ESB based Services Composition platform for Cloud PaaS

Canh Ngo, Yuri Demchenko, Cees de Laat
System and Network Engineering Group
University of Amsterdam
Amsterdam, The Netherlands
e-mail: { c.t.ngo, y.demchenko, delaat}@uva.nl

Mary Grammatikou
NTUA
Athens, Greece
e-mail: mary@netmode.ntua.gr

Pedro Martínez Juliá
University of Murcia
Murcia, Spain
e-mail: pedromj@um.es

Antonio D. Pérez Morales
RedIRIS, Spain
e-mail: antonio.perez@rediris.es

Diego R. Lopez
Telefonika I+D, Spain
e-mail: diego@fid.es

***Abstract*—Cloud Platform as a Service (PaaS) provides an environment for creating and deploying applications using one of popular development platforms. This poster paper present a practical solution for building a service composition platform based on the Web Services and Enterprise Service Bus (ESB) technologies. The ESB is widely used as a platform for SOA and Web Services based integrated enterprise solutions. However in existing practices ESB design is still based on manual development, configuration and integration. ESB can be considered as a logical choice for creating Cloud PaaS services composition and provisioning platform but needs to support automatic or interactive services composition and deployment. The paper proposes an approach how to make interactive services composition using general ESB functionality and services management framework. The paper refers to how this approach has been implemented and used in the GEMBus (GEANT Multidomain Bus) and GEYSERS Logical Infrastructure Composition Layer (LICL) that can be considered as two complementary solutions. The presented work motivates how the common GEMBus and LICL framework can be used for building effective services composition and provisioning platform for Cloud PaaS offering widely accepted ESB Platform as a Service.**

Keywords-Services Composition; Cloud Platform as a Service; Enterprise Service Bus (ESB); GEMBus (GEANT Multidomain Bus); GEYSERS Logical Infrastructure Composition Layer (LICL)

I. INTRODUCTION

Cloud computing [1, 2] defines three basic service models: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). Cloud PaaS provides an environment for creating and deploying applications using one of popular development platforms.

Customers use PaaS services to deploy applications on controlled, uniform execution environments available through the network. IaaS gives a way to bind hardware, operating systems, storage and network capacity over the Internet. The service delivery model allows the customer to acquire virtualized servers and associated services. We will first discuss how a distributed service-oriented infrastructure can ease the deployment of service instances in the Cloud, and how it can facilitate the usage of Cloud infrastructural services as well.

The paper introduces the Composable Services Architecture (CSA) that provides a basis for flexible integration of component services. The CSA provides a framework for the design and operation of composite services, provisioned on-demand. Since it is based on the virtualisation of component services, which in its own turn is based on the logical abstraction of the (physical) component services and their dynamic composition, it does naturally fit in the cloud distributed virtualization philosophy.

Reference CSA implementation is based on the GÉANT Multi-domain service Bus (GEMBus) [3] that is being developed in the GÉANT3 Project as CSA middleware. CSA also provides a basis for GEYSERS Logical Infrastructure Composition Layer (LICL) [4] that allows for combined network+IT infrastructure services virtualisation.

GEMBus is an interoperability and integration platform that extends the functionality of traditional enterprise-wide service-oriented architectures to a distributed multi-domain environment - therefore enabling them to be located within the Cloud. It acts as enabler for new services that can be deployed in the Cloud using a well-defined API, as well as integrating enterprise services and Cloud based services. In this way, GEMBus intends to provide a middleware platform to support the integration of Cloud-based applications, services and systems.

II. CLOUD AND SOA FOR SERVICES COMPOSITION

There are two main directions in which mutual influence in the evolution of Cloud infrastructures and Service-Oriented Architecture (SOA) can translate into benefits for the maturity and usability of both technologies. First of all, service deployment and operation can greatly benefit from a supporting Cloud infrastructure able to transparently provide elastically and on-demand computational and storage resources. On the other hand, Cloud infrastructure services are essentially service-oriented and therefore suitable to take advantage from supporting services, such as messaging, security, accounting, composition and therefore simplifying their integration into business processes. In any of the above directions, multi-domain issues have to be considered from

the beginning: service deployment in any Cloud infrastructure beyond enterprise limits, as well as access to Cloud interfaces out of those limits require mechanisms spanning several management domains. Other, more complicated use cases like collaborating services supported by different infrastructures, or access to different Cloud providers imply much more complicated settings although they are clear application environments in the short term, if not already required.

III. COMPOSABLE SERVICES ARCHITECTURE (CSA)

Composable Services Architecture (CSA) provides a framework for Cloud based services design, composition, deployment and operation. CSA allows for flexible services integration of existing component services. The CSA infrastructure provides functionalities related to Control and Management planes, allowing the integration of existing distributed applications and provisioning systems, thus simplifying their deployment across network and Cloud infrastructures. The GÉANT Multi-domain service Bus (GEMBus), being developed in the GÉANT3 Project, is the realization of a CSA middleware layer.

CSA provides a framework for the design and operation of distributed composite services provisioned on-demand. It is based on the virtualisation of component services, which in its own turn is based on the logical abstraction of the component services and their dynamic composition. Composition mechanisms are provisioned as CSA infrastructure services.

The following diagram shows the major functional components of the proposed CSA and their interaction. The central part of the architecture is the CSA Middleware (CSA-MW), which supports message exchange through a normalized container that provides seamless access to a set of general infrastructure services supporting reliable and secure delivery and operation of composite services:

- A Service Lifecycle Metadata service (MD SLC) that stores service management metadata and code.
- A Registry service that holds information about service instances.
- Security services that ensure the proper operation of the infrastructure.
- Logging mechanisms able to provide operational information for monitoring and accounting purposes.

It must be noted that both logging and security services can be also provided as component services that can be composed with other services in a regular way.

The Logical Abstraction Layer (LAL) defined by CSA eases service relocation across highly distributed infrastructures that can span different management domains, enabling service developers to simply fit them to satisfy the requirements to make them able to be seamlessly deployed in the Cloud, as shown in "Component Services & Resources" in the diagram above. Composite services offer compatible interfaces through the Service Composition layer, which in a simple case can be provided by standard workflow management systems adapted through the Logical Abstraction Layer.

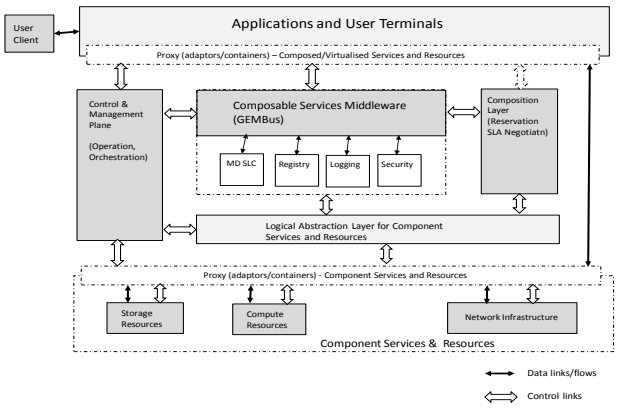


Figure 1. Composable Service Architecture and main functional components.

While this architecture provides a good basis for creating and composing services, making them suitable to be support and make advantage of the dynamical re-configurability associated with cloud infrastructures also requires them to rely on a well-defined Services Lifecycle Management (SLM) model. Most of existing services development and lifecycle management frameworks and definitions are oriented towards rather traditional human-driven services development and composition [5, 6]. Dynamically provisioned and re-configured services will require re-thinking of existing models and proposing new security mechanisms at each stage of the provisioning process.

The proposed service lifecycle includes the following main stages, depicted in the diagram below:

- Service Request. It relies on service metadata and supports SLA negotiation, described in terms of QoS and security requirements.
- Composition/Reservation. It provides support for complex reservation process in potentially multi-domain, multi-provider environment. This stage may require access control and SLA/policy enforcement.
- Deployment. This stage begins after all component resources have been reserved and includes distribution of the common composed service context (including the security context) and binding the reserved resources.
- Operation. This is the main operational stage of the provisioned on demand services.
- Decommissioning. It ensures that all security contexts are terminated, and data are cleaned.

To take advantage of a distributed infrastructure, two additional stages can be initiated from the Operation stage based on the running service state, such as its availability or SLA requirements from its user composite services:

- Re-composition or Modification, allowing incremental infrastructure changes.
- Recovery/Migration, initiated both by the user and the provider. It may also require re-composition/modification

Defining different lifecycle stages allows using different level of service presentation and description at different

stages, and addressing different aspects and characteristics of the provisioned services. However, to ensure integrity of the service lifecycle management, consistent service context management mechanisms should be defined and used during the whole service lifecycle, including the corresponding security mechanisms to protect integrity of the services context. The problem here is that such mechanisms are generically stateful, what imposes problems for a SOA environment, which is defined as generically stateless. The SLMD shown in the diagram is intended to address these issues.

IV. GEMBUS

The GÉANT Multi-domain service Bus (GEMBus), under development in the GÉANT3 Project, is the realization of a CSA middleware layer. GEMBus is a SOA multi-domain middleware, able to support the deployment and composition of services spanning different management domains, applying the federation mechanisms that have come to play a key role in several application areas, like digital identity or distributed computing itself. Federation preserves management independence for each party as long as they obey the (minimum) set of common policies and technological mechanisms that define their interoperation. Metadata constitute the backbone of such federations, as they describe the components provided by each party and their characteristics in what relates to the federated interfaces.

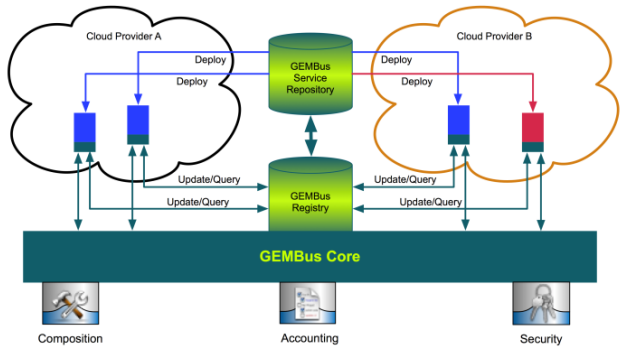


Figure 2. GEMBus as platform for Cloud services integration.

Figure 2 above shows the intended usage of GEMBus in a Cloud infrastructure. A common service registry and a service repository provide the metadata backbone for the federated SOA, together with service code. Service descriptions are updated at the common registry and made known at the participant instances through it. Service instances are deployed within the supporting infrastructure from the repository, allowing for re-composition, modification and migration. The GEMBus core supports the LAL through a common messaging infrastructure plus a few infrastructural services for security, accounting and composition.

The GEMBus core is constituted by those elements that provide the functionality required to maintain the federation infrastructure, allowing the participant SOA frameworks to interoperate in accordance with the principles previously described. The GEMBus core comprises two types of elements, combined to provide the functional elements described below according to the functionalities of the service frameworks connected to GEMBus:

- The core components that form the federation fabric, enforcing its requirements in regard to service definition and location, routing of requests/responses and security. These elements are implemented by specific software elements and by extending and profiling the service frameworks to be connected.
- A set of core services that provide direct support to any service to be deployed in GEMBus, such as the STS or the Workflow Server described below. These core services are invoked by the core elements as part of their functions. They can be called from the code implementing any service deployed in GEMBus. Furthermore, as any other service taking part in the infrastructure, they are suitable to be deployed anywhere, and integrated within composite services.

V. TESTBED FOR ESB BASED PAAS PLATFORM

The proposed solutions and GEMBus/ESB based platform for services composition has been implemented as a Cloud PaaS tested at University of Amsterdam. The testbed provides a facility for testing technologies and developing them as an open Cloud PaaS platform.

Figure 3 shows the testbed structure and implementation details. The lower layer infrastructure uses OpenNebula Virtual Machines (VM) management environment. Each VM runs a Fuse ESB [7] instance that may host one or more services. Services interconnections is realised based on such ESB functional components as Message Broker (based on Apache ActiveMQ) and Message Router (based on Apache Camel). Component services can be deployed in ESB environment using VM's with preinstalled and pre-configured ESB instances. Final services interconnection topology can be created by pre-configuring ESB instances or dynamically changing their configuration after deployment and during run-time, what is supported by ESB functionality.

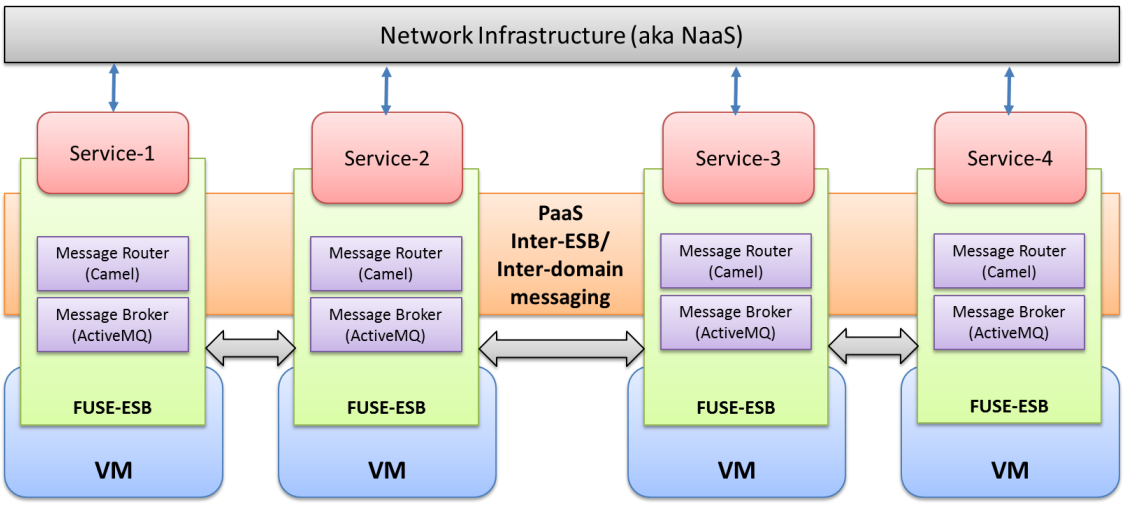
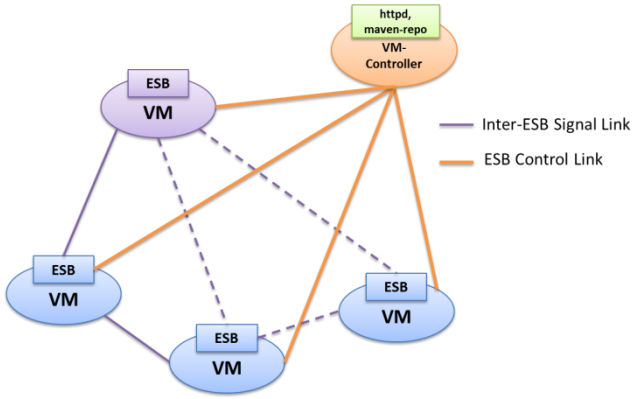


Figure 3. Testbed for GEMBus/ESB based services composition as a Cloud PaaS

Figure 4 provides graphical illustration of the services topology realization using Message Router and network of Brokers. The listing below provides example of Message Broker configuration in bean.xml.




```
<route>
    <from uri="jms:S1_Out"/>
    <to uri="jms:S3_Out"/>
</route>
<route>
    <from uri="jms:S2_Out"/>
    <to uri="jms:S3_Out"/>
</route>
<route>
   <from uri="jms:queue:S3_Out"/>
   <to uri="bean:logger?method=log”/>
</route>
```

Figure 4, Message Router and network of Brokers. Example of bean.xml configuration.

VI. FUTURE DEVELOPMENT

The described above testbed provides initial implementation of the proposed solution for ESB based PaaS platform. Future research and development will include integration with the GEMBus/GEANT3 Composable Services testbed and GEYSERS Infrastructure Services Virtualisation testbed, including extension for dynamically provisioned security services.

The presented research is planned to be contributed to the recently created the Open Grid Forum Research Group on Infrastructure Services On-Demand provisioning (ISOD-RG) [8], where the authors play active role.

ACKNOWLEDGEMENTS

This work is supported by the FP7 EU funded project GEANT3 (FP7-ICT-238875), and the FP7 EU funded Integrated project The Generalised Architecture for Dynamic Infrastructure Services (GEYSERS, FP7-ICT-248657).

REFERENCES

[1] NIST SP 800-145, “A NIST definition of cloud computing”, [online] Available: http://csrc.nist.gov/publications/drafts/ 800-145/Draft-SP-800-145_cloud-definition.pdf

[2] GFD.150 Using Clouds to Provide Grids Higher-Levels of Abstraction and Explicit Support for Usage Modes. [Online]. http://www.ogf.org/documents/GFD.150.pdf

[3] GEANT Project. [Online] http://www.geant.net/pages/home.aspx

[4] Generalised Architecture for Dynamic Infrastructure Services (GEYSERS Project). [Online] http://www.geysers.eu/

[5] Generic Architecture for Cloud Infrastructure as a Service (IaaS) Provisioning Model, Release 1. SNE Techn. Report SNE-UVA-2011-03, 15 April 2011. [Online] http://staff.science.uva.nl/~demch/worksinprogress/sne2011-techreport-2011-03-clouds-iaas-architecture-release1.pdf

[6] Demchenko, Y., J. van der Ham, M. Ghijsen, M. Cristea, V. Yakovenko, C. de Laat, "On-Demand Provisioning of Cloud and Grid based Infrastructure Services for Collaborative Projects and Groups", The 2011 International Conference on Collaboration Technologies and Systems (CTS 2011), May 23-27, 2011, Philadelphia, Pennsylvania, USA

[7] FUSE ESB Platform. [Online]. http://fusesource.com/products/enterprise-servicemix/

[8] Open Grid Forum Research Group on Infrastructure Services On-Demand provisioning (ISOD-RG). [Online]. http://www.ogf.org/gf/event_schedule/index.php?event_id=17