Software reliability: Experiences in European scientific research projects and new trends

D.C. Duma, E. Ronchieri, P. Orviz Fernandez, M. David, J. Gomes, D. Salomoni

***Abstract*—In this work we describe the trends in scientific software reliability by analyzing past European Commission funded software development projects in the context of distributed computing field, such as Enabling Grids for E-sciencE, European Middleware Initiative and the ongoing INDIGO-DataCloud.**

I. Evolution of Software Reliability

In the European scientific research arena, the analysis of some of the last decade software development projects depicts a continuous increase of the prominence and robustness of testing and validation procedures. The observed trends are aligned with the evolution of the software engineering techniques (e.g. rise of DevOps practices [1]) over the last years. This is tied to the parallel advancements in the ICT field (with virtualization and container technologies) and the advent of automation and event-response tools. In the following we introduce a set of European Commission funded projects, starting in 2001 up to now, where the authors were and are currently involved. For each project we explain how they addressed the challenge in software reliability.

The DataGrid [2] project (2001 – 2003) was devoted to develop software to provide Grid computing functionalities and related management tools for widely distributed scientific communities, such as the Large Hadron Collider (LHC) experiments, Earth observations and Bioinformatics. In this project, a large effort was dedicated in the definition and implementation of academic software engineering practices due to the unusual context in which DataGrid aims, starting from the large geographical distribution of the teams involved in the development tasks, evolving requirements coming from the different application areas and the design of a new technology: the Grid [3].

The three phases of Enabling Grids for E–sciencE (EGEE, 2004 – 2010) [4] projects brought together scientists and engineers from more than 240 institutions in 45 countries worldwide to provide a seamless Grid infrastructure for e–Science. The *gLite* middleware [5] was the ultimate official distribution of EGEE as of 2006, after two years of prototyping and re–engineering efforts. Starting in EGEE–II, there was a need to move to a more dynamic release process. At this stage no software metrics were collected, while functional and integration testing were performed manually, and only on the last project, EGEE–III, an automatic build system was adopted; the E–Infrastructure for Testing, Integration and Configuration of Software (*ETICS*) service.

The ETICS and ETICS 2 (2006 – 2010) [6] projects aimed at defining and implementing a framework able to integrate a combination of new technologies in test management, release management tools and resource virtualization to improve quality, reliability and interoperability of large–scale scientific software. Both projects supported mechanisms to measure static software metrics, number of faults, number of defects and number of bugs; execute unit testing; deploy and test software on a virtual set of resources. The *ETICS* portal was the first automated service for delivering software products in distributed environments such as the Grid.

The EMI (2010 – 2013) project [7] aimed to deliver a consolidated set of middleware products based on the four major middleware providers in Europe – *ARC*, *dCache*, *gLite* and *UNICORE*. The EMI Quality Model used ISO/IEC 9126 standard [8] to guide the definition of a set of metrics and Key Performance Indicators (KPIs). EMI was one of the first projects in using a Continuous Integration (CI) system, leveraging on *ETICS* tool for building, packaging and testing the various components.

The EGI–InSPIRE (Integrated Sustainable Pan–European Infrastructure for Researchers in Europe, 2010 – 2014) project [9] maintained a production–ready Grid computing middleware distribution called *UMD* (Unified Middleware Distribution). The role of *UMD* was to enforce the fulfilment of a set of quality criteria definitions [10] in the software being delivered. *UMD* is still being used and deployed in the European scientific e–Infrastructures as a result of its continuation under a follow–up project, the EGI–Engage (Engaging the EGI Community towards an Open Science Commons) [11]. This distribution is currently complemented by a Cloud–specific one called *CMD* (Cloud Middleware Distribution).

INDIGO–DataCloud (INtegrating Distributed data Infrastructures for Global ExplOitation) [12] is an ongoing software development project that started on April 1^{st} 2015 with the aim at providing solutions to address the existing gaps in the three cloud levels; IaaS, PaaS and SaaS, helping developers, e–Infrastructures and scientific communities to exploit cloud computing resources. The project ends on September 2017.

D. C. Duma, E. Ronchieri, and D. Salomoni are with INFN - CNAF, Bologna, Italy.

P. Orviz Fernandez, is with CSIC, Santander, Spain, (e-mail: orviz@ifca.unican.es)

M. David, and J. Gomes are with LIP, Lisbon, Portugal

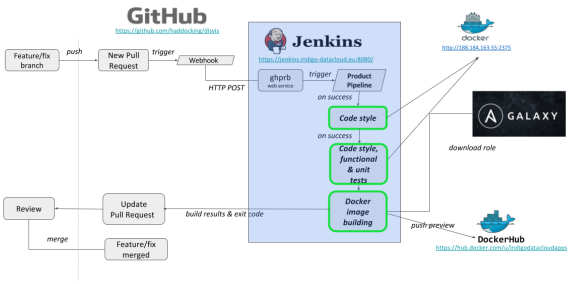

Fig. 1. Continuous Delivery workflow for Docker container images.

II. New Trends in Software Reliability

The INDIGO–Datacloud project is a reflection of the progress made in software quality and reliability aspects throughout the past scientific European experiences. Nevertheless, the projects Quality Assurance (QA) procedures are also highly influenced by the insights of current big worldwide collaborations of software development. These collaborations have continuously inspired new technologies and procedures to increase the robustness and quality of the software being delivered. In this scenario, the DevOps culture is one of the major outcomes that has been progressively adopted in the INDIGO–DataCloud project.

A. DevOps practices

The DevOps methods emphasize on exercising QA techniques to avoid infrastructure disruption whenever new developments are deployed into production. The INDIGO–DataCloud project promoted from its early stages the application of a CI scenario to enforce the QA requirements during the development phase. As the project matured, it evolved towards a Continuous Delivery (CD) implementation, automating the packaging of the software. Figure 1 showcases the workflow followed by the products distributed as container images.

B. Software Quality Procedures

The software quality procedures [13] cover: 1) the identification and description of the *quality requirements* that the software need to comply with, and 2) the *quality metrics*.

The QA requirements define that the source code shall be publicly available to augment its visibility while promoting external contributions. The developed software is compliant with community de–facto style standards, automatically checked in the CI infrastructure together with the analysis of the unit and functional testing coverage. As the last step, a human-based code review is performed to detect potential incoherences that may appear during the automated process.

The evaluation of the software quality is performed by measuring the values of the metrics and KPIs defined based upon the ISO/IEC 9126. These metrics cover the development, release and maintenance phases of the software lifecycle.

C. Integration, preview and early adoption

Two pilot infrastructures are at the disposal of developers and scientific communities involved in the project. The aim of these testbeds is to test the level of integration between the components involved in the INDIGO–DataCloud solution and use cases validation, by deploying and executing the applications with the last stable version of the software.

The released software is also tested in production environments through the stage–rollout process. Selected resource providers are requested to install the most updated stable versions of the software, accessed by their users. The stage–rollout process is key to detect and mitigate issues that could only appear in production.

III. Conclusion

It is a fact that the European software produced for scientific purposes is evolving to a more sustainable model where the quality and reliability of software is being prioritized. Recent software engineering insights, such as DevOps, are gradually being applied as the open–source collaborative tools evolve, allowing tighter integrations. The focus of the Software QA procedures has been integrated in the development stage, trying to detect and correct issues early in the software lifecycle. Automation and metric analysis are at the base of prompt issue solving. However, post–release validation, such as integration testbeds and the stage–rollout process, are also needed to strengthen the software reliability for scientific usage.

Acknowledgment

The authors would like to thanks European Commission with the various funded projects.

References

[1] M. Walls, *Building a DevOps Culture*, in O'Reilly Media, 1 edition (April 15, 2013)
[2] *DataGrid project*, CORDIS Europe, http://cordis.europa.eu/project/rcn/53665_en.html
[3] [3] L. Momtahan et al, *e-Science Experiences: Software Engineering Practice and the EU DataGrid*, in Proc. Asia-Pacific Software Engineering Conference, 2002, pp. 269-275, IEEE Press
[4] *EGEE* project, CORDIS Europe, http://cordis.europa.eu/project/rcn/80149_en.html, EGEE-II – http://cordis.europa.eu/project/rcn/99189_en.html, EGEE-III – http://cordis.europa.eu/project/rcn/87264_en.html
[5] E. Laure et al, *Programming the Grid with gLite*, in Computational Methods in Science and Technology, vol. 12(1), pp. 3345, 2006, Scientific Publishers OWN
[6] *ETICS* project, CORDIS Europe, ETICS – http://cordis.europa.eu/project/rcn/80138_en.html, ETICS-2 – http://cordis.europa.eu/project/rcn/86604_en.html
[7] *EMI* project, CORDIS Europe, http://cordis.europa.eu/project/rcn/95311_en.html
[8] *ISO/IEC 9126 Software Engineering - Product Quality*, International Organization for Standarization, https://www.iso.org/standard/22749.html
[9] *EGI-InSPIRE* project, CORDIS Europe, http://cordis.europa.eu/project/rcn/95923_en.html
[10] *EGI Quality Criteria*, http://egi-qc.github.io/
[11] *EGI-ENGAGE* project, CORDIS Europe, http://cordis.europa.eu/project/rcn/194937_en.html
[12] *INDIGO-DataCloud* project, CORDIS Europe, http://cordis.europa.eu/project/rcn/194882_en.html
[13] *Initial Plan for WP3* INDIGO-DataCloud Deliverable 3.1, https://www.indigo-datacloud.eu/documents/initial-plan-wp3-d31