file catalogue. DaVinci was introduced to the gLite framework and subsequently a physics analysis on the $B_S \to J/\Psi\Phi$ channel [7][8] was carried out.

Using the gLite package manager, analysis jobs were submitted to exploit available Grid resources and test the framework. This required some additional effort but did lead to successful use of the system.

2 DaVinci

DaVinci is the analysis program of LHCb, which is based on the Gaudi Framework[9] and LHCb core packages[10]. Programmed in C++, DaVinci is a collection of distinct packages that are managed using CMT[11]. By using binary releases of the software, currently released as package tarballs, the dependency on CMT can be removed.

In this way, DaVinci depends on five distinct packages, which are: Gaudi, LHCb Software[10], FieldMap[12], ParamFiles [13] and XmlDDDB[14]. The typical DaVinci user will generally only need to modify the DaVinci package itself. This procedure is simplified through the use of options files which steer DaVinci. As such, any additional user-specific libraries may be included using only the options files and the dependent packages may effectively be ignored from the perspective of the user.

DaVinci will eventually be interfaced to the Grid to provide a mechanism for physicists to analyse the data from the LHC. This document describes how DaVinci was included in the gLite framework to run typical jobs.

3 The gLite Framework

The gLite Framework is a reduced version of the Enabling Grids for E-Sciences in Europe (EGEE) Grid Middleware. The first instance of this was the gLite Prototype[15] which uses the AliEn file catalogue. As stated in [16], the gLite Prototype was designed to accommodate an iterative sequence of user interactions in an analysis context. After a review of existing projects, AliEn[6] was chosen on the basis of showing the most complete distributed analysis functionality. A re-factoring of AliEn and other services into ARDA (A Realisation of Distributed Analysis for LHC)[17] led to the creation of the gLite prototype.

The gLite Middleware prototype consists of the following core services[18]:

- File catalogue
- Authentication module
- Task queue
- Meta-data catalogue
- Package manager
- Grid access service

To access gLite users must first have a valid X.509 Grid certificate registered in a supported Virtual Organisation (VO) such as LHCb. This allows the user to become part of a well defined group sharing resources on the Grid.

An interactive shell is provided for users in order to access Grid services. As outlined in [19], this shell is implemented as a client within which the user can issue commands similar to those in a standard Unix shell. The file catalogue is organised in a hierarchical way which is similar to a file system. This has advantages because familiar commands such as *ls* and *rm* may be used in a transparent way for the user. This masks, for example, the relationship between Logical File Names (LFNs) and Physical File Names (PFNs). Files may be added to the catalogue by either specifying a URL or by adding a reference to an already existing file in an accessible storage element.

Around seventy commands are available in the gLite shell. In principle, these provide all the functionality necessary to successfully submit user jobs and retrieve output. However, for the user, this is still very far from what a standard Unix shell provides and the system can feel rather restrictive.

Jobs may be submitted from the gLite shell using a job description script (JDL file) which specifies an executable. For successful submission both the JDL file and the executable must be accessible via the filesystem on the Grid. In practice, the user must manually insert the JDL file and executable file for each specific job into the file catalogue themselves.

In order to retrieve output from gLite to the local file system users must execute a command which brings a copy of the closest PFN to a temporary directory. From this the user can copy the file to their local directory.

The current release of gLite, Release 1, has lost some of the functionality of the prototype. The gLite management taskforce decided upon this course of action in order to focus on key services. As a result, for example, the package manager and Grid access service developed in the AliEn framework have now been removed from the current release.

4 $B_S \to J/\Psi\Phi$ Channel

The LHCb experiment will look into evidence of asymmetries in the decay of B and $\bar{B}$ B mesons, in order to understand the mechanism of CP violation thought to be responsible for the prevalence of matter over antimatter. The $B_S \to J/\Psi\Phi$ channel will have an annual signal yield of 100,000 events and is sensitive to new physics effects. This made it an ideal candidate for performing a typical analysis using DaVinci in the gLite Framework. The 100K events refers to the $J/\Psi \to \mu\mu$ decay (as does the following analysis), there is an additional 20K of $J/\Psi \to ee$ which was not considered.

The final state of the $B_S \to J/\Psi\Phi$ decay, consisting of two vector mesons, implies that there are three contributions to the decay [7]. The angular analysis is greatly simplified by considering the transversity basis [20] where it is possible to disentangle the two CP even and one CP odd contributions through the transversity angle.

The angular distribution allows the extraction of the CKM property $\delta\gamma$ which could signal new physics if a large enough value is detected.

5 Analysis Using gLite

5.1 Overview

The $B_S \rightarrow J/\Psi\Phi$ channel was chosen to provide a typical, generic base on which to test the gLite framework. For the purposes of this analysis, DaVinci v12r3 was deployed using binary tarballs of all dependants.

From the perspective of the user, Figure 1 highlights the analysis dataflow. A typical analysis using DaVinci involves the creation of user specific options files and algorithms and will also include a whole host of standard options files for configuration purposes. These serve as input to the gLite Framework, along with a script to run DaVinci and a JDL file to control job submission. The latter two files are quite generic and could easily be standardised for other LHCb users.

To use DaVinci in the gLite Framework it is necessary to condense all of the options files. This is most easily achieved using JOE, the Job Options Editor[21] which is released with GANGA[22].

There are two options available at this point, one could use a single tarball of all relevant software or could utilize the gLite package manager described in Section 5.3. The choice of this only impacts on the script to run DaVinci and the JDL file. Ideally, it is recommended that an LHCb administrator or super user would insert several versions of DaVinci into the gLite file catalogue using the package manager. This would allow all potential LHCb users to make use of a particular version without having to insert it themselves.

Once this is decided and all files are added to the file catalogue, job submission is possible. gLite handles everting from the point at which the user submits the job. One can observe the job status using the shell-like behaviour inherent to the gLite prototype and then gather the output as desired. For a typical analysis using DaVinci, output comes in three forms, histograms, ntuples and the standard output from DaVinci.

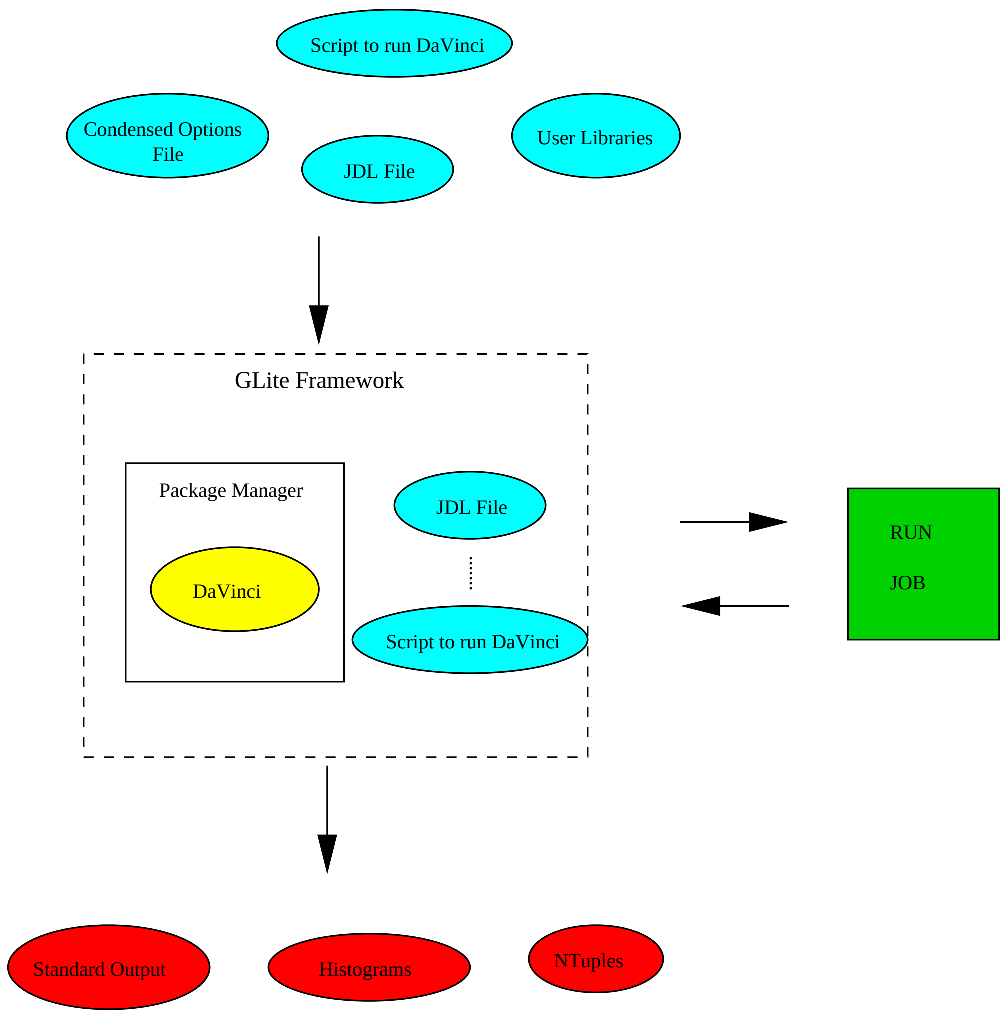

Figure 1: This diagram shows the dataflow during an analysis job using DaVinci through gLite from a user's perspective. The user provides the files at the top and after adding them to the catalogue, gLite will return the output.

5.2 Job Splitting

Job splitting is possible in the gLite Framework [23]. Inside the job description (JDL file) it is possible to specify a flag to enable splitting and this results in a master job being created after submission. From this point, any operations made on the master job will also affect the sub-jobs so, for example, killing the master job would result in the termination of all sub-jobs.

Job splitting was applied to DaVinci jobs, some of which ran successfully. In the event of sub-job failure it is necessary to resubmit either the failed sub-jobs independently or the original job again. Also, at this point, there is no mechanism in place to merge sub-jobs after completion therefore using this method of job submission was found to be overly user-intensive at this time.

5.3 Use of the gLite Package Manager

Instead of sending one large, manually constructed tarball of all the software necessary to run DaVinci jobs it is preferable to take the individual tarballs of dependent packages from the LHCb release area and insert them into the gLite file catalogue. DaVinci in this sense directly depends on Gaudi, XmlDDDB, ParamFiles, FieldMap and the LHCb packages.

Inside the gLite prototype it is possible to insert tarballs as packages with each having specific setup commands specified by the user. In this sense it was possible to create the proper environment for the software to run with each package being installed independently in different locations. The structure of the packages is implied by that of CERN and so the tarballs can be inserted directly from the LHCb release area.

By having an LHCb administrator to set up several versions of DaVinci, a typical user would be completely unaware of where the packages are installed and how they are set up. They would only need to assume that the software is set up in a similar manner to what they would be used to at lxplus and only need to supply the input outlined in Figure 1. One point to note is that there is little change to the environment between versions of DaVinci so the mechanism in place is quite scalable.

Since working on this, the gLite package manager has evolved to become far more streamlined and is now very easy to use [24].

6 Results

The results presented here are based on a selection of B_S events run on DaVinci v12r3 in the gLite Framework using a 100,000 event sample. Figures 2, 3 and 4 show the reconstructed J/Ψ, ϕ and B_S respectively. The overall selection efficiency of the B_S was 8.2%.

The transversity distribution, Figure 5 shows a very good correlation with the plot on page 12 of [8] which was obtained using a fast parameterized 'toy' Monte Carlo experiment. The resulting distribution shows what one would expect from the admixture of helicity states but some investigation into the event generator is required in order to determine that all states are being accounted for.

The gLite Middleware prototype was very much in its infancy when carrying out this analysis. Unfortunately the system could be down for a period of days or even weeks at a time due to many factors. The system was prone to hanging and often needed to be rebooted. Getting real estimates of system performance and efficiency was also hampered by issues of reliability with individual commands and job submission.

An attempt at robustness tests was made (submission of fifty 25,000 event jobs were sent every day over several days) but unfortunately the system would either execute some or all of the jobs sent or none at all due to stability issues. This was further compounded by the fact new Worker Nodes were being added frequently and these did not always behave as was initially expected.

Castor access was also a problem, originally all available datasets were picked up without any issues but towards the end of this work the system failed inexplicably and the cause of this was not determined. Another issue was with the Worker Nodes only having 20Gb disks. Unfortunately this made it impossible to run over large numbers of datasets directly. This effectively forces a user to either split their jobs themselves or via the system (see Section 5.2).

Overall, using the gLite prototype to perform user analysis was possible although it required some additional effort from the user when compared to standard batch systems. Unfortunately it often seemed to be unclear whether the user was at fault or the system itself. There is much scope for improvement however and when the system was working, user analysis was

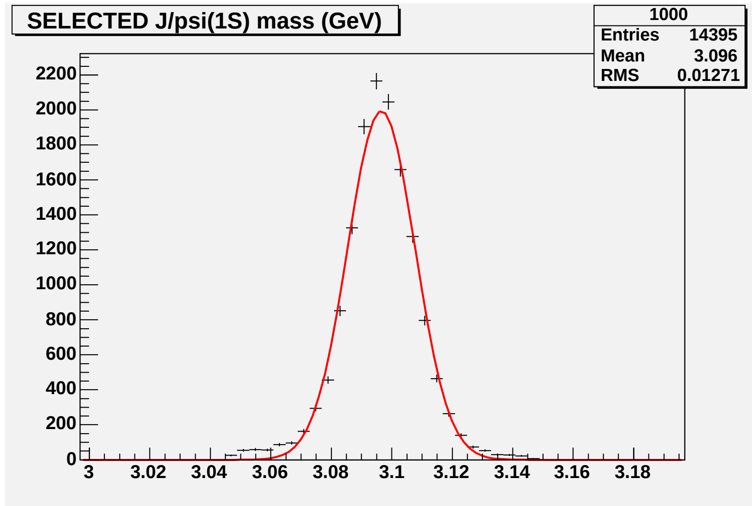

Figure 2: Reconstructed J/Ψ mass distribution after applying J/Ψ selection cuts, run over 100,000 events using DaVinci through the gLite Framework.

certainly possible.

7 Conclusions

The gLite prototype is a reduced version of the Grid middleware, this infrastructure was tested by carrying out a physics analysis using the LHCb DaVinci software on the Grid. The importance of this was two-fold, firstly, the tests were used to determine where improvements could be made to the framework. Secondly, the utilisation of Grid resources is becoming increasingly important as the start of the LHCb experiment approaches and it is necessary for new mechanisms of analysis to be explored.

Overall, analysis is possible using DaVinci in the gLite Framework. When the system works it can be relatively painless to use. However, since the system is experimental there were some reliability issues and teething problems. Nonetheless, large jobs were successfully executed using the gLite prototype and this led to the exploitation of Grid resources. Furthermore, as it stands, LHCb users could utilise the gLite prototype.

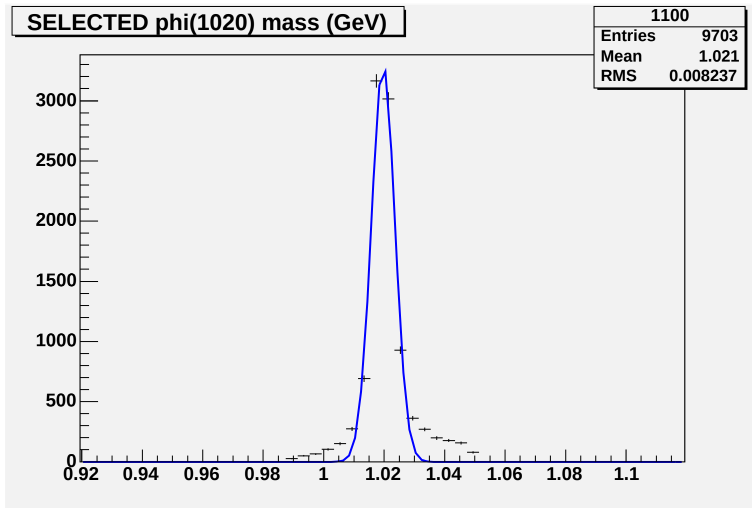

Figure 3: Reconstructed Φ mass distribution after applying J/Ψ and Φ selection cuts, run over 100,000 events using DaVinci through the gLite Framework.

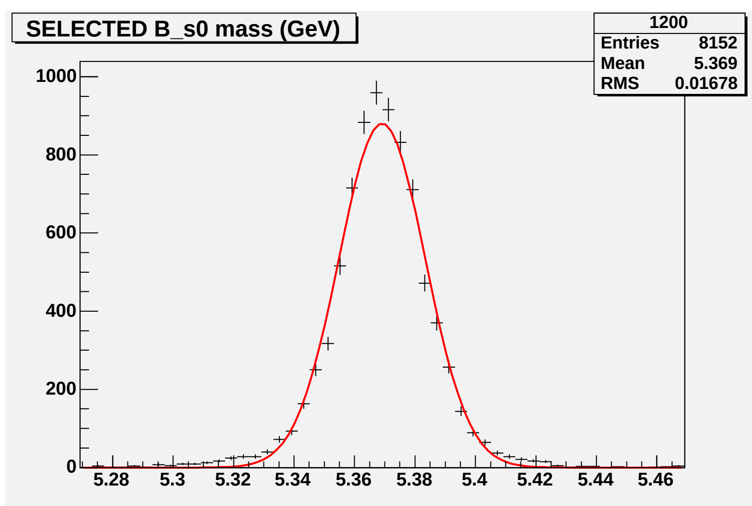

Figure 4: Reconstructed Bs mass distribution after applying all selection cuts, run over 100,000 events using DaVinci through the gLite Framework. The selection efficiency for this was 8.2%.

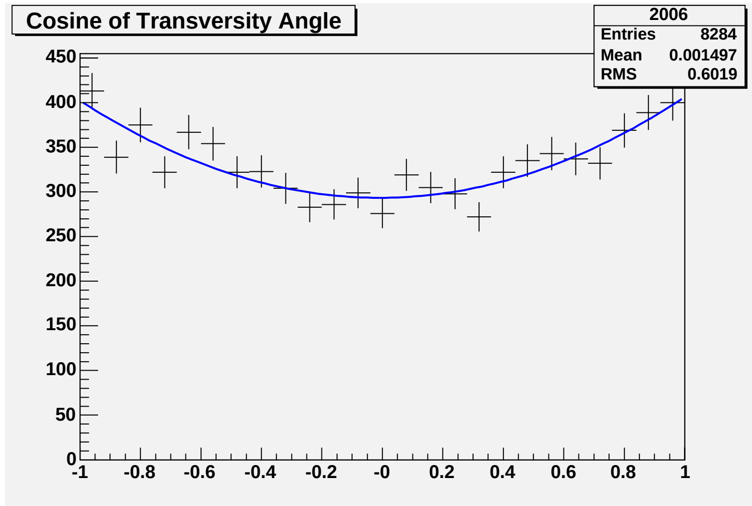

Figure 5: This is a plot of the cosine of the transversity distribution $\cos\theta_{tr}$ run over all selected Bs events. The distribution shows the projection of all contributions including CP odd and CP even and shows a good correlation with the plot on page 12 of [8], which was obtained using a fast parameterized 'toy' Monte Carlo experiment.

Acknowledgements

I would like to thank Andrew Maier, Jakub Moscicki and Pablo Saiz for their help and advice during this work.

References

[1] S.Amato et al., LHCb Technical proposal, CERN/LHCC 98-4.

[2] DaVinci - The LHCb Analysis Program, http://lhcb-comp.web.cern.ch/lhcb-comp/Analysis/default.htm.

[3] gLite Homepage, http://glite.web.cern.ch/glite/.

[4] EGEE Middleware Architecture, EU Deliverable DJRA1.1, EGEE-DRJA1.1-476451-v1.0, August 26, 2004.

[5] Design of the EGEE Middleware Grid Services, EU Deliverable DJRA1.2, EGEE-DRJA1.2-487871-v1.0, October 6, 2004.

[6] AliEn Homepage, http://alien.cern.ch/.

[7] Raven,G., LHCb Technical Note, LHCb-2003-118.

[8] Raven,G., LHCb Technical Note, LHCb-2003-119.

[9] Gaudi Framework, http://lhcb-comp.web.cern.ch/lhcb-comp/Frameworks/Gaudi/.

[10] LHCb Core Packages, http://lhcb-comp.web.cern.ch/lhcb-comp/Frameworks/LHCbSys/default.htm.

[11] Arnault,C., CMT Homepage, http://www.cmtsite.org/.

[12] Magnetic Field Map, http://lhcb-comp.web.cern.ch/lhcb-comp/Frameworks/DetDesc/default.htm.

[13] Parameter Files, http://lhcb-comp.web.cern.ch/lhcb-comp/.

[14] Xml Detector Description Database, http://lhcb-comp.web.cern.ch/lhcb-comp/Frameworks/DetDesc/default.htm.

[15] gLite Prototype Testbed, http://egee-jra1.web.cern.ch/egee%2Djra1/Prototype/testbed.htm.

[16] Architectural Roadmap Towards Distributed Analysis, Technical Report, CERN-LCG-2003-033, October 31, 2003.

[17] The ARDA Project, http://lcg.web.cern.ch/LCG/peb/arda/Default.htm.

[18] J.Andreeva et al., The ARDA Prototypes, CHEP04, Interlaken, Switzerland, 2004.

[19] B.Koblitz et al., First Experience with the EGEE Middleware, CHEP04, Interlaken, Switzerland, 2004.

[20] Dighe,A.S., Proceedings of UK Phenomenology Workshop on Heavy Flavours and CP Violation, Journal of Physics G: Nuclear and Particle Physics 27, (2001), p1341-1344.

[21] Gaudi/Athena Job Options Editor (JOE), http://ganga.web.cern.ch/ganga/user/v2/JOE/UserManual.html.

[22] Gaudi/Athena Grid Alliance,http://ganga.web.cern.ch/ganga/.

[23] gLite Prototype Documentation, http://egee-jra1.web.cern.ch/egee-jra1/Prototype/Documentation/glite_tutorial.html.

[24] Saiz,P., AliEn Documentation, http://alien.cern.ch/Alien/main?task=doc&section=PackMan.