Integrated Solutions For Information Sharing In Health Care Applications

Saverio Da Ronco[1], Guillermo Diez-Andino Sancho[2], Lorenzo Dini[2], Elisabetta Ronchieri[3], Matteo Selmi[3]
[1] INFN, University of Padova, Via Marzolo 8, I-35131 Padova, Italy
[2] INFN-CNAF, Via Ranzani 13/2, I-40126 Bologna, Italy
[3] CERN, CH-1211, Geneva 23, Switzerland
elisabetta.ronchieri@cnaf.infn.it

Abstract

The interoperability between various standards and the exchange of data amongst different resources constitutes an important issue in Health Care Applications. Standards allow institutes to transmit clinical data and images from the electronic patient folder to different hospital sites, requiring these elements for diagnosis, medical studies and treatment. Medical image examination is a subset of the medical image folder that includes a request, images, prescription, reports of specimen analysis, and other elements.

In this paper, we introduce a system called ETICS, which is also designed to automate the investigation of this kind of problems. ETICS provides a service for software projects by integrating well-established procedures, tools and resources in a coherent framework and adapting them to the special needs of distributed software. An implementation of how the ETICS system supports interoperability issues is also described, taking into consideration the standard Digital Imaging and Communications in Medicine. Finally, we provide the obtained results achieved by using the ETICS capabilities.

1. Introduction

In the latest years, interoperability has become important in European and International collaborations, where different parts of Grid and distributed systems must interoperate amongst themselves and with different middleware implementations in different networks [1]. Interoperability and compliance to standards are important quality attributes of software developed for Grid and distributed environments, because many different parts of an interconnected system have to interact. One of the major factors in making sure that interoperating parts of a distributed system can actually interconnect and exchange information, is compliance with standards. Considering the case of the Grid environment [2], most of the projects that are developing software have not reached the maturity level of other communities yet and have difficulties identifying and adopting standards. Validating the compliance with standards often requires the design of custom test suites and a firm care to any proposed changes. Interoperability among middleware and application software developed in order to be used on Grid and other distributed environments is usually a very complex issue.

Interoperability issues can be also found in Health Care systems, where achieving interoperability is much more than an engineering design problem. In Health Care, the end users (i.e., doctors and lab technicians) can help to identify and address interoperability issues, actively collaborating in the development process, and being involved with the vendors and personnel who are making decisions about their systems. In addition, they can plan to adopt new processes and technologies only if they are not too complicated: it is very difficult for doctors or lab technicians to justify the time they use to learn a new software application when patients are waiting for their attentions. Technologies in Health Care have risen continuously in the past years. Medical devices are able to communicate data in digital formats. A number of de facto standards have been developed, which ensures that medical devices from different manufacturers are able to feed information into the same type of information system. The best known of standards are Health Level Seven (HL7) [3] and Digital Images and Communications in Medicine (DICOM) [4, 5, 6, 7].

In this paper, we present ETICS [8, 9], a Configuration, Build and Test System, that provides a number of services and tools to access. They represent an integrated infrastructure for the automated build, configuration, integration and testing of software. These tools help to improve the interoperability of software applications and

1530-1605/08 $25.00 © 2008 IEEE

middleware components, evaluate software by comparing it with published standards and validate the portability of the software across different platforms. The ETICS system is addressing projects developing Grid and distributed software, especially to help them define processes, methodologies and tools to implement static and dynamic analysis (such as evaluating the adherence to user defined coding conventions, making custom analysis like IPv6 compliance, evaluating the code coverage for unit tests). Nevertheless, the ETICS system is equally oriented towards non-Grid projects. Currently, several fixed-term research projects such as EGEE II [10], DILIGENT [11], and OMII-Europe [12] are using or planning to use the ETICS system to solve their software build problems. Recently, also long-term research projects such as Condor [13] and Metronome [14] are using the ETICS system to evaluate the reliability of the ETICS test process. Therefore, the ETICS system is able to support middleware and applications addressed to different communities, like High Energy Physics through the supporting of EGEE. In addition, ETICS can be used by several communities for its dynamicity.

This paper is organized as follows. Section 2 describes some medical standards, whilst Section 3 focuses on a specific standard for medical imaging, called DICOM. Section 4 details interoperability issues in Health Informatics. Section 5 describes the basic architecture of the ETICS system used to support the interoperability issues in the Health Care Applications. Section 6 details some ETICS functionalities. Then, Section 7 describes the implementation done to prove the usability of the ETICS system in order to verify interoperability between devices. Section 8 provides information about the current usage of the ETICS system and plans for evaluation within or after the implementation phase of the system. Finally, Section 9 reports conclusions.

2. Medical standards

This section summarizes standards in Health information technology, as schematized in Table 1. Basically, there are two distinct families of standards, respectively relating to hardware and software. In this section, we only describe the major software standards.

In the software branch, ISO TC 215 [15] constitutes the international standard of Health Information and Communications Technology (ICT), to allow for compatibility and interoperability between independent systems. A European standard, called CEN TC 251 [16], has been published recently. It is a group working on standardization in the field of Health ICT in the European Union, aiming to achieve compatibility and interoperability between independent systems and to enable modularity in Electronic Health Record systems. In 1993, the specialist standard for medical imaging and communications, DICOM, was issued. Since then, this standard has been updated with a new release each year. The HL7 and EDI [17] standards cover the field of messaging and are maintained by independent committees.

Table 1. Standards in Health Information Technology.

Standards	*Descriptions*
ISO TC 215	International standard
CEN TC 251	European standard
DICOM	Standard For Medical Imaging and Communications
HL7	Standard For Messaging
EDI	Standard For Messaging

Medical image examination is a subset of the medical image folder. The standards involved are DICOM, HL7, Corbamed [18], and EDI, to facilitate admission, order entry, image acquisition, results reporting, and billing. Due to the large number of different and overlapping standards, it is quite obvious that there is a need for further integration and for standards to evolve.

3. DICOM

The DICOM standard was created in 90s by the National Electrical Manufactures Associations to aid the distribution and viewing of medical images in Computed Tomography scans, Magnetic Resonance, and computer radiology. The idea was to create a standard designed to ensure interoperability of systems used to produce, store, display, process, send, retrieve, query or print medical images and the derived structured documents as well to manage related workflow [19]. The DICOM standard is widely used by manufacturers, but many medical acquisition devices are weakly conforming to it. DICOM is complex, involving several elements

such as images, storage services, image folder retrieval, and information and communication technology networking support. DICOM was extended to cardiology applications in 1995; ultrasound, nuclear medicine, and PET in 1996; radiotherapy and visible light in 1997; digital x-ray mammography and radiology, and vital signs in 1998; audio and oral waveform, electrocardiograms, and visible light for endoscopy and pathology in 1999. Currently, there are about twenty working groups in the DICOM committee, two of which are recent and very important. The first group is responsible for providing a method to flag images for clinical trial and education purposes, trying to integrate them into the research protocol. The second one is working on the integration of imaging and information systems, using the standards HL7, ISO TC 215, and CEN TC 251.

DICOM uses as a standard for compression JPEG LS at present, but work is in progress to adopt the JPEG 2000 standard too. The new supplement 254 is now on public command. It is called DICOM Multipurpose Internet Mail Extensions (MIME) [20, 21], which allows encapsulating DICOM objects into e-mail by means of the MIME protocol. This is the DICOM Web and e-mail initiative in order to integrate DICOM with Web technology. Links are being established between ISO TC 215, HL7 and DICOM to avoid inactivity in the standardisation process.

The DICOM standard defines also the communication protocol for image transfer; therefore a DICOM server/client architecture can be defined. The DICOM servers are software applications residing on computers with a disk and/or tape backend able to store and retrieve DICOM images, whilst DICOM clients can be identified with almost all medical image acquisition devices. There is no standardization on DICOM storage: DICOM servers are implementing their own policy of data storage. The DICOM security model is rather weak (e.g., DICOM files are not encrypted and transported unencrypted). The DICOM server security is based on the fact that all users having access to DICOM client applications have right to access the information they are accessing. This protection limitation is often exceeded in hospitals by isolating the server from outside the world.

A DICOM file contains a header (which stores information for instance related to the patient's name, the type of scan, the image dimensions), as well as all the image data. The image content is usually stored in raw format but also compressed formats are allowed. The header is composed by fields: some of them are mandatory (even though not all devices use them); other fields can be defined by manufacturers to store desired information. There exists a lot of software [22] able to provide browsing and viewing functionalities of DICOM images: some of them are free, whilst others are proprietary. Moreover, most of them are distributed for different platforms such as Mac OS, Windows, UNIX, Linux.

3.1. An example of DICOM application

Projects are also in place with the aim of allowing a better communication amongst hospitals. Also, they provide an integrated framework allowing all the medical community the access to medical images. One of them is the Medical Data Manager (MDM) [23, 24], a software interface able to interconnect the Grid Storage Resource Management (SRM) (i.e., gLite SRM) [25] to DICOM servers.

MDM was developed with the support of the EGEE[1] European IST project. It aims to provide accessing medical data sources for computing without interfering with the clinical practice, to ensure transparency so that accessing medical data does not require any specific user intervention, and to ensure a high data protection level to respect patients privacy. MDM is able to register data produced by DICOM sources in the Grid Data Management System, and to retrieve medical images on dedicated machines. In practice, the MDM has to allow a query done by DICOM client to access DICOM servers outside the hospital and retrieve the DICOM files needed. Image search can be Logical File Name-based or metadata-based. Clearly, security must be enhanced in order to protect the patient privacy, and inter-communicability has to be provided between the DICOM transactions and the gLite SRM requests. The system includes an interface to DICOM, a metadata management tool, a SRM interface for Grid compliance, and Grid data catalogues. The metadata can be distributed over the various acquisition sites.

A prototype was successfully deployed and tested in a controlled computing environment. Medical databases were accessed without interfering with clinical practice. In fact, data are kept on clinical sites and transparently

[1] Enabling Grids for E-sciencE, http://www.eu-egee.org

transferred to the Grid only when needed. The MDM prototype ensured a high level of security to preserve patients privacy. The core MDM development is not finished yet and additional functionalities will be included to enrich the service. The next step will see interfacing to medical imagers inside hospitals. It will require to simplify the installation and configuration procedures as most as possible.

4. Interoperability issues

It is necessary to comply to standards because interoperability has to be ensured. In addition, users should be informed of this compliance through the principles of conformance statement, standard body mark, and certification and conformity assessment. There are two types of conformance: the former, called dynamic, applies to all those requirements which are permitted by the standard; the latter, called static, is a statement that a particular application corresponds to all that is implemented by the standard. Software coming from different projects or different development clusters within the same project is often written in different programming languages, deployed on different architecture and operating systems: this increases the possibility of having run-time problems when the programs try to interoperate. In addition, distributed software often has to deal with communication and interoperation problems due to the lack of a real test system and the difficulties of running complete and validated automatic tests on the projects' infrastructures. Moreover, when middleware and applications are deployed on certification testbeds, a lot of time is usually spent trying to ensure interoperability among the different parts of the system, due to the adoption of different file formats or protocol specifications.

In Section 3, we have reported that there are a lot of programs implementing partially or completely the DICOM standard. Some of them provide both a client and a server and therefore they probably do not need to test interoperability with themselves, but there are also some applications providing either a client or a server service. Moreover, DICOM images are often shared amongst different institutes and it is necessary to transfer them through the network between machines that, most likely, have different environments, both as operating system and as DICOM software. Therefore, the interoperability is an important issue in these distributed systems. In particular, this issue has to be deal with in all the environments in which data has to be transferred amongst different machines, having different software installed as it is for the Health Care institutes.

5. ETICS architecture

The basic architecture of the ETICS system is illustrated in Figure 1. It is based on the requirements for the design of the build, configuration, integration and testing framework for distributed software [8]. It also addresses portability issues and interoperability testing [9]. It is organized into four layers, as shown on Figure 1: (1) client layer; (2) business layer; (3) scheduler layer; (4) data layer.

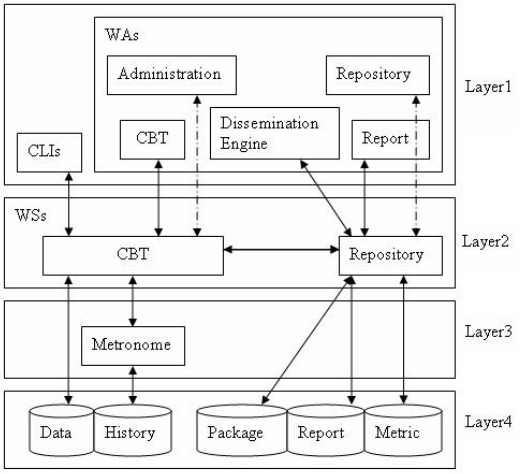

Figure 1. Etics architecture

At the client layer level, the Command Line Interfaces (CLIs) and the Configuration, Build and Test (CBT) Web Application (WA) are the tools provided to the user in order to access metadata information handled by the CBT WS, which manages the user requests and contacts the execution engine to commit build and test jobs. The Administration WA is a tool used to register users in the ETICS infrastructure and give permissions to them. Then, the Repository and Report WAs are tools provided to the user to access results and packages that have been produced during the build and test activities. Finally, the Dissemination Engine (DE) WA is an extensible framework composed by different disseminators that shows several dynamic views of the metrics collected during the build and test

process. The storage of the results and packages are handled by the Repository WS, which manages the user and CBT WS requests. The CBT WA provides to the user the tools to view, monitor, configure and execute automated builds and tests. The CLIs implement a similar functionality as the CBT WA and make use of the same CBT WS interface for simplicity and symmetry (i.e., they share certain functionalities).

At the business layer, the CBT WS is the entity providing logic for the entire service, used by both the CLIs and the CBT and Administration WAs. The CBT WS acts as an intermediate layer between the front-end applications available to the user and the data storage backend, which holds the ETICS data model. It is, in fact, the abstract interface that manages all the data exchange between the other entities. For simplicity and better scalability, the WS is stateless, that is it does not use a stateful WS paradigm, such as Web Services Resource Framework (WSRF). At the business layer level, there is also the Repository WS that is the entity providing logic for the entire service, used by both the Report and Repository WAs. The Repository WS interacts with the CBT WS every time the CBT WS requests to save new packages and reports.

At the scheduler layer, Metronome allows the CBT WS to offer to the user the automation of builds and tests, possibly on a regular schedule, on a large set of different resources and platforms. The engine is provided by Metronome build and test framework, which builds on top of Condor, a specialized workload management system for computing intensive jobs. The CLIs can be used directly by the user on local resources (e.g., a developer machine), but also allow to interact with remote ones. In fact, the same client is used in an almost identical context by the Metronome build and test framework for the execution of the remote builds and tests. This similarity is crucial to avoid context switching between local and remote builds/tests, which would reduce the usability and reliability of the system.

At the data layer, the Data model is designed to organize a software project by using high-level entities such as the project structure, the build configuration, the security information, the job schedule information. The Data model describes explicitly the objects and the relationships amongst them. Furthermore, the model allows representing the results of running a build and test job in a way that can be consumed by the Report WA. These results are stored in a Report model used to generate reports. Then, the History model is used by Metronome in order to store and retrieve the information of remote builds and tests. Moreover, the Package model is designed to collect all the packages produced during the build process. Finally, the Metric model is populated by the Report information used to generate static (e.g., screenshot) and trend (e.g., period graphics) analysis for the quality assurance of software.

6. Some of the ETICS functionalities

This section describes the different features ETICS provides in order to make possible the execution of interoperability tests.

The CBT mechanism is fully provided in ETICS [8]. Then, in order to validate the interoperability amongst different systems it is required, previously to the execution of the tests, the deployment of the different systems to be validated. This is achieved in ETICS by the co-scheduling mechanism described in this section. Once all the systems are deployed, the specific interoperability tests are executed by using the ETICS extensibility mechanism known as the Plugin Framework**.** Finally, in order to execute the whole process on a regular basis, the Scheduler service is provided.

6.1. Co-Scheduling

The co-scheduling (CO-S) mechanism can be used in ETICS in order to automatically deploy and run distributed tests in different platforms. By co-scheduling we refer to the automation of the deployment of the different services and clients, the execution of the tests and the gathering of information such as test results, metrics, and logs. A typical scenario to be faced by the co-scheduling is composed of several services that interoperate amongst them, clients that contact the various services and tests that require servers and clients to be in place.

Figure 2 represents a deployed scenario composed of three services, one client and a testsuite. The arrows represent the dependencies amongst the different elements and the numbers associated to each element define the sequence that the co-scheduler should follow when deploying and executing them. The dotted squares indicate that the elements inside it run on the same machine. The distributed deployment

and testing is a complex problem that requires several issues to be solved. In the first place it is necessary to model how the different parts (i.e., servers, clients, tests) interact. Once this has been defined, it is required to synchronize the deployment and execution of the different instances. In order to do this, it is necessary to have a mechanism by which the different instances can publish information and make it available to the other co-scheduled instances (e.g., ServiceA has been deployed in the host with IP 131.12.2.2). Finally, the different tests need to be executed and the system has to report all the generated information (e.g., logs, test results, and metrics).

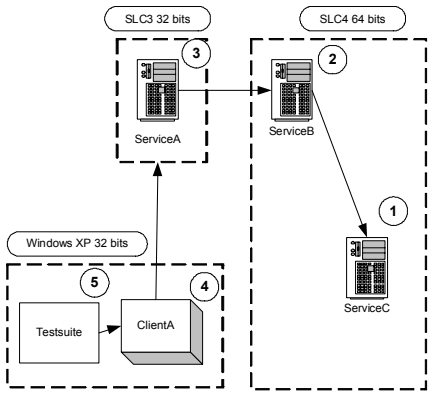

Figure 2. Example of a distributed test scenario.

ETICS solves the modeling problem by using its data model. Services, clients and tests can be represented by a set of Configurations. A Configuration contains all the required information to deploy it (e.g., platform and dependencies). Using configurations we can easily define a deployment scenario, specifying how many nodes it is composed of, what services we need to deploy in each node, and what are the dependencies amongst the different nodes and services. In order to exchange information amongst the nodes, the ETICS system provides a mechanism to publish information and make it publicly available to the rest of the distributed deployment. This mechanism for data interchange includes a Set operation and two Get operations, for blocking and non blocking calls in case the required information is not yet available. This allows the synchronization amongst the nodes according to the data dependencies present amongst the services.

Once the scenario has been modeled it is necessary to orchestrate the automatic deployment and execution. The deployment is achieved by using the Metronome execution engine. Metronome will synchronize the deployment of the different elements (e.g., services, clients, and tests) in the different platforms. In addition it will publish all the necessary information that any element might require (e.g., a test suite needs the host IP where the server to test was deployed). Finally, the ETICS client will collect all the reports and metrics produced.

6.2. Plugins

The Plugin framework is an extensibility mechanism used in ETICS that allows adding extra functionalities on top of the ones provided by the system, and executing quality validation tools using what are called plugins.

A plugin is a module that runs on ETICS as part of the build or test process. Currently, different types of plugins are available that are wrappers to specialized tools in analyzing possible problems in the code or to validate standard compliance of the source code. Their functionalities vary from code checkers to the generation of metrics (i.e., SLOC count, IPv6 and WSI compliance, bug tracking connectors). Users can select from the built-in set which metrics are important to them, therefore ETICS automatically calculates them during the build and test process and makes them visible in the reports.

6.3. Scheduler

This service allows the users to define the periodicity of the builds/tests. It will guarantee that in case any problem is encountered the user will be immediately notified.

7. Implementation

In this section some ETICS configurations are defined to test interoperability between two software that are DCMTK [26] and dgate-server [27], implementing the DICOM standard. The former is a collection of libraries and applications implementing DICOM. It includes software for examining, constructing and converting DICOM image files, handling offline media, sending and receiving images over a network connection. The latter is a fully featured

DICOM server heavily extending the public domain UCDMC DICOM software.

DICOM has been chosen mainly because it is one of the most used standard for sharing Health Care images and there are lot of applications implementing it, completely or partially. Therefore, ETICS can be very useful for testing whether its implementations (DCMTK and dgate) are able to interoperate.

The first step executed consists of inserting and configuring the two components in the ETICS system, including the definition of the repository where to find the code (source or binary) and the commands to download and, if needed, compile it. After having correctly put the software configuration information in ETICS, the system is able to build it, create the packages (e.g. rpms, and debs) and store it in the software repository. The next step consists of defining the interoperability test script that will be run on the ETICS test infrastructure.

The test is run on two nodes using the co-scheduling mechanism that allows users to execute different operations on different nodes. The two nodes, called node1 and node2, contain respectively the installation of the DCMTK client and server, and the installation of the dgate-server whose configuration file requires the hostname of node1. In order to achieve the exchange of information, and the synchronization of node1 and node2, the ETICS Set and blocking Get operations have been used.

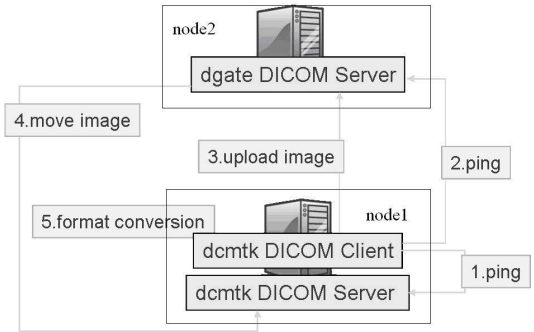

Figure 3. DICOM interoperability amongst the DCMTK client and two servers (i.e., DCMTK and dgate).

Figure 3 describes in five steps how the interoperability verification is tested: (1) the client of node1 sends a ping message to the local DICOM server on node1; then (2) the same client pings a remote preinstalled DICOM server on node2; next (3) the client of node 1 uploads a DICOM image to the server on node2; (4) the preinstalled DICOM server of node2 moves the image from node2 to the server on node1; finally (5) the client of node1 converts the image in a web-compliant format (e.g., from dcm to jpg). At the end of the test, the image is published in the report pages of the test, as shown in Figure 4.

Figure 4. Image converted in a web-compliant format.

This simple test is completely automatic and allows users to test the interoperability between two different implementations of the DICOM standard (DCMTK and dgate-server). It uses some methods defined in the DICOM standards (i.e., C-ECHO, C-STORE, C-MOVE) and checks if the two programs are able to communicate through these methods. This is only one simple test that we have implemented using the ETICS system but a lot of different and even more complicated tests can be issued using ETICS. Therefore, it is also possible to provide a complete test suite for all the DICOM methods checking whether a specific program implements correctly the DICOM standard. In addition, it is also possible to provide some tests checking the interoperability amongst different implementations of the same standard.

8. ETICS usage and evaluation

ETICS is already used by twenty-eight projects (as shown by the ETICS Web Application at CERN https://etics.cern.ch/etics) either in production or in an evaluation phase. The major projects that have already adopted ETICS as the official build and test service for their software, include EGEE/gLite, OMII-Europe and DILIGENT, whilst UNICORE [28], Quattor [29], and CASTOR [30] have chosen ETICS as prototype preliminary framework. They are among the most important European Grid projects that involve a large number of

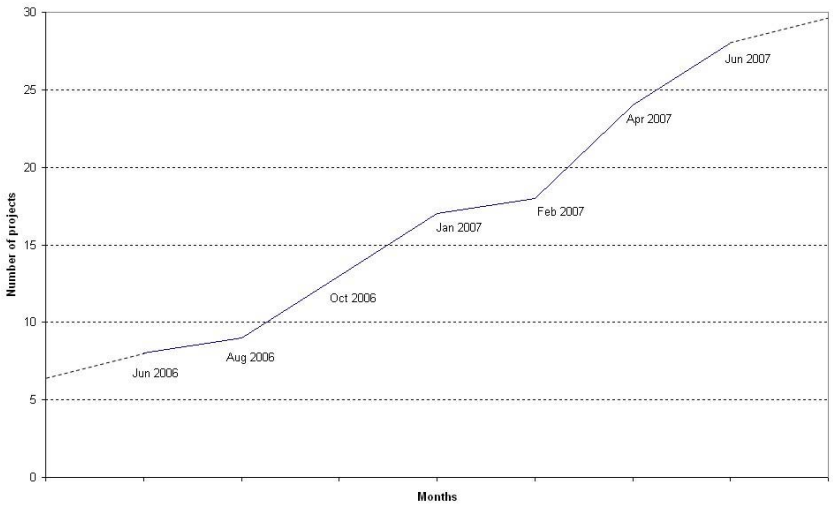

Figure 5. Trend for the ETICS usage.

research institutes aiming to develop a Grid for scientists. Figure 5 shows the registration trend of projects in the ETICS system. The trend has grown as soon as the ETICS system has become trustier within the ETICS community.

In what follows, some details are given about the different projects use the ETICS framework. EGEE/gLite uses ETICS to build packages for specific platforms such as SLC4 (Scientific Linux Cern 4), SLC3 and Debian on 32 and 64 bit architectures. Their service meta-packages are built every night on the resources provided by ETICS, and they are tested dynamically using *yum* or *apt* repository servers on all supported platforms. OMII-Europe uses ETICS to build its re-engineered middleware components on all platforms (mainly SLC4, SLC3, Debian), supported by the original providers. OMII-Europe also performs interoperability testing of services on ETICS to validate the compliance of such components with reference standards and interoperability between different implementations. OMII-Europe has a dedicated ETICS installation at the University of Southampton and also uses the public ETICS distributed testbed. DILIGENT uses ETICS to build its digital library software based on gLite and to perform deployment tests of its services.

The people that work in these projects have been the main testers of the ETICS services and they have provided a lot of useful information to improve the functionalities, the performance and the work plan of ETICS. This feedback from the ETICS user community has been achieved throughout the access to a dedicated ticketing support, via the participation to training/demonstration meetings and to other dissemination events. Therefore, ETICS has defined a good evaluation plan, collaborating with projects people deeply involved in estimating the ETICS performance. Currently, ETICS has also started a re-engineering plan, aiming to analyze the performance of the ETICS services in order to optimize the overhead introduced by the system itself, therefore speeding up the build and test processes.

9. Conclusions

In this paper, we presented how the ETICS system can be used in order to help Health Care applications to deal with interoperability issues. We summarized some of the medical imaging standards, such as the international standard ISO TC 215, the European standard CEN TC 251, the specialist standard for medical imaging and communications DICOM, and the HL7 and EDI standards. DICOM was particularly described, giving a brief description of a Grid Health application, called MDM. We explained why the software integration is an important issue in the Health Care applications. Then, the ETICS architecture was explained, together with the mechanisms adopted by ETICS in order to provide the adequate tools for supporting interoperability and compliance to standards. Again, we described how ETICS supports the DICOM interoperability. Finally, we provided the ETICS evaluation describing how potential

users were involved during the system development.

10. Acknowledgement

We would like to thank colleagues from the ETICS project, who have provided useful suggestions to complete this paper. This work is partially funded by the European Commission under contract number INFSOM-RI-026753.

References

[1] F. Gagliardi, "The EGEE European Grid Infrastructure Project", Lecture Notes in Computer Science, Vol. 3402, pp: 194-203, 2005.

[2] I. Foster, and C. Kesselman, "The Grid 2: Blueprint for a New Computing Infrastructure", Morgan Kaufmann Publishers Inc., 2003, San Francisco, CA, USA.

[3] Health Level Seven, http://www.hl7.org/.

[4] DICOM, http://medical.nema.org.

[5] DICOM Standards Committee, http://www.nema.org/prod/med/upload/DICOM%20STANDARDS%20COMMITTE-3.pdf.

[6] DICOM Brochure, What is DICOM?, http://medical.nema.org/dicom/geninfo/Brochure.pdf.

[7] A Digital Imaging and Communications in Medicine (DICOM) Part 1: Introduction and Overview, ftp://medical.nema.org/medical/dicom/2007/07_01pu.pdf.

[8] M. Bégin, *et al.*, "Build, Configuration, Integration and Testing Tools for Large Software Projects: ETICS", In Springer Verlag Lecture Notes in Computer Science (LNCS) Series, LNCS 4401, pp: 81-97, 2007.

[9] M. Bégin, *et al.*, "Analysis of Requirements for Automated Interoperability Testing", In Proceedings of the Tenth World Conference on Integrated Design & Process Technology, IDPT 2007, Antalya, Turkey, June 3-8, 2007.

[10] EGEE Middleware Architecture, August 2004, https://edms.cern.ch/file7476451/1.0/architecture.pdf.

[11] D. Castelli, *et al.*, "DILIGENT: a DL infrastructure for supporting joint research", *Proceedings of the 2nd IEEE - CS International Symposium Global Data Interoperability*, IEEE Computer Society, 2005, pp: 56-59.

[12] OMII Europe, http://omii-europe.org/OMII-Europe/.

[13] D. Thain, *et al.*, "Distributed Computing in Practice: The Condor Experience", Concurrency and Computation: Practice and Experience, Vol. 17, No. 2-4, pp: 323-356, February-April, 2005.

[14] A. Pavlo, *et al.*, "The NMI Build & Test Laboratory: Continuous Integration Framework for Distributes Computing Software", In Proceedings of LISA'06: Twentieth Systems Administration Conference, Washington, DC, December 2006, pp: 263-273.

[15] ISO/TC 215 Working Group 3 - Health Informatics - Semantic Content, http://www.tc215wg3.nhs.uk/.

[16] CEN/TC 251 - European Standardization of Health Informatics, http://www.centc251.org/.

[17] R. Clarke, "Electronic Data Interchange (EDI): An Introduction", http://www.anu.edu.au/people/Roger.Clarke/EC/EDIIntro.html.

[18] S. Vinoski, "CORBA: Integrating Diverse Applications Within Distributed Heterogeneous Environments", IEEE Communications Magazine, Vol. 14, No. 2, February 1997.

[19] G. Schaefer, J. Huguet, Shao Ying Zhu, P. Plassmann, and F. Ring, "Adopting the DICOM standard for medical infrared images", In Proceedings of the 28th Annual International Conference of the IEEE - Engineering in Medicine and Biology Society, Aug. 2006, pp: 236-239.

[20] N. Freed, N. Borenstein, "Multipurpose Internet Mail Extensions, Part I: Format of Internet Message Bodies", November 1996, http://www.ietf.org/rfc/rfc2045.txt.

[21] N. Freed, N. Borenstein, "Multipurpose Internet Mail Extensions, Part I: Media Press", November 1996, http://www.ietf.org/rfc/rfc2046.txt.

[22] The DICOM Standard, http://www.sph.sc.edu/comd/rorden/dicom.html.

[23] J. Montagnat, *et al.*, "Bridging clinical information systems and grid middleware: a Medical Data Manager", In Proceedings of HealthGrid conference (HealthGrid'06), pp: 14–24, Spain, June 2006.

[24] J. Montagnat, T. Glatard, D. Lingrand, R. Texier, "Exploiting production grid infrastructures for medical images analysis", In Proceedings of the First

Singaporean-French Biomedical Imaging Workshop (SFBI'06), Singapore, Oct 2006.

[25] D. Budgen, *et al.*, "Managing healthcare information: the role of the broker", In Healthgrid'05, Oxford, UK, April 2005.

[26] M. Eichelberg, J. Riesmeier, T. Wilkens, P. Jensch, "One decade of medical imaging standardisation and implementation: a short review of DICOM and the OFFIS DICOM Toolkit", In Proceedings of EuroPACS-MIR 2004 in the Enlarged Europe, pp: 253-256.

[27] Conquest DICOM software, http://www.xs4all.nl/~ingenium/dicom.html.

[28] M. Rambadt, *et al.*, "DEISA and D-Grid: using UNICORE in production Grid infrastructures", In Proceedings of German e-Science Conference 2007, Baden-Baden, May, 2007.

[29] R. García Leiva, *et al.*, "Quattor: Tools and Techniques for the Configuration, Installation and Management of Large-Scale Grid Computing Fabrics", In Journal of Grid Computing, Vol. 2, No. 4, pp: 313-322.

[30] G. Lo Presti, *et al.*, "CASTOR: A Distributed Storage Resource Facility for High Performance Data Processing at CERN", In IEEE MSST 2007, San Diego, USA, September 2007.