A Testing Framework for Assessing Grid and Cloud Infrastructure Interoperability

Thomas Rings, Jens Grabowski
Institute of Computer Science, University of Göttingen
Göttingen, Germany
Email: {rings,grabowski}@cs.uni-goettingen.de

Stephan Schulz
Conformiq Software OY
Espoo, Finland
Email: stephan.schulz@conformiq.com

***Abstract*—The composition of grid and cloud computing infrastructures using equipment from different vendors to allow service enrichment and increase productivity is an important need in industry and for governmental institutions. Interoperability between equipment can be achieved using the gateway approach or the standardized interface approach. These approaches, as well as equipment need to be engineered and developed with the goal to allow problem-free interoperations between involved equipment. A step towards such interoperation is the assessment of interoperability. Focusing on technical interoperability, we present a testing framework for the assessment of interoperability of grid and cloud computing infrastructures. This also includes the assessment of application deployment onto several infrastructures provided by different vendors, which is a key driver for market success. This testing framework is part of an initiative for standardizing the use of grid and cloud technology in the context of telecommunication at the *European Telecommunications Standards Institute* (ETSI). Following the test development process developed and used at ETSI, we developed a test architecture, test configurations, compliance levels, test purposes, interoperability test descriptions, test applications, and a test selection method that together build the testing framework. Its application is exemplified by the assessment of resource reservation and application deployment onto grid and cloud infrastructures based on standardized Grid Component Model descriptors. The presented testing framework has been applied successfully in an interoperability event. In this article, we present a testing framework for the assessment of the interoperability of grid and cloud infrastructure.**

***Keywords*-standardization; interoperability; GCM; grid; cloud; testing**

I. Introduction

Accessing globally distributed data and computing power independent from locations becomes more and more an obligatory requirement from customers in order to store data and utilize computing power. Systems for efficient usage of idling resources, which are physically located all over the world are needed. Grid computing systems, but also recently cloud computing systems, offer methodologies for achieving such a goal. Both offer services for obtaining, providing and selling computing power on demand. This is especially interesting in application domains where computing power is needed spontaneously and in unpredictable time intervals. Many grid and cloud providers have recently appeared on the market offering their own custom-made solutions to address this need. However, from a customer point of view, it is required to access several systems offered by different providers to use more resources, for example, for replication on different systems, but also to save money choosing the best solution. This allows service enrichment by integrating services only available in another infrastructure and to increase productivity by consuming such extended services. A way to achieve this goal is to make these systems interoperable.

Interoperability can be leveraged to open new markets, to foster innovation, and to enable mass markets. Interoperability allows the creation of new and innovative systems through composition of interoperable systems. Furthermore, it increases system availability and reliability. Interoperability provides a great mean to success. However, equipments from which the systems are composed can be developed by different vendors. These equipments need to be engineered to be interoperable. An interim approach is the gateway solution that allows communication between equipments. A gateway converts messages received by one equipment into a representation understandable by another equipment to allow their interoperation. The long-term approach to achieve interoperability is the implementation of agreed specifications, i.e., standards, which capture requirements and functionality. In addition, they define architectures as well as interfaces and specify protocols to be used for communication via these interfaces. Ideal specifications are independent from implementations and leave space for innovation. Even if specifications are assumed as unambiguous, which is rarely the case, testing is needed to validate that implementations follow the specifications. A further step is to test if implementations are able to interoperate.

Proper testing - similar as specification - requires a well defined process and guidelines to be effective in its application. The purpose of testing frameworks is to define a structured approach to test specifications in a given domain, i.e., what to test as well as how to test certain aspects of a *System Under Test* (SUT). They help to increase the quality of test specifications.

At the *European Telecommunications Standards Institute* (ETSI), an initiative for standardizing the use of grid and cloud computing technology in the context of telecommunication, the *Technical Committee CLOUD* (TC CLOUD) (previously *Technical Committee GRID* (TC GRID)) [1], has been formed. Under its umbrella, standards and interoperability in grid, cloud and telecommunication systems are analyzed, developed and forwarded. These include the following: analysis of interoperability gaps between grid and cloud [2]; surveys on grid and cloud standards [3]; comparisons between grid, cloud and telecommunication systems [4]; the grid standard *Grid Component Model* (GCM) [5]–[7], analysis of architectural options for combining grid and the *Next Generation Network* (NGN) [8]. As part of this initiative towards standardization, we present an interoperability testing framework for grid and cloud computing systems following the systematic test development process developed and used at ETSI [4]. The testing framework unifies testing in diverse domains such as grids and *Infrastructure as a Service* (IaaS) clouds systems, which are dominated by proprietary interfaces for similar functionalities. The framework should be applied in the focus of interoperability events.

This paper extends our previous work [9] with a discussion on differences and similarities of grid and cloud computing systems. Furthermore, concepts on interoperability are presented and a extended description about test configurations and compliance levels, which belong to the presented test framework is given. As a result of applying the testing framework in a cloud and grid interoperability test event, we added consideration about needs of standardizing cloud computing systems.

This article is structured as follows: In Section II, we present the three forms of cloud computing and discuss briefly the differences between grid and cloud computing systems. Afterwards, in Section III, we discuss types of interoperability and approaches on achieving interoperability in software systems. In Section IV, we consider different types of testing in the context of standardization including conformance and interoperability testing. In Section V, we introduce the ETSI GCM standard, which provides the main context for our testing framework that is illustrated afterwards. This testing framework, our main contribution, is applicable for resource reservation and application deployment in grid and cloud infrastructures. In Section VI, the application of the framework is exemplified by the grid middleware Globus Toolkit and the cloud computing system of Amazon. An event around the presented testing framework organized by ETSI is described in Section VII. Resulting from the event, we discuss standardization needs towards interoperability of cloud computing infrastructures. Afterwards, in Section VIII, we provide an overview and a comparison with related work in the domains of grid and cloud computing. Finally, we conclude with a summary and outlook in Section IX.

II. Cloud Versus Grid Computing

Grid computing has a complementary but independent relationship to cloud computing. Both are highly distributed systems and fulfill needs for large computational power and huge storage resources.

However, cloud systems utilize virtualization to offer a uniform interface to dynamically scalable underlying resources. Such a virtualization layer hides heterogeneity, geographical distribution, and faults of resources. By the nature of virtualization, a cloud system provides an isolated and custom-made environment. Clouds are classified in a layered model containing the following layers from bottom to top as depicted in Figure 1: *Infrastructure as a Service* (IaaS), *Platform as a Service* (PaaS), and *Software as a Service* (SaaS).

In the IaaS layer, the user is responsible for *Virtual Machine* (VM) management. Physical hardware that includes disks, processors, and networks are virtualized and configured according to the needs of customers. It is not needed to purchase or manage physical data center equipment. The benefit is its scalability because resources can be added or removed on demand while continuing business operations in a rapid and highly dynamic way. It targets especially system administrators.

The PaaS layer provides a higher abstraction than the IaaS layer and is usually deployed on the virtualized infrastructure of IaaS. PaaS provides a development platform, e.g., an *Application Programming Interface* (API) to request VMs, which are handled transparently by the PaaS. It can also include databases, message queues, or object data stores. PaaS is especially used by software developers and system architects.

SaaS is the instance on top of the cloud model. It provides the application in the cloud, which is only customizable within limits. It focuses on end-user requirements and allows the end-user to access applications intuitively, e.g., via a web browser. The application or software is used on demand over a high-speed network and runs on VMs, which are deployed on the cloud providers' physical hardware. Therefore, current cloud solutions offer dedicated access, i.e., the cloud customer is bound to a company or institution or requires duplicated effort to repeat the deployment process for additional cloud environments.

In contrast, a grid computing environment aggregates heterogeneous resources offered by different providers. It aims to provide a standard set of services and software that enable the collaborative sharing of federated and geographically distributed computing and storage resources. It provides a security framework for identifying inter-organizational parties (both human and electronic), managing data access and movement, and utilization of remote computing resources.

Grid computing can benefit from the development of cloud computing by harnessing new commercially available computing and storage resources, and by deploying cloud technology on grid-enabled resources to improve the management and reliability of those resources via the virtualization layer.

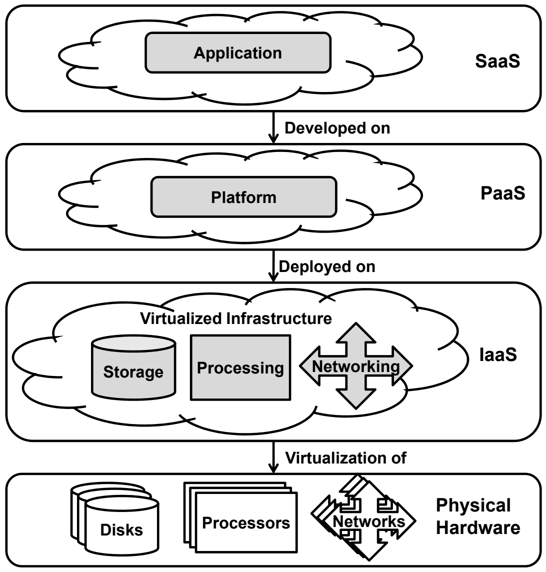


Figure 1. Three Forms of Cloud Computing

The more advanced state of interface standardization within grid technology allows some degree of choice between various software and hardware systems. Cloud computing still lacks any substantive standards or possibilities for interoperation [2], [4].

The GCM standards, on which the presented testing framework is based on, was originally developed for grid environments. However, it can be extended with clouds. Therefore, the testing framework presented in this article is applicable for grid computing and IaaS cloud computing infrastructures. The GCM standards are described in detail in Section V-A.

III. Achieving Interoperability

Interoperability is crucial to ensure delivery of services across systems from different vendors. To achieve technical interoperability [10], different types according the system interoperation can be identified. This section identifies these types and describes approaches for achieving technical interoperability.

A. *Types of Technical Interoperability*

Depending on the view on distributed systems, such as grid or cloud computing environments, technical or functional interoperability can be interpreted differently. In general, three different types of technical interoperability including interoperability within an infrastructure, between the same form of infrastructures, and between different forms of infrastructures can be distinguished as depicted in Figure 2 [3].

Interoperability within an infrastructure means that the services provided by an infrastructure or entities using and implementing them are able to communicate by well defined interfaces (Figure 2–1a). This means that the services within a specific infrastructure are able to interoperate through common, standardized (or otherwise agreed) interfaces inside the infrastructure. A practical example is the requirement to utilize two different components such as a billing and a monitoring service implemented by different vendors that need to communicate within one infrastructure.

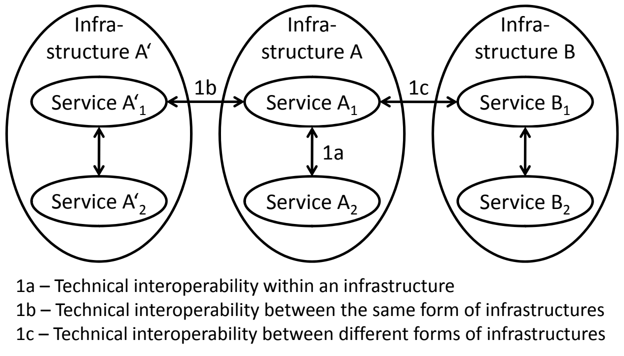

Infra-structure A'
Infra-structure A
Infra-structure B
Service A'1
Service A'2
Service A1
Service A2
Service B1
Service B2
1a
1b
1c
1a – Technical interoperability within an infrastructure
1b – Technical interoperability between the same form of infrastructures
1c – Technical interoperability between different forms of infrastructures

Figure 2. Types of Technical Interoperability

Interoperability between different infrastructures is usually located at user domain level, i.e., interoperability between end users (Figure 2–1b). An infrastructure A and an infrastructure A' need to be able to communicate and exchange data through one or more standardized interfaces. More specifically, the services provided by infrastructure A understand the services provided by infrastructure A'. For example, a service is able to use an execution service of another infrastructure to reduce computing time. However, this also involves interoperability of other services such as authentication and authorization.

Another type of technical interoperability is interoperability of an infrastructure A with an infrastructure of another form B (Figure 2–1c). Despite other considerations to apply this type, it needs to be determined if the services that need to interoperate for certain functionalities are provided by both infrastructures. The infrastructure should be able to interact in order to exchange information and data, or provide access to resources. For example, a grid system can be extended with storage offered by a cloud computing environment.

Within this paper, we consider technical but also syntactical interoperability between the same form of infrastructures (e.g., grid–grid, cloud–cloud), and between different forms of infrastructures (e.g., cloud–grid).

B. *Approaches on Achieving Interoperability*

Several approaches to achieve interoperability between computing and storage infrastructures exist. In general, these approaches are classified in gateways and standardized interfaces [3].

A gateway contains several translators and adapters as depicted in Figure 3. The translator transforms data from a

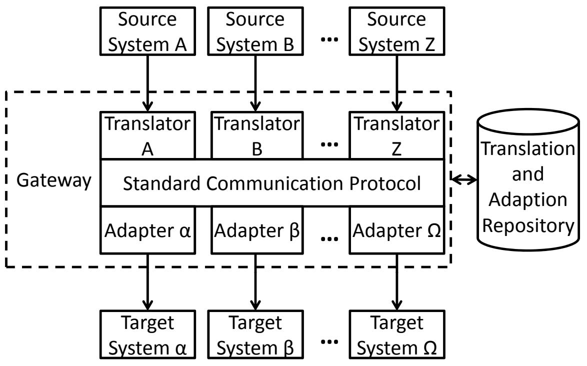


Figure 3. Gateway Approach for Achieving Interoperability

source system into a common and agreed representation, e.g., an *eXtensible Markup Language* (XML) scheme to allow systems using different protocols to be connected to the gateway. The adapter takes this translation and converts it to a specific protocol that is used by the target system. The adapter communicates the translated information to the target system. If the target system replies, it takes the role of a source system. Note that in a one-to-one scenario, it is possible to translate and adapt directly into the required protocol of the target or source system instead of into an agreed intermediate representation. The data of the involved protocols and the translation schemes can be stored in a translation and adaptation repository that can be accessed for translating purposes. Figure 3 shows a specific many-to-many scenario where the gateway resides independent from the involved systems, e.g., in the network. In the one-to-one scenario, the translator and the adapter can reside at the respective system side. Therefore, they do not consolidate a gateway.

Gateways should be considered as interim solutions, as they do not scale well. If the number of systems increases, the gateway performance decreases. It is an expensive approach, because for each protocol, a translator and an adapter need to be developed and integrated. Therefore, gateway solutions are not viable in ad-hoc scenarios or emergency cases.

The long-term approach to address interoperability is the use of open and standardized interfaces. The interfaces that need standardization can evolve from the gateway deployment since the mapping to different infrastructures has already been identified. However, the drawback of this approach is that an agreement on a common set of standard interfaces that also meet production system requirements is very time consuming. However, standardization can enable interoperability in a multi-vendor, multi-network, and multi-service environment without scalability problems as in the gateway approach. Standards need to be engineered for interoperability as described in the next section.

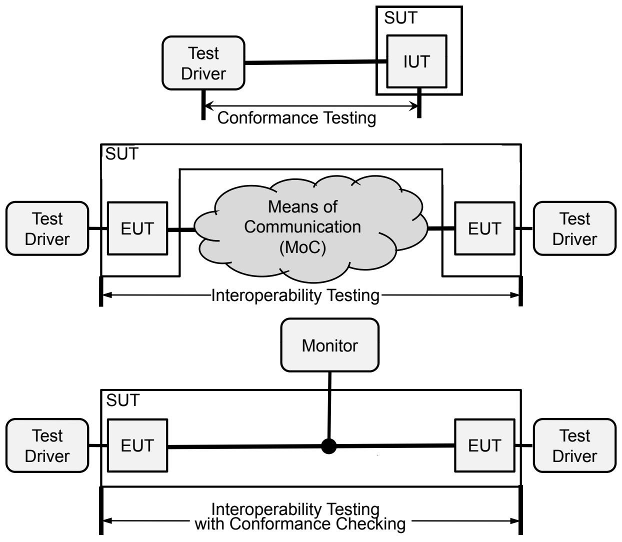


Figure 4. Three approaches to testing in standardization

IV. Conformance vs. Interoperability Testing

Interoperability testing demonstrates that implementations provide end-to-end functionality as described or implied by a specification. In this section, the differences and commonalities among conformance and interoperability testing are described.

Conformance testing is generally used to check that an implementation follows the requirements stated in a specification whereas interoperability testing checks that equipments from different vendors provide an end-to-end service or functionality. In the following, we consider such a specification to be standardized and published by an organization like *Open Grid Forum* (OGF), *World Wide Web Consortium* (W3C) or ETSI.

At ETSI, conformance testing, interoperability testing, or interoperability testing with conformance checking [11] are formally used to test implementations of standards. The three approaches are illustrated in Figure 4. In the following, each approach is considered with its benefits and limitations.

In conformance testing, one *Implementation Under Test* (IUT) is tested with functional black-box tests. Hereby, it is checked if the IUT is conform to a standard. The IUT is embedded by the SUT, which is a testing environment that also includes parts that are required by the IUT to provide its service or functionality to the user. Usually, the development and implementation of sometimes sophisticated testing tools based, e.g., on the *Testing and Test Control Notation* (TTCN-3) [12] is required by conformance testing. Such tools support the simulation of the environment, which is needed for a proper execution of the IUT. However, even if the IUT passes the conformance tests, it does not automatically prove that the IUT is interoperable with other systems implementing the same standard. Standards need to be engineered for interoperability, because they may contain implementation options and leave space for interpreting requirement specifications, which can lead to interoperability problems.

End-to-end functionality specified or implied by a standard between two or more *Equipment Under Test*s (EUTs) is checked with interoperability testing. Each EUT corresponds to a complete system that can consist of several soft- and hardware components. EUTs interoperate via an abstract *Means of Communication* (MoC). It is generally assumed that the communication services used between EUTs are compliant to underlying standards. Interoperability testing is usually driven manually because of the proprietary nature of end user interfaces and does per se not require any testing tool support. However, from our experiences most interoperability problems in practice are often caused by incorrect or non compliant use of communication interfaces.

To address this problem, ETSI endorses a form of interoperability testing that includes conformance checking, i.e., a hybrid of the two testing approaches. This approach extends end-to-end interoperability testing with the monitoring of the communication among the EUTs. Monitors are used to check the conformance of the EUTs with the relevant protocol specifications within the SUT during the interoperability test. The ETSI experience with applying this hybrid approach during interoperability events [13] is that this approach provides valuable feedback to standardization. Even if end-to-end interoperability has been observed, EUTs did not communicate in a number of cases according to underlying standards. Although this approach is not a replacement for conformance testing, it offers an economic alternative to gain insights about the conformance of equipment participating in an interoperability test to a standard.

V. Testing framework

In this section, we present a testing framework for resource reservation and application deployment onto grid and cloud infrastructures, which is based on the generic ETSI interoperability testing methodology. The framework is based on the ETSI GCM standards [5], [6], which provide a starting point for the extraction of testable requirements for an application deployment. However, the consolidating nature of the GCM standards allows the application of the presented framework outside of the specific context of GCM on other grid and cloud infrastructures. The presented testing framework includes interoperability test configurations and test descriptions, which provide a basis for test specifications that can be applied in the context of an interoperability event. The presented testing framework can be used to assess interoperability within and between grid and cloud infrastructures.

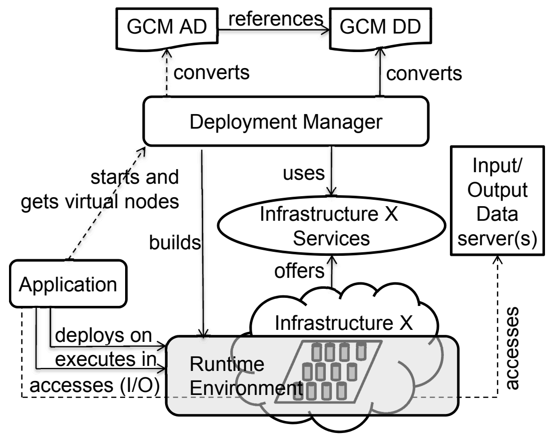


Figure 5. GCM Architecture

A. ETSI Grid Component Model

For users of grid or cloud communities, a provision of common interfaces for the allocation of resources for application deployment in different infrastructures becomes a crucial requirement, since the users wish to access multiple resources of several infrastructures simultaneously and in the most cost saving way. An approach towards such an interface is described in the ETSI GCM standards. The main objective of GCM is the creation of a uniform interface for allocating resources for applications whereas resources may be provided across different grid and cloud infrastructures. The GCM is a gateway approach with a standardized communication protocol based on XML descriptors. The XML descriptors, i.e., in GCM the *Deployment Descriptor* (DD) and the *Application Descriptor* (AD) specify resource information of involved infrastructures in a standardized way.

The content and concepts used in the GCM DD have been derived by abstracting different proprietary interfaces offered by commercial products in the grid, cloud, and cluster computing domains. The key aspect of the GCM specification is the mapping of this abstract interface to different proprietary interfaces of these systems as well as interfaces standardized for this purpose outside of ETSI, e.g., *Open Grid Service Architecture-Basic Execution Service* (OGSA-BES) [14]. Figure 5 shows a generic GCM architecture, which focuses on the GCM AD and DD. It also introduces deployment manager and infrastructure entities to illustrate the likely separation of GCM descriptor processing and provision of the actual resource. Here, the user is assumed to provide a (test) application, a DD XML file, as well as optionally an AD XML file.

The GCM DD describes the resources that can be requested for the deployment of an application on one or more infrastructures. It is converted by the deployment manager into the invocation of specific infrastructure services or commands to reserve resources from the specified infras-

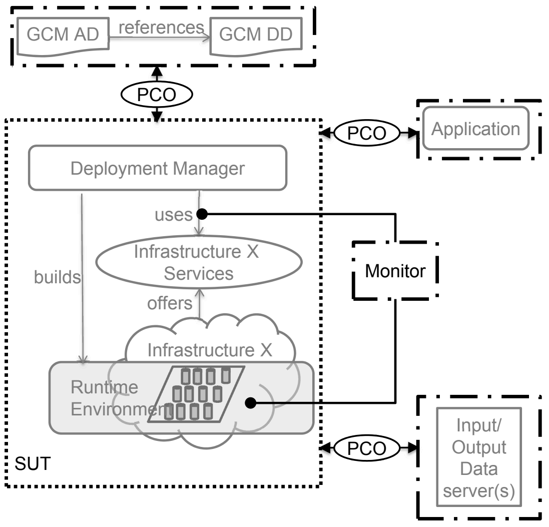


Figure 6. A test architecture for GCM-based deployment

tructure(s). The GCM differentiates between infrastructures with direct access to their computing resource - as in the case of a cloud computing system or a set of desktop computers - and indirect access by using a job scheduler - as in the case of a cluster, or a grid middleware. More information and examples of different types of infrastructures can be found in [3].

For the application deployment, a GCM AD specifies the mapping of virtual nodes to real resources as well as the location of input and output data server(s). If a GCM AD is provided, it is used to establish the runtime environment, which is required for application execution.

B. Test architecture

A test architecture for GCM-based application deployment, which follows the concepts defined in [11], [15] is shown in Figure 6. The SUT consists of the deployment manager and at least one infrastructure. The different types of entities that compose the means of testing handle the provision of the GCM DD and AD files to the deployment manager. These entities associated with the infrastructure to be tested evaluate responses from the deployment manager, and analyze the output produced by the application via their *Point of Control and Observation* (PCO). In addition, the processes that run on each infrastructure as well as their interface(s) to the deployment manager and the input/output server(s) are monitored during tests execution. The monitors are *Points of Observation* (PoOs).

The presented testing architecture can also be used to access other standards related to the deployment and execution of applications on grid or cloud infrastructures, e.g., an OGSA-BES web service based interface between the deployment manager and the infrastructure.

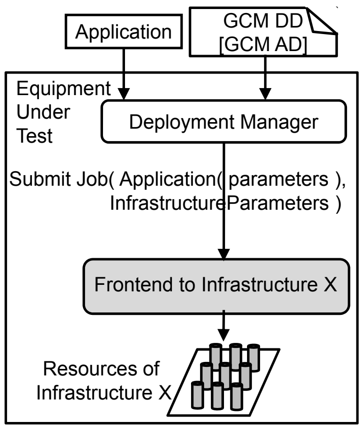

Application
GCM DD [GCM AD]
Equipment Under Test
Deployment Manager
Submit Job(Application(parameters), InfrastructureParameters)
Frontend to Infrastructure X
Resources of Infrastructure X

Figure 7. Single infrastructure

C. Test configurations

Test configurations are a refinement of the test architecture. They specify structural aspects of a test and define in detail all equipments participating in a test as well as their communication. Test configurations are then referenced during the specification of tests, which mainly specify behavioral aspects. In this section, we introduce GCM test configurations [16].

1) Single infrastructure: In the test configuration "Single infrastructure", which is depicted in Figure 7, the EUT contains a single infrastructure and the deployment manager. Access to the deployment manager, the infrastructure, the application, the GCM DD, and the GCM AD are available from one single physical machine. The purpose of this test configuration is to keep the complexity low to allow basic testing with minimal effort to establish the test configuration. The user uses the deployment manager to load the GCM DD and in case the test application is a GCM application, also the GCM AD as input. The user is logged locally into the infrastructure to establish the GCM runtime environment and submit jobs related to the application and the infrastructure. If an infrastructure provides indirect access to its resources, e.g., a grid system, a frontend is used to access its resources.

2) Single infrastructure with a bridge: The test configuration "Single infrastructure with a bridge" depicted in Figure 8 has two EUTs, whereas EUT A contains a deployment manager, which is connected via a bridge to EUT B, which contains a single infrastructure. In contrast to the test configuration presented in the previous clause, access to the deployment manager, the infrastructure, the test application to be executed, the GCM DD, and the GCM AD are distributed across two different physical machines. The user is connected remotely to the infrastructure in order to establish the GCM runtime environment and to submit jobs related to the application from the remote machine. This test configuration can be extended with several infrastructures, which are then mapped to EUTs as described in the next

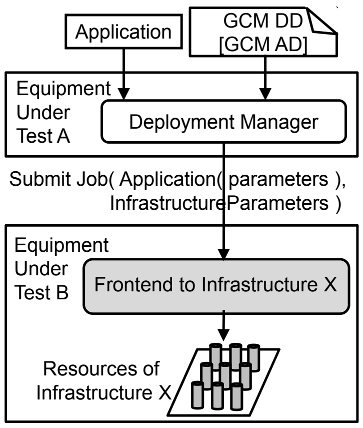

Application
GCM DD [GCM AD]
Equipment Under Test A
Deployment Manager
Submit Job(Application(parameters), InfrastructureParameters)
Equipment Under Test B
Frontend to Infrastructure X
Resources of Infrastructure X

Figure 8. Single infrastructure with a bridge

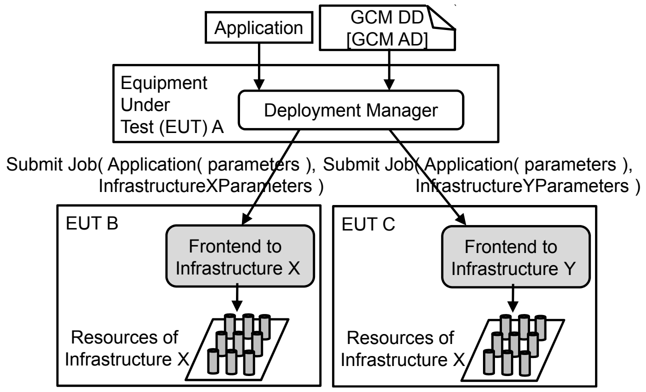

Application
GCM DD [GCM AD]
Equipment Under Test (EUT) A
Deployment Manager
Submit Job(Application(parameters), InfrastructureXParameters)
Submit Job(Application(parameters), InfrastructureYParameters)
EUT B
Frontend to Infrastructure X
Resources of Infrastructure X
EUT C
Frontend to Infrastructure Y
Resources of Infrastructure X

Figure 9. Two infrastructures and bridges

clause.

3) Two infrastructures and bridges: This test configuration is depicted in Figure 9 and extends the test configuration described in the previous clause with a second infrastructure. This test configuration has three EUTs, whereas EUT A contains the deployment manager, EUT B contains the infrastructure X, and EUT C contains the infrastructure Y. Since the deployment manager controls both infrastructures at the same time, it has to be connected to each infrastructure via a bridge.

4) Single infrastructure with a bridge and I/O servers: This test configuration is depicted in Figure 10 and extends the test configuration described in clause V-C2 with input and output data servers. The application can access the input/output data servers from the infrastructure.

D. Compliance Levels

The main purpose of the test framework is the assessment of the standardized GCM AD and DD. The general test objective is to check that applications can be deployed and executed on a given infrastructure based on the information provided in GCM AD and DD. An infrastructure can either provide direct or indirect resource access. To access an

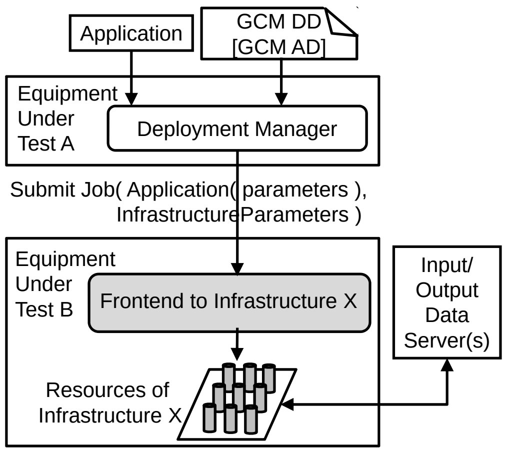

Application
GCM DD [GCM AD]
Equipment Under Test A
Deployment Manager
Submit Job(Application(parameters), InfrastructureParameters)
Equipment Under Test B
Frontend to Infrastructure X
Input/ Output Data Server(s)
Resources of Infrastructure X

Figure 10. Single infrastructure with a bridge and I/O servers

infrastructure, its protocol need to be followed as specified in the GCM standard [5]. For a classification of functionalities that are provided by a SUT, we define compliance levels as follows:

Compliance by the infrastructure:

1) An infrastructure does not support properties described in GCM AD and DD.
2) An infrastructure supports properties described in GCM AD and DD but are converted in a manual manner.
3) An infrastructure supports properties described in GCM AD and DD and are converted in an automated manner.

Compliance by the deployment manager:

1) Support of multiple infrastructures fulfilling infrastructure compliance level 2.
2) Support of multiple infrastructures where at least one of them fulfills infrastructure compliance level 3 and the others infrastructure compliance level 2 (at least one).
3) Support of multiple infrastructures fulfilling infrastructure compliance level 3.

E. Test purpose specification

The first step in the development of ETSI test specifications is to analyze the base standard and extract testable requirements that are used to specify test purpose. A test purpose specifies if and how a catalogued requirement can be assessed in the context of a given test architecture, i.e., in the form of pre-conditions that are relevant to the requirement, and (a) stimulus and response pair(s). Each test purpose includes at least one reference to the clause in a specification where the requirement to be assessed is described. It should have a unique identifier reflecting its place in the test suite structure.

The GCM standard defines the specification of deployment information and not an interface for the deployment.

TP ID:	TP_GCM_DD_DA_PA_001
Clause Ref:	ETSI TS 102 827 V1.1.1 clause 7.1
Configuration:	Single infrastructure or single infrastructure with a bridge
Summary:	Ensure that an infrastructure with direct resource access provides a single processor as specified in the GCM DD

Figure 11. Test purpose "Single processor with direct resource access"

TP ID:	TP_GCM_AD_VN_001
Clause Ref:	ETSI TS 102 828 V2.1.1 clause 5.2.2
Configuration:	Single infrastructure or single infrastructure with a bridge
Summary:	Ensure that a specific capacity of a virtual node (VN) is enforced as specified in the GCM AD

Figure 12. Test purpose "Specific capacity of a single virtual node"

Therefore, the specification of test purposes for GCM descriptors is not a trivial task. In the case of GCM DD, the primary source of testable requirements is general information associated with resources, such as the number of offered processors or the number of threads per processor available for execution. The secondary source of testable requirements includes parameters that are common to a number of standardized GCM mappings to different infrastructures, e.g., wall time or maximum memory. However, these might be not be supported by each infrastructure. Therefore, a test purpose should not be specific to a single mapping. A third source for additional test purposes includes variations of the requirements mentioned above based on different resource access methods, i.e., direct versus indirect as well as local versus remote access. In the presented testing framework, each test purpose is dedicated to one aspect of a specific requirement or concept defined in the GCM standard.

In Figure 11, an exemplified test purpose for GCM DD is depicted. In this case, the support of the direct resource access is a precondition and a GCM DD with a single processor reservation is the stimulus. The success of the application execution determines the success of the resource reservation.

A test purpose for GCM AD is exemplified in Figure 12. For this test, the support of the GCM AD is required. A GCM AD with a virtual node reservation is the stimulus. The test was successful if the test application is able to allocate the capacity of a virtual node as specified in the GCM AD.

In the development of GCM AD test purposes, (re)assessing of GCM DD information should be avoided. For example, the test purposes for GCM AD should be applicable independently from the method the resources of an infrastructure are accessed (direct or indirect). This means that these test purposes focus on information and concepts specified in the GCM AD. Example source for test purposes is the handling of virtual nodes and input/output data location.

F. Specification of interoperability test descriptions

A test description is a detailed but informal specification of the pre-conditions and test steps needed to cover one or potentially more given test purposes. A test description shall contain the following information:

- **Identifier**: A unique identifier that relates a test to its group and sub-group.
- **Summary**: A unique description of the test purposes covered by this test.
- **Configuration**: A reference to all the equipments required for the execution of this test as well as their connection.
- **Specification References**: One or more references to clauses in the standard for which the test purposes have been specified
- **Test application**: A reference to the test application, which is required to execute this test.
- **Pre-test conditions**: A list of all conditions that have to be fulfilled prior to the execution to a test. These conditions should identify the features that are required to be supported by participating equipment to be able to execute this test, requirements on GCM descriptors, as well as requirements on the parameterization of the test application.
- **Test sequence**: A test sequence is written in terms of external actors and their ability to interact and observe the services provided by the infrastructure, i.e., end-to-end behavior. Based on its success, a test verdict reflecting the interoperability of all EUTs in a test is derived.

The test description can also include a list of checks that should be performed when monitoring the EUT communication on standardized interfaces during the end-to-end test. In the case of GCM testing, this option is not directly relevant since the GCM standard does not intentionally define the interfaces between a deployment manager and infrastructures. However, checks can be formulated if an infrastructure implements interfaces standardized for resource reservation and application execution by other standardization organization, e.g., OGF or *Distributed Management Task Force, Inc.* (DMTF).

An exemplified test description for the GCM DD test purpose shown in Figure 11 is depicted in Figure 13. This test description details a test to check if an infrastructure with direct resource access provides a single processor as specified in the GCM DD. A complete list of the test descriptions can be found in [16].

G. Test applications

For the assessment of the success and validity of each application deployment, a test application is executed on all involved infrastructures. The purpose behind these applications is not to perform complex, real world, computational

Interoperability Test Description		
Identifier:	TD_GCM_DD_DA_PA_001	
Summary:	Ensure that an infrastructure with direct resource access provides a single processor as specified in the GCM DD	
Configuration:	Single Infrastructure or single Infrastructure with a bridge	
Specification References:	GCM DD clause 7.1	
Test Application:	Single process batch job	
Pre-test conditions:	• Infrastructure provides direct resource access • GCM DD contains a direct group description with `hostList` containing one host and host description with `hostCapacity=1` for the infrastructure • Infrastructure has a processor available for use	
Test Sequence:	**Step**	**Description**
	1	User loads the GCM DD and starts the test application on the infrastructure using the deployment manager
	2	Verify that the infrastructure has created and executed the process
	3	Verify that returned application output is correct

Figure 13. Test description "Single processor with direct resource access"

tasks but to produce output that allows determining the real usage of resources and the behavior relevant to a test purpose covered by a test. The test application is parameterizable to allow its reuse across multiple tests.

We determined four different kind of test applications: single process batch job, parallel job, virtual node GCM application, and data manipulation GCM application [16].

The single process batch job starts a single process on a single processor and consumes CPU and memory for a given amount of time. The application's behavior including its execution time, the amount of memory to allocate, and the number of threads can be controlled by parameters. The application prints all information required to determine if a test execution has succeeded or failed either to the standard output or a file. This includes the application start time, the value of each parameter, and the identifier of the application. With this test application, resource deployment and usages can be evaluated.

The parallel job starts a job that uses multiple processes. Each process is mapped to a single processor. The multiple processor application consists of one master process and multiple worker processes. The worker processes communicate with the master process so that the master process receives notifications from all worker processes. A notification should include the host name where the worker process runs and a timestamp. The number of worker processes to be created by the parallel application should be parameterizable. By default, the master process starts up as many worker processes as processors are available, i.e., one node less than specified in the GCM DD. That means that a parallel application requests all available resources. The parallel job prints all the information required to determine if a test execution has succeeded or failed either to the standard output or a file.

The virtual node GCM application starts a deployment as specified in the GCM AD and DD. Once the deployment has been performed, it prints the information provided by each virtual node either to the standard output or a file. For each virtual node, the virtual node name, current number of nodes, and the information about each node used is required.

The data manipulation GCM application starts a deployment as specified in the GCM AD and DD. It deploys a worker on each available node. Each worker reads the same input file from the remote or local input location as specified in the GCM AD. It creates a file with the same content as the input file into the remote or local output location as specified in the GCM AD. Workers should avoid file name conflicts and collisions in the output directory.

H. *Test selection and execution*

To determine the applicability of a test, all pre-conditions need to be evaluated. To speed up this process, an *Implementation Conformance Statement* (ICS) should be established to allow infrastructure providers to specify supported features prior to a test execution and support automatic test selection. A test should be selected for execution if all of its pre-conditions have been ensured. Common types of pre-conditions in the GCM tests include constraints on:

- the GCM DD and/or AD specifications,
- the infrastructure relating to the type of resource access, features that need to be supported by the EUTs, and available amount of resources,
- and the test application parameterization.

A test should not be selected and recorded as being not applicable if one of its pre-conditions is not met by one (or more) equipment part of the SUT.

A collection and specification of *Protocol Implementation Extra Information for Testing* (PIXIT) can be used to capture infrastructure specific aspects of a GCM DD such as the access details to an infrastructure and resource identifiers, and used to significantly speed up the execution of tests. The developer of an IUT/EUT states the PIXIT that includes information according the IUT/EUT and its testing environment to enable runs of an appropriate test suite against the IUT/EUT [17].

Each grid and cloud infrastructure will be assessed under the same conditions based on a standardized ETSI test specification. Applicable tests are executed by uploading a test specific application, providing infrastructure and deployment information, e.g. via GCM descriptors, and observing the execution of the application as specified in the test specification.

VI. Example test session

This section describes how the tests of the framework can be applied. For this, we apply the test configuration "Two infrastructures and bridges" more specifically as depicted in Figure 14. EUT B contains the grid middleware Globus Toolkit [18] whereas EUT C includes the cloud computing

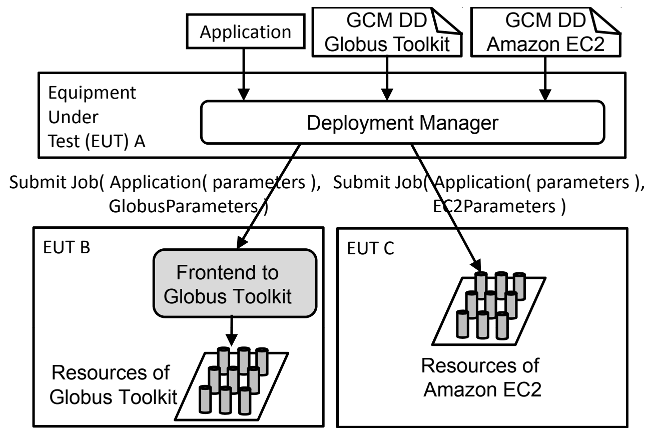

Application
GCM DD Globus Toolkit
GCM DD Amazon EC2
Equipment Under Test (EUT) A
Deployment Manager
Submit Job(Application(parameters), GlobusParameters)
Submit Job(Application(parameters), EC2Parameters)
EUT B
Frontend to Globus Toolkit
Resources of Globus Toolkit
EUT C
Resources of Amazon EC2

Figure 14. Test configuration "Two infrastructures and bridges" exemplified by Globus Toolkit and Amazon *Elastic Compute Cloud* (EC2)

system Amazon *Elastic Compute Cloud* (EC2) [19], which is an IaaS. For each infrastructure, a GCM DD is needed.

The attributes for describing a DD for Globus Toolkit have already been specified in the GCM standard [5]. The DD used in this test session is depicted in Listing 1. Globus Toolkit is contacted via a *Secure SHell* (SSH)-bridge in order to access the Globus Toolkit frontend. An UNIX-based operating system runs on each computing node whereas each contains four processors, which are represented by the element `hostCapacity` of the attribute `host`. Since Globus Toolkit is an infrastructure with indirect resource access, the total number of available processors is not specified.

A DD for the Amazon EC2 is depicted in Listing 2. A scheme for this infrastructure has not been specified yet, but to experience the application of Amazon EC2 and GCM, this example DD has been developed. In case of its successful deployment, it gives a base for an extension of the GCM standard by the specification of a GCM DD scheme for Amazon EC2. Since Amazon EC2 is an infrastructure with direct resource access, the number of included computing node needs to be specified. In our example, the Amazon EC2 contains ten computing nodes, which are based on a Windows operating system. The infrastructure is accessed via an SSH-bridge. Both presented DDs can be merged into one DD file.

In this test session, we exemplify the test specified in the test description depicted in Figure 15. It will be checked if both infrastructures provide multiple processors for a parallel application. Therefore, the parallel application allocates more than one processor in each infrastructure. The execution of the application will be logged in order to evaluate the result of the test. The Amazon EC2 infrastructure includes ten nodes as described in the DD and the number of nodes in the Globus Toolkit cannot be determined from its DD. Therefore, the parallel test application needs to start

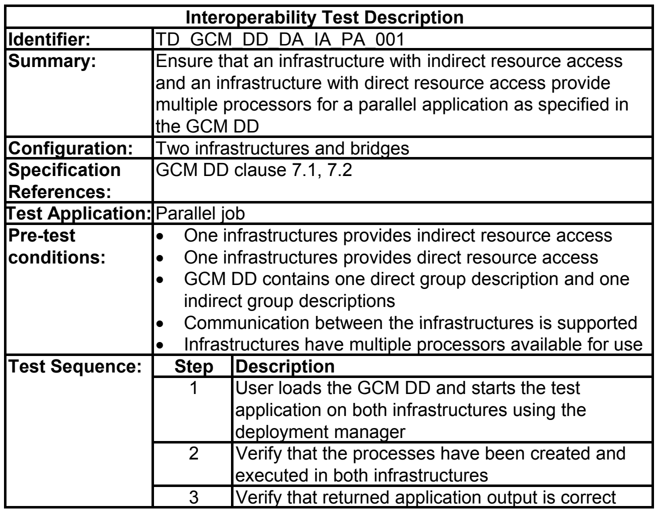
```
<?xml version="1.0" encoding="UTF−8"?>
<GCMDeployment xmlns="urn:gcm:deployment:1.0"
  xmlns:xsi="http: // www.w3.org/2001/XMLSchema−instance"
  xsi:schemaLocation="urn:gcm:deployment:1.0
    http: // etsi .org/schemas/GCMDDSchemas/extensionSchemas.xsd ">
  <environment>
    <javaPropertyVariable name="user.home" />
  </environment>
  <resources>
    <bridge refid="globusGateway" />
    <group refid="globusGrid">
      <host refid="ComputeNodeUnix" />
    </group>
  </resources>
  <infrastructure>
    <hosts>
      <host id="ComputeNodeUnix" os="unix" hostCapacity="4">
        <homeDirectory base="root" relpath="${user.home}" />
      </host>
    </hosts>
    <groups>
      <globusGroup id="globusGrid"
        hostname="globus.grid.local"
        bookedNodesAccess="ssh"
        queue="free">
        <maxTime>5</maxTime>
        <stdout>./output</stdout>
        <stderr>./error</stderr>
      </globusGroup>
    </groups>
    <bridges>
      <sshBridge id="globusGateway"
        hostname="grid.informatik.uni−goettingen.de"
        username="globususer" />
    </bridges>
  </infrastructure>
</GCMDeployment>
```

Listing 1. GCM DD for Globus Toolkit

```
<GCMDeployment>
  <environment>
    <javaPropertyVariable name="user.home" />
  </environment>
  <resources>
    <bridge refid="amazonCloudGateway" />
    <group refid="amazonCloud">
      <host refid="ComputeNodeWindows" />
    </group>
  </resources>
  <infrastructure>
    <hosts>
      <host id="ComputeNodeWindows"
        os="windows" hostCapacity="1">
        <homeDirectory base="administrator" relpath="${user.home}" />
      </host>
    </hosts>
    <groups>
      <amazonCloudGroup id="amazonCloud"
        hostList="node−[01−10]">
      </amazonCloudGroup>
    </groups>
    <bridges>
      <sshBridge id="amazonGateway"
        hostname="aws.amazon.com"
        username="amazonuser" />
    </bridges>
  </infrastructure>
</GCMDeployment>
```

Listing 2. GCM DD for Amazon EC2

ten processes on the Amazon EC2 system and secondly four processes in the Globus Toolkit environment. If all



Interoperability Test Description		
Identifier:	TD_GCM_DD_DA_IA_PA_001	
Summary:	Ensure that an infrastructure with indirect resource access and an infrastructure with direct resource access provide multiple processors for a parallel application as specified in the GCM DD	
Configuration:	Two infrastructures and bridges	
Specification References:	GCM DD clause 7.1, 7.2	
Test Application:	Parallel job	
Pre-test conditions:	• One infrastructures provides indirect resource access • One infrastructures provides direct resource access • GCM DD contains one direct group description and one indirect group descriptions • Communication between the infrastructures is supported • Infrastructures have multiple processors available for use	
Test Sequence:	**Step**	**Description**
	1	User loads the GCM DD and starts the test application on both infrastructures using the deployment manager
	2	Verify that the processes have been created and executed in both infrastructures
	3	Verify that returned application output is correct

Figure 15. Test description "Multiple processors in infrastructures with indirect and direct resource access"

the processes have been started successfully and if the test application writes its output as expected, the test can be evaluated as successful.

VII. Interoperability Event

At ETSI, the validation of interoperability test specifications usually takes place in testing events. Such testing events also provide opportunities for system vendors of the technology under test to assess and demonstrate their interoperability with systems from other vendors. For the GCM testing framework, this validation took place in November 2009 as part of the ETSI Grids, Clouds, and Service Infrastructures event [20]. It provided a unique opportunity for standardization experts, operators, IT service providers, telecom equipment vendors to see available systems running.

A. Application of the Framework in an Interoperability Event

A requirement of the application of the presented testing framework is that a possible SUT needs to include an implementation of the GCM standard. However, the interoperability event included a variety of state-of-the-art systems that implement grid, cloud, cluster, or related technologies, which fit into the idea of the GCM standard. Therefore, as a first step, we compared and executed different state-of-the-art systems to determine and evaluate their similarities and differences. The goal was to feed the result of the demonstrations of the systems of the participated vendors back into the standard to make GCM applicable for these systems. However, this framework is independent of this event and can be applied at other ones as well as in-house.

B. Event Summary

The interoperability event attracted various actors of grid and cloud computing systems from academics, industry, and other standard organizations. In total, six exhibitors demonstrated their grid or cloud environments. The demonstration included resource reservation and application deployment onto different infrastructures. The basis for the evaluation was a questionnaire as well as use case scenarios defined in the ETSI GCM test specification. The questionnaire assessed interfaces for resource reservation and preparation of infrastructure as well as standard support. The use cases included scenarios for infrastructures offering direct and indirect access. Infrastructures were not required to support ETSI GCM standards so that custom-made interfaces were used for application deployment. In addition, capabilities beyond the requirements of the ETSI use cases were shown.

The systems of the vendors who participated in the event mainly implemented a deployment manager for IaaS. However, there was also a deployment manager for PaaS and a cloud management system. All the solutions provided a portal or *Graphical User Interface* (GUI) for resource reservation. For automated use, these have been realized either as RESTful *Web Service* (WS), *Command Line Interface* (CLI), or a Java/XML based API. In general, all the demonstrated systems shield users from complex details of resource reservation. Resource provision and resource requests were handled separately. Handling of data transfer to/from the computing node was mostly done by the application, e.g., using the *Secure Copy Protocol* (SCP) or FTP.

1) Resource Request and Access: Resource request and access were based on different requirements on resources such as computing, storage, or network resources and assessed by the test application referenced in the test description. The resources were selected based on requirements such as performance, *Service Level Agreement* (SLA), application types, or objectives. Fixed or common concepts between the systems could not be identified. The reason may be the application domain specific implementations of the systems. For example, while one system needs a detailed specification of resource requirements, another system only requires a specification of a class defined for resource requirements. In addition, the transparency of the resource management of the systems differed. Therefore, there is a need for an appliance independent hypervisor that manages resources independent of the application.

2) Standard Support: For resource request and access, mainly ETSI GCM, OGF *Distributed Resource Management Application API* (DRMAA), and DMTF *Open Virtualization Format* (OVF) have been implemented. However, most systems use non compliant default configuration, but also allow adaptation to DMTF OVF.

A few basic standards are supported by commercial cloud systems, since cloud computing is an emerging technology and standards are only slowly evolving. Most of the cloud systems provide proprietary RESTful WS and XML based interfaces to resources. This provides a simple basis for further standardization and extensions.

Weak points of existing standards are that they allow too many options such as in the OGF *Job Submission Description Language* (JSDL) or that they require to fix the location of resources. Most desired is a standardized API for virtual machine and resource management.

3) Test automation: The presented testing framework has been mainly developed for the application in interoperability events. The tested systems can be seen as black boxes connected and accessible only by their interfaces. Automating test executions in such a configuration is challenging since agreed standards are not available for the usage of grid or could infrastructures. Furthermore, in the setting of such events participants are usually known as late as two weeks before the event. However, test specification efforts usually are concluded months before an event. Therefore, and due to time and budget limitations, test execution cannot be automated by the organizer. For participants, this would be an extra cost they are not willing to spend. Therefore, the tests have been conducted manually.

4) Reflections: Key areas for standardization, i.e., clear boundaries of the state-of-the-art systems is an API for the provision of resources and for requests of resources. Open issues are the achievement of portable appliances of the hypervisor, i.e., the management of different virtual machine images and resources. Minor concerns include lack in agreed terminology and the need for a strong common denominator. Cloud standardization needs are considered in detailed in the following section.

C. Identified Needs in Cloud Standardization

In the conducted interoperability event, we identified several standardization needs for cloud computing infrastructures related to interoperability. These needs are related to the forms of cloud computing as described in Section II.

For IaaS clouds, functionalities such as the management of resources, application, and data but also common security (authentication and authorization), billing, and accounting interfaces need to be standardized. A cloud resource management standard should consider interfaces for the deployment of virtual machines including their start, stop, status requests, image format, monitoring, performance measurement, and access. Similar to the resource management, cloud application should be management in a common way. This includes their deployment, start, stop, and status. Cloud data management includes especially their access.

On the PaaS level, the platform uses the standardized interfaces described above. Such cloud platforms should offer further features such as dynamic resource allocation and abstraction of resources through a standardized API. In addition, it should be possible to import and use other PaaS interfaces. The SaaS is then implemented using such a standardized cloud platform API.

According to interoperable grid and cloud infrastructures, an application should be able to use them simultaneously. For this, commonly agreed protocols are required to exchange information and to allow their management. A result would be a cloud/grid broker, which the user accesses to use functionalities of grid and cloud systems. Further consideration on cloud standardization requirement can be found in [21].

VIII. Related work

ETSI developed and published a methodology for interoperability testing [11] and automated interoperability testing [15]. In the latter, guidelines and best practices for automated interoperability testing are presented. This methodology has been applied successfully for the development of interoperability test specifications for various technologies, e.g., IPv6 [22] and *Internet Protocol* (IP) *Multimedia Subsystem* (IMS) [23], and put into practice in ETSI interoperability events [13]. Previously, this approach had not been applied to grid or cloud computing technology.

Several interoperability and standard initiatives for grid and cloud computing systems exit. For cloud systems, these include the OGF *Open Cloud Computing Interface* (OCCI) Working Group [24], the IEEE Standards Association [25], the DMTF *Cloud Management Standards* [26], and the *Open Cloud Consortium* (OCC) [27]. The activities of major cloud standardization initiatives have been summarized in a report by the *International Telecommunication Union* (ITU) [28]. These standardization activities are diverse and each initiative chooses the flavors of cloud computing that fit best to their requirements. This is one reason why the concepts of cloud computing are not fully agreed on. However, no interoperability test specifications for grid and cloud computing systems have been published, yet.

Bernstein et. al. identified areas and technologies of protocols and formats that need to be standardized to allow cloud interoperability [29]. They call this set of protocols and formats Intercloud protocols because they should allow cloud computing interoperability. If this set of protocols will be commonly accepted, the GCM and the interoperability testing framework presented in this paper could be adapted to improve cloud interoperability.

Merzky et. al. present application level interoperability between clouds and grids based on SAGA, a high-level interface for distributed application development [30]. The interoperability is achieved by cloud adapters. These adapters are specific to the Amazon Cloud and the Globus Toolkit.

Interoperability initiatives such as OGF *Grid Interoperability Now* (GIN) and standards bodies in grid computing are described in [4]. OGF interoperability test specifications for grid are rarely available and only for selected standards such as GridFTP. Also, they follow rather ETSI's notion of conformance testing than interoperability testing. Due to our knowledge, an interoperability testing framework for such diverse domain of grid and cloud computing infrastructures has not been published.

IX. Summary and Future Research

In this paper, we discussed differences between grid and cloud computing infrastructures. Furthermore, we presented generic approaches on achieving interoperability of distributed systems and different types of testing including interoperability testing with conformance checking.

Our main contribution is a testing framework around the GCM standard developed by ETSI that is applicable for grid, cloud, and cluster management systems. This framework has been developed by following the ETSI test development process. We described the GCM architecture, an applicable test architecture, developed test configurations, and test applications. Test purpose specification and interoperability test descriptions have been explained. Furthermore, we considered test selection and test execution according to the presented testing framework. The application of the framework has been exemplified by Globus Toolkit and Amazon EC2 as EUTs. As a result from the application of the framework in the ETSI Grids, Clouds, and Service Infrastructures event, we identified needs toward standardized grid and cloud infrastructures.

We believe that the GCM standard is a first step towards the use of resources from different infrastructures simultaneously in a standardized way. Infrastructure adapters and translators can easily be incorporated into the standard. With the presented testing framework, it is possible to enhance and expand the standard to allow a wide adaption of several systems provided by different vendors.

The described testing framework is part of an initiative for standardizing the use of grid and cloud technology in the context of telecommunication at ETSI. We believe that this testing framework is a step towards systematic interoperability testing of grid and cloud computing environments in general [3].

The specification of executable test cases from GCM test descriptions is considered as future work. This step includes the further concretization of GCM test configurations including equipment user and monitor test components as well as the specification of their behavior. We consider to automate the tests following the methodology on automated interoperability testing [15], which is a challenging task. Furthermore, we plan to extend the testing framework to make it applicable for upcoming cloud standards.

Acknowledgment

The work carried out here is co-financed by the EC/EFTA in response to the ECs ICT Standardisation Work Programme.

References

[1] ETSI Technical Committee CLOUD (TC CLOUD), previously TC GRID, [Online; http://portal.etsi.org/cloud fetched on 10-06-11].

[2] “ETSI TR 102 659: GRID;Study of ICT Grid interoperability gaps,” European Telecommunications Standards Institute (ETSI), Sophia-Antipolis, France, 2009.

[3] “ETSI TR 102 766: GRID; ICT Grid Interoperability Testing Framework and survey of existing ICT Grid interoperability solutions,” European Telecommunications Standards Institute (ETSI), Sophia-Antipolis, France, 2009.

[4] T. Rings, G. Caryer, J. Gallop, J. Grabowski, T. Kovacikova, S. Schulz, and I. Stokes-Rees, “Grid and Cloud Computing: Opportunities for Integration with the Next Generation Network,” *Journal of Grid Computing: Special Issue on Grid Interoperability, JOGC*, vol. 7, no. 3, pp. 375 – 393, 2009.

[5] “ETSI TS 102 827: GRID;Grid Component Model (GCM);GCM Interoperability Deployment,” European Telecommunications Standards Institute (ETSI), Sophia-Antipolis, France, 2008.

[6] “ETSI TS 102 828: GRID;Grid Component Model (GCM);GCM Application Description,” European Telecommunications Standards Institute (ETSI), Sophia-Antipolis, France, 2010.

[7] “ETSI TS 102 829: GRID;Grid Component Model (GCM);GCM Fractal Architecture Description Language (ADL),” European Telecommunications Standards Institute (ETSI), Sophia-Antipolis, France, 2009.

[8] “ETSI TR 102 767: GRID;Grid Services and Telecom Networks;Architectural Options,” European Telecommunications Standards Institute (ETSI), Sophia-Antipolis, France, 2009.

[9] T. Rings, J. Grabowski, and S. Schulz., “On the Standardization of a Testing Framework for Application Deployment on Grid and Cloud Infrastructures,” in *Proceedings of the 2nd International Conference on Advances in System Testing and Validation Lifecycle (VALID 2010)*. IEEE Computer Society, 2010, pp. 99–107.

[10] H. van der Veer and A. Wiles, “Achieving Technical Interoperability - the ETSI Approach,” White Paper, European Telecommunications Standards Institute (ETSI), Sophia-Antipolis, France, 2008.

[11] “ETSI ES 202 237: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT);Generic approach to interoperability testing,” European Telecommunications Standards Institute (ETSI), Sophia-Antipolis, France, 2007.

[12] ETSI, “ETSI Standard (ES) 201 873 V3.2.1: The Testing and Test Control Notation version 3; Parts 1-8,” European Telecommunications Standards Institute (ETSI), Sophia-Antipolis, France, also published as ITU-T Recommendation series Z.140, 2007.

[13] ETSI, “Plugtests^{TM}Interop Events,” [Online; http://www.etsi.com/WebSite/OurServices/plugtests/home.aspx fetched on 10-06-11].

[14] I. Foster, A. Grimshaw, P. Lane, W. Lee, M. Morgan, S. Newhouse, S. Pickles, D. Pulsipher, C. Smith, and M. Theimer, "OGSA Basic Execution Service Version 1.0, GFD-R.108," Open Grid Forum, 2008.

[15] "ETSI EG 202 810: Methods for Testing and Specification (MTS);Automated Interoperability Testing;Methodology and Framework," European Telecommunications Standards Institute (ETSI), Sophia-Antipolis, France, 2010.

[16] "ETSI TS 102 811: GRID;Grid Component Model (GCM);Interoperability test specification," European Telecommunications Standards Institute (ETSI), Sophia-Antipolis, France, 2010.

[17] ISO/IEC, "Information Technology – Open Systems Interconnection – Conformance testing methodology and framework," International ISO/IEC multipart standard No. 9646, 1994-1997.

[18] I. Foster, "Globus Toolkit Version 4: Software for Service-Oriented Systems," in *Proceedings of the IFIP International Conference on Network and Parallel Computing (NPC05)*, ser. LNCS, vol. 3779. Springer, 2005.

[19] "Amazon Elastic Compute Cloud (Amazon EC2)," [Online; http://aws.amazon.com/ec2/ fetched on 10-06-11].

[20] ETSI, "Grids, Clouds & Service Infrastructures: Plugtests^{TM} and Workshop," [Online; http://www.etsi.com/plugtests/GRID09/GRID.htm fetched on 10-06-11].

[21] "ETSI TR 102 997: CLOUD;Initial analysis of standardization requirements for Cloud services," European Telecommunications Standards Institute (ETSI), Sophia-Antipolis, France, 2010.

[22] "ETSI TS 102 517: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT): IPv6 Core Protocol;Interoperability Test Suite (ITS)," European Telecommunications Standards Institute (ETSI), Sophia-Antipolis, France, 2008.

[23] "ETSI TS 186 011-2: Technical Committee for IMS Network Testing (INT);IMS NNI Interworking Test Specifications;Part 2: Test descriptions for IMS NNI Interworking," European Telecommunications Standards Institute (ETSI), Sophia-Antipolis, France, 2009.

[24] OGF, "Open Cloud Computing Interface Working Group," [Online; http://forge.ogf.org/sf/projects/occi-wg fetched on 10-06-11].

[25] IEEE Standards Association, "P2301 - Guide for Cloud Portability and Interoperability Profiles (CPIP)," [Online; http://standards.ieee.org/develop/project/2301.html fetched on 10-06-11].

[26] DMTF, "Cloud Management Standards," [Online; http://www.dmtf.org/standards/cloud fetched on 10-06-11].

[27] "Open Cloud Consortium," [Online; http://opencloudconsortium.org/ fetched on 10-06-11].

[28] ITU Telecommunication Standardization Bureau, "Activities in Cloud Computing Standardization - Repository," 2010, [Online; http://www.itu.int/dms_pub/itu-t/oth/49/01/T49010000020002PDFE.pdf fetched on 10-16-11].

[29] D. Bernstein, E. Ludvigson, K. Sankar, S. Diamond, and M. Morrow, "Blueprint for the Intercloud - Protocols and Formats for Cloud Computing Interoperability," in *ICIW*, M. Perry, H. Sasaki, M. Ehmann, G. O. Bellot, and O. Dini, Eds. IEEE Computer Society, 2009, pp. 328–336.

[30] A. Merzky, K. Stamou, and S. Jha, "Application Level Interoperability between Clouds and Grids," in *GPC Workshops*. IEEE Computer Society, 2009, pp. 143–150.