Composable Services Architecture (CSA) as a platform for Dynamically Re-Configured Virtualised Services

Yuri Demchenko, UvA

GN3 Concertation meeting
16 September 2010
TERENA Offices, Amsterdam
Outline

- Composable Services Architecture
- Composable Services Lifecycle and Workflow
- GEMBus as CSA middleware
- Issues in CSA and GEMBus

- Additional information (TMF SDF, WS vs REST)
Composable Services

- Composable services defined as “dynamically re-configured virtualised services”
  - In accordance with SOA and OSIMM (Open Group Services Integration Maturity Model) composable services can be positioned as a highest level

- GEMBus (GEANT Multidomain Enterprise Bus) will address multidomain issues and distributed services composition and orchestration
### Composable Services Layered Model

<table>
<thead>
<tr>
<th>Layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Layer</td>
</tr>
<tr>
<td>Virtualisation Layer</td>
</tr>
<tr>
<td>Composition &amp; Orchestration Layer</td>
</tr>
<tr>
<td>Logical Abstraction Layer</td>
</tr>
<tr>
<td>Messaging Layer</td>
</tr>
<tr>
<td>Network &amp; Transport Layer</td>
</tr>
</tbody>
</table>

- **Application Layer** hosts application related protocols
- **CSA primary focus on**
  - **Messaging Layer**
  - **Virtualisation (Composition & Orchestration) Layer**
- **Network & Transport Layer** should allow using/binding to standards communication and security protocol
- **Composable services are defined as** *“dynamically re-configured virtualised services”* according to OSIMM model
GEMBus provides common dynamically configurable messaging infrastructure for Composable Services communication

Logical Abstraction Layer includes/relies on component/physical services/resources adapters

* Adaptation (sub) layer

Composable Services Architecture (CSA) –
Version 0.11 – Simplified/Deprecated
Composable Services Architecture – Version 0.13

Composable Services lifecycle/provisioning stages

1. Request
2. Composition/Reservation
3. Deployment
4. Operation
5. Decommissioning

Applications and User Terminals

Proxy (adaptors/containers) – Composed/Virtualised Services and Resources

Control & Management Plane
(Management, Orchestration)

Composable Services Middleware (GEMBus)

Logical Abstraction Layer for Component Services and Resources

Proxy (adaptors/containers) - Component Services and Resources

Control/Management Links
Data Links

Component Services & Resources

Storage Resources
Compute Resources
Network Infrastructure

User Client

MD SLC Registry Logging Security

Composition Layer/Serv
(Reservation SLA Negotiation)
Composable Services Lifecycle/Provisioning Workflow

- Main stages/phases
  - Service Request (including SLA negotiation)
  - Composition/Reservation (aka design)
  - Deployment, including Registration/Synchronisation
  - Operation (including Monitoring)
  - Decommissioning

- Additional stages
  - Re-Composition should address incremental infrastructure changes
  - Recovery/Migration can use SL-MD to initiate resources resynchronisation but may require re-composition

- The whole workflow is supported by the Service Lifecycle Metadata Service (SL MD)
Composable Services Architecture – Lifecycle stages workflow

Composable Services lifecycle/provisioning stages

1. Request
2. Composition/Reservation
3. Deployment
4. Operation
5. Decommissioning

MD SLC – Service Lifecycle Metadata

GEMBus – GEANT Multidomain Bus

User Client

Applications and User Terminals

Proxy (adaptors/containers) – Composed/Virtualised Services and Resources

Control & Management Plane
(Operation, Orchestration)

Composable Services Middleware
(GEMBus/GESB)

Composition Layer /Serv
(Reservation SLA Negotiation)

Logical Abstraction Layer for Component Services and Resources

MD SLC

Registry

Logging

Security

Storage Resources

Compute Resources

Network Infrastructure

Component Services & Resources

Proxy (adaptors/containers) – Component Services and Resources

Control/Management Links

Data Links
CSA functional elements interaction

(1) Request
- User Client -> Control and Management

(2) Composition/Reservation
- Control&Mngnt -> Registry -> Composition/Reservation Serv -> (Logical Abstract -> Resr Adapters) -> LC Metadata Serv

(3) Deployment
- Control&Mngnt -> Composition/Reservation Serv -> (Logical Abstract -> Resr Adapters) -> LC Metadata Serv -> User Client

(4) Operation
- User Client -> Control&Mngnt (Orchestration) -> Rsr Adapters -> Virtualised/Composed Applications

(5) Decommissioning
- Control&Mngnt -> LC Metadata Serv -> (Logical Abstract -> Resr Adapters)
GEMBus Infrastructure for Composable Service

GEMBus Component Services

Service 1 (CSrvID, SesID)
Service 2 (CSrvID, SesID)
Service 3 (CSrvID, SesID)
Service 4 (CSrvID, SesID)

ESB

GEMBus Infrastructure Services

GEMBus Registry
Composition & Orchestration
Security Service
Logging Service

Message Handling
Routing
Configuration
Interceptors (AspOrient)

GEMBus Messaging Infrastructure (GMI)

GMI provides common dynamically configurable messaging infrastructure for Composable services communication
Example Service Composition – Service NX

Role and place for Composition and Orchestration

* Composable services or GEMBus infrastructure service

CSrvID, SesID – bind component services into the on-demand provisioned Composed service NX

UserClient
(CSrvID, SesID)

Service 3
(CSrvID, SesID)

Service 1
(CSrvID, SesID)

Service 2
(CSrvID, SesID)

Service 4
(CSrvID, SesID)

Composed Service NX
(CSrvID, SesID)

User Interface

UserClient
(CSrvID, SesID)

Frontend CompSrvNX
(CSrvID, SesID)

Service 3
(CSrvID, SesID)

Service 1
(CSrvID, SesID)

Service 2
(CSrvID, SesID)

Service 4
(CSrvID, SesID)

Composed Service NX
(CSrvID, SesID)
GEMBus Messaging Infrastructure – Functional Components

- Message Processor provides actual message parsing, analysis and transformation
- Composable/Component services are connected via Adaptors
- GEMBus services are connected via Requestor/Responder (RR)
  - GEMBus security services invoked from RR or called out
- Composition&Orchestration as GEMBus services

Composable Service Lifecycle Metadata

Requester Service

Provider Service

Ext Security Services

GEMBus SecServ

Requester/Responder (RR)

Configurat Manager

Events Handler

Resolver Router

Req/Respder

Registry Service

Provider Service

LogConfiguratn

GEMBus Messaging Infrastructure – Functional Components

- Message Processor provides actual message parsing, analysis and transformation
- Composable/Component services are connected via Adaptors
- GEMBus services are connected via Requestor/Responder (RR)
  - GEMBus security services invoked from RR or called out
- Composition&Orchestration as GEMBus services

Composable Service Lifecycle Metadata

Requester Service

Provider Service

Ext Security Services

GEMBus SecServ

Requester/Responder (RR)
Multidomain issues in CSA

- Can be addressed in the properly layered infrastructure and message routing
  - *CSA middleware and GEMBus Messaging Infrastructure*
- End-Point Reference (EPR) and use of Fully Qualified Names (FQN)
- Dynamic security federation/associations and provisioning session management
Full convergence is possible if CSA will integrate topology based network infrastructure provisioning

- Subject of current cooperation with GEYSERS Project the develops infrastructure services virtualisation architecture and don-demand provisioning

CSA should investigate Cloud middleware use, in particular, RESERVOIR and Claudia Frameworks, future GEYSERS infrastructure
Additional Information

- TMF SDF Lifecycle Management model
- Proposed Security SLM model
- WS vs REST -> UPR vs URI

- 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

Service Delivery Lifecycle

- Concept
- Design
- Develop
- Deploy
- Operate
- Retire
SDF Reference Architecture (refactored from SDF)

1 – Service Instance
2 - Service Management Interface
3 – Service Functional Interface
4 - Management Support Service (SDF MSS)
8 - Infrastructure Support Service (ISS)
DESIGN stage
9 - Service Repository
10 - Service Lifecycle Metadata Repository
16 - Service Design Management
DEPLOYMENT stage
10 - Service Lifecycle Metadata Repository
11 - Service Lifecycle Metadata Coordinator
17 - Service Deployment Management Coordinator
OPERATION stage
5 - Service Provisioning Management
6 - Service Quality/Problem Management
7 - Service Usage Monitor
12 - Service State Monitor
13 - Service Resource Fulfillment
14 - Service Resource Monitor
15 - Resource Usage Monitor

GN3-JRA3-T3
Composable Services Architecture
GEYSSERS Service Delivery Workflow
(WP2 Deliverable D2.1)

- Geysers SDF supports both Geysers infrastructure development and deployments and its operation for on-demand Infrastructure services provisioning by VIO.

Service Request/SLA Negotiation

Planning/Design

Deployment (Instant&Config&Synchro)

Operation &Monitoring (by VIO)

Decommissioning

Re-planning

Recovery/Migration

Network+IT Services Provisioning Workflow by VIO

Service Request/SLA Negotiation

Composition/Reservation

Deployment

Registr&Synchro

Operation (Monitoring)

Decommissioning

Re-Composition

Recovery/Migration

Composable Services Architecture
Proposed Security Services Lifecycle Management Model

- **Security Service request and generation of the GRI** that will serve as a provisioning session identifier and will bind all other stages and 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 GRI as a common provisioning session ID.

- **Registration & Synchronisation stage** (optional) specifically targets possible scenarios with the provisioned services migration or failover/interruption. 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.

- **Operation stage** - security services provide access control to the provisioned services and maintain the service access or usage session.

- **Decommissioning** stage ensures that all sessions are terminated, data are cleaned up and session security context is recycled.

---

**Diagram:**

- Service Request (GRI)
- Reserv Session Binding
- Deployment & RsrBind
- Registr Synchro (Opt)
- Operatn Access
- Decommission

**Composable Services Architecture**

---

**Discussion**

**GN3-JRA3-T3**
## Relation between SSLM/SLM stages and supporting general and security mechanisms

<table>
<thead>
<tr>
<th>SLM stages</th>
<th>Request</th>
<th>Design/Reservation Development</th>
<th>Deployment</th>
<th>Operation</th>
<th>Decommissioning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process/Activity</td>
<td>SLA Negotiation</td>
<td>Service/Composition Reservation</td>
<td>Resource Composition Configuration</td>
<td>Orchestration/Session Management</td>
<td>Logoff Accounting</td>
</tr>
</tbody>
</table>

### Mechanisms/Methods

<table>
<thead>
<tr>
<th>SLA</th>
<th>Workflow</th>
<th>Metadata</th>
<th>Dynamic Security Association</th>
<th>AuthZ Session Context</th>
<th>Logging</th>
</tr>
</thead>
<tbody>
<tr>
<td>V</td>
<td>(V)</td>
<td>V</td>
<td>(V)</td>
<td>V</td>
<td>(V)</td>
</tr>
</tbody>
</table>
Relation between SSLM and general SLM

(a) Services Lifecycle Stages
- Service Request (GRI)
- Design Development Reservation
- Deployment
- Operate Maintain
- Decommission

(b) Security Services Lifecycle Stages
- SecServ Request (GRI)
- Reserv Session Binding
- Deploy- ment & RsrBind
- Registr Synchro (Ext)
- Operatn Access
- Decommission

- Service Request stage may include SLA negotiation
  - *Security service instantiation may use SLA security context*
REST vs Web Services = URI vs EPR

- GEMBus uses own registered namespace branch
  - urn:geant:gembus
- CSA requires state management and communication between services
  - Basic functionality to be supported by WSRF and EPR
  - Keep EPR and URI/REST mapping
- End Point Reference is a part of WS-Addressing and complements WSDL to support the following scenarios
  - "Dynamic generation and customization of service endpoint descriptions.
  - Identification and description of specific service instances that are created as the result of stateful interactions.
  - Flexible and dynamic exchange of endpoint information in tightly coupled environments where communicating parties share a set of common assumptions about specific policies or protocols that are used during the interaction."
Example EPR and mapping to URI

EPR example with service properties and parameters

```xml
<wsa:EndpointReference>
  <wsa:Address>http://clarin.geant.net/registry</wsa:Address>
  <wsa:ReferenceProperties>
    <gembus:domainID>clarin.geant.net</gembus:domainID>
    <gembus:ServiceRegistryKey>K2349456076</gembus:ServiceRegistryKey>
  </wsa:ReferenceProperties>
  <wsa:ReferenceParameters>
    <gembus:sessionID>173945623490764234854</gembus:sessionID>
    <gembus:sessionDuration>8460</gembus:sessionDuration>
  </wsa:ReferenceParameters>
  <wsa:PortType>gembus:RegistryPortType</wsa:PortType>?
  <wsa:ServiceName PortName="RegistryUpdate">
    urn:geant:gembus:registry:update-service
  </wsa:ServiceName>
</wsa:EndpointReference>
```

Mapping to URI string:

```
http://clarin.geant.net/registry/update-service/ServiceRegistryKey=
K2349456076/sessionID=173945623490764234854;sessionDuration=8460
```