Project Acronym	***iMarine***
Project Title	***Data e-Infrastructure Initiative for Fisheries Management and Conservation of Marine Living Resources***
Project Number	***283644***
Deliverable Title	***Software Release Activity Report***
Deliverable No.	D7.4
Delivery Date	***September 2014***
Author	***Paolo Fabriani (E-IIS)*** ***Gabriele Giammatteo (E-IIS)*** ***Massimiliano Assante (CNR)*** ***Nikolas Laskaris (NKUA)***

Abstract: *This document reports on activities related to the gCube system integration, testing, distribution and documentation performed during the third reporting period of iMarine project.*

iMarine (RI – 283644) is a Research Infrastructures Combination of Collaborative Project and Coordination and Support Action (CP-CSA) co-funded by the European Commission under the Capacities Programme, Framework Programme Seven (FP7).

DOCUMENT INFORMATION

PROJECT	
Project Acronym	iMarine
Project Title	Data e-Infrastructure Initiative for Fisheries Management and Conservation of Marine Living Resources
Project Start	1st November 2011
Project Duration	30 months
Funding	FP7-INFRASTRUCTURES-2011-2
Grant Agreement No.	283644
DOCUMENT	
Deliverable No.	D7.4
Deliverable Title	Software Release Activity Report
Contractual Delivery Date	30 September 2014
Actual Delivery Date	11 November 2014
Author(s)	Paolo Fabriani, E-IIS Gabriele Giammatteo, E-IIS Massimiliano Assante, CNR Nikolas Laskaris, NKUA
Editor(s)	Gabriele Giammatteo, E-IIS
Reviewer(s)	Fabio Simeoni, FAO

Contributor(s)	Paolo Fabriani, E-IIS Gabriele Giammatteo, E-IIS Massimiliano Assante, CNR Nikolas Laskaris, NKUA
Work Package No.	WP 7
Work Package Title	Enabling-technology Integration and Distribution
Work Package Leader	E-IIS
Work Package Participants	E-IIS, CNR, NKUA
Estimated Person Months	1
Distribution	Public
Nature	Report
Version / Revision	1.0
Draft / Final	Final
Total No. Pages (including cover)	30
Keywords	software, integration, testing, distribution, documentation, release

CHANGE LOG

Reason for Change	Issue	Version	Date	Partner / Responsible
Initial content		0.1	27/10/2014	E-IIS
CNR contribution integrated		0.2	29/10/2014	E-IIS
NKUA contribution integrated		0.3	29/10/2014	E-IIS
Internal Review		0.3_Reviewd	11/11/2014	FAO
Comments accepted		1.0	11/11/2014	E-IIS

DISCLAIMER

iMarine (RI – 283644) is a Research Infrastructures Combination of Collaborative Project and Coordination and Support Action (CP-CSA) co-funded by the European Commission under the Capacities Programme, Framework Programme Seven (FP7).

The goal of iMarine*, Data e-Infrastructure Initiative for Fisheries Management and Conservation of Marine Living Resources*, is to establish and operate a data infrastructure supporting the principles of the Ecosystem Approach to Fisheries Management and Conservation of Marine Living Resources and to facilitate the emergence of a unified Ecosystem Approach Community of Practice (EA-CoP).

This document contains information on iMarine core activities, findings and outcomes and it may also contain contributions from distinguished experts who contribute as iMarine Board members. Any reference to content in this document should clearly indicate the authors, source, organisation and publication date.

The document has been produced with the funding of the European Commission. The content of this publication is the sole responsibility of the iMarine Consortium and its experts, and it cannot be considered to reflect the views of the European Commission. The authors of this document have taken any available measure in order for its content to be accurate, consistent and lawful. However, neither the project consortium as a whole nor the individual partners that implicitly or explicitly participated the creation and publication of this document hold any sort of responsibility that might occur as a result of using its content.

The European Union (EU) was established in accordance with the Treaty on the European Union (Maastricht). There are currently 27 member states of the European Union. It is based on the European Communities and the member states' cooperation in the fields of Common Foreign and Security Policy and Justice and Home Affairs. The five main institutions of the European Union are the European Parliament, the Council of Ministers, the European Commission, the Court of Justice, and the Court of Auditors (http://europa.eu.int/).

GLOSSARY

ABBREVIATION	DEFINITION
iMarine	Data e-Infrastructure Initiative for Fisheries Management and Conservation of Marine Living Resources
IDE	Integrated Development Envrionment
gCF	gCore Framework
ETICS	E-infrastructure for Test, Integration and Configuration of Software
E-IIS	Engineering Ingegneria Informatica S.p.A.
FAO	Food and Agriculture Organization of the United Nations
NKUA	National and Kapodistrian University of Athens
CNR	Consiglio Nazionale delle Ricerche
D4Science-II	Data Infrastructure for Science-II
GWT	Google Web Toolkit
BTRT	Build and Test Report Tool
SVN	Subversion

TABLE OF CONTENTS

INDEX OF TABLES

TABLE OF FIGURES

EXECUTIVE SUMMARY

This document describes activities performed and results achieved in Work Package 7 during the third (and last) reporting period of the iMarine project.

Work Package 7 activities are aimed at integrating software produced in the project, releasing good quality, well-tested, and documented software. It is possible to categorize them in two groups: a) one-off activities undertaken with the aim of making changes to release procedures and/or tools used in the Work Package and b) routine activities performed in the context of release procedures.

Concerning the first group, since this report covers only the last period of the project, the **procedures** were, at that point, well-established and tested. Therefore, little effort has been spent in this activity. There are only two significant changes to report: i) the introduction of a procedure to formally execute functional tests of the iMarine portal components, and ii) a variation to the documentation validation procedure. Also work on the infrastructure and the tools was limited, except for some updates to the ETICS tool used for integration of software.

The size of the **build and testing infrastructure** (three nodes for the build infrastructure and 12 nodes for the testing infrastructure) remained unchanged since the previous reporting period. However, the infrastructure was maintained and regularly updated with new software releases and project needs. The two infrastructure have been used for regular building and testing of the gCube software.

This document also contains a general overview of all **gCube releases** delivered in the entire iMarine project. Twenty-four releases (from gCube 2.7.3 to gCube 3.5.0) have been integrated and delivered to Work Package 5 (in charge of deployment in production) during the project. On average, except for the summer period, one gCube release per month has been rolled out. However, there is a high variance concerning the duration of the integration activities. In particular, there have been three main technological changes with high impact on gCube components that led to releases that required an extra-effort for being integrated: i) the switch from Ant to Maven as building tool (gCube 2.9.0 and gCube 2.10.0), ii) the introduction of FeatherWeight Stack (2.17.0) and iii) the switch from Java 6 to Java 7 (gCube 3.0.0).

With regard to the **documentation**, the three gCube guides (Administrator, Developer and User) have been reviewed and updated to be aligned with new/changed functionalities of the system. In particular the indexes of the guides have been completely rethought to make it easier for readers to find information they need.

The document is structured as follow:

- 1. **Release procedures Update**: describes the updates to the Work Package procedures during the reporting period;
- 2**. Build infrastructure**: describe the status of and the updates to the build infrastructure performed during in the reporting period;
- 3. **Testing infrastructure**: describe the status of and the updates to the build infrastructure performed during in the reporting period;
- 4. **Release history**: contains overview and statistics for all project releases, as well as a description of each release in the reporting period;

- 5**. Documentation validation**: describe the work done to the documentation during the reporting period;

1 RELEASE PROCEDURES UPDATE

This section describes the changes to procedures adopted in the Work Package during the reporting period. Since this report covers only the last period of the project, there are no significant changes to the main procedures that were, at that point, stable and well-tested. There is only a significant change to report: the introduction of a new procedure to execute functional tests against portlets. Indeed, portlet testing was not covered by the previous testing procedures and the **Portlets Functional Testing procedure** fills the gap. A minor change (described in section 5) has been done also to the **Documentation Validation procedure**.

1.1 PORTLETS FUNCTIONAL TESTING PROCEDURE

Starting with the gCube 3.2.0. Release (June 2014), the Portlets composing the iMarine Gateway have been required to pass a functional testing procedure in order to be made available for end users. The actors involved in the procedure are: the portal manager, the infrastructure manager, and the portlet application domain experts. Portlet application domain experts are expert users of a specific application. The domain expert knows very well all the possible interactions between the end user and the user interface. In some specific cases, they could be application developers or application owners.

The procedure starts by analysing the software artefacts included in the release along with their software dependencies. This task is carried out manually by the portal manager. Each portlet artefact which is released must be part of the functional testing procedure. Also, if anything in the dependency chain of a Portlet artefact X is released, than X has to be functionally tested too.

The Portlets Functional Testing procedure consists of 4 stages:

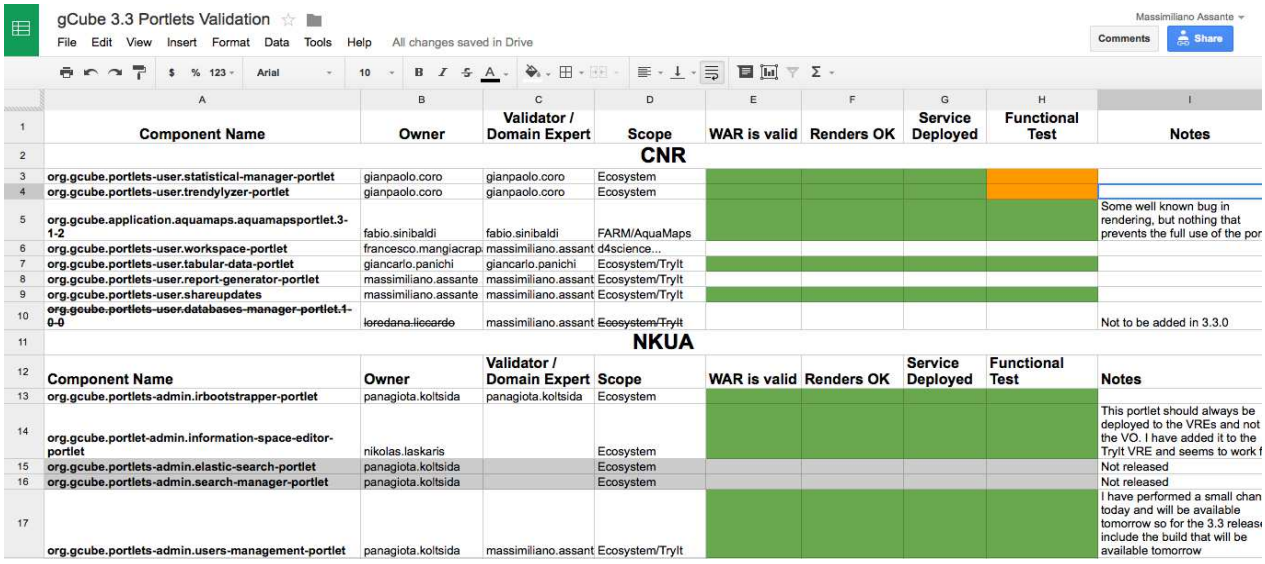
1. *Web Archive check*: every Portlet web archive released is verified to contain all the necessary files needed for its correct deployment. If the check is not passed the Portlet developer is notified through a new ticket of type integration issue;
2. *Back-end Service check:* The portal manage and the infrastructure manager make sure that the back-end services of the Portlet application are ready and fully functional in the testing / pre-production infrastructure.
3. *Rendering check*: every Portlet web archive passing stage no. 1 is then actually deployed into the pre-production portal to check if it renders correctly. In this stage there are few things that could make the Portlet not render correctly. For instance, the GWT compilation could fail during Web Archive creation, or the developer could have used some CSS classes that clash with those of the iMarine Gateway. If the check is not passed the Portlet developer is notified through a new ticket of type integration issue;
4. *Functional Test*: when stage 1 and 2 are completed the related Portlet application domain expert is assigned to the functional test by opening a new ticket of type 'functional test'; This functional test ticket contains also a link to the **functional test master table**: an *online* table created by the portal manager (see Figure 1 for column names) and shared with each actor involved in the functional testing procedure. Each row contains a Portlet artefact having to pass the functional testing procedure. With respect to the table header reported in Figure 1 the columns indicate the following:
    - **Component Name**: Portlet unique identifier;
    - **Owner**: generally the developer of the Portlet or the responsible of the application;
    - **Domain Expert**: an expert user of the application;

- **Scope**: the infrastructure scope where the Portlet has to be functionally tested;
- **Web Archive validity**: indicates if the software package produced by the build system can be correctly deployed in the iMarine Gateway. The portal manager compiles this part;
- **Renders OK**: indicates whether the Portlet displays correctly in the iMarine Gateway. The portal manager compiles this part;
- **Service Deployed**: indicates whether the infrastructure services composing the back-end of the Portlet application are ready and fully functional. The infrastructure manager compiles this part;
- **Functional Test:** compiled by the Application Domain Expert user, indicates if the functional test was performed;
- **Notes:** if the functional test cannot be performed the Application Domain Expert can explain the reasons in the notes;
- **Ticket:** the TRAC ticket associated to the functional test;

Component Name	Owner	Domain Expert	Scope	Web Archive check	Rendering check	Service Deployed	Functional Test	Notes	Ticket

Figure 1: The functional test master table header

The aim of the **functional test master table** is for the portal manager to have at a glance the status of the functional tests running for each portlet. This is also because each stage is assigned a different meaningful colour to indicate whether it passed or not. A screenshot of the functional test master table for the gCube release 3.3.0 is shown in *Figure 2*.



gCube 3.3 Portlets Validation

File Edit View Insert Format Data Tools Help All changes saved in Drive

Massimiliano Assante

Comments Share

	A	B	C	D	E	F	G	H	I
1	Component Name	Owner	Validator / Domain Expert	Scope	WAR is valid	Renders OK	Service Deployed	Functional Test	Notes
2				CNR					
3	org.gcube.portlets-user.statistical-manager-portlet	gianpaolo.coro	gianpaolo.coro	Ecosystem					
4	org.gcube.portlets-user.trendylyzer-portlet	gianpaolo.coro	gianpaolo.coro	Ecosystem					
5	org.gcube.application.aquamaps.aquamapsportlet.3-1-2	fabio.sinibaldi	fabio.sinibaldi	FARM/AquaMaps					Some well known bug in rendering, but nothing that prevents the full use of the por
6	org.gcube.portlets-user.workspace-portlet	francesco.mangiacrap	massimiliano.assant	d4science...					
7	org.gcube.portlets-user.tabular-data-portlet	giancarlo.panichi	giancarlo.panichi	Ecosystem/TryIt					
8	org.gcube.portlets-user.report-generator-portlet	massimiliano.assante	massimiliano.assant	Ecosystem/TryIt					
9	org.gcube.portlets-user.shareupdates	massimiliano.assante	massimiliano.assant	Ecosystem/TryIt					
10	~~org.gcube.portlets-user.databases-manager-portlet.1-0-0~~	~~loredana.liccardo~~	massimiliano.assant	~~Ecosystem/TryIt~~					Not to be added in 3.3.0
11				NKUA					
12	Component Name	Owner	Validator / Domain Expert	Scope	WAR is valid	Renders OK	Service Deployed	Functional Test	Notes
13	org.gcube.portlets-admin.irbootstrapper-portlet	panagiota.koltsida	panagiota.koltsida	Ecosystem					
14	org.gcube.portlet-admin.information-space-editor-portlet	nikolas.laskaris		Ecosystem					This portlet should always be deployed to the VREs and not the VO. I have added it to the TryIt VRE and seems to work f
15	org.gcube.portlets-admin.elastic-search-portlet	panagiota.koltsida		Ecosystem					Not released
16	org.gcube.portlets-admin.search-manager-portlet	panagiota.koltsida		Ecosystem					Not released
17	org.gcube.portlets-admin.users-management-portlet	panagiota.koltsida	massimiliano.assant	Ecosystem/TryIt					I have performed a small chan today and will be available tomorrow so for the 3.3 releas include the build that will be available tomorrow

Figure 2: The functional test master table during the validation phase

2 BUILD INFRASTRUCTURE

This section focuses on tools and infrastructure facilities that support build activities. The infrastructure remained essentially unchanged during the reporting period, only regular maintenance and service operation activities have been carried out. In this section we briefly present the status of the build infrastructure (section 2.1), the main maintenance activities (section 2.3) and the most relevant improvements and updates received by the building tools, mainly ETICS (section 2.2).

2.1 INFRASTRUCTURE OVERVIEW

The build infrastructure in use within the project is spread across three sites:

- The **ETICS system** [1], which maintains metadata for all gCube components and provides a number of build nodes where developers can submit private builds before releasing changes to the daily build process and to other developers;
- Two external **build servers** are configured to build the development version and the current release candidate versions of entire gCube system. One server is hosted at E-IIS premises, the other one at CNR premises;

To ensure coherence and consistency of the build process, both external build servers rely on the project's meta-model maintained and exposed by the ETICS system. Actually an ETICS agent runs on them and builds software as ETICS would do on its own infrastructure. The need for such external build services is motivated by a number of further quality assurance processes and distribution peculiarities that cannot be fulfilled easily on the ETICS infrastructure.

2.2 ETICS

Since the beginning of the project, gCube integration activities have relied on the ETICS system. During the previous reporting period, iMarine successfully switched ETICS instance used to keep the configuration and run builds of gCube software (from the instance hosted at CERN to the instance hosted at E-IIS premises).

The ETICS system is actively developed and evolved accordingly with feedback received from user communities (including the iMarine community). During the reporting period, ETICS received some improvements that have been deployed in production in order to be used by iMarine community. Next sections briefly describe the most relevant updates.

2.2.1 GUI IMPROVEMENTS

On the GUI side, a number of improvements have been done in order to consolidate the access to the tool old and new functionalities with the aim of improving the user friendliness and user experience. The new portal is available at https://etics.res.eng.it/tools/etics-gui/ and has the following layout:

- A **left panel** that contains a summary of the status of ETICS: open issues (see section 2.2.2), status of the pool, latest build submission, and latest changes;
- The **main panel** that can show (by selecting the tab on the top of the page):
    - The **Submission tab** (Figure 3): shows the list of builds submitted to the system. For running builds live logs are available, for completed builds logs and reports are available;
    - The **Issues tab** (Figure 4): shows the issues found by the system in the built components. See section 2.2.2 for further details;

- The **Configuration tab** (Figure 5): this tab embed a portion of the old portal for the editing of the project configuration;

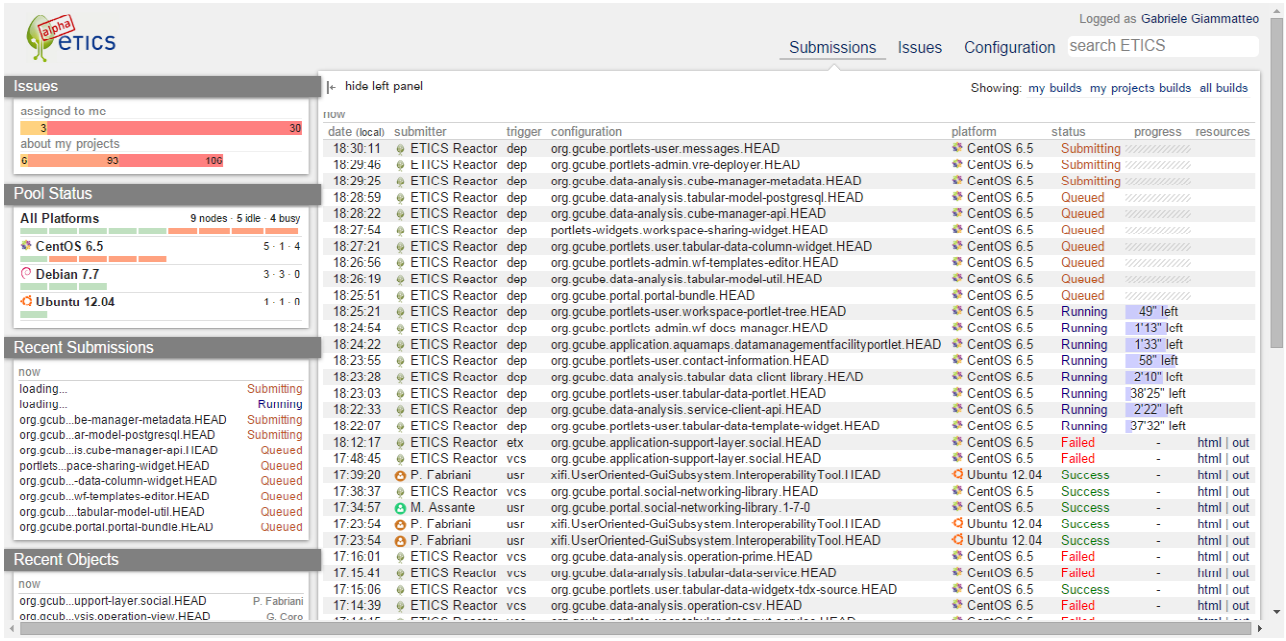


Figure 3: ETICS Submission tab

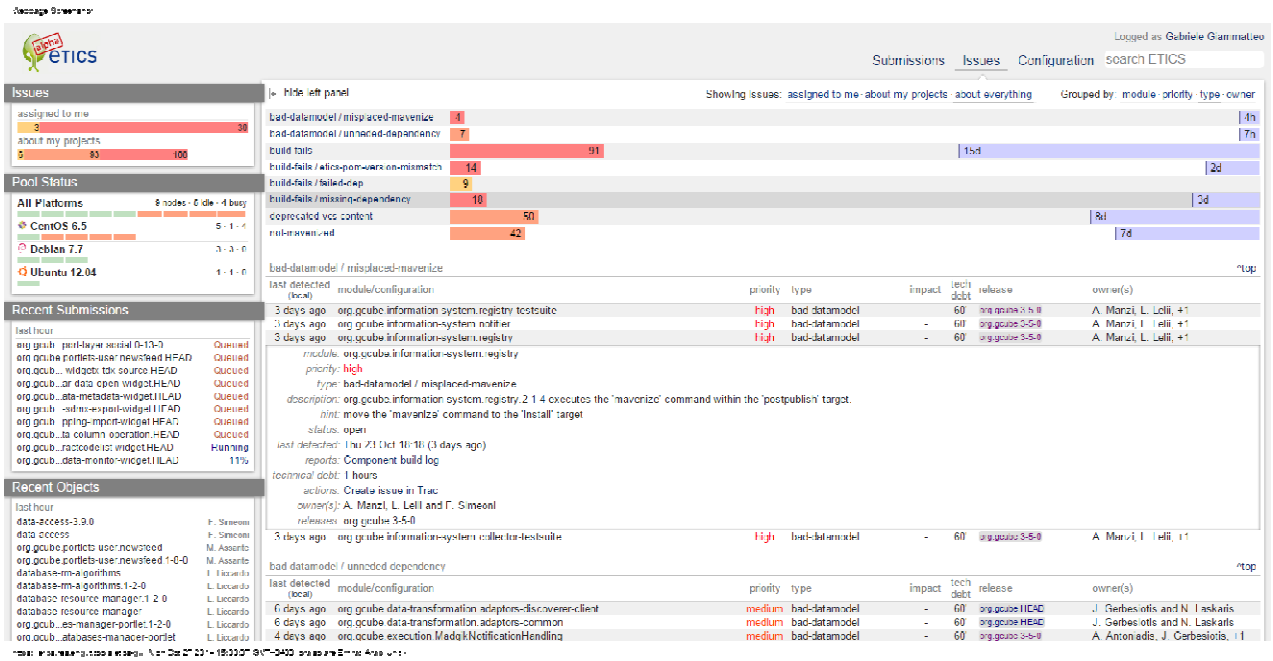


Figure 4: ETICS Issues tab

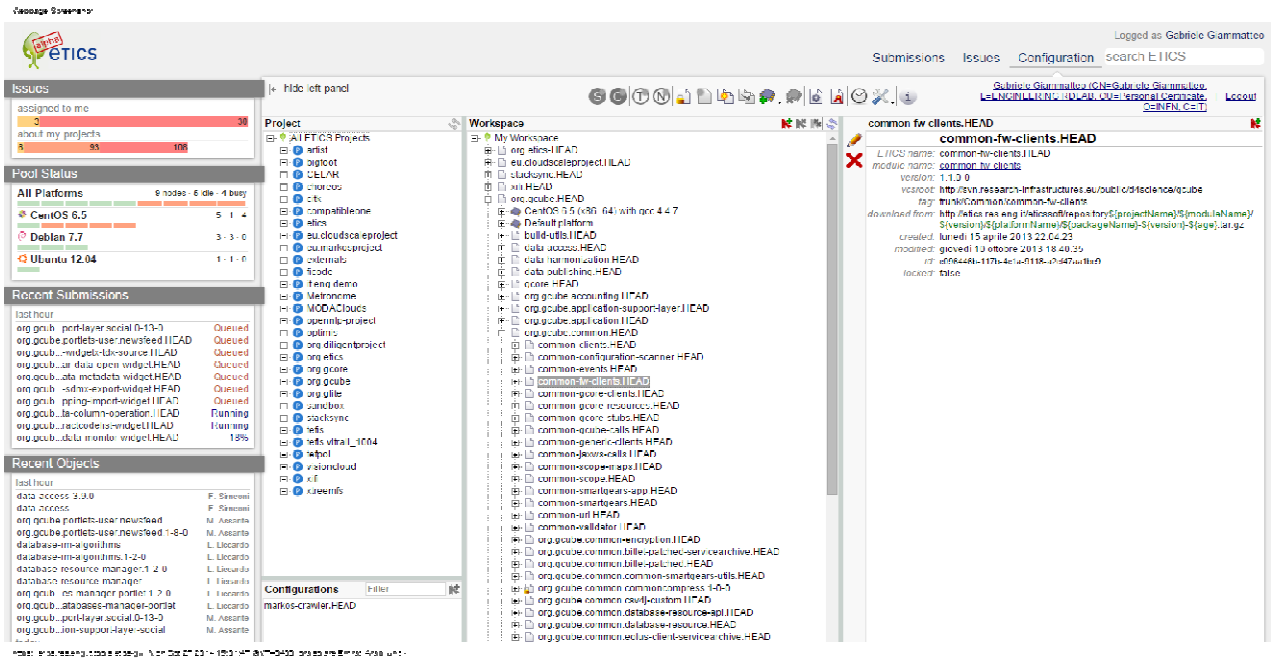


Figure 5: ETICS Configuration tab

2.2.2 INTEGRATION ISSUES

The experience gained from the analysis of builds during the integration activity showed that the types of issues occurring is quite limited and that most of them are easy to guess from build reports. This led to the implementation of a module to automatically process the build reports, discover the issues, and publish them. A GUI has been implemented to show this information, and it has been integrated in the new ETICS portal (see section 2.2.1).

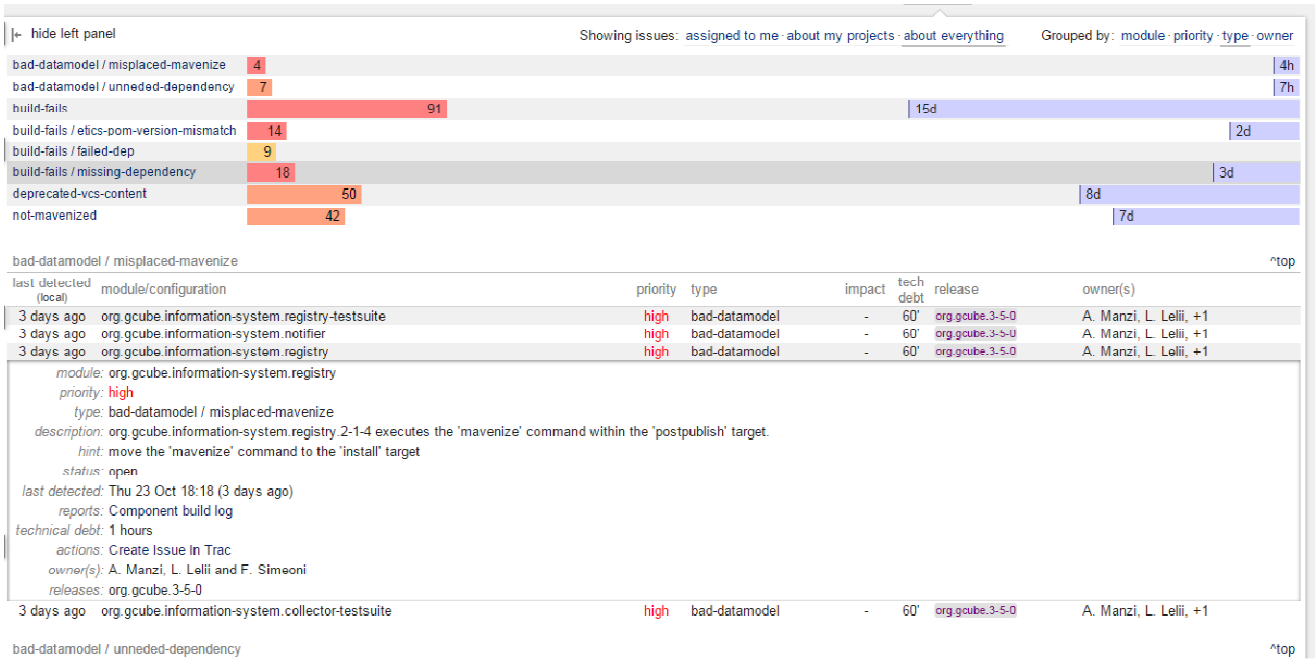


Figure 6: Issue tab detail

Figure 6 shows the Issue tab. In the top right there are a set of filters to decide which issues should be shown (i.e., only the ones assigned to me, all the issues related to a project I'm working in, all issues) and grouping options (i.e., by component, by type, by owner, by priority). In the middle section for each group

(that will depend from the grouping option selected) a summary of the number of issues for that group and an estimation of the overall technical debt is reported. Finally at the end of the page, there is the list of all issues. By clicking on an issue, the details of that issue are shown, including also hints and useful links for the solution.

With this new feature, every developer can see the issues related to his/her components grouped in a single page, along with detailed information to fix. This an improvement over the past, when the developer had to manually inspect the numerous build reports in order to find the same information.

The analysis of issues is done at the end of each build: new issues are opened and existing issues are closed if they are solved. Additionally, when an issue is open or closed an e-mail notification is sent to the owner. At the moment only a limited set of issues are generated, the complete list is in Table 1.

Type:Subtype	Description
build-fails	The build of the component fails, for unknown reasons
build-fails:failed-dep	The build of the component fails because one of its dependencies is failing
build-fails:etics-pom-version-mismatch	The version expressed in the pom.xml is different from the verion in ETICS
Build-fails:missing-dependency	The build of the component fails because of a missing dependency in the pom.xml
technology-upgrade:gwt-upgrade-needed	Version of GWT used is older than the version defined at gCube project level
technology-upgrade:xstream-upgrade-needed	Version of XStream used is older than the version defined at gCube project level
bad-datamodel:missing-etics-dependency	A needed dependency is not expressed in ETICS
bad-datamodel:unnedded-dependency	A dependency expressed in ETICS in not needed
bad-datamodel:misplaced-mavenized	The *mavenize* script is invoked in the wrong place. Should be invoked in *postpublish* target
bad-datamodel:service-archive-not-needed	The servicearchive component in ETICS is deprecated for Maven components and should be removed
not-mavenized	The component still uses Ant for building
release-from-trunk	The component points to the */trunk* folder on SVN

deprecated-vcs-content	The source code repository contains deprecated files like .jar archives
no-revision-number	The checkout command does not specify any revision number
dev-out-of-date	The development version of this component has a lower version than the latest released version
new-version-in-head	The development version is higher than the latest released version
not-in-head	No development version for this component has been found in ETICS
not-in-release	This component has not any released version

Table 1: Current automatically discovered issues

2.2.3 QUALITY ASSURANCE REPORTS

This update introduces a new report for software quality assurance. Every time a component is built in ETICS a number of quality plug-ins run and collects data from source code files. In the past, this data was not accessible directly from the build reports. Moreover, it was not easy to understand due its fine grain of detail. In order to make this data exploitable by developers as well as project managers, a new quality model has been developed.

The new quality model implemented in ETICS is based on the SQALE method [2] which is compliant with the two main standards in software quality: the ISO/IEC 9126 and the ISO 25010. This method is based on the computation of the technical debt for each component (the effort estimated for fixing all quality issues detected in the source code) and a quality rating based on the technical debt and the size of the component.

A new ETICS module has been implemented that runs during remote builds and receives in input all the quality data extracted from the other plug-ins, aggregates it and computes SQALE metrics for each component.

A new report (in Figure 7) has been created for each remote build and it is available together with the build reports. The new report shows the technical debt, the quality rating and the quality issues at various hierarchy level in the source code: component, file, class and method level.

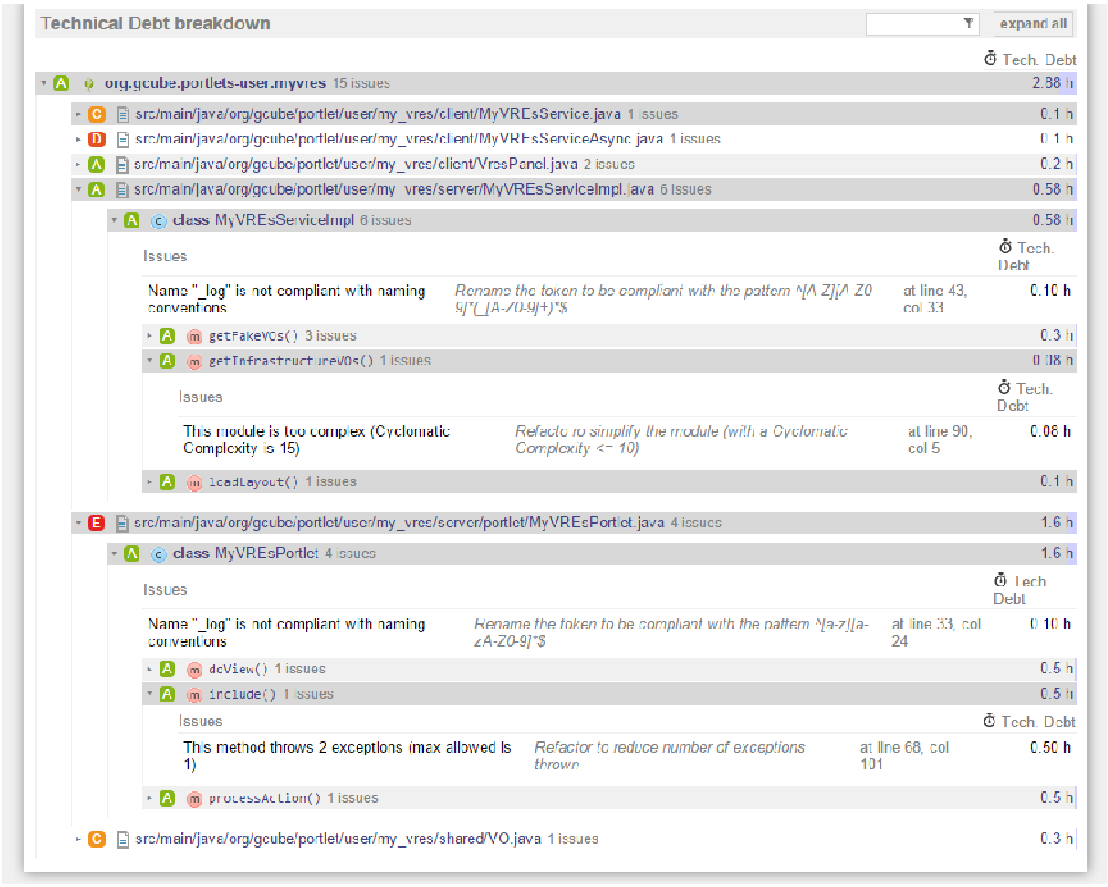


Figure 7: SQALE Report

2.3 SERVICES OPERATION AND MAINTENANCE

The iMarine build infrastructure consists of three main components: the ETICS instance and two external build servers. All of them are under the responsibility and operated by E-IIS. The main activities in this area include:

- Deployment of any update to the ETICS service and/or build servers software;
- Upgrade of software on ETICS and externals build nodes in order to offer latest version of supported operating systems (i.e. CentOS, Debian and Ubuntu);
- Monitoring the availability of hardware resources (e.g. disks, ram, network) and provisioning of additional resources on demand;
- Routine activities during the gCube candidate release integration (i.e. activation/deactivation of builds, configuration of repositories).

3 TESTING INFRASTRUCTURE

One of the activity of Work Package 7 is to install, maintain, and keep up to date a testing infrastructure for gCube services. This infrastructure is used during the integration of gCube releases to run deployment and functional tests of the components under release.

3.1 INFRASTRUCTURE STATUS AND UPDATES

In this last reporting period, the size of the testing infrastructure remained unchanged. It counts twelve nodes. The nodes are organized as follows:

- 3 nodes (*im-octopus*, *im-goldfish* and *im-seahorse*) are dedicated to Enabling Services for the '*/testing*' infrastructure.
- 3 nodes (*gcube-testing-node[1..3]*) are dedicated to Enabling Services for the '*/testing/vo1*' virtual organisation.
- 5 nodes (g*cube-testing-node[4..8]*) run gCube containers ready to to host gCube services
- 1 further node (*gcube-testing-portal*) hosts the gCube portal and it is used to test gCube portlets and to interact with deployed gCube services for end-to-end tests. to test end-user use cases.

Activities performed on the testing infrastructure were:

- Maintenance of infrastructure nodes: ordinary and extraordinary interventions to keep the virtual machines running and updated in the infrastructure;
- Upgrade of gCube Enabling services. This had a twofold objective: a) test the deployment of Enabling service and b) keep the infrastructure aligned with gCube releases;
- Schedule of automatic deployment tests and analysis of results. In case of issues, developers have been involved to solve them. All gCube services have been tested with this modality for all gCube releases;

4 RELEASE HISTORY

This section reports on gCube software releases, which represent the main outcome of work package 7., along with a detailed report for all releases (section 4.2) and the development version (section 4.3) in the reporting period. Since this is the latest report in the Work Package 7, it also possible to give some statistics concerning of all releases in the iMarine project (section 4.1).

4.1 OVERVIEW OF ALL IMARINE RELEASES

During iMarine project 24 gCube releases have been rolled out including major, minor and maintenance releases. Table 2 reports, for each release, some essential size indicators: a) when the integration of the release started, b) the number of days needed to complete the integration (end of WP7 activity), c) total number of components in ETICS and d) the number of new and/or updated components with regards to the previous release.

Release	Start	Duration (gg)	Total components	New/Updated
2.7.3	28 Jan 2012	10	526	18
2.8.0	20 Feb 2012	18	548	148
2.8.1	6 Apr 2012	14	554	51
2.9.0	04 May 2012	67	512	245
2.9.1	24 Jul 2012	4	511	22
2.10.0	14 Sept 2012	32	524	246
2.11.0	19 Oct 2012	24	547	118
2.11.1	1 Dec 2012	8	548	60
2.12.0	17 Jan 2013	41	561	233
2.13.0	05 Mar 2013	20	558	80
2.14.0	22 Apr 2013	30	551	172
2.15.0	31 May 2013	19	550	69
2.16.0	9 Jul 2013	22	461	161
2.16.1	13 Sep 2013	12	461	42
2.17.0	21 Oct 2013	40	470	251
2.17.1	14 Dec 2013	26	472	24
2.17.2	30 Jan 2013	4	472	14
3.0.0	27 Jan 2014	63	497	237
3.1.0	31 Mar 2014	52	511	183
3.1.1	22 May 2014	12	511	14
3.2.0	04 Jun 2014	16	518	184

3.3.0	30 Jun 2014	18	537	137
3.4.0	10 Sep 2014	16	546	122
3.5.0	--	--	--	--

Table 2: gCube releases during iMarine project

In addition, the charts in Figure 8 and Figure 9 shows for each release the number of new, updated and unchanged components with regards to previous release. Figure 8 shows all releases from gCube 1.1.0 in D4Science project, while Figure 9 shows only releases in the iMarine lifespan. For increased readability, and since they do not provide meaningful data for the charts, maintenance releases are omitted.

It is not meaningful to compute metrics like average or standard deviation over this data because each release has singularities (in terms of historical period and/or amount of new functionalities released) that make them really heterogeneous. It is possible, however, to highlight the most notable trends.

Looking at the biggest releases, both in number of released components and duration in time, it is possible to associate them to technological changes with an high impact on an all components in gCube:

- In gCube 2.9.0 and gCube 2.10.0, the new building system based on Maven has been introduced. In these two releases most of components had to migrate their building system and be re-released. The amount of components released along with the tuning of integration, release and distribution tools to work with Maven led to a very long integration time (67 and 32 days);
- In gCube 2.17.0 two important technological changes took place: a) migration from GWT 2.4 to GWT 2.5 and b) introduction of a new lightweight runtime environment for gCube services called the FeatherWeight Stack (FWS). While the former change required most of web components (i.e., the portlets) to be released to make them compatible with GWT 2.5, the latter required several services to be released to use the new libraries;
- In gCube 3.0.0 the transition from Java 6 to Java 7 took place. Since Oracle ended the support to Java 6 in 2014, the entire gCube codebase has been migrated to Java 7. Only few components needed to actually update their source code, but several components needed to update their Maven project descriptor (i.e., pom.xml);

An additional insight that can be derived from this data is a reduction of components in the release between gCube 2.15.0 (550 components) and gCube 2.16.0 (461 components). In fact in gCube 2.16.0 several obsolete and not used components have been removed including two subsystems (“dir” and “ontology-management”). Furthermore several services in “execution” subsystem (due to a refactoring of subsystem services) and components implementing the old D4Science-II project use cases have been dropped. This made possible to have a smaller, easier to integrate releases.

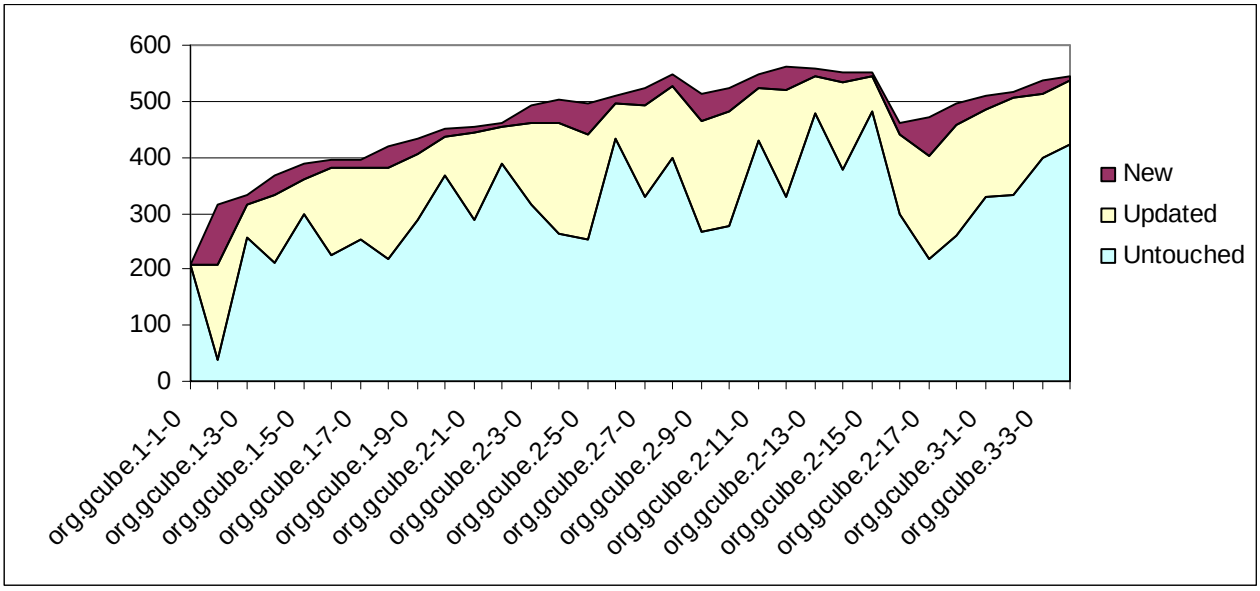


Figure 8: gCube releases since D4Science project

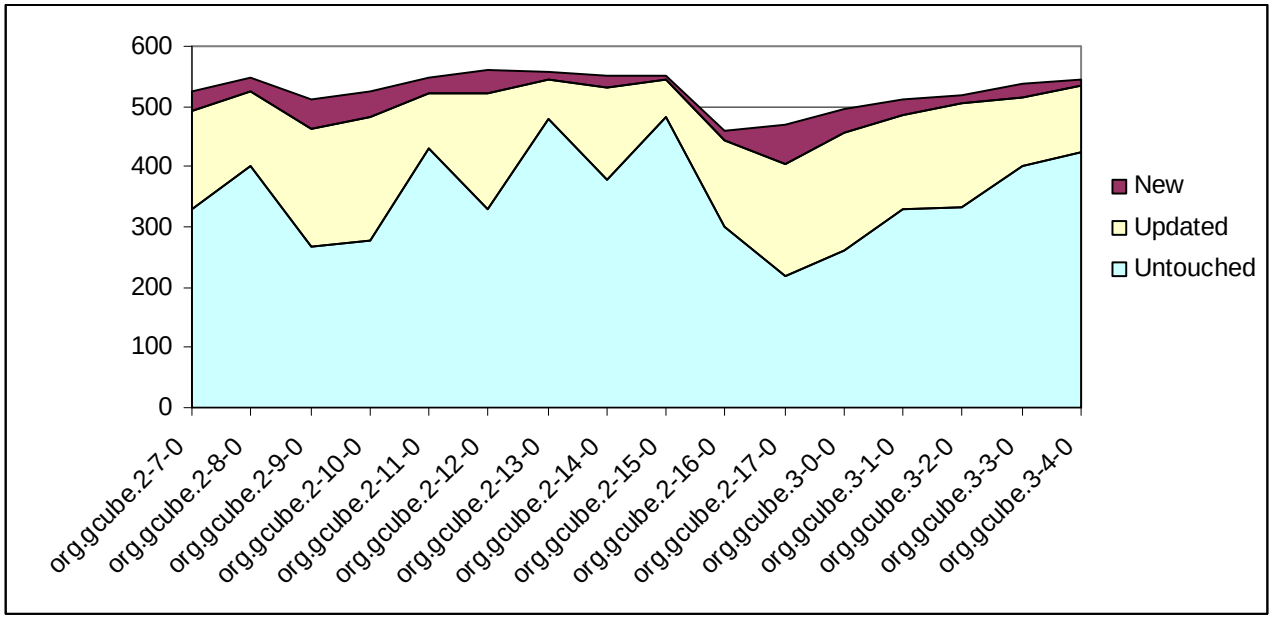


Figure 9: gCube releases in iMarine project

Figure 10 shows in a Gantt chart the distribution of releases over the time. It is clear how, excepting for the June/August period (vacations period for most of project partners), almost one release per month has been rolled out from January 2012 to October 2014.

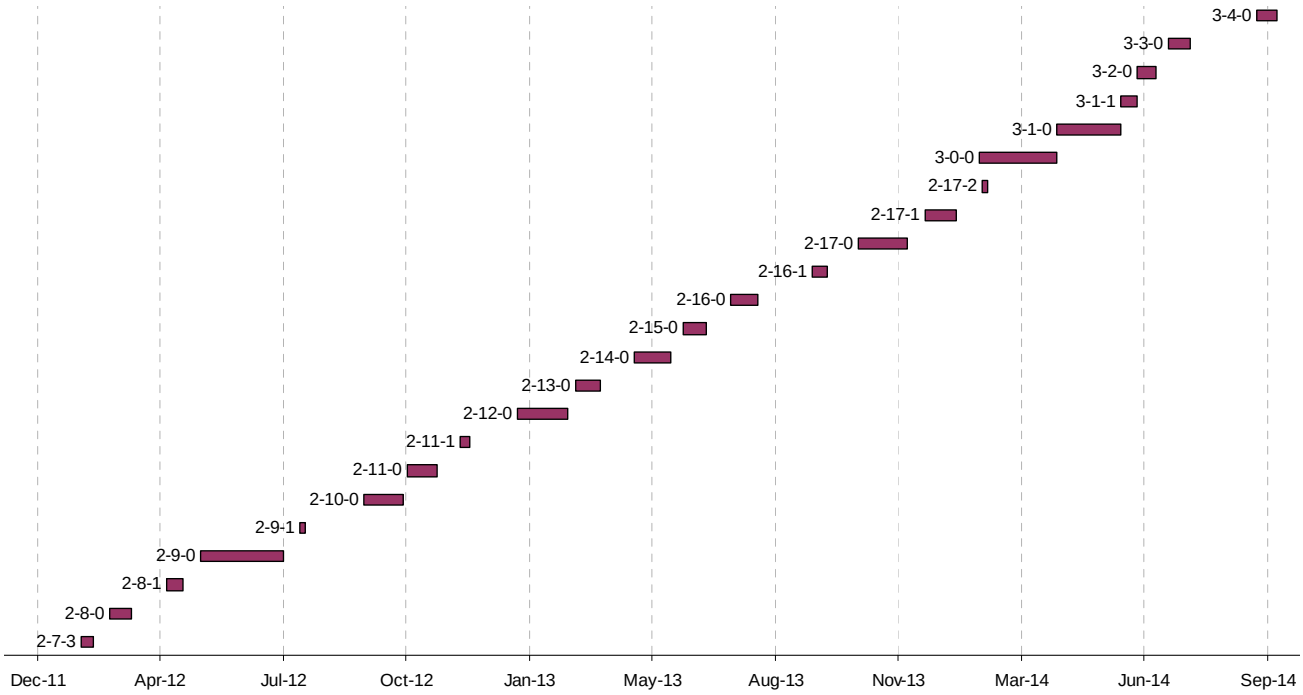


Figure 10: Distribution of gCube releases in time

4.2 3RD REPORTING PERIOD RELEASES

This section reports all the gCube releases rolled out **during the last iMarine project reporting period**. Older releases can be found in previous activity reports D7.2 [3] and D7.3 [4]. The description for each release will focus on characteristics of each release that required special attention and effort from the Work Package 7 point of view.

A more detailed description of releases can be found at following sources:

- **Release News**: a feature-based description of each release is available at the gCube website [5] in the "News" section;
- **Release Notes**: a component-based description of each release is available in the Work Package 7 wiki at *Release Log* page [6];

4.2.1 GCUBE 2.16.0

In gCube release 2.16.0, a review of components in previous releases has been done in order to discover outdated and unused components with the aim of reducing the size of the release. As result, 89 components have been dropped. Among the main changes shipped with this release are: several *Portlets* have been improved (especially *Search* and *Workspace* portlets), *Data Transfer* service has been better integrated within the system, some bugs have been fixed in the *Data Access* layer and in the *Enabling* layer.

In total, 40 integration issues have been reported in the Trac [7] system, and 20 full release builds have been necessary before the release was ready for the deployment in production.

4.2.2 GCUBE 2.16.1

This maintenance releases fixed several issues found in production after deployment of gCube 2.16.0. The components which were most affected were: *Aquamaps, Workspace Portlet, News Feed, Species Product Discovery*.

4.2.3 GCUBE 2.17.0

Release gCube 2.17.0 marked an important milestone in gCube system because the new *FeatherWeight* stack was released to replace, gradually, the old gCore stack. The new stack is more lightweight and more flexible than the older one and starting from the release many services have been updated so as to use it.

Additionally, on the portal side, several portlets received an update to implement new features of fix bugs.

Given the impact of the change, almost half of components in gCube have been upgraded in this release: 251 out of 470. This led to longer integration times (40 days) with the need of 35 full builds and 30 integration issues reported in Trac.

4.2.4 GCUBE 2.17.1

This maintenance release fixed few high priority bugs found concerning *OAI-TM Harvester, Workspace, HomeLibrary, XSearch* and *User Management*.

4.2.5 GCUBE 2.17.2

This further maintenance release of gCube 2.17.0 fixed some more bugs that caused incidents in production and, therefore, categorized as high priority.

4.2.6 GCUBE 3.0.0

gCube 3.0.0 is the only major release in iMarine project and one of the most significant milestones of gCube system from the point of view of software integration. In fact, in this release all the gCube software has been integrated to run in Java 7 runtime environment. All previous releases had been integrated with Java 6, but since Java 6 reached its End Of Life deadline (i.e. no further updates were planned), the project decided to switch to Java 7.

The switch impacted not only the release process (Work Package 7) but also the development of components and the update of the project infrastructures (Work Package 5). This is because the Java version should be the same in all infrastructures (development, integration, testing and production). A collaboration with other Work Packages and an integrated migration plan was then necessary.

As to the release integration phase, several components have been released in order to make their source code or (most of the times) their project descriptor file (i.e., pom.xml) compatible with Java 7. In fact, in the release, 237 out of 497 components have been released and the integration phase was 63 days long with 55 full gCube builds.

Additionally, following components released also new features: *Workspace, Tabular Data Manager, Data Transformation* and *VRE Management.*

4.2.7 GCUBE 3.1.0

In contrast with the previous release that was focused on technological aspects, gCube 3.1.0 has been focused on delivering new functionalities to gCube portal and its applications. In particular, *Tabular Data Management, User Management, SmartGears, Statistical Management, Smart Fish*.

The release required 52 days to be completed with 44 full builds executed and 18 issues reported in Trac.

4.2.8 GCUBE 3.1.1

This maintenance release focused on bugs in the *Home Library* components found in production.

4.2.9 GCUBE 3.2.0

This release focus on *Tabular Data Manager* service and portlets with new plug-ins and widgets added, a set of new libraries (called SwisSQL) to interface with different database systems for indexing data, *Workspace* sharing capabilities.

In addition the new procedure for functional tests of portlets ran for the first time, testing 12 portlets.

The integration of the release took 16 days and 19 full integration builds.

4.2.10 GCUBE 3.3.0

Also this release, like the previous one, focused on shipping new functionalities for *Tabular Data Management and Workspace* components.

Additionally improvements to the *VME* component has been done and a new set of portlets functional tests has been executed on six new portlets.

4.2.11 GCUBE 3.4.0

In this release several components received an update: *Tabular Data Management, Workspace, Search portlet, Data Transformation service, Ecological Modelling*. Five new portlets have been tested by the portlet functional test procedure.

4.2.12 GCUBE 3.5.0

gCube 3.5.0 is the last gCube release planned for iMarine project and the integration has not been started yet at the time of writing.

4.3 GCUBE.HEAD

gCube.HEAD is the name of the development version of the gCube project. This configuration contains development version of all gCube components. It is automatically built on a daily basis to check the integration between development versions of components. Following the philosophy of Continuous Integration, daily builds of gcube.HEAD help developers stay aligned with the rest of project developments, and to make their components work with the rest of the system. Given its nature, this “version” of gCube is never superseded and it is continuously updated with the latest development versions of gCube components.

Integration builds of gCube.HEAD are executed on the build server hosted at CNR. The generated reports and artefacts generated are kept on the build server for a limited amount of time, mainly to preserve disk space (the current policy is to only keep last three builds). On the same build server an instance of BTRT has been installed to access results of builds from the web [8].

During the reporting period almost 450 builds have been executed and for most problematic situations either ticket on gCube Trac have been opened or developers have been contacted directly and asked for fixes.

5 DOCUMENTATION VALIDATION

The validation procedure of the Wiki documentation (also known as **Wikidoc**) was performed over this period, to ensure the high quality and validity of each of the three subsections, namely:

- the **Developer's Guide** [9];
- the **Administrator's Guide** [10];
- the **User's Guide** [11];

The Wikidoc structure has been reformed towards a more intuitive design. Previously, Wikidoc guides were restricted to a listing of the available components - with some groupings - but with no further semantic connection between the groups and components. In order to improve that and make easier to reach the information inquired, the Wikidoc guides structure is now extended with "How-to-do" wiki pages.


page discussion edit history move watch

Administrator's Guide

This guide is conceived to provide its users with detailed instructions for installing, configuring and operating a gCube-based Hybrid Data Infrastructure.

It is organised in five main parts:

- an Overview that gives basic information on a gCube HDI;
- a section dedicated to the basic deployment of a gCube Infrastructure;
- a section dedicated to the deployment of a portal to act as a GUI on top of the Infrastructure;
- a section dedicated to the creation and operation of Virtual Organisations;
- a section dedicated to the creation and operation of Virtual Research Environments;

Overview

gCube Architecture

Infrastructure Deployment and Daily Management

How to set up a gCube infrastructure
How to install the Core Services
How to create a Spatial Data Infrastructure
How to deploy a gCube Hosting Node
How to create Runtime Resources
How to install Jackrabbit
How to manage Resources
How to monitor the infrastructure via Nagios
How to monitor Resources Consumption

Portal Management

How to install and configure a gCube Portal

Virtual Organisation Management

How to create a Virtual Organization
How to manage a Virtual Organisation **(to be revised)**
How to monitor a Virtual Organisation **(to be revised)**
How to create Indices
How to use the IR Bootstrapper
How to manage Indices
How to configure Search Fields **(to be revised)**
How to manage Resources
How to manage Users
How to manage Roles
How to upload a new Software Component

Virtual Research Environments Management

How to create a Virtual Research Environment
How to manage Users
How to manage Roles

Figure 11: Administrator's Guide Index

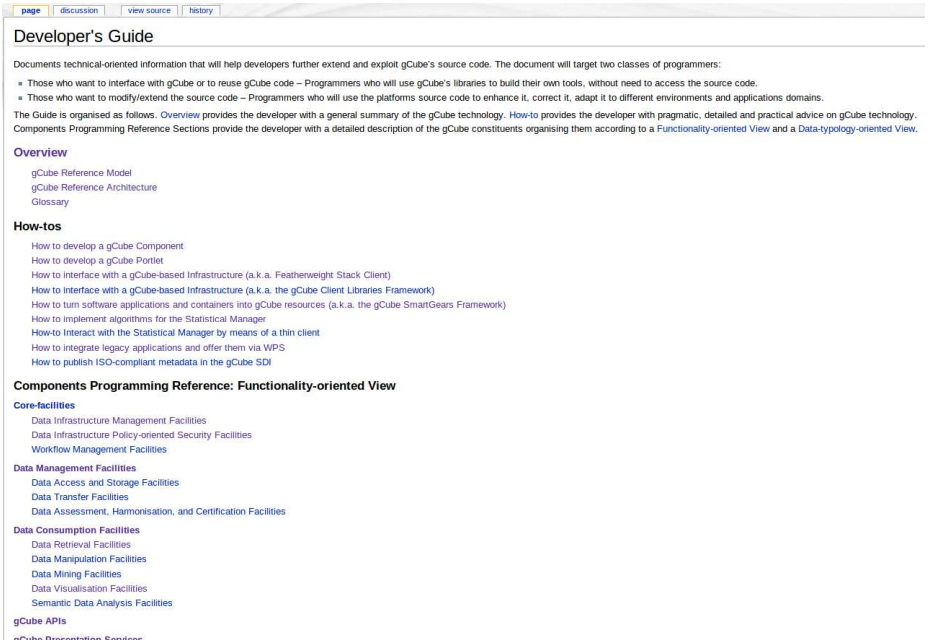


Figure 12: Developer's Guide index

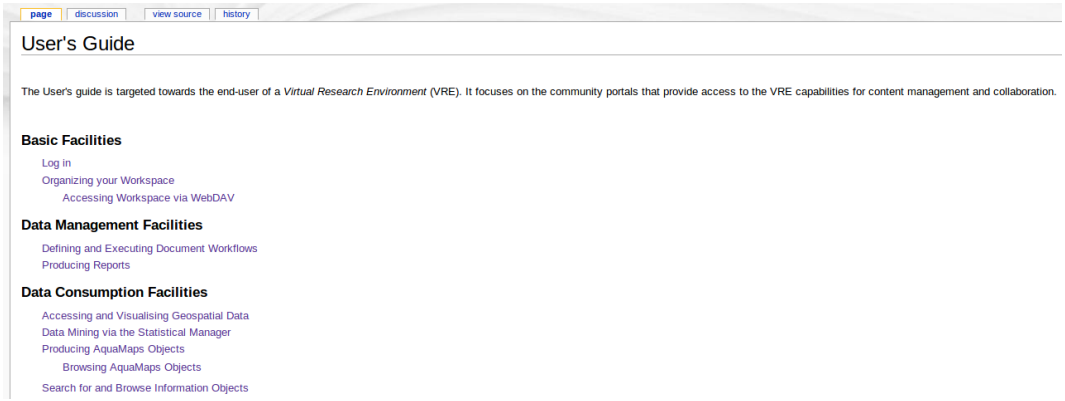


Figure 13: User's Guide index

All wiki pages referring to obsolete components have not been linked within the new guides, ensuring that the guides are up to date.

The validation procedure of the Wikidoc content has been performed by the procedure described on the D7.3 deliverable. A slight deviation in favour of efficiency, is the decision to monitor the validation progress entirely through the state of the issued tickets and not through the dedicated (validation) wiki page.

6 REFERENCES

[1]: ETICS tool website – http://etics.res.eng.it/

[2]: SQALE method - https://sqale.org/

[3]: D7.2 Software Release Activity Report, iMarine project, M 10

[4]: D7.3 Software Release Activity Report, iMarine project, M 20

[5] gCube website - http://www.gcube-system.org/

[6]: Release Log at Work Package 7 wiki - https://gcube.wiki.gcube-system.org/gcube/index.php/Software_Integration_and_Distribution:_Release_Log

[7]: The iMarine issue tracker - https://issue.imarine.research-infrastructures.eu

[8]: BTRT instance at CRN - http://eticsbuild2.research-infrastructures.eu/BuildReport

[9]: Developer's Guide - https://gcube.wiki.gcube-system.org/gcube/index.php/Developer%27s_Guide

[10]: Administrator's Guide - https://gcube.wiki.gcube-system.org/gcube/index.php/Administrator%27s_Guide

[11]: User's Guide - https://gcube.wiki.gcube-system.org/gcube/index.php/User%27s_Guide