Composable Services Architecture (CSA)

GN3-JRA3-T3 Technical document, Version 0.2

Editor: Yuri Demchenko, UvA

Contributors:
Yuri Demchenko, UvA
Diego Lopez, RedIRIS

1 Introduction

Composable Services Architecture is a part of the general GEMBus (GEANT Multidomain Service Bus) [1] development that is based on the industry adopted Enterprise Services Bus (ESB) [2] which is extended to support dynamically reconfigurable virtualised services in the GEANT3 services infrastructure environment.

(Open) Composable Services Architecture (CSA/OCSA) provides a framework and a foundation for building composable services and corresponding infrastructure to support on-demand services provisioning based on Services Oriented Architecture (SOA) [3].

This document describes the overall architecture for Composable Services and provides a framework for defining functionality and operation of all other GEMBus components.

The document provides also short overview of the related industry standards and discusses GSA basic design principles, in particular, related to Services Oriented Architecture (SOA) [3], Open Service Environment (OSE) concept, as defined by ITU-T Next Generation Network (NGN) [7, 11, 12].

In the following sections, the keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in RFC 2119.

These keywords are thus capitalised when used to unambiguously specify requirements over protocol and application features and behaviour that affect the interoperability and security of implementations. When these words are not capitalised, they are meant in their natural language sense[1].

2 Background and Basic Design Principles

2.1 Compliance with the standard SOA frameworks

CSA and GEMBus are by definition based on, and should support, the basic SOA architectural principles and services interaction models [2, 3-5].

[1] Note. Current version of requirements temporary use only SHOULD word; after discussion requirements will be updated to use different keywords SHALL, SHOULD and MAY

As correctly noted in [5] “SOA is based on business requirements” and “aligns IT and business so that IT systems work the way the business does, helping to ensure that IT produce business value”. From the service design point of view this means that SOA based services and the services design and deployment process should allow and use different levels of abstraction and ensure the whole services delivery lifecycle.

In its evolution and gradual development GEMBus should adopt SOA best practices and comply with the Open Group Services Integration Maturity Model (OSIMM) (see [6] and Appendix B). The OSIMM defines a grid of the 7 maturity level and 7 dimensions that describe provisioned services and SOA related properties. The 7 OSIMM maturity levels include:

(OSIMM1) Silo;

(OSIMM 2) Integrated;

(OSIMM 3) Componentised;

(OSIMM 4) Services,

(OSIMM 5) Composable services;

(OSIMM 6) Virtualised services;

(OSIMM 7) Dynamically re-configurable services.

The 7 dimensions define different presentation layers and aspects of the services such as Business view, Governance and Operations, Methods, Applications, Architecture, Information, Infrastructure and Management.

In Applications dimension the SOA based applications develop/evolve from functional modules to dynamic services/applications assembly. They deal with the different components and building blocks mapped to the above defined maturity levels: modules (OSIMM1), objects (OSIMM2), components (OSIMM3), services (OSIMM4), applications comprised of services (OSIMM5), process integration via services (OSIMM6), and dynamic application assembly (OSIMM7).

2.2 CSA and NGN Open Service Environment (OSE)

Composable Services Architecture (CSA) has been motivated and can be positioned as a framework for implementing the Open Service Environment (OSE) concept, as defined by ITU-T Next Generation Network (NGN), which demonstrates present trend to using service-oriented concepts in modern telecommunication industry [7, 8]. Compatibility with the ITU-T standards will ensure future sustainable CSA development and evolution.

A number of the recent ITU-T recommendations related to the Next Generation Network (NGN) provide a basis for transport/network and Information Technology (IT) convergence based on NGN. The NGN reference model, according to ITU-T Y.2011 Recommendation [8], suggests separation of the transport network and application services and defines them as NGN service stratum and NGN transport stratum consisting of User plane, Control plane and Management plane. The ITU-T Recommendations Y.2012 and Y.2201 specify high level requirements and functional architecture of the NGN Release 1 [9, 11]. The described NGN service architecture implements services and network separation principle and defines functional components of the Transport stratum and Service stratum. The NGN Y.2012 architecture defines also Application Network Interface (ANI) that provides an abstraction of the network capabilities and is used as a channel for applications to access network services and resources.

The NGN convergence service model is defined by ITU-T Recommendation Y.2232 and suggests the major scenario with using Web Services [12]. The NGN Open Service Environment (OSE) defined by ITU-T Recommendation Y.2234 [13] is based on Web Services and actually implements basic SOA principles in defining a services integration model. The definition of the OSE and Web Services convergence model is targeted to provide common framework for both applications developers and provider services developers.

The Y.2234/Y.2201 NGN OSE is required to satisfy such requirements as independence from transport network providers, independence from manufactures, location transparency, network transparency, and

protocol transparency. The OSE should provide the following capabilities to support effective services integration and operation: service coordination; interworking with service creation environment; service discovery; service registration; policy enforcement; and development support. The latter capability actually suggests that OSE should "support the full lifecycle of components, ranging from installation, configuration, administration, publishing, versioning, maintenance and removal.

More detailed information about referenced standards is provided in Appendix A.

2.3 Composable Services Lifecycle Management

The SOA based technologies provide a good basis for creating composable services which, in case of advancing to dynamically re-configurable services, should also rely on the well-defined Service Lifecycle Management (SLM) model. Most of existing service development and lifecycle management frameworks and definitions are oriented on rather traditional human-driven services development and composition [17, 18, 20]. Dynamically provisioned and re-configured services will require re-thinking of existing models and propose new security mechanisms at each stage of the typical provisioning process.

The typical service lifecycle includes the following stages:

1. Service request
2. Design or development
3. Deployment or implementation
4. Operation
5. Retire or disposal.

Defining different lifecycle stages allows using different level of the 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, the consistent service context management mechanisms should be defined and used during the whole service lifecycle, including corresponding security mechanisms to protect integrity of the services context. The problem here is that such mechanisms are generically stateful, what imposes problems for an SOA environment, which is defined as generically stateless.

Appendix C provides additional information about existing standards and recommendations on the general services lifecycle management and security services lifecycle management in particular.

2.4 Architecture Layers

GEMBus as a multidomain communication environment should adopt the layering approach defined in the Web Services Architecture [22, 23, 24] and extend it with additional lower and upper layers to reflect use cases specific for Internet and telecommunications services. This should include the following layers:

Networking Layer: this layer provides a possibility to apply technologies typical for distributed enterprise applications, such as VPN

Transport Layer: this layer may define specific for service communication functionality such as transport layer security (using TLS/SSL protocols), assigning service (types) to specific ports, etc.

Messaging Layer: this layer defines message handling related functionality such as message routing, message format transformation, etc.

Virtualisation Layer (that actually consists of Logical Abstraction Layer and Composition and Orchestration layer): this layer provides functionality to compose services and support/drive their interaction (e.g. with workflows) to ensure application interactions.

Application Layer: this layer represents applications, where the major goal is application related data handling

Security services are applied at multiple layers to ensure consistent security. Management functions are also present at all layers and can be seen as the management plane, similar to the NGN reference model.

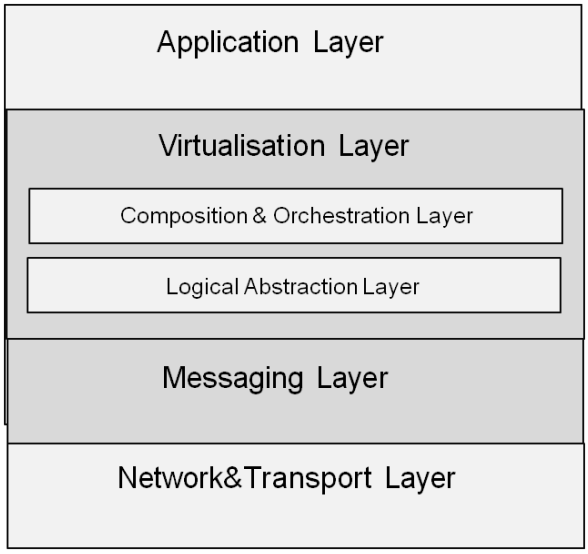

Figure 1. Composable Service Architecture Layers

3 The Proposed Composable Services Architecture

The proposed CSA provides a framework for the design and operation of the composite/complex services provisioned on-demand. It is based on component services virtualisation, which in its own turn is based on the logical abstraction of the component (physical/real) services and their dynamic composition. Composite services may also use the Orchestration service provisioned as a CSA infrastructure service to operate composite service specific workflow.

One of the important components of the proposed architecture is the CSA middleware that should ensure smooth service operation during all stages of the composable services lifecycle.

To be included into the CSA infrastructure, component/physical services need to implement a special adaptation layer that is capable of supporting major CSA provisioning stages, in particular, service identification, service metadata including security context, and provisioning session management.

CSA must define and implement also special adaptation layer interfaces providing the necessary functionality to support dynamically provisioned control and management plain/functionality.

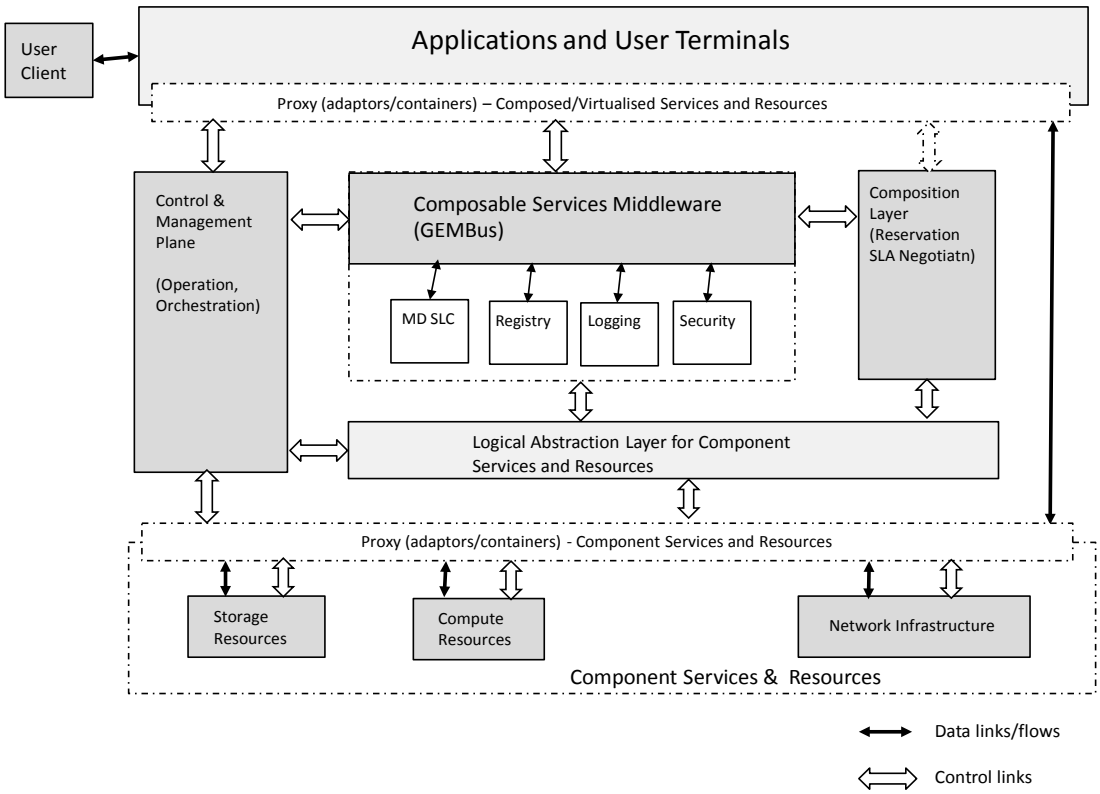

Figure 2. Composable Service Architecture and main functional components

Composable Services Middleware (CSA-MW) is an important component of the CSA that provides common interaction environment for both (physical) component services and complex, composite services, built with them. Besides exchanging messages, CSA-MW also contains/provides a set of basic/general infrastructure services required to support reliable and secure (composite) services delivery and operation.

It is suggested that the ongoing GEMBus development will provide a generic/reference CSA middleware implementation. CSA implementation suggestions and details are discussed in the next chapter.

4 CSA Implementation Suggestions

4.1 CSA Middleware and GEMBus

Figure 3 below illustrates the suggested GEMBus architecture. GEMBus infrastructure includes three main groups of functionalities:

- GEMBus Messaging Infrastructure (GMI) that includes, first of all, messaging backbone and other message handling supporting services such as message routing, configuration services, secure messaging, event handler/interceptors. The GMI is built on and extends the generic Enterprise Service Bus (ESB) functionality to support dynamically configured multidomain services as defined by GEMBus.
- GEMBus infrastructure services that support reliable and secure composable services operation and the whole services provisioning process. These include such services as Composition, Orchestration,

Security, and the also important Lifecycle Metadata Service, which are provided by the GEMBus environment/framework itself.

- Component services, although typically provided by independent parties, need to implement special GEMBus adaptors or use special “plug-in sockets” that allow their integration into the GEMBus/CSA infrastructure.

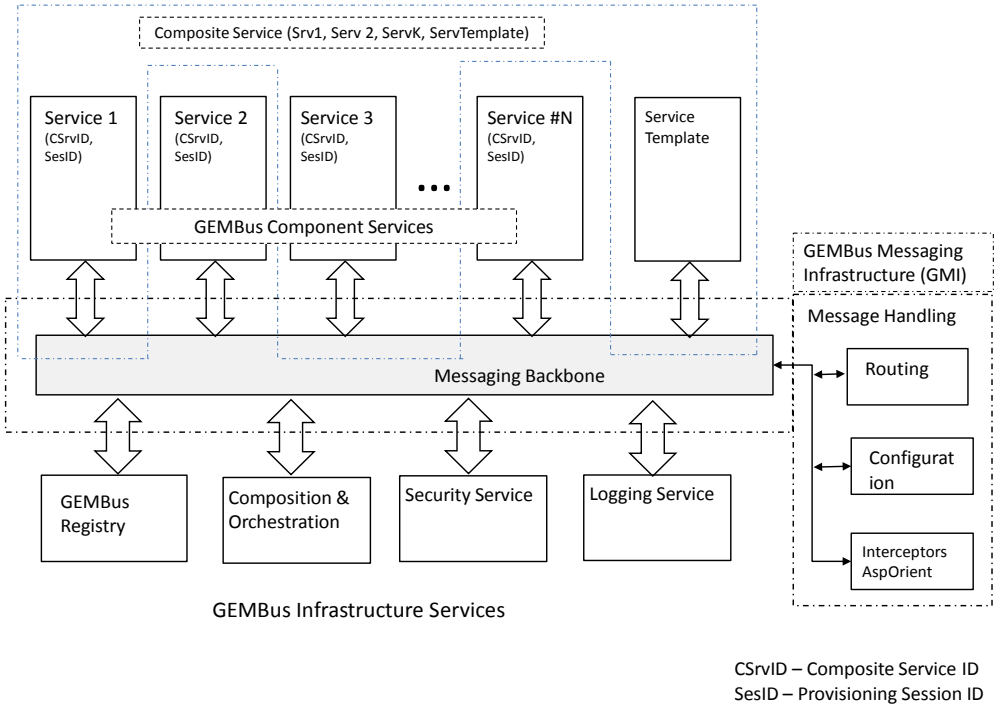

Figure 3. GEMBus infrastructure, including component services, service template, infrastructure services, and core message-processing services

Figure 4 illustrates two examples of the composite services that are composed of four component services. In the second case the composite service contains a special Frontend service that is created of the corresponding service template that should be available for specific kind of applications. Examples of such services templates can be a user terminal (or rich user client), or a visualisation service. Requiring the GEMBus framework or toolkit to provide a number of typical service templates will provide more flexibility in delivery/provisioning composite services.

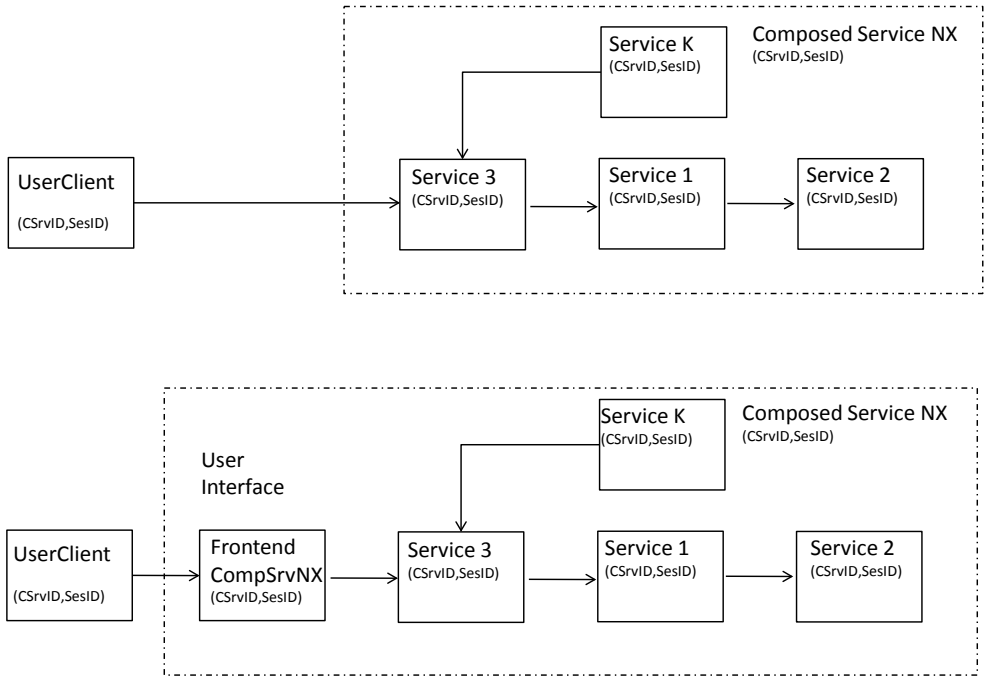

Figure 4. Example composite service composed of services Service 1, Service 2, Service 3, Service 4.

4.2 General CSA Middleware Requirements

GEMBus as a CSA middleware should provide an open environment for composable service integration and operation. The following are specific requirements related to such environment.

- GEMBus configuration facility/functionality should allow service virtualisation and dynamic configuration (without the whole GEMBus infrastructure re-deployment)
- GEMBus should support services virtualisation by supporting configurable logical abstraction of the Composition/Orchestration/Virtualisation layer and dynamically configurable message routing
- GEMBus should support different stages of the composable services lifecycle, in particular: service discovery, composition, operation, and de-commissioning. This can be provided via special configuration services and supported by GEMBus metadata services that should allow dynamic services re-configuration.
- GEMBus should support event driven services and allow configurable event recognition, alarming and recording. This functionality is specifically required for, and targeted to, support service orchestration, workflow management, logging, and accounting.

5 Security suggestions

CSA should support/provide security services to address the following security concerns/issues:

- Data and communication security
- Access control (e.g. Authentication, Authorisation, Identity Management)
- Policy enforcement
- Additionally, auditing/logging and accounting

Security and security services in the CSA specification and GEMBus design are applied at different layers and in different functional components.

There are two general areas/domains for which the security measures are defined:

1. Composable security services that may be composed with other component services

2. Security services to support normal GEMBus operation and (services) management

Security services are called from different functional components based on the message properties that are extracted by the aspect-based message interceptors. Security services are governed by related security policies.

Security services can be designed as pluggable services operating at the messaging layer and can be added as composable services to the other composable services.

Security services should support the whole provisioned/composable services lifecycle and consequently support session-related security context management.

CSA-Security should implement multi-layer security services including transport, messaging and application/data security, and additionally network layer security for distributed VPN based enterprise domains.

It is suggested that (due to the typical for ESB collapsed messaging backbone architecture) the intra-domain messaging exchange environment is secured and trusted at the messaging level. Therefore intra-domain message exchange doesn't require using transport level security, however this part of the security design is still under discussion.

CSA-Security and GEMBus should implement basic infrastructure security services to ensure its normal operation such as Requestor identity and message authentication, message content protection including confidentiality and integrity, requestor and request authorisation, and others

CSA-Security should support policy based security services configuration and management and allow policy modification without security services re-deployment

CSA-Security should provide measures for security session context management by securely binding session security context to the session ID. At least 3 types of session should be considered:

- Message level protocol sequence, e.g. trust/key negotiation, attributes/security tokens exchange, etc.
- Provisioning session that could be identified by so-called GRI (global resource/ resource ID) – that starts from reservation and ends with the service access/use;
- Service resource access/use – uses the same GRI but may include multiple/multi-resource access sessions

6 Remaining issues and future developments

The CSA specification is considerably based on the GEMBus implementation and will evolve with the GEMBus development.

It is also considered that CSA will be proposed to other GN3 activities and other projects as a common framework for building composed and/or provisioned on-demand infrastructure services.

The following issues should be addressed in the ongoing CSA development:

1. Extension to network infrastructure services provisioning allowing also network topology instantiation and control

2. Provisioning dynamically configurable services for multi-domain composable services

3. Applicability for Cloud based infrastructure services provisioning and use of corresponding open interfaces

4. CSA compatibility with and positioning against NGN services architecture

It is suggested that the CSA will be proposed as a contribution to the ISOD-RG (On-demand Infrastructure Services provisioning Research group) at Open Grid Forum (OGF).

Appendix A. ITU-T Next Generation Network standards overview

The Next Generation Networks (NGN) is introduced by ITU-T as a next step in creating Global Information infrastructure. The NGN principles and the general reference model specified in the ITU-T Recommendation Y.2011 separate NGN services from the NGN transport network what allows for more service oriented approach in designing both transport network and network based services. Modern networking environment is characterised by integration between services and network infrastructure, increasing use of Internet protocols for inter-service communication, services “digitising”, and integration with the higher-level applications.

It is a natural step that NGN technology are moving to adopting SOA concepts and Web Services based services integration model to build Open Service Environment (OSE) as pre-scribed by another set of ITU-T standards defining NGN convergence model based on Web Services [13] and required NGN capabilities to support OSE [14]. Web services enabled NGN transport networks provide a native environment for integrating applications, services and resources that can be provisioned on-demand.

The NGN reference model according to ITU-T Y.2011 Recommendation suggests separation of the transport network and application services and defines them as NGN service stratum and NGN transport stratum consisting of User plane, Control plane and Management plane. The ITU-T Recommendations Y.2012 and Y.2201 specify high-level requirements and functional architecture of the NGN Release 1. The described NGN service architecture implements services and network separation principle and defines functional components of the Transport stratum and Service stratum. The NGN Y.2012 architecture defines also Application Network Interface (ANI) that provides an abstraction of the network capabilities and is used as a channel for applications to access network services and resources.

The NGN convergence service model is defined by ITU-T Recommendation Y.2232 and suggests the major scenario with using Web Services. The NGN Open Service Environment (OSE) defined by ITU-T Recommendation Y.2234 is based on Web Services and actually implements basic SOA principles in defining services integration model. The definition of the OSE and Web Services convergence model is targeted to provide common framework for both applications developers and provider services developers.

The Y.2234/Y.2201 NGN OSE is required to satisfy such requirements as independence from transport network providers, independence from manufactures, location transparency, network transparency, protocol transparency. The OSE should provide the following capabilities to support effective services integration and operation: service coordination; interworking with service creation environment; service discovery; service registration; policy enforcement, development support. The latter capability actually suggests that OSE should “support the full lifecycle of components, ranging from installation, configuration, administration, publishing, versioning, maintenance and removal.

Appendix B. Open Group Service Integration Maturity Model (OSIMM)

In its evolution and gradual development the GEMBus should adopt SOA best practices and comply with the Open Group Services Integration Maturity Model (OSIMM) [6], The OSIMM defines a grid of the 7 maturity level and 7 dimensions that describes provisioned services. The 7 OSIMM maturity levels include:

(1) Silo;

(2) Integrated;

(3) Componentised;

(4) Services,

(5) Composable services;

(6) Virtualised services;

(7) Dynamically re-configurable services.

The 7 dimensions define different presentation layers and aspects of the services such as Business view, Governance and Operations, Methods, Applications, Architecture, Information, Infrastructure and Management.

In Applications dimension the SOA based applications deal with the different components and building blocks mapped to the above defined maturity levels: modules (OSIMM1), objects (OSIMM2), components (OSIMM3), services (OSIMM4), applications comprised of services (OSIMM5), process integration via services (OSIMM6), and dynamic application assembly (OSIMM7).

Starting from the level "OSIMM4 - Services" the information or data are represented as Information as a service (OSIMM4), Data dictionary and repository (OSIMM5), Virtualised data services (OSIMM6), Semantic data vocabularies, correspondingly (OSIMM7). Table 1 provides summary of the services presentation models at the different OSIMM levels.

The OSIMM also defines so-called domains that are specific problem areas projected into the Maturity – Dimensions grid. Starting from the "Level 4 – Services" the security services are considered as a basic service that according to the OSIMM model can be composed, virtualised and dynamically reconfigured. This implies more requirements to defining the CSA discussed in this document.

Table 1. SOA components presentation at different OSIMM levels

OSIMM levels & Dimensions	OSIMM1 Silo	OSIMM2 Integrated	OSIMM3 Componentized	OSIMM4 Services	OSIMM5 Composable services	OSIMM6 Virtualised services	OSIMM7 Dynamically re-configurable services
Business view	Isolated business lines	Business process integration	Componentized business	Componentized business offers services	Processes through services composition	Geographical independent service centers	Mixed match business and context aware-capabilities
Organisation	Ad hoc IT strategy & Governance	Ad hoc enterprise strategy & Governance	Common Governance process	Enabling SOA Governance	SOA and IT Governance Alignment	SOA and IT Infrastructure Governance Alignment	Governance through policy
Methods	Structured analysis	Object Oriented	Component based developme	Service Oriented	Service Oriented Modeling	Service Oriented Modeling for	Business Grammar Oriented

	and Design	Modeling	nt	Modeling		Infrastructure	Modeling
Applications	Modules	objects	components	services	applications comprised of services	process integration via services	dynamic assembly, context-aware invocation
Architecture	Monolithic architecture	Layered architecture	Component architecture	Emerging SOA	SOA	Grid based SOA	Dynamically re-configurable architecture
Information	Application specific	LOB or enterprise specific	Canonical models	Information as a service	Enterprise Business Data dictionary and repository	Virtualised data services	Semantic data vocabularies
Infrastructure (and Management)	LOB Platform specific	Enterprise standards	Common re-usable infrastructure	Project based SOA environment	Common SOA environment	Virtual SOA environment, S&R	Dynamic sense, Decide& Respond
	OSIMM1	OSIMM2	OSIMM3	OSIMM4	OSIMM5	OSIMM6	OSIMM7

Appendix C. Services Lifecycle Management

C.1. Existing Service Lifecycle Development and Management Frameworks

The SOA-based technologies provide a good basis for creating composable services, which in case of advancing to dynamically re-configurable services should also rely on a well-defined services lifecycle management (SLM). Most of existing SLM frameworks and definitions are oriented on rather traditional human-driven services development and composition. Dynamically provisioned and re-configured services will require re-thinking of existing models and proposing new security mechanisms at each stage of the typical provisioning process.

Figure C.1 illustrates typical sequence of stages defining the provisioned service lifecycle that includes service request, design or development, deployment or implementation, operation, retire or disposal. These stages are quite in tact with the proposed in [27] Complex Resource Provisioned (CRP) model which was used for defining multi-domain authorisation infrastructure for on-demand Network resources provisioning in the Phosphorus project 34].

Defining different lifecycle stages allows using different level of the services 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, the consistent services context management mechanisms should be defined and used during the whole service lifecycle. In particular case of the security services, the security services should ensure integrity of the service context management together with ensuring integrity of the security context itself. The problem here is that such mechanisms are generically stateful what impose problems for SOA environment, which is defined as generically stateless.

The NIST Special Publication 800-14 “Generally Accepted Principles and Practices in Systems Security” [28] together with SP 800-27 “Engineering Principles for Information Technology Security” [29] define a basic set of the generally accepted principles and practices for designing and managing security services. The defined security services lifecycle includes the following phases: Initiation, Development/Acquisition, Implementation, Operation/Maintenance, and Disposal. Providing a good basis for security services management, these principles still reflect the traditional approach to services and systems design driven by engineering forces.

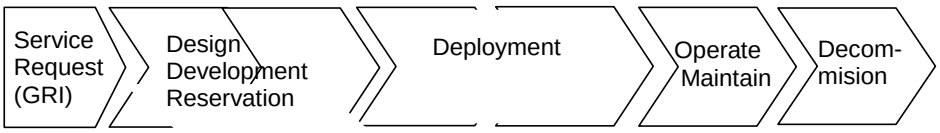

Figure C.1. General Services Lifecycle Management model

C.2. Service Delivery Framework by TeleManagement Forum (SDF)

To answer dynamic character of the New Generation Networks (NGN) concept, the TeleManagement Forum (TMF) [23] proposed the Service Delivery Framework (SDF) [30] as a part of their New Generation Operations Systems and Software (NGOSS) solutions framework [31]. The main motivation behind developing SDF is achieving automation of the whole service delivery and operation process, in particular:

- End-to-end service management in a multi-service providers environment

- End-to-end service management in a composite, hosted and/or syndicated service environment
- Management functions to support a highly distributed service environment, for example unified or federated security, user profile management, charging etc.
- Any other scenario that pertains to a given phase of the service lifecycle challenges, such as on boarding, provisioning, or service creation.

SDF services lifecycle corresponds to the general services lifecycle management model shown on Figure C.1. Figure C.2 illustrates the major SDF components and their interactions during 3 main stages Design, Deployment, Operation. The SDF defines two basic supporting systems: Management Support Service (SDF MSS) and Infrastructure Support Service (SDF ISS). The Service instance, delivered or provisioned on demand, is presented as containing: (2) Service Management Interface, (3) Service Functional Interface, and (3a) Service Consumer Interface.

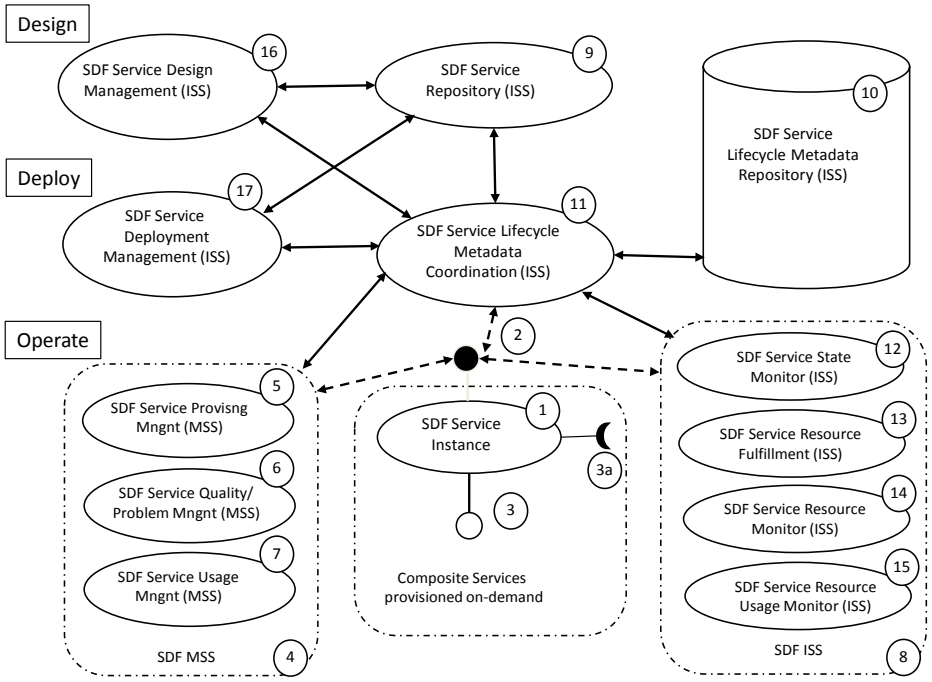

Figure C.2. SDF main components and their interaction during Design, Deployment and Operation Stages

The framework specifies the following components involved into the service provisioning at different lifecycle stages:

SDF Stages				
REQUEST	DESIGN stage	DEPLOYMENT stage	OPERATION stage	DECOMMISSION
SLA Negotiation	9 - Service Repository	10 - Service Lifecycle Metadata Repository	5 - Service Provisioning Management	10 - Service Lifecycle Metadata Repository
Provisioning	10 - Service Lifecycle	11 - Service Lifecycle	6 - Service Quality/Problem	11 - Service

Session ID assignment	Metadata Repository	Metadata Coordinator	Management	Lifecycle Metadata Coordinator
11 - Service Lifecycle Metadata Coordinator	16 - Service Design Management	17 - Service Deployment Management	7 - Service Usage Monitor	14 - Service Resource Monitor
			11 - Service Lifecycle Metadata Coordinator	15 - Resource Usage Monitor
			12 - Service State Monitor	
			13 - Service Resource Fulfillment	
			14 - Service Resource Monitor	
			15 - Resource Usage Monitor	

It is important to mention that the key component of the SDF model is Services/resource Lifecycle Metadata management. This system and a process is designated to ensure the consistency of the service management and in particular important to support consistent security services provisioning and management.

C.3. The Proposed Security Services Lifecycle Management Model

Most of the existing security lifecycle management frameworks, such as defined in the NIST Special Publication 800-14 “Generally Accepted Principles and Practices in Systems Security” [29], provide a good basis for security services development and management, but they still reflect the traditional approach to services and systems design driven by engineers force. The defined security services lifecycle includes the following typical phases: Initiation, Development/Acquisition, Implementation, Operation/Maintenance, and Disposal.

Figure C.3 (b) illustrates the proposed Security Services Lifecycle Management (SSLM) model that reflects security services operation in generically distributed multidomain environment and their binding to the provisioned services and/or infrastructure. The SSLM includes the following stages:

- Service request and generation of the GRI that will serve as a provisioning session identifier and will bind all other stages and the related security context.
- Reservation session binding that provides support for complex reservation process including required access control and policy enforcement.
- Deployment stage begins after all component resources have been reserved and includes distribution of the security context and binding the reserved resources or services to the Global Reservation ID (GRI) as a common provisioning session ID.
- Registration & Synchronisation stage (that can be considered as optional) that specifically targets possible scenarios with the provisioned services migration or failover. In a simple case, the Registration

stage binds the local resource or hosting platform run-time process ID to the GRI as a provisioning session ID.

- During the Operation stage the security services provide access control to the provisioned services and maintain the service access or usage session.
- The Decommissioning stage ensures that all sessions are terminated, data are cleaned up and session security context is recycled.

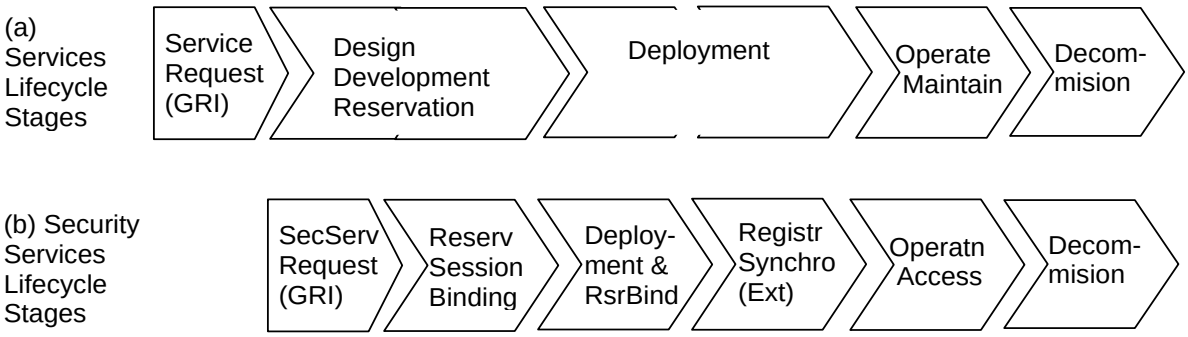

Figure C.3. The proposed Security Services Lifecycle Management model and its relation to the general Services Lifecycle Management model

The proposed SSLM model extends the existing SLM frameworks and earlier proposed by authors the CRP model [3] with the new stage “Registration & Synchronisation” that specifically targets such security issues as the provisioned services/resources restoration (in the framework of the active provisioning session) and provide a mechanism for remote data protection by binding them to the session context.

7 References

1. Deliverable DJ3.3.1: Composable Network Services use cases. GEANT3 Project Deliverable. January 11, 2010. http://www.geant.net/Media_Centre/Media_Library/Media%20Library/GN3-09-198-DJ3_3_1_Composable_Network_Services_use_cases.pdf
2. Chappell, D., "Enterprise service bus", O’Reilly, June 2004. 247 pp.
3. [SOA1] OASIS Reference Architecture Foundation for Service Oriented Architecture 1.0, Committee Draft 2, Oct. 14, 2009. http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/soa-ra-cd-02.pdf
4. [SOA2] IBM's SOA Foundation: An architectural introduction and overview. Whitepaper. November 2005. http://download.boulder.ibm.com/ibmdl/pub/software/dw/webservices/ws-soa-whitepaper.pdf
5. Exploring the Enterprise Service Bus, Part 2: Why the ESB is a fundamental part of SOA. By Greg Flurry, Rachel Reinitz. http://www.ibm.com/developerworks/webservices/library/ar-esbpat2/
6. [OSIMM] The Open Group Service Integration Maturity Model (OSIMM). https://www.opengroup.org/projects/osimm/ uploads/40/17990/OSIMM_v0.3a.pdf
7. ITU-T Recommendation Y.2001 (2004) - General overview of Next Generation Networks (NGN). http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-Y.2001-200412-I!!PDF-E&type=items
8. ITU-T Recommendation Y.2011 (10/2004) - General principles and general reference model for Next Generation Networks. [Online] http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-Y.2011-200410-I!!PDF-E&type=items

9. ITU-T Recommendation Y.2012 (09/2006) - Functional requirements and architecture of the NGN release 1. [Online] http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-Y.2012-200609-S!PDF-F&type=items

10. ITU-T Recommendation Y.2013 (12/2006): Converged services framework functional requirements and architecture. http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-Y.2013-200612-I!!PDF-E&type=items

11. ITU-T Recommendation Y.2201 (09/2006) – NGN release 1 requirements. [Online] http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-Y.2201-200909-I!!PDF-E&type=items

12. ITU-T Recommendation Y.2232 (01/2008) - NGN convergence service model and scenario using web services. [Online] http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-Y.2232-200801-I!!PDF-E&type=items

13. ITU-T Recommendation Y.2234 (12/2004) - Open service environment capabilities for NGN. [Online] http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-Y.2234-200809-I!!PDF-E&type=items

14. ITU-T Recommendation Y.2703 (01/2009): The application of AAA service in NGN. [Online] http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-Y.2703-200901-I!!PDF-E&type=items

15. ITU-T Recommendation Y.2704 (01/2010): Security mechanisms and procedures for NGN. [Online] http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-Y.2704-201001-I!!PDF-E&type=items

16. ITU-T Recommendation Y.2720 (01/2009): NGN identity management framework. http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-Y.2720-200901-I!!PDF-E&type=items

17. Deliverable D.A1a Framework Architecture. European research project SLA@SOI: Empowering the service industry with SLA-aware infrastructures. http://sla-at-soi.eu/wp-content/uploads/2009/07/D.A1a-M12-Framework_Architecture.pdf

18. Microsoft Security Development Lifecycle (SDL) - Version 5.0. http://download.microsoft.com/download/F/2/0/F205C451-C59C-4DC7-8377-9535D0A208EC/Microsoft%20SDL_Version%205.0.docx

19. ITIL (IT Infrastructure Library): ITIL Version 3 - Service Improvement. URL: http://www.itil-officialsite.com

20. ITIL Version 3 Lifecycle Process Model. http://www.best-management-practice.com/officialsite.asp?DI=597910&trackID=002192

21. The Open Group, “TOGAF (The Open Group Architecture Framework)”. http://www.opengroup.org/architecture/

22. Web Services Architecture. W3C Working Group Note 11 February 2004. [Online]. Available: http://www.w3.org/TR/ws-arch/

23. Web Services Security Roadmap (2002). [Online]. Available: http://www.ibm.com/developerworks/library/specification/ws-secmap/

24. GFD.80 "The Open Grid Services Architecture, Version 1.5," I. Foster, H. Kishimoto, A. Savva, D. Berry, A. Grimshaw, B. Horn, F. Maciel, F. Siebenlist, R. Subramaniam, J. Treadwell, J. Von Reich. Open Grid Forum, Sept. 5, 2006.

25. M. Gudgin and Marc Hadley (ed.), Web Services Addressing 1.0 - Core, W3C Candidate Recommendation 9 May 2006, http://www.w3.org/TR/ws-addr-core/

26. GFD.131 "Secure Addressing Profile 1.0," D. Merrill. Open Grid Forum, June 13, 2008.

27. Demchenko, Y., M. Cristea, C. de Laat, E. Haleplidis, Authorisation Infrastructure for On-Demand Grid and Network Resource Provisioning, Proceedings Third International ICST Conference on Networks for Grid Applications (GridNets 2009), Athens, Greece, 8-9 September 2009. ISBN: 978-963-9799-63-9

28. NIST Special Publication 800-14 - Generally Accepted Principles and Practices for Securing Information Technology Systems. National Institute of Standards and Technology. September 1996. http://csrc.nist.gov/publications/nistpubs/800-27/sp800-27.pdf

29. NIST Special Publication 800-27 - Engineering Principles for Information Technology Security (A Baseline for Achieving Security), June 2001. National Institute of Standards and Technology. - http://csrc.nist.gov/publications/nistpubs/800-27/sp800-27.pdf

30. TMF Service Delivery Framework. http://www.tmforum.org /servicedeliveryframework/4664/home.html

31. TMF New Generation Operations Systems and Software (NGOSS). http://www.tmforum.org/BestPracticesStandards/ SolutionFrameworks/1911/Home.html

32. WS-I Basic Profile Version 1.1 (2006), WS-I Basic Profile Version 1.1. http://www.ws-i.org/Profiles/BasicProfile-1.1.html

33. WS-I Basic Security Profile Version 1.0 (2007), Basic Security Profile Version 1.0. http://www.wsi.org/Profiles/BasicSecurityProfile-1.0.html

34. Phosphorus Project. [Online]. Available: http://www.ist-phosphorus.eu/

35. TeleManagement Forum. http://www.tmforum.org/

36. ESB-oriented architecture: The wrong approach to adopting SOA, by Bobby Woolf, WebSphere SOA and J2EE Consultant, IBM. http://www.ibm.com/developerworks/webservices/library/ws-soa-esbarch/