GEMBus based Services Composition Platform for Cloud PaaS

Yuri Demchenko, University of Amsterdam

on behalf of

Yuri Demchenko, Canh Ngo, Cees de Laat

Pedro Martínez-Julia, Elena Torroglosa, Antonio D. Perez-Morales

Mary Grammatikou, Jordi Jofre, Steluta Gheorghiu, Joan A. Garcia-Espin

ESOCC2012, 19-21 September 2012

Bertinoro, Italy

connect • communicate • collaborate

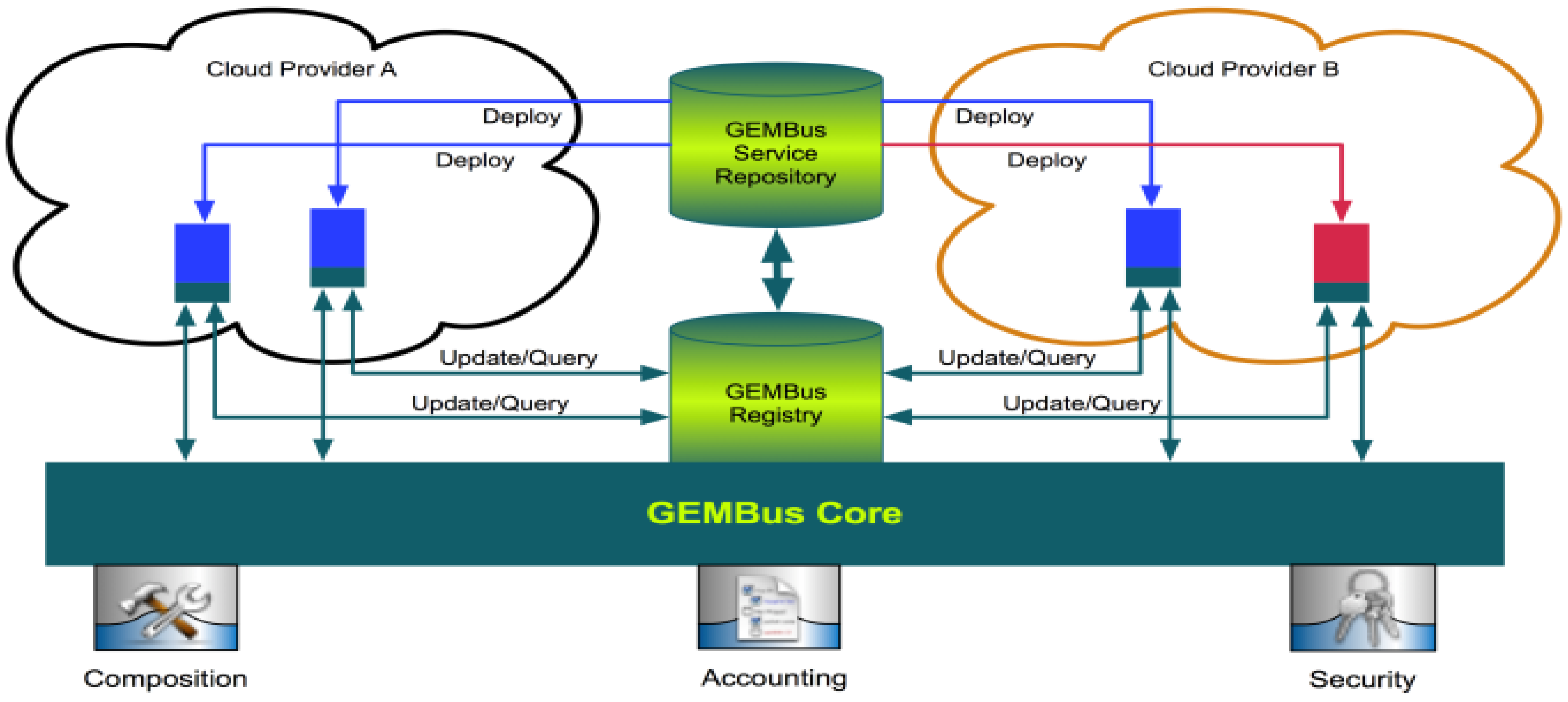
Outline

- GEMBus – GEANT Multi-domain service Bus
- Composable Services Architecture (CSA) as a GEMBus architecture basis
  - GEMBus as a Platform for Cloud based service provisioning
- GEMBus components and core services
  - Composition Service, Secure Token Services (STS), Accounting, Repository, Registry
- GEMBus testbed and demonstrators
  - Collaborative development platform for GEANT/NREN community
- Future developments
  - Contribution to OGF ISOD-RG activity, other community activities

GEMBus (GEANT Multi-domain Bus)

- GEMBus (GEANT Multi-domain Bus) is a service-oriented middleware platform that allows flexible services composition and their on-demand provisioning to create new specialized task-oriented services and applications
  - Part of EU Project GN3 JRA3 Task 3 “Composable Services”
- GEMBus is built upon state-of-the-art Enterprise Service Bus (ESB) technology and extends it with new functionalities that allow dynamic component services *composition*, *deployment*, and *management*
- Implements the Composable Services Architecture (CSA)
- Evolves as a cloud Platform as a Services (PaaS)
  - Cloud based services virtualisation
  - Community oriented
- Provides core GEMBus/PaaS services to support multi-domain services and users federation, including network connectivity
  - eduGAIN, Identity federation, federated network access (GEANT based)

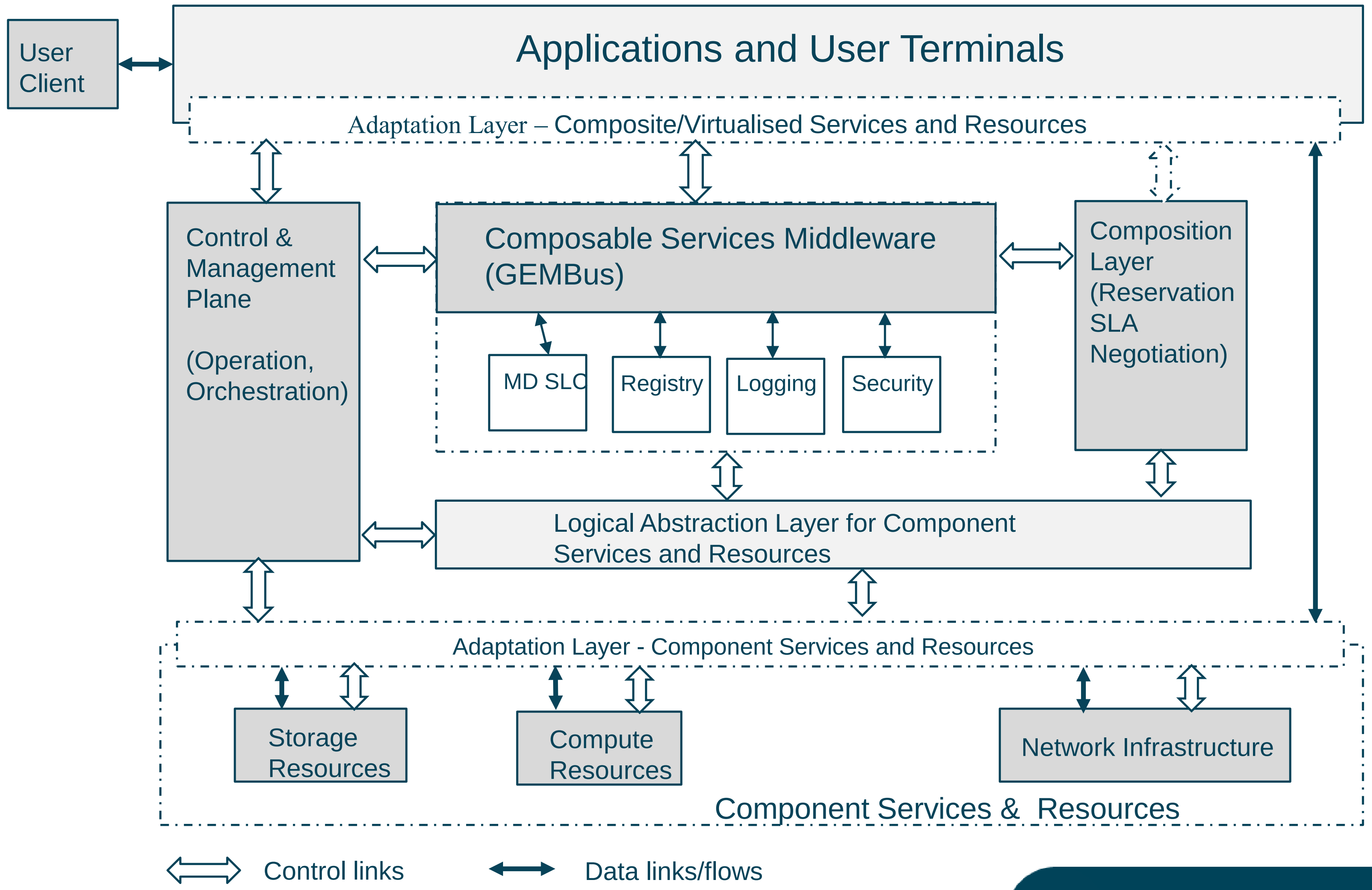
Inter-domain Services Integration (Federation)



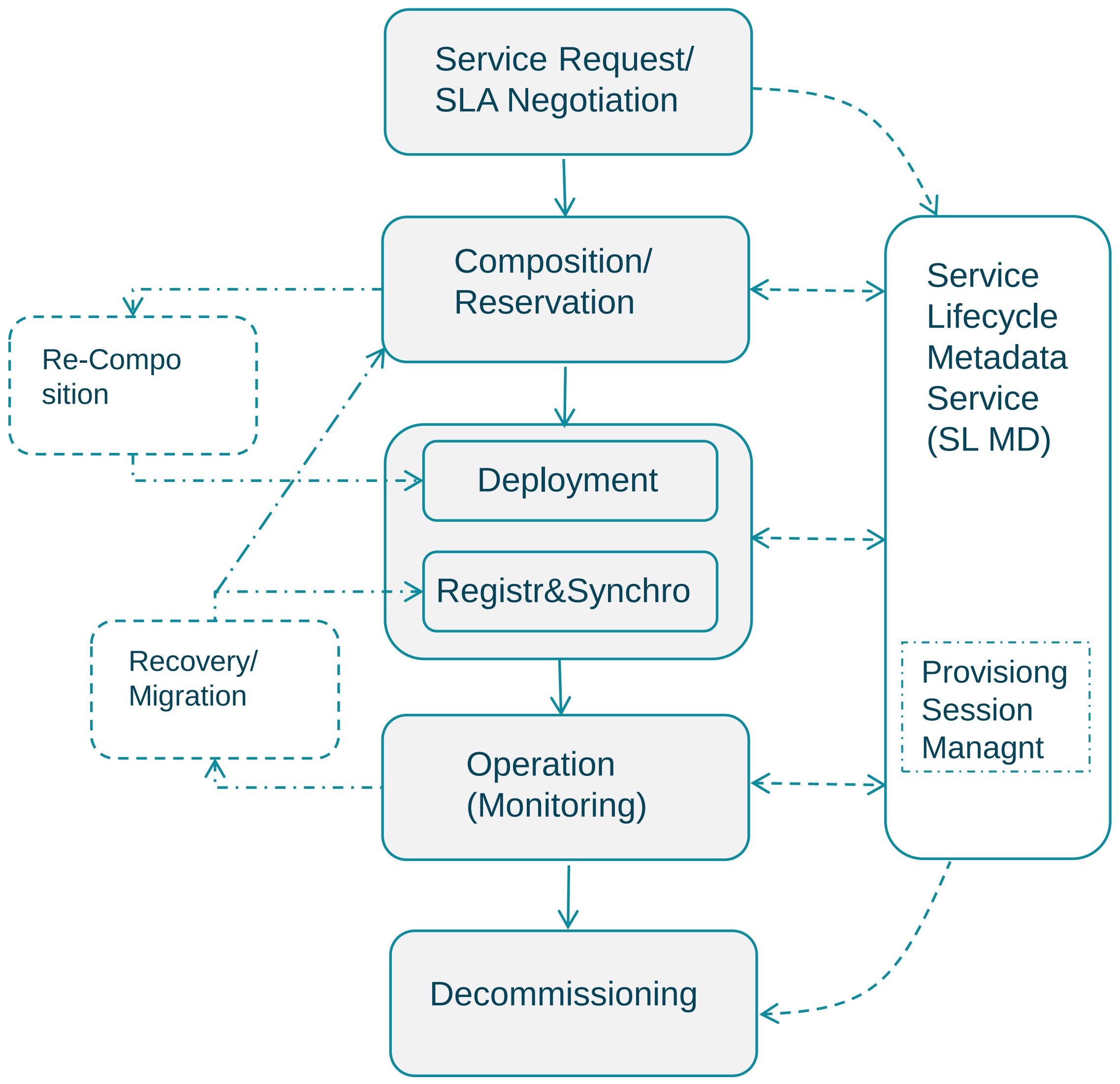
Composable Service Architecture (CSA)

- Composable Services Architecture (CSA) provides a basis for flexible services composition, integration and operation of existing component services
  - Defines functionalities related to Control and Management planes, allowing the integration of existing distributed applications and provisioning systems
  - Based on the virtualisation of component services
- Supports (composable) Services Lifecycle Management (SLM)
  - (1) Service Request, (2) Reservation, (3) Deployment, (4) Operation, (5) Decommissioning
- Provides a basis for defining GEMBus middleware
  - Leverages the SOA ESB architecture and Web Services platform

Composable Services Architecture

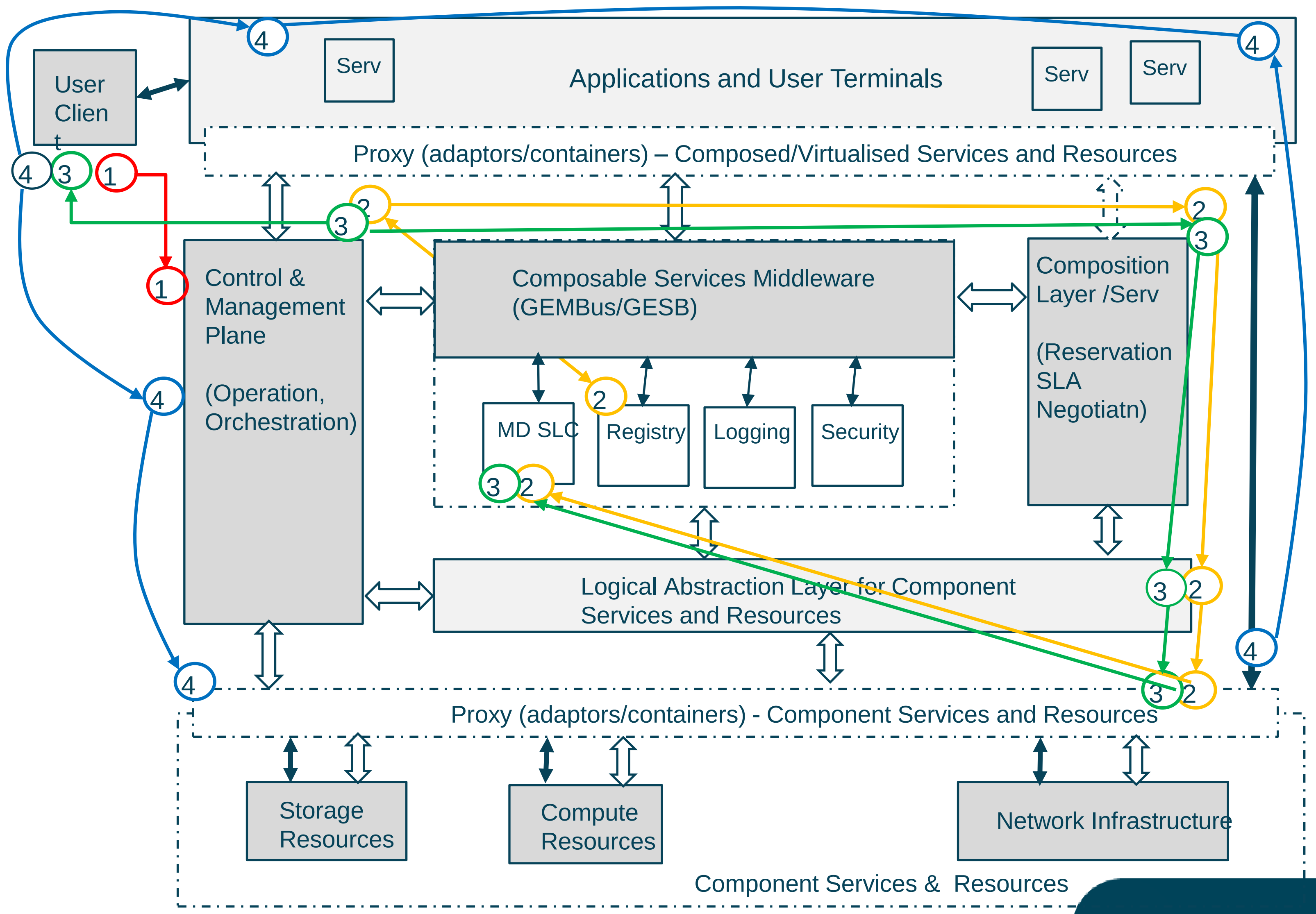

User Client
Applications and User Terminals
Adaptation Layer – Composite/Virtualised Services and Resources
Control & Management Plane
(Operation, Orchestration)
Composable Services Middleware (GEMBus)
MD SLC
Registry
Logging
Security
Composition Layer (Reservation SLA Negotiation)
Logical Abstraction Layer for Component Services and Resources
Adaptation Layer - Component Services and Resources
Storage Resources
Compute Resources
Network Infrastructure
Component Services & Resources
Control links
Data links/flows

Composable Services Lifecycle/Provisioning Workflow



- Main stages/phases
  - Service Request (including SLA negotiation)
  - Composition/Reservation (aka design)
  - Deployment, including Reqistration/Synchronisation
  - Operation (including Monitoring)
  - Decommissioning
- Additional stages
  - Re-Composition should address incremental infrastructure changes
  - Recovery/Migration can use SL-MD to initiate resources re-synchronisation but may require re-composition
- The whole workflow is supported by the Service Lifecycle Metadata Service (SL MD)

Composable Services Architecture Lifecycle stages workflow



4
User Client
Serv
Applications and User Terminals
Serv
Serv
4
4
3
1
Proxy (adaptors/containers) – Composed/Virtualised Services and Resources
2
3
2
3
1
Control & Management Plane
(Operation, Orchestration)
Composable Services Middleware (GEMBus/GESB)
Composition Layer /Serv
(Reservation SLA Negotiatn)
4
MD SLC
2
Registry
Logging
Security
3
2
Logical Abstraction Layer for Component Services and Resources
3
2
4
4
3
2
Proxy (adaptors/containers) - Component Services and Resources
Storage Resources
Compute Resources
Network Infrastructure
Component Services & Resources

Composable Services lifecycle/provisioning stages

(1) Request

(2) Composition/ Reservation

(3) Deployment

(4) Operation

(5) Decommissioning

MD SLC – Service Lifecycle Metadata

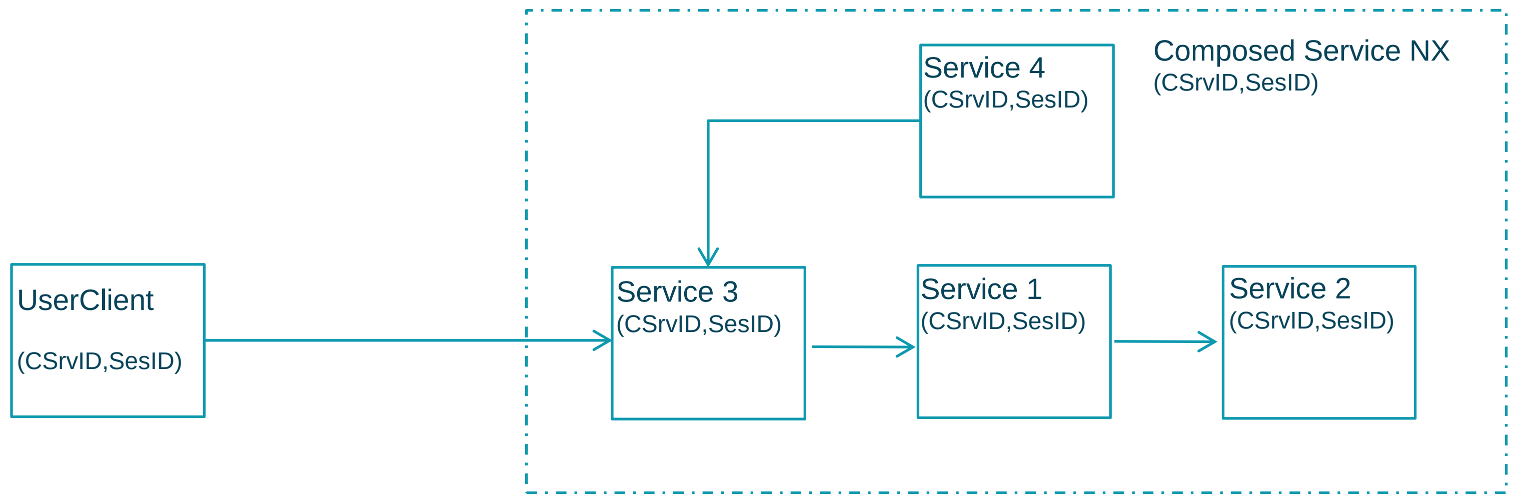
GEMBus – GEANT Multidomain Bus

GESB – Geysers ESB

⟺ Control/

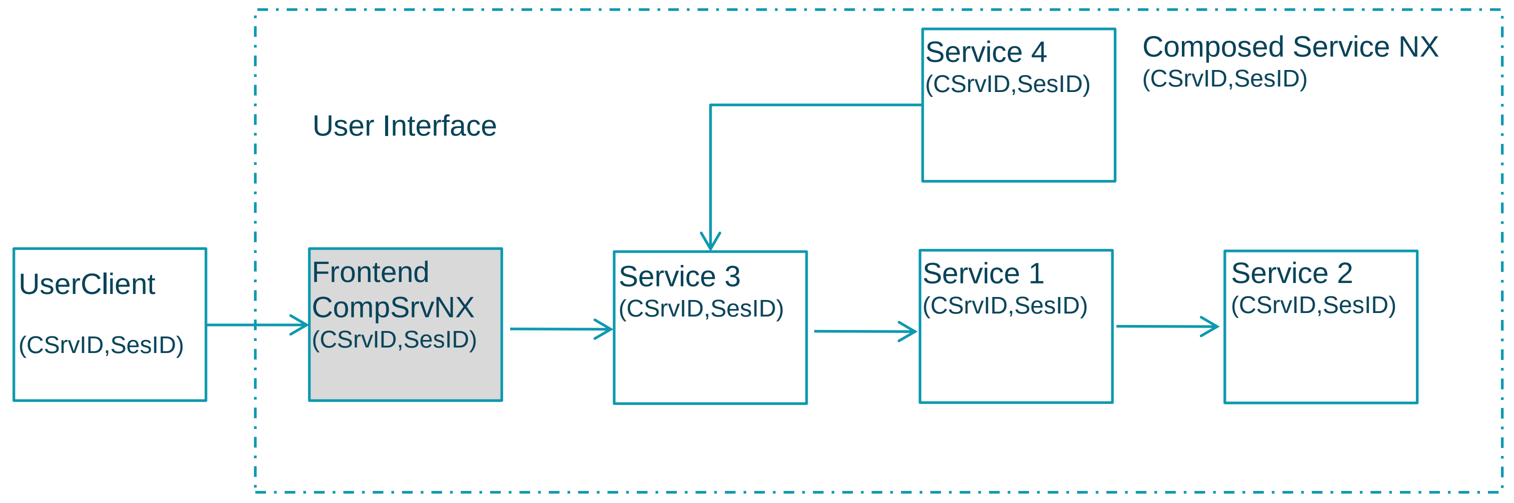
connect • communicate • collaborate

Example Service Composition – Service NX



Role and place for Composition and Orchestration

* Composable services or GEMBus infrastructure service

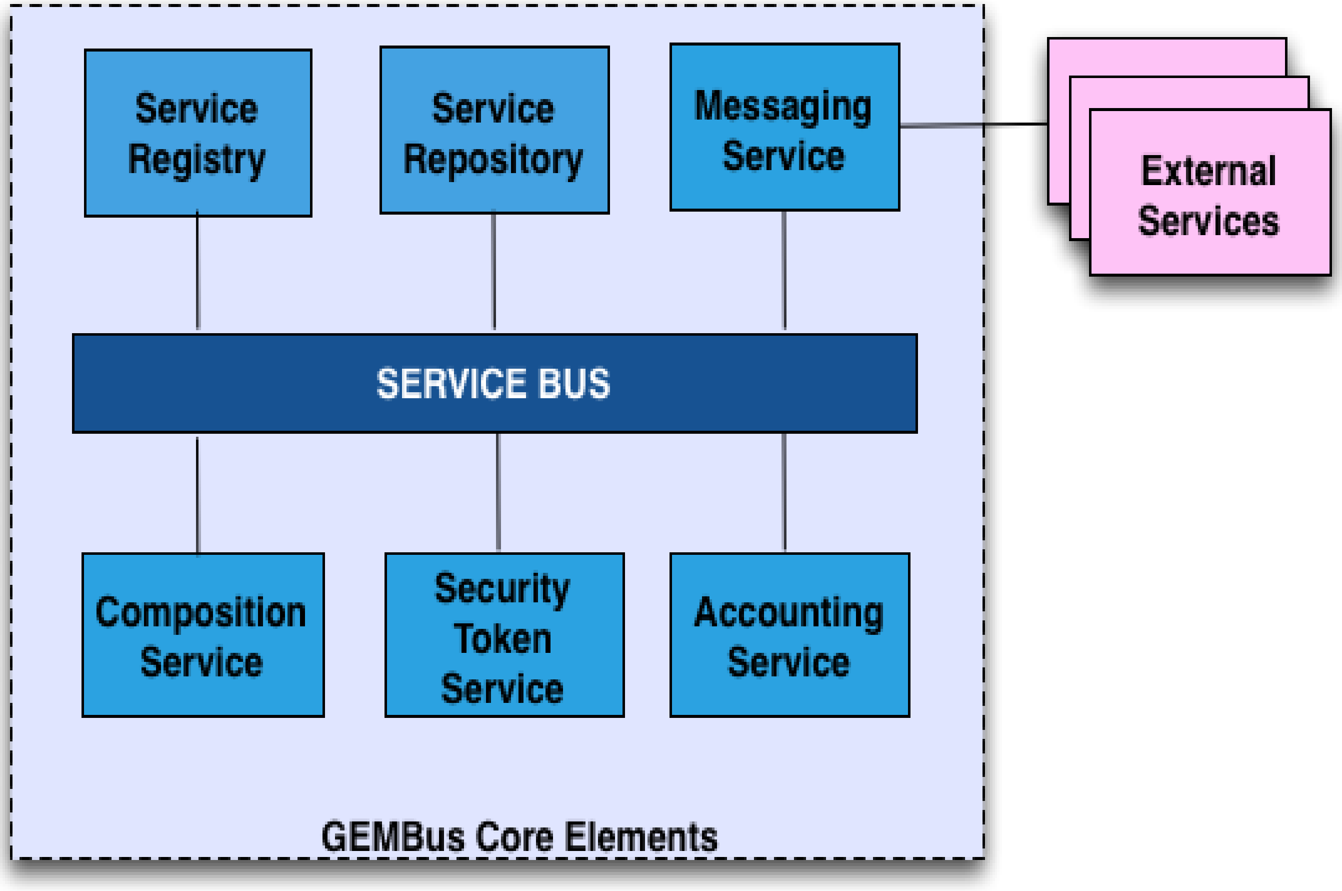


CSrvID, SesID –
bind component services into the on-demand provisioned Composed service NX

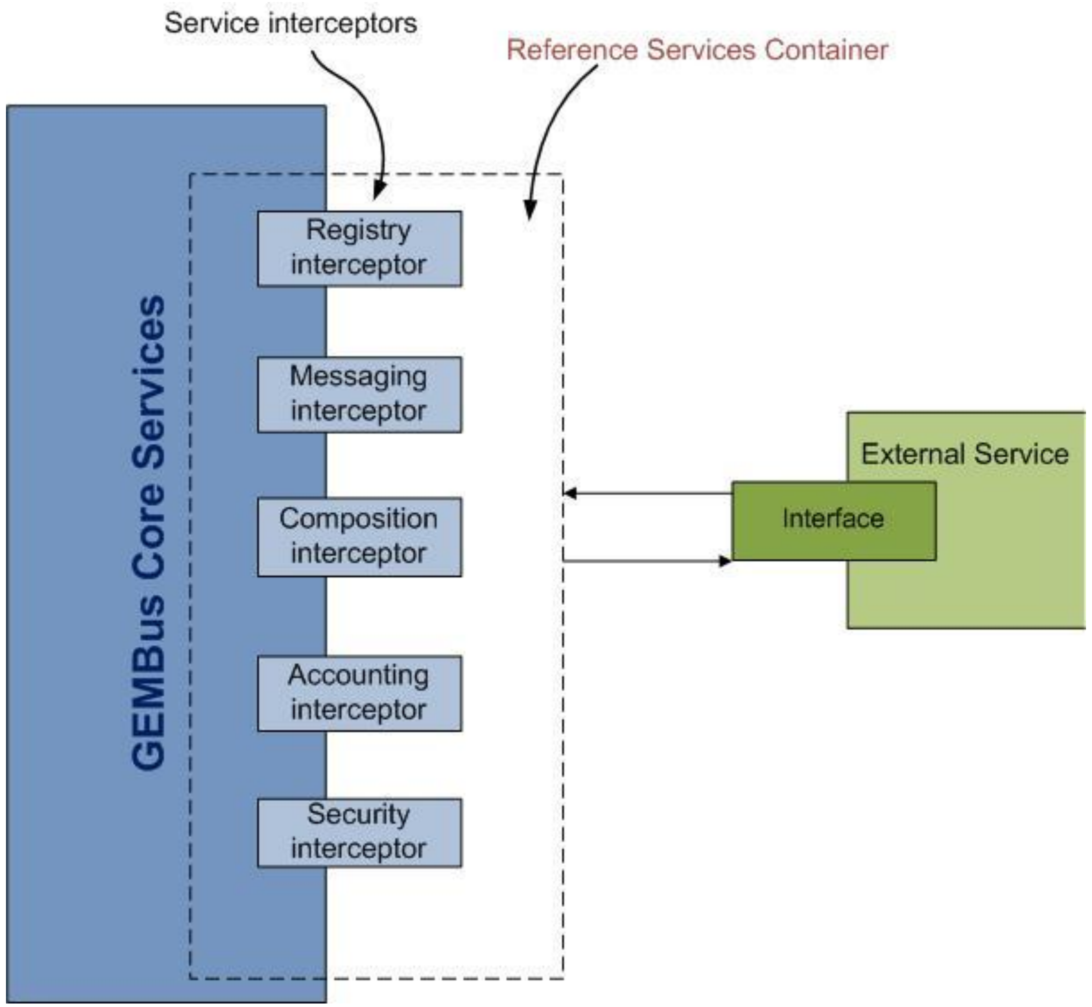
connect • communicate • collaborate

UvA Universiteit van Amsterdam

GEMBus as a Service Bus for heterogeneous services Integration



GEMBus Reference Container



- Reference service template (or Reference Container) is defined by an API that needs to be followed in order to connect to GEMBus platform.
- Provides set of configurable interceptors to GEMBus core services
- Provided as a standard piece of code for developing new and adapting any external APIs to GEMBus
- Provided as part of preconfigured GEMBus VMs for cloud PaaS services integration

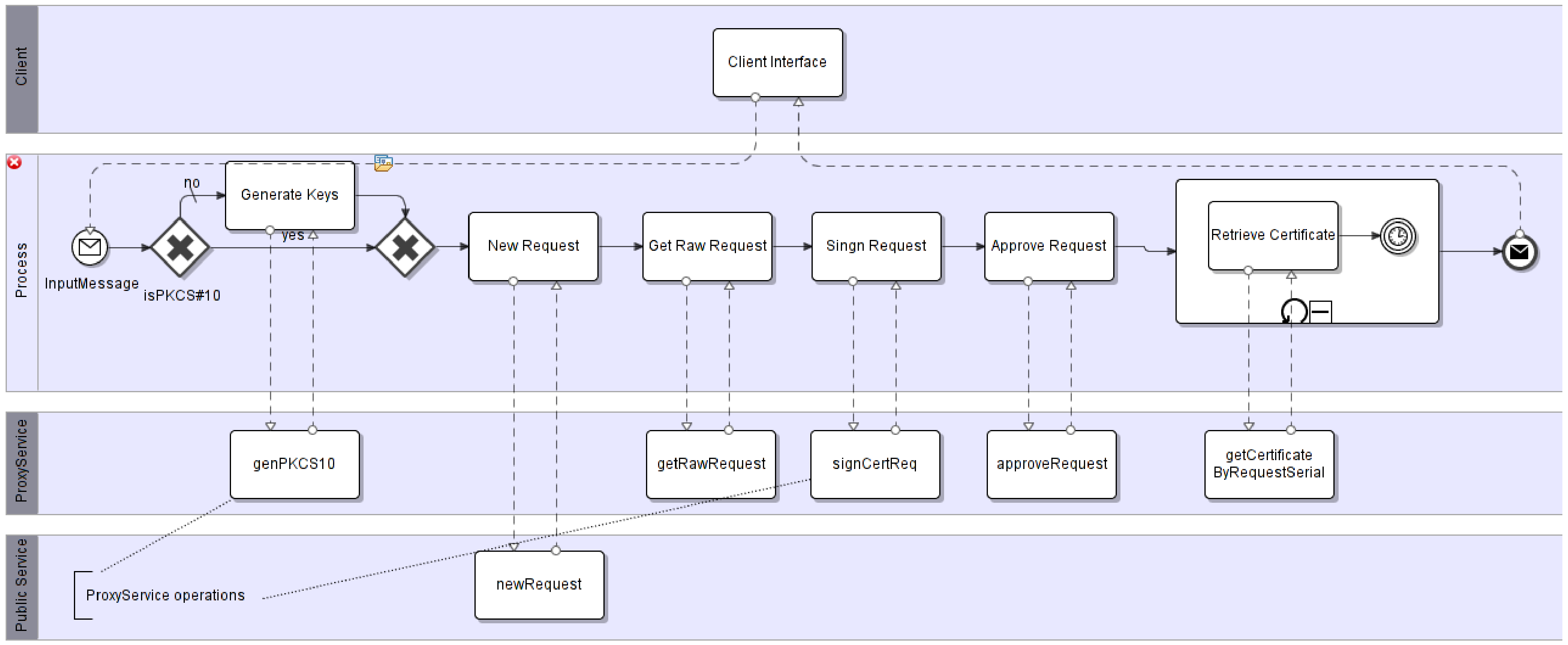
GEMBus Core Services

- **Federated Service Registry**: stores and provides information about GEMBus services.
- **Service Repository**: stores service bundles, thus allowing their deployment via GEMBus.
- **Composition Service**: enables services composition and supported by the orchestration engine.
- **Security Token Service (STS)**: issues, verifies and translates security tokens to allow the authentication of requesters in a federated, multi-domain environment.
- **Accounting Service**: provides configurable and aggregated access to the GEMBus login service to support monitoring, auditing, diagnostics, and troubleshooting.
- **Messaging service** that provides underlying infrastructure for composable services interaction, integration and QoS management
  - GEMBUs Messaging Infrastructure (GMI)

Composition Service

- Uses Business Process Execution Language (BPEL) and available ESB-based workflow execution engines to enable services orchestration
  - Connected with a specific description and control tools based on Business Process Modeling Notation (BPMN)
    - *From the analysis of a business processes to implementation and management*
  - First prototype implemented as an Eclipse plug-in and provides mapping to the underlying BPEL execution language
  - Allows integration of other workflow engine and support other workflow execution languages
- Orchestration service based on Taverna workflow management system
  - Cross-platform set of tools enabling users to compose web services supporting also local Java services (Beanshell scripts), local Java APIs, R scripts and data import from Excel or in CVS format
- Moving to Intalio|BMPS platform – full compatibility

Example services composition with Intalio|BMPS designer

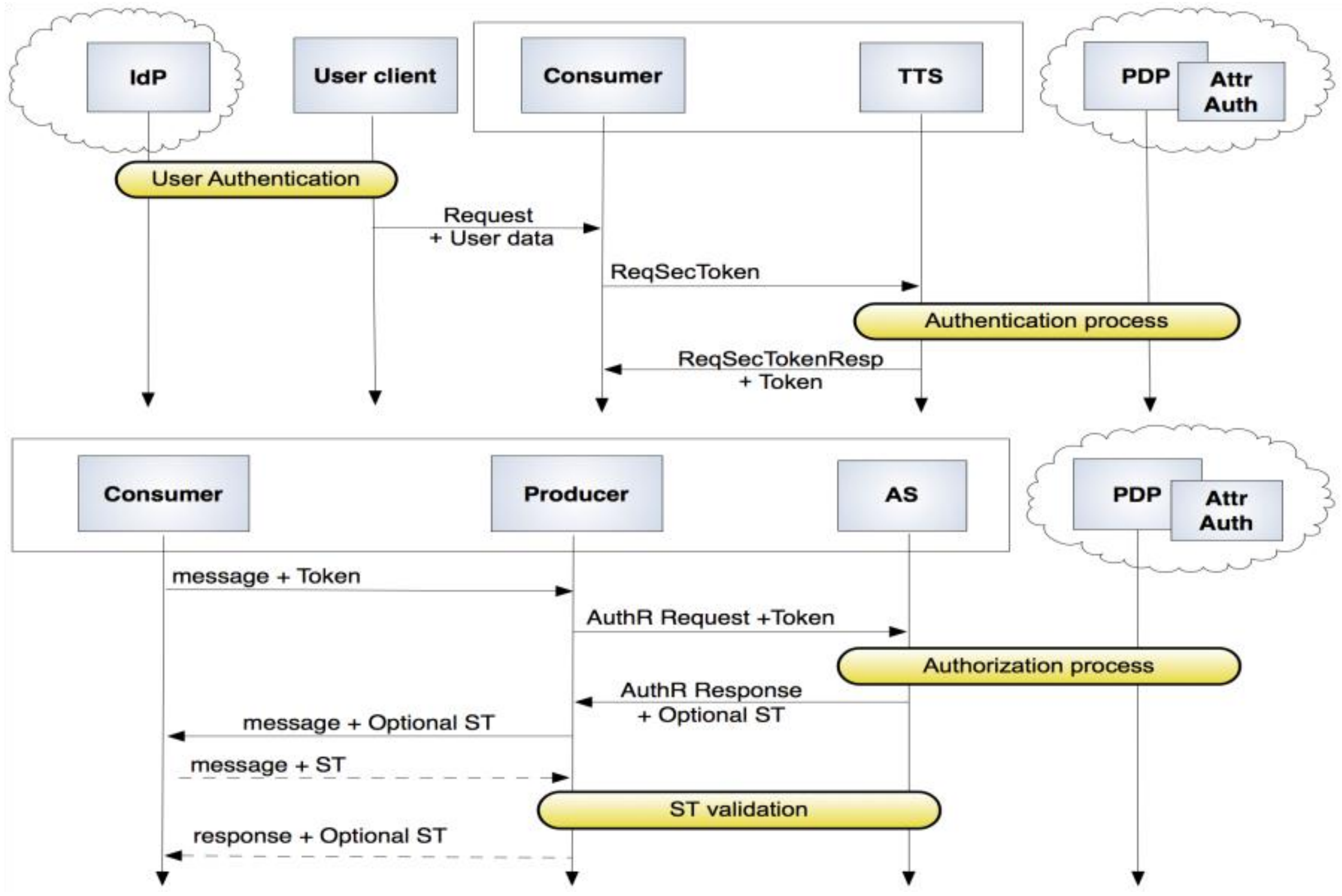


- Intalio|BPMS Business Process Management System (BPMS)
- Based on the open source Eclipse BPMN Modeler, Apache ODE BPEL engine, and Intalio Tempo WS-Human Task service

Security Token Service (STS)

- GEMBus Security Token Services (STS) implements the generic STS as defined in WS-Security and WS-Trust as a security mechanism to convey security information between services and supporting federated services infrastructure
  - STS supports issue and validation of the basic security tokens such as SAML, JWT, and X.509 certificates
  - STS supports services (identity) federation and federated identity delegation
- GEMBus STS support token issues and validation
  - Ticket Translation Service (TTS) responsible for generating valid tokens in the system according to the received credentials, renewing and converting security tokens
  - Authorisation Service (AS) supporting token validation (on request from a service) and can also retrieve additional attributes or policy rules from other sources to perform the validation

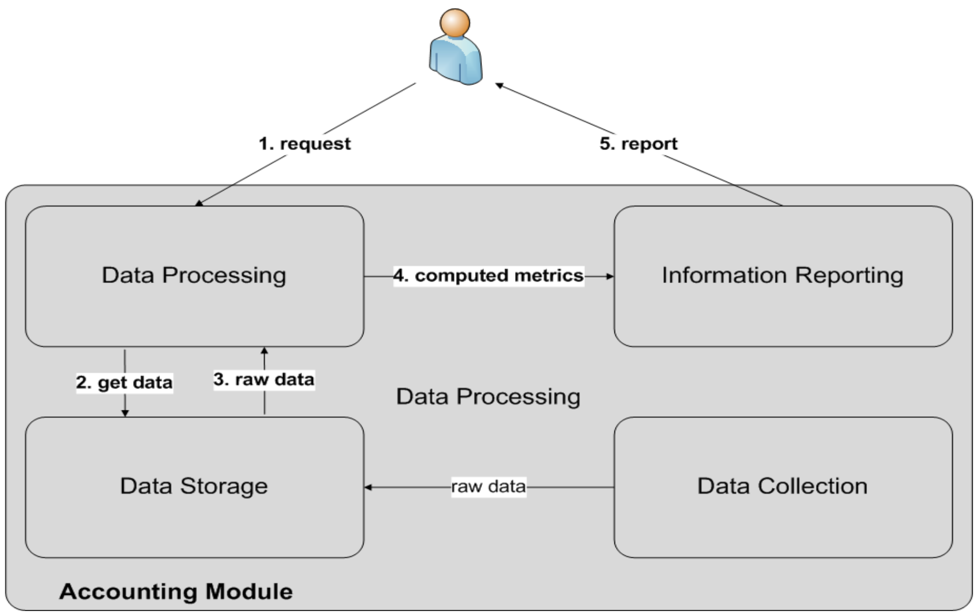
Security Token Service (STS) operation in AuthN and AuthZ



GEMBus Accounting Service (1)

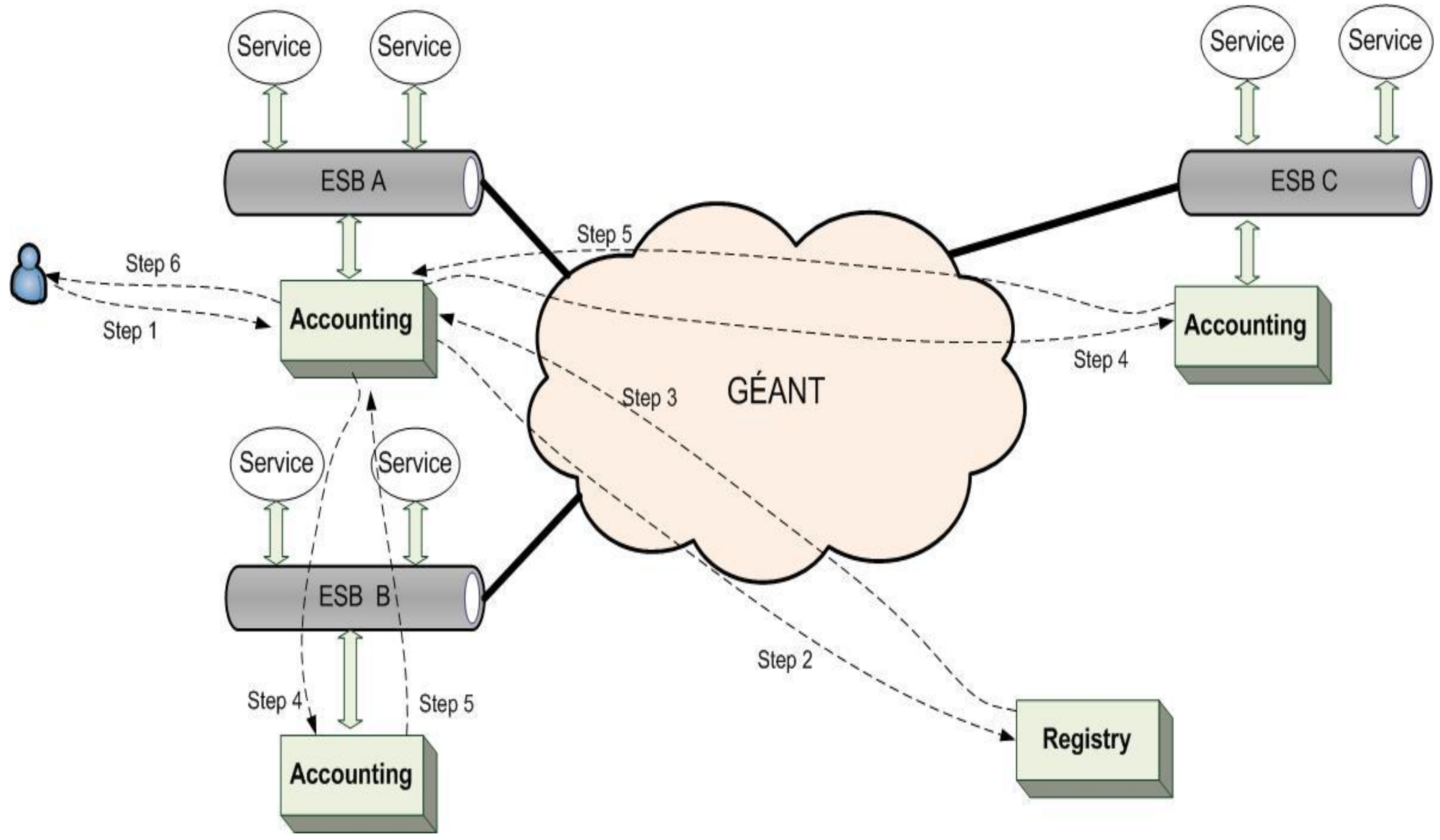
Consists of a Common Accounting Repository and an Accounting Module (AM) deployed at every GEMBus instance. The Accounting Module includes the following main building blocks:

- Data Collection: raw log, event, state data collection from services at ESB level
  - Works asynchronously and triggered each time a service is being called
- Data Storage: stores raw data provided by the Data collection module
- Data Processing: filter/select data for computing specific metrics of interest related to each service
  - Can work both continuously and on request from other services
  - Performs aggregation of accounting data from all available domains



- Information Reporting: produces reports in a human-readable format
  - Can aggregate data from different domains over the monitored GEMBus infrastructure

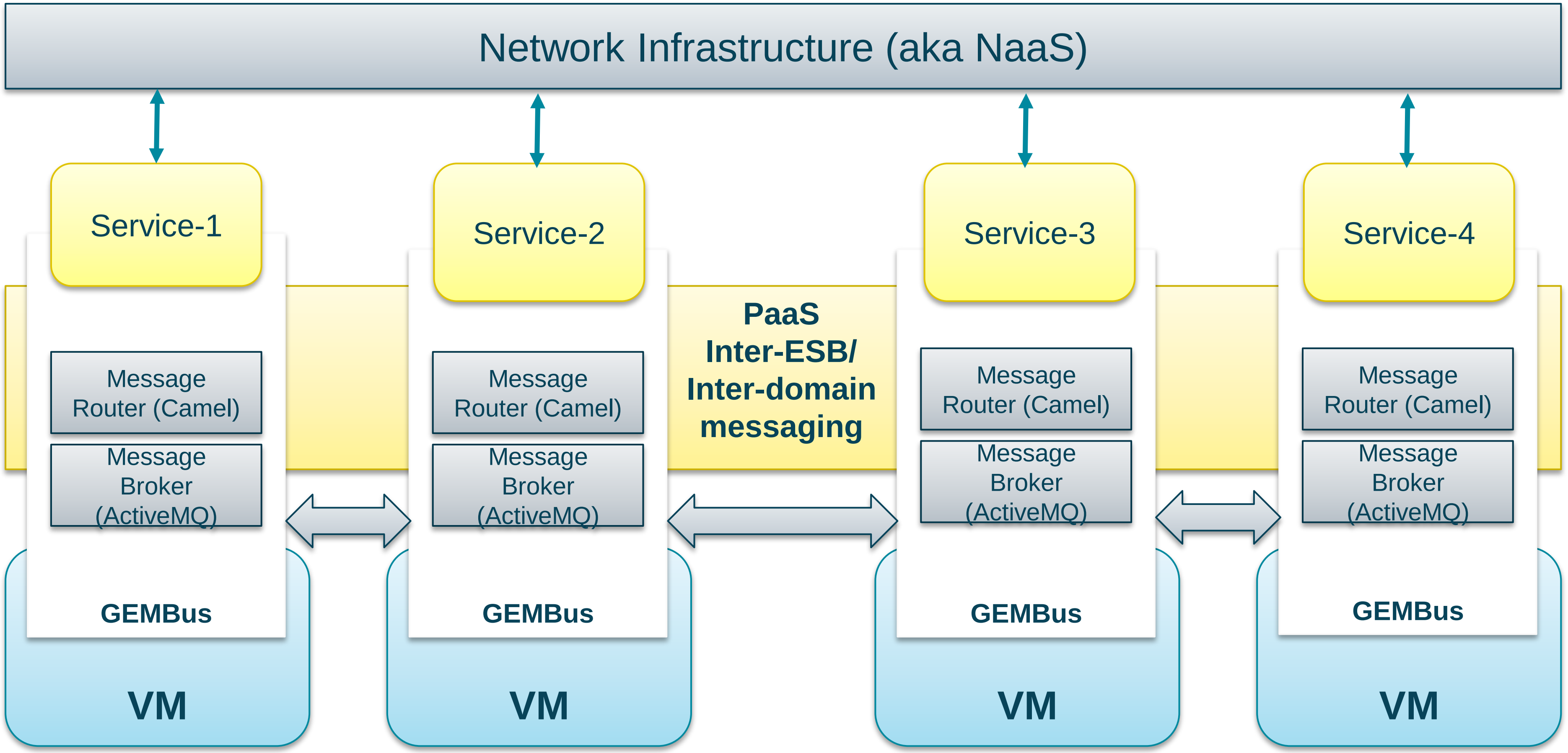
GEMBus Accounting Service (2)



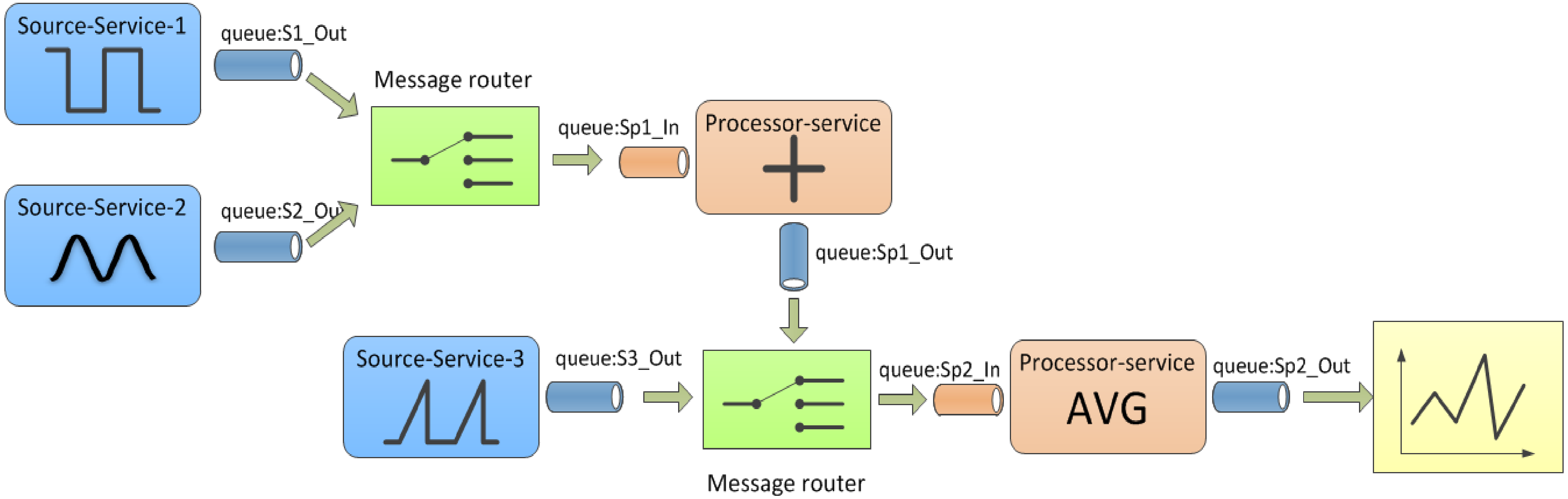
GEMBus/ESB testbed

- Fuse ESB (Servicemix and Fuse Fabric as adopted for clouds): service deployment environment
- Fuse Message Broker: connect message brokers in intra-domains and inter-domains
- Fuse Mediation Router: configurable message routes in in composable services
- Java OSGi Bundles for service deployment
- VM management platform
    - OpenNebula and VirtualBox

Testbed Architecture for GEMBus based Cloud PaaS


Network Infrastructure (aka NaaS)
Service-1
Service-2
Service-3
Service-4
PaaS Inter-ESB/ Inter-domain messaging
Message Router (Camel)
Message Broker (ActiveMQ)
GEMBus
VM

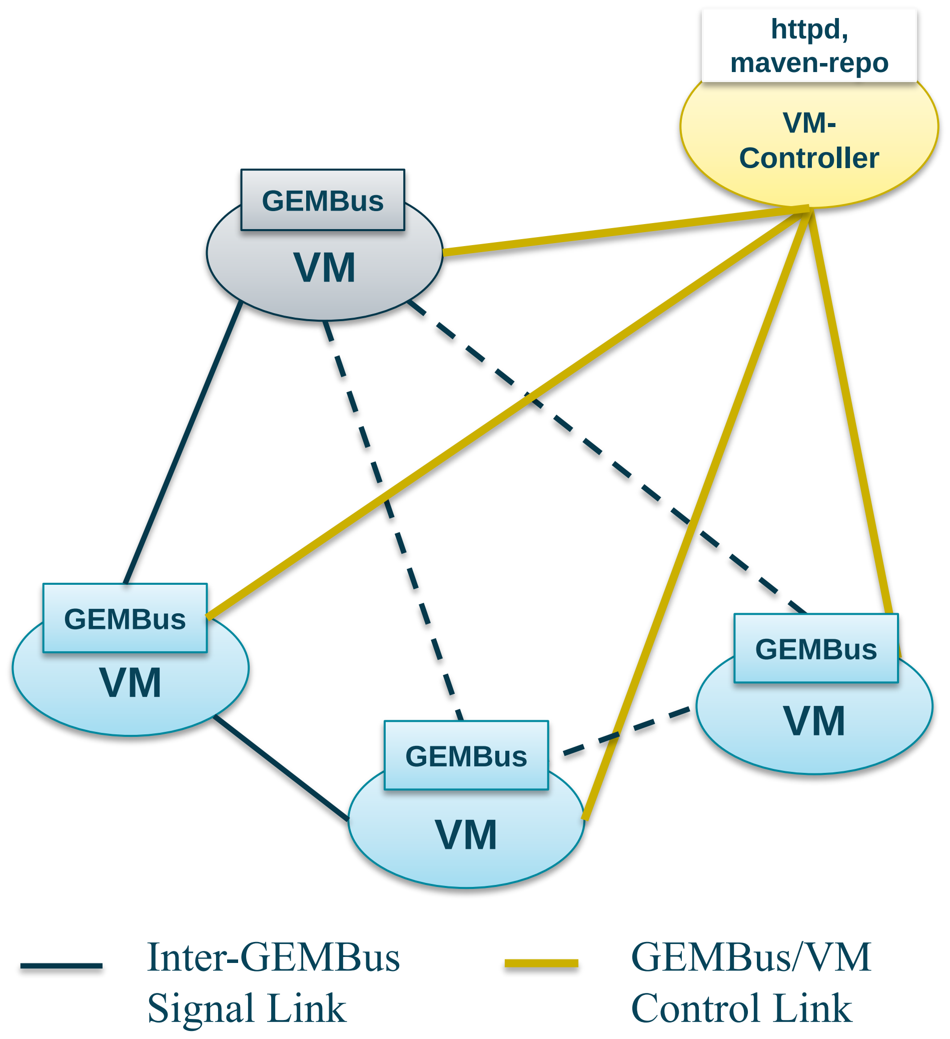
Example Service composition workflow



(A v B v (A+B)) + C => Average service => Visualisation

Presented at SuperComputing 2011

Controller and Network of Brokers



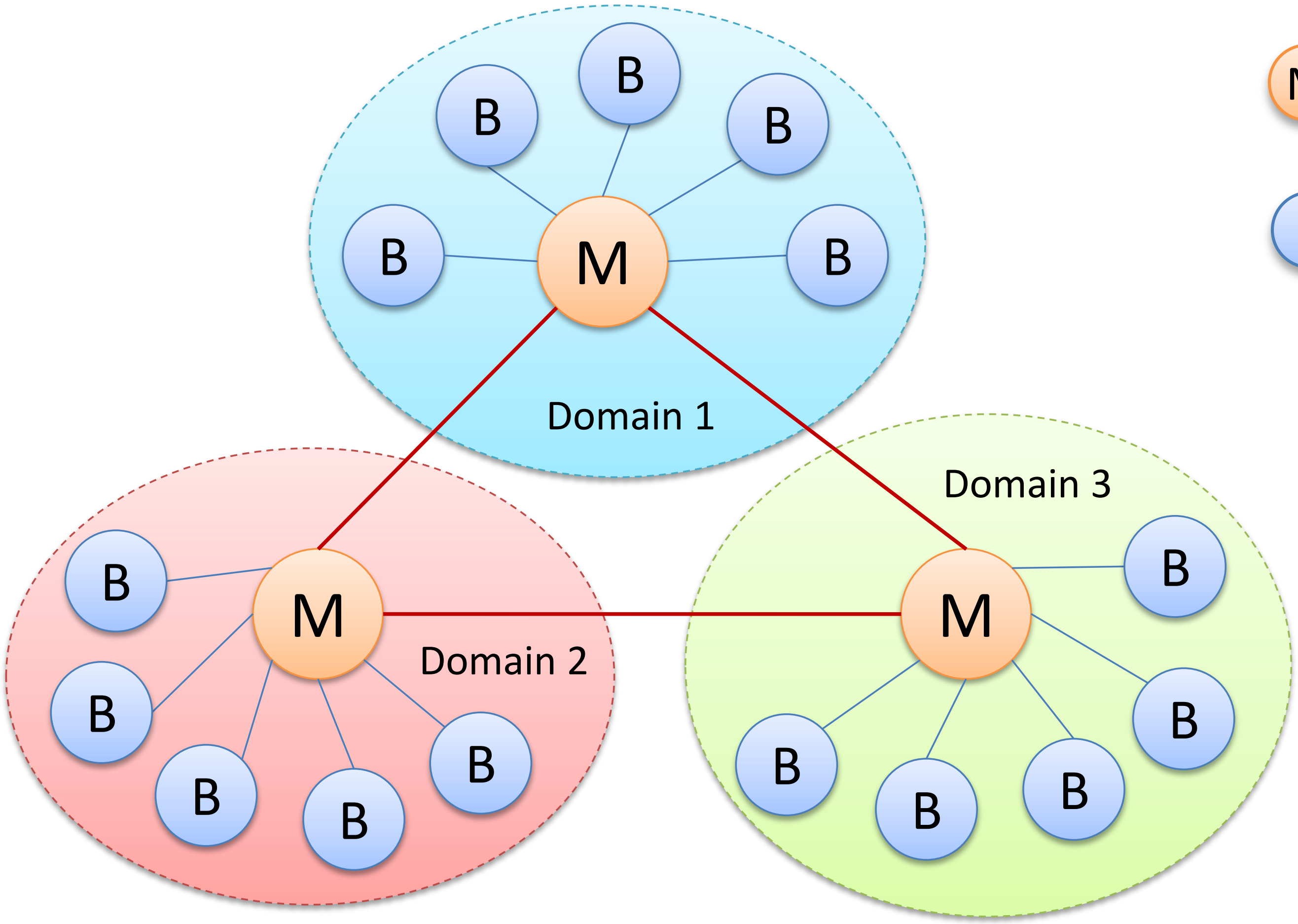
Beans.xml

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

Java DSL

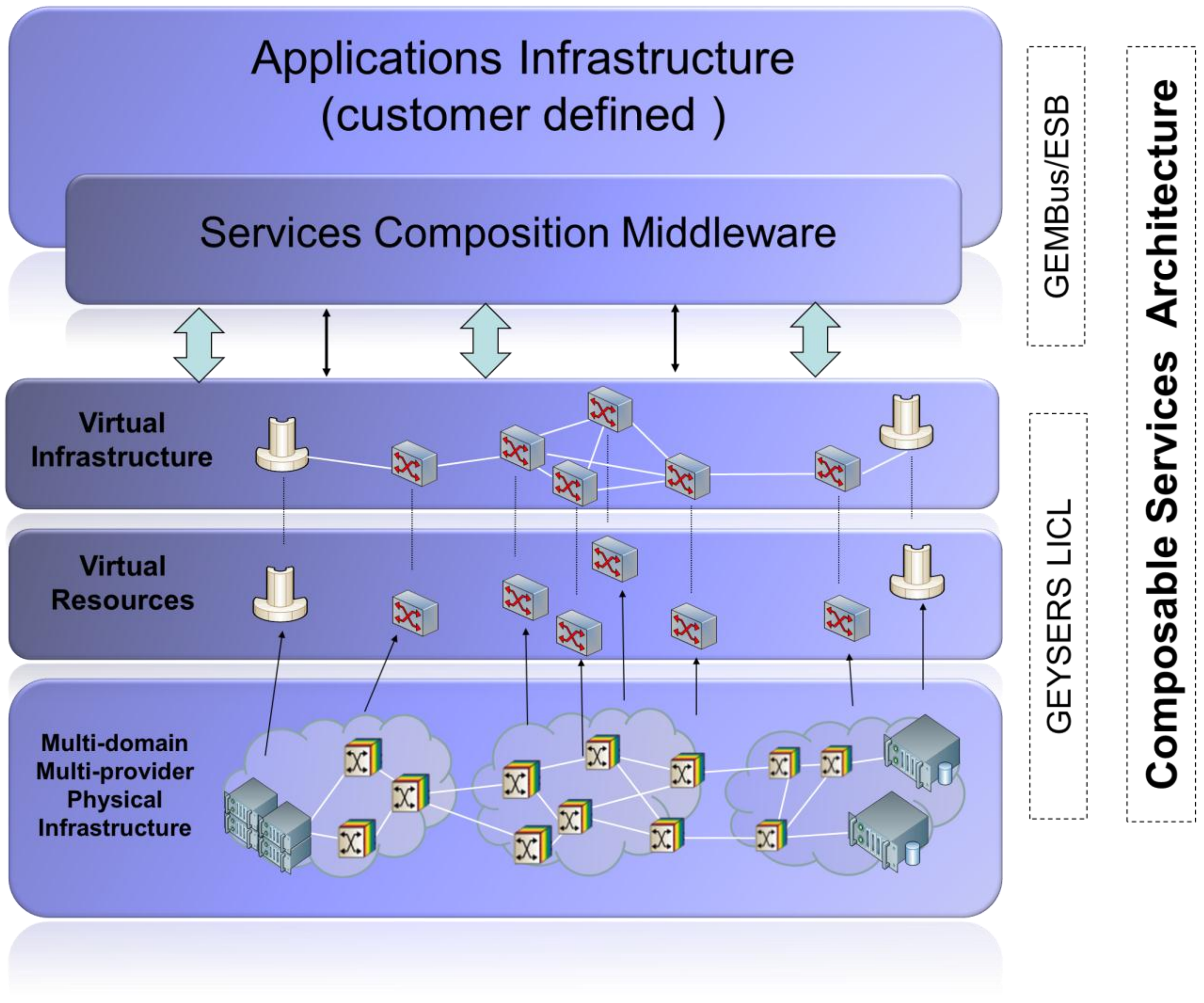
```
from("jms:S1_Out").to("jms:S3_In");
from("jms:S2_Out").to("jms:S3_In");
from("jms:S3_Out").
bean(LogSignal.class,
"log(${body})");
```

Inter/multi-domain message routing/brokering in GEMBus



M	Master broker node to connect administrative domains
B	ESB broker instance to mediate local services

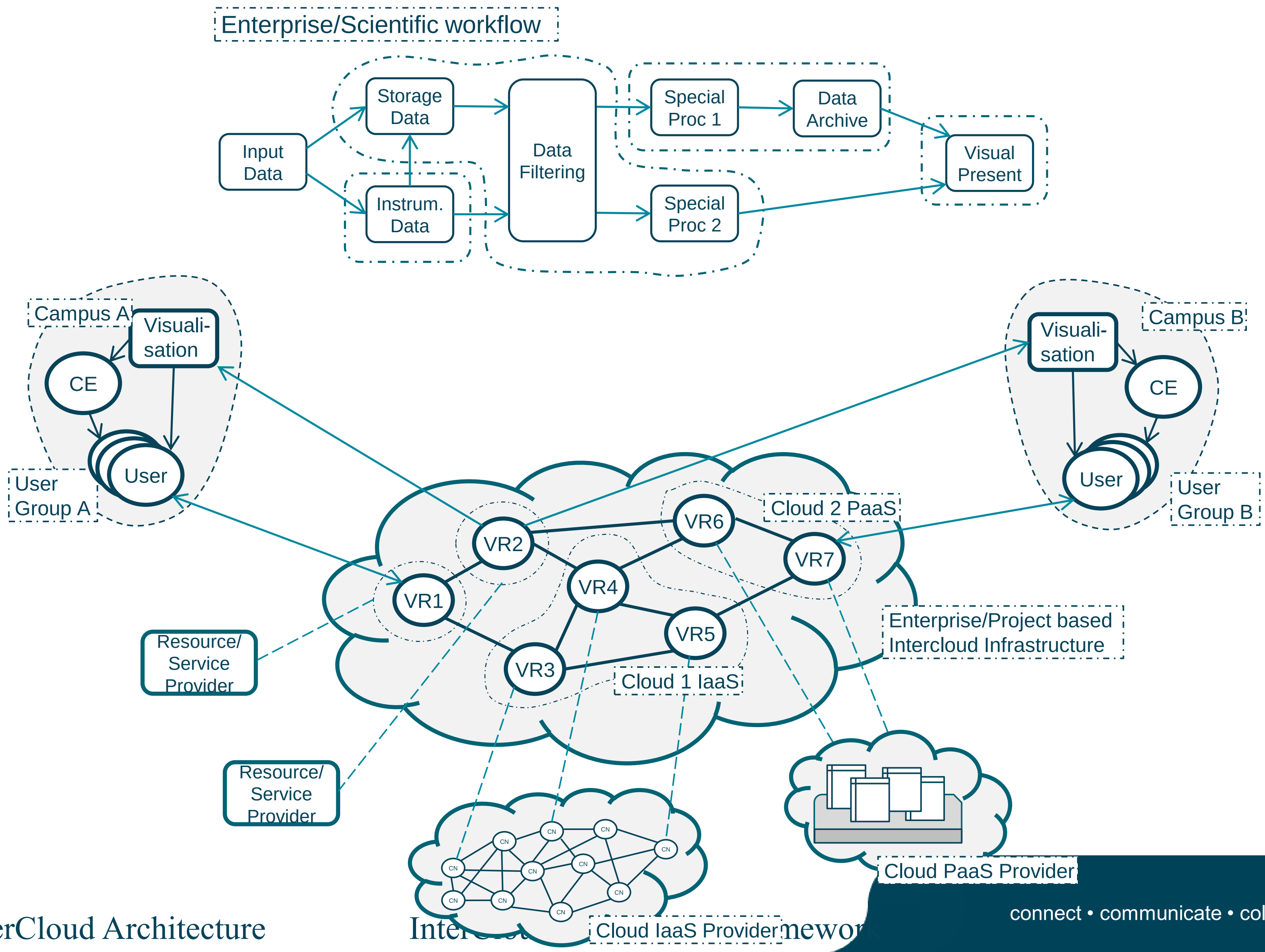
GEMBus in a multi-layer infrastructure services provisioning



Applications Infrastructure (customer defined)
Services Composition Middleware
Virtual Infrastructure
Virtual Resources
Multi-domain Multi-provider Physical Infrastructure
GEMBus/ESB
GEYSERS LICL
Composable Services Architecture

- Network topology/QoS aware infrastructure services provisioning
  - GEYSERS Project technology
- Services composition platform for community oriented services and/or applications
  - GEMBus@GN3 Project

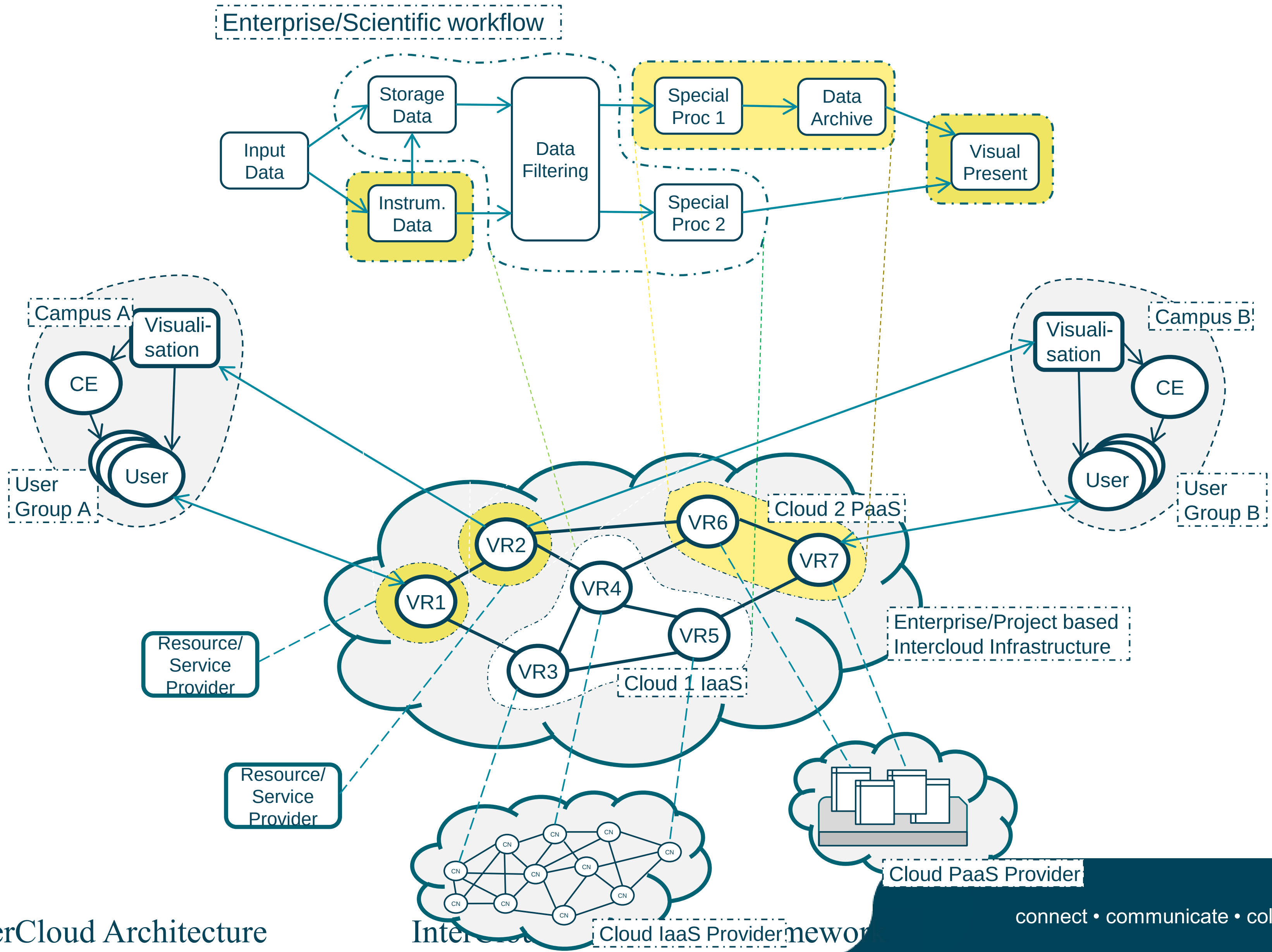
General infrastructure provisioning: Workflow => Logical (Cloud) Infrastructure (1)



InterCloud Architecture

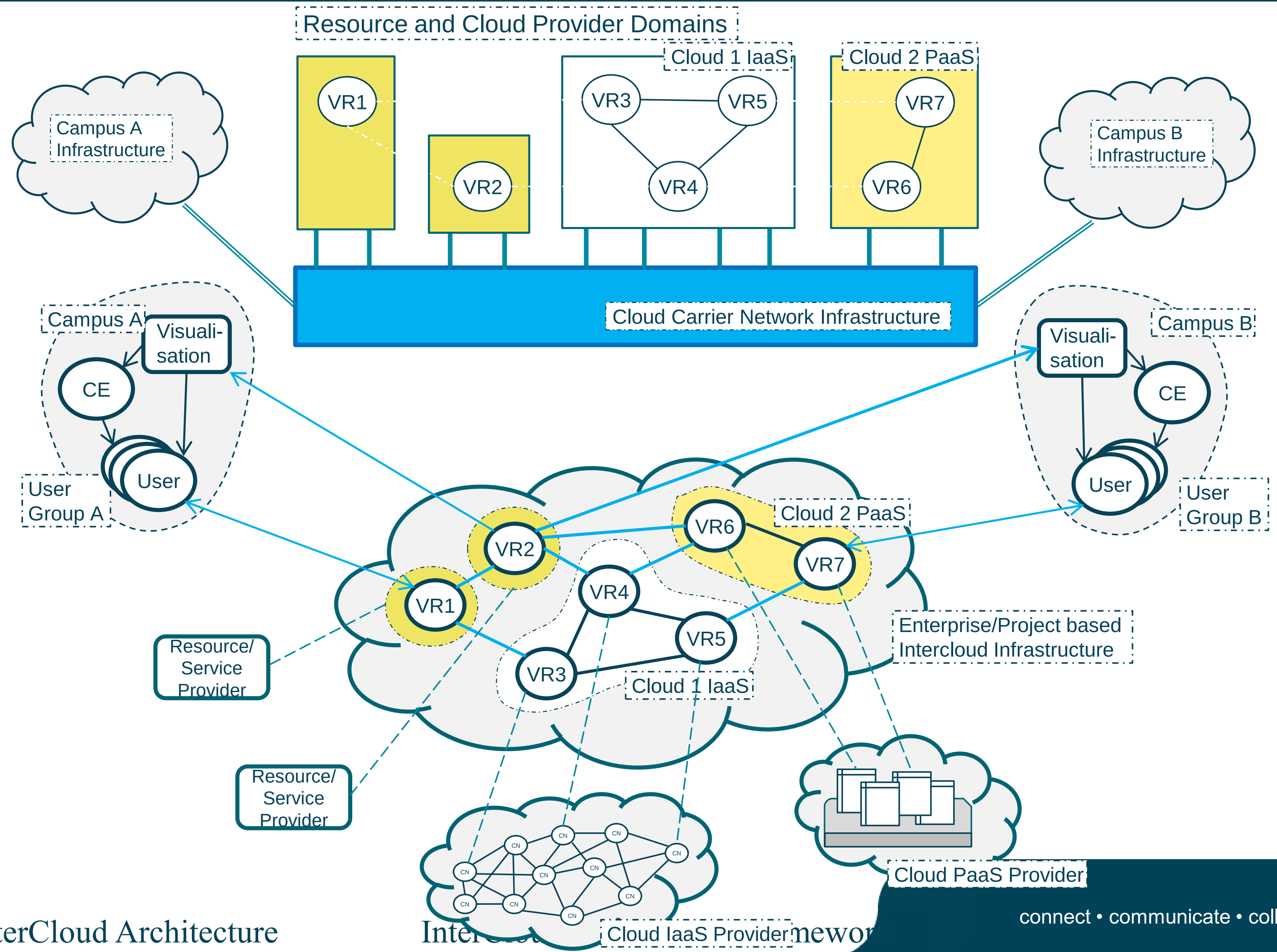
Inter[illegible]mewor

General infrastructure provisioning: Workflow => Logical (Cloud) Infrastructure (2)



InterCloud Architecture   InterCloud ... Framework

General infrastructure provisioning: Workflow => Network Infrastructure



Future work and Discussion

- Promoting GEMBus as a platform Cloud PaaS to wider community
- Research on Semantic services composition based on Semantic Services description and semantic enabled Service Registry
- Support REST/SOAP integration with OAuth2lib and GemSTS and mutual delegation
  - JSON (JavaScript Object Notation) and SOAP token mapping
- Integration of the core GEANT network services
  - Authobahn – On-demand network connectivity provisioning
  - PerfSONAR – Network monitoring system (open source)
  - eduGAIN – Inter-federation Identity management and Access control system (GEANT/NREN oriented)
  - eduPKI – Personal certificates management system for GEANT/NREN

- Questions and Discussion

Semantic Services composition

- Services can be semantically described and populated via GEMBus Registry that uses RDF and ontology to store service descriptions
  - This allows semantic searching with complex queries based on non-trivial attributes and service relations
  - External ontologies can be used via Registry infrastructure federation and mapping
- A composed service may be semantically described in terms of standard ontology. The composition engine will get the semantic description and find the necessary services in the GEMBus Registry by performing the semantic searches.
- Example: We have three services: A service (S1) that produces messages of type A which may be sent as a flow. A transformation service that receives messages of type A, transforms them to type B, and sends them out. And a service (S2) that consumes messages of type B.
  - The composed service may be defined by using a simple description:
  - Connect S1 to S2
  - The composition engine will know that S1 and S2 are incompatible and will automatically request a transformation service from type A to type B.

Additional information

- GEMBus namespace and services naming
- First GEMBus/ESB demo (SC2011) details

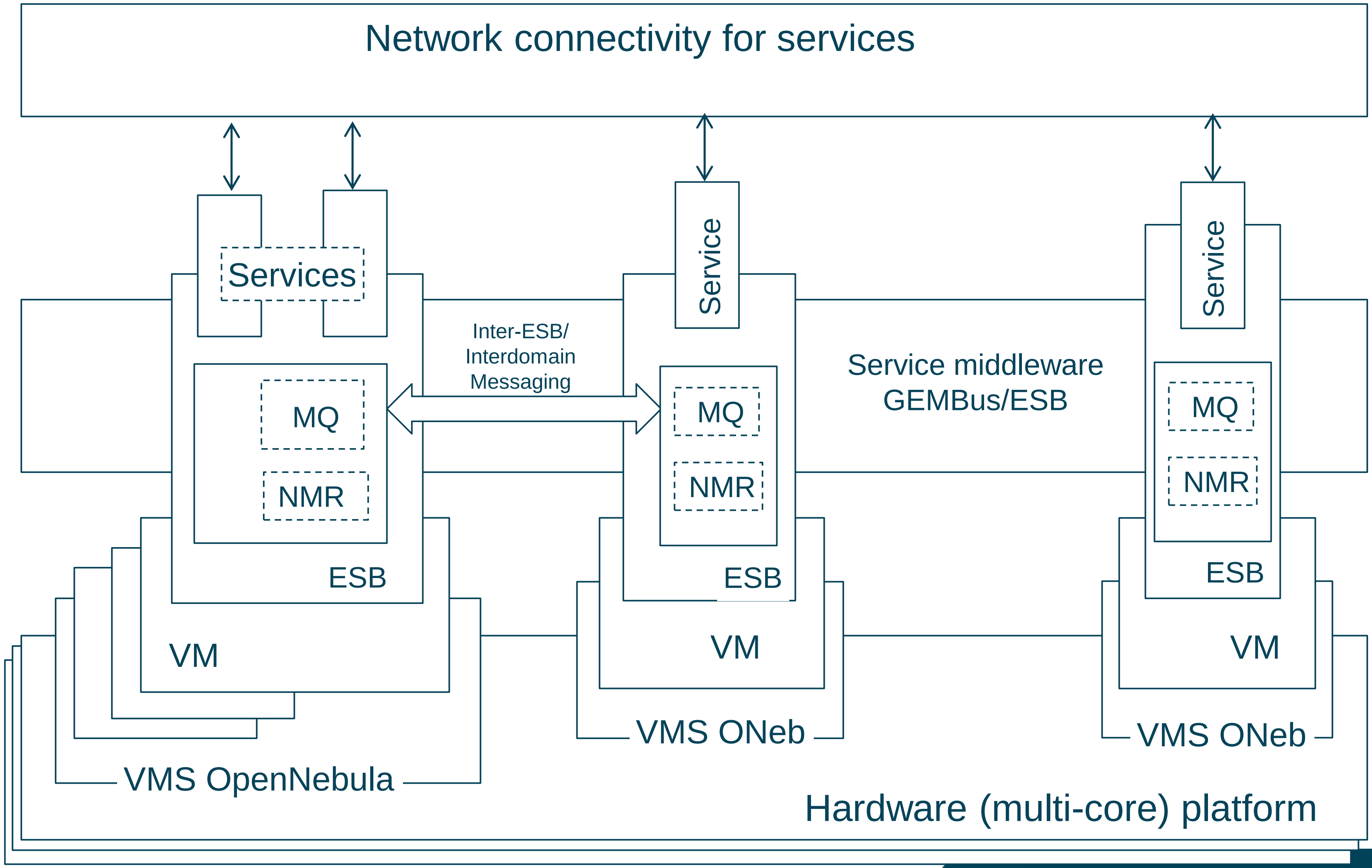
GEMBus Messaging Infrastructure (GMI) Functionality

- Provides configurable message handling and routing functionality for composed services
    - Supports GEMBus namespace and EPR/URI profiles
    - Service and security context handling, including services lifecycle management
        - *Context dependent rules to be defined*
- Primarily uses SOAP with future extension to REST
    - Currently REST doesn't have well defined architecture model
- Built on the top of the standard Apache/Fuse messaging infrastructure
    - Fuse Message Broker (Apache ActiveMQ) messaging processor and cross-domain integration
    - Fuse Mediation Router (Apache Camel) normalised message router
        - *Rule based message routing*
- Message level security and transport level security can be used as part of Fuse/Apache platform

GEMBus Namespace and naming

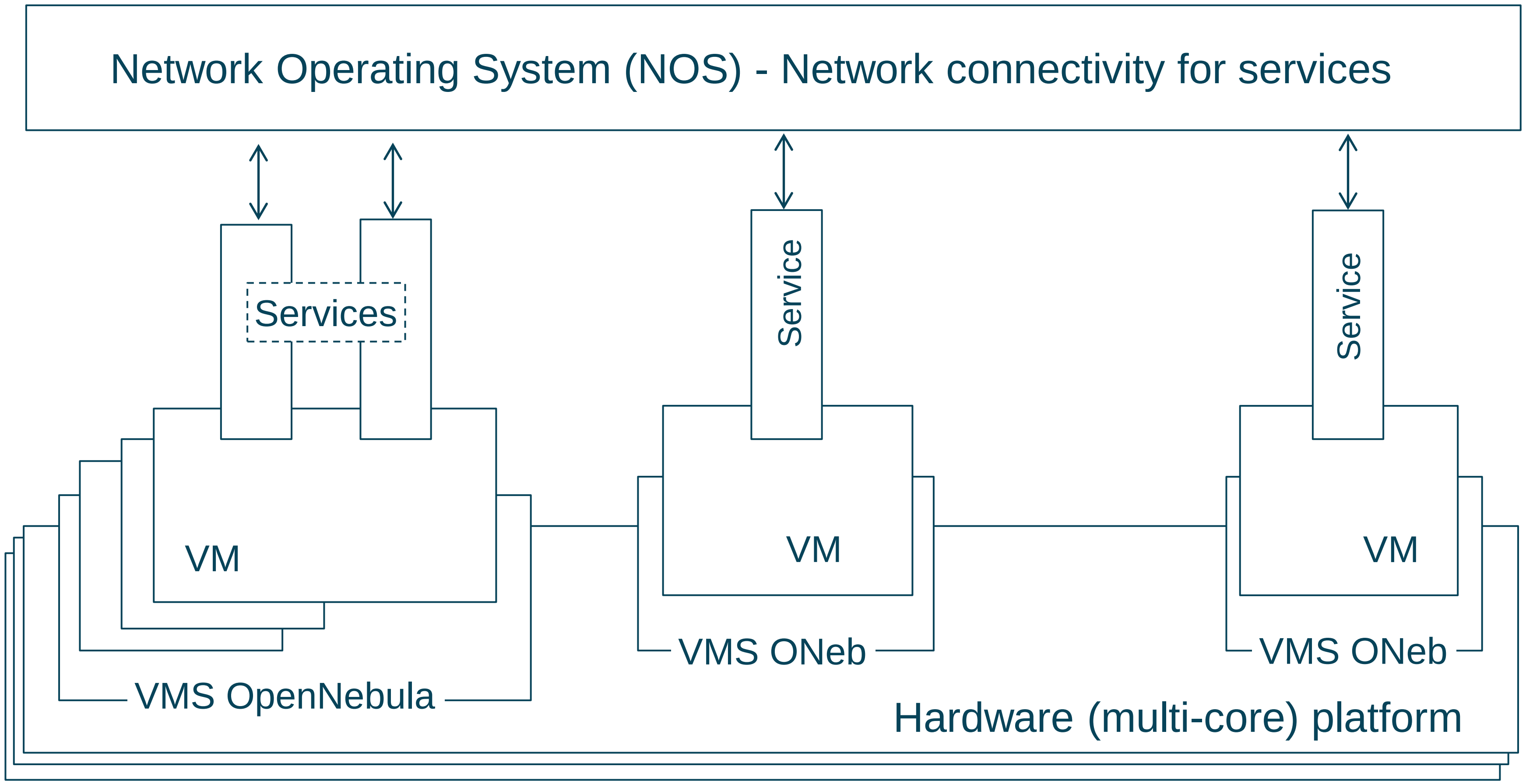
- GEMBus uses designated namespace “urn:geant:gembus” and defines the following initial namespace branches
  - urn:geant:gembus:protocol
  - urn:geant:gembus:service
  - urn:geant:gembus:security
- All GEMBus defined names and attributes should have prefix “gembus”
  - Branches may use specific prefix in a form “gembus-{branch}”
- All GEMBus identifiers should use FQN (Fully Qualified Name) format
  - urn:geant:gembus:service:{service_type}:{attribute_name}
- GEMBus should allow export of external namespaces in a standard way
- GEMBus namespace should be supported by (centralised) GEANT namespace registry

UvA Testbed – Cloud IaaS and GEMBus/CSA



- Planned for SC12

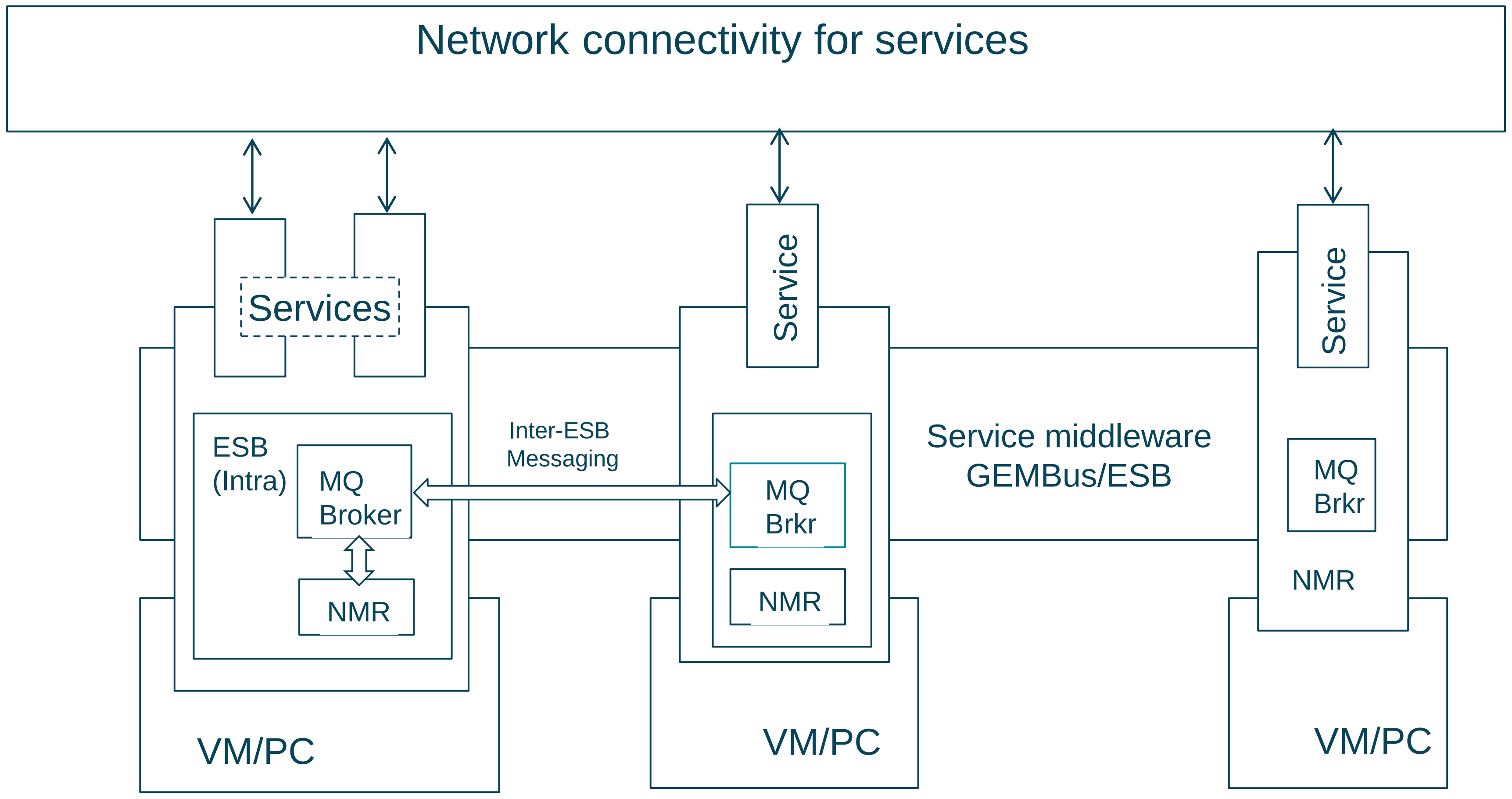
UvA Testbed – phase 1.1 Cloud/Network Infrastructure Layer



Network Operating System (NOS) - Network connectivity for services
Services
Service
Service
VM
VM
VM
VMS ONeb
VMS ONeb
VMS OpenNebula
Hardware (multi-core) platform

- Phase 1.1 to create and test network infrastructure with dummy services

UvA Testbed – phase 1.2 GEMBus/ESB/CSA Service Layer



- Phase 1.2 to create and test inter-ESB messaging

Current Demo setup - Topology

Signal_source_s1 (message producer) -> VM1-Broker (Jupiter) <---> VM2-Broker (Neptune) -> signal_sink_s3 (message consumer)

- S1 = a + bt, where t – time with step = 0.1
- S2 = sin (t)
- S3 = S1 + S2

VM1 Jupiter - Signal sources S1 and S2

- Signal_source_s1 sends messages to message queue “jms:SignalS1”
- Signal_source_s2 sends messages to message queue “jms:SignalS2”
- Jupiter Message Broker registers with Neptune Message Broker

VM2 Neptune – Signal sink and Camel router

- Camel Router – Receives messages from “jms:SignalS1_Out” and “jms:SignalS1_Out” and sends to “jms:SignalS3_In”
- Signal_sink_s3 subscribes messages from a message queue “jms:SignalS3_In”

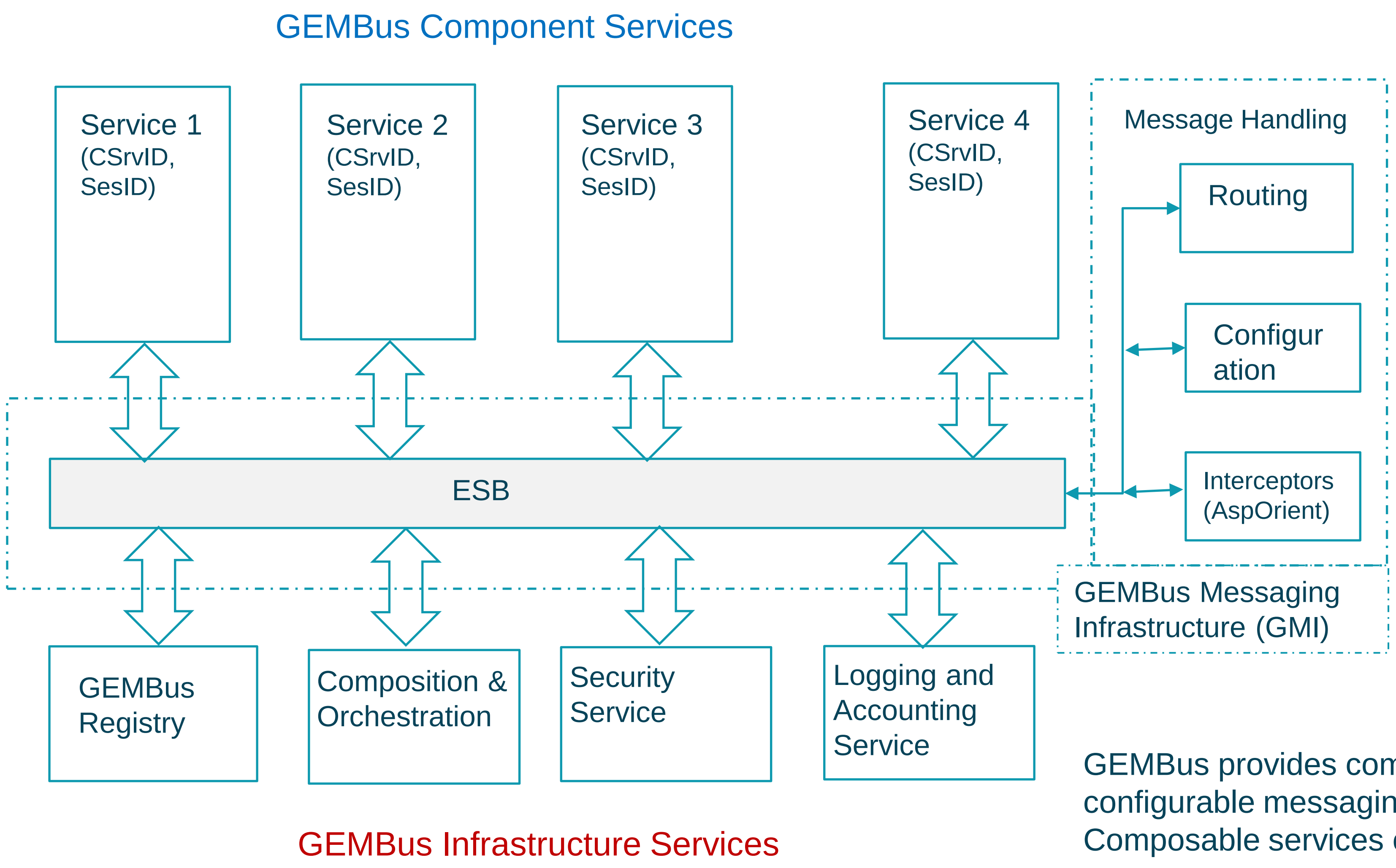
ActiveMQ Message Broker Configuration

```
<blueprint ...>
  <broker ...>
    <networkConnectors><networkConnector name="ncNeptue“
                    uri="static:(tcp://192.168.56.102:62001)“ duplex="true">
      </networkConnector></networkConnectors>
    <transportConnectors>
      <transportConnector name="openwire" uri="tcp://localhost:61616"/>
      <transportConnector name="stomp" uri="stomp://localhost:61613"/>
    </transportConnectors>
  </broker>
  <bean id="activemqConnectionFactory"
class="org.apache.activemq.ActiveMQConnectionFactory">
        <property name="brokerURL" value="tcp://localhost:61616" />
  </bean>
  <service ref="pooledConnectionFactory"
interface="javax.jms.ConnectionFactory">
    <service-properties>
      <entry key="name" value="localhost"/>
    </service-properties>
  </service>
</blueprint>
```

Camel NMR Router Configuration

```
<beans ….>
<bean id="jms" class="org.apache.camel.component.jms.JmsComponent">
  <property name="connectionFactory">
    <bean class="org.apache.activemq.ActiveMQConnectionFactory">
      <property name="brokerURL" value="tcp://localhost:61616"/>
    </bean>
  </property>
</bean>
<camelContext xmlns="http://camel.apache.org/schema/spring">
  <!- Route for S1 messages -->
  <route>
    <from uri="jms:S1_Out"/>
    <to uri="jms:incomingQueueS3"/>
  </route>
 <!- Route for S2 messages -->
  <route>
    <from uri="jms:S2_Out"/>
    <to uri="jms:incomingQueueS3"/>
  </route>
</camelContext>
</beans>
```

GEMBus Messaging Infrastructure (GMI) in overall GEMBus architecture


GEMBus Component Services
Service 1 (CSrvID, SesID)
Service 2 (CSrvID, SesID)
Service 3 (CSrvID, SesID)
Service 4 (CSrvID, SesID)
Message Handling
Routing
Configur ation
Interceptors (AspOrient)
ESB
GEMBus Messaging Infrastructure (GMI)
GEMBus Registry
Composition & Orchestration
Security Service
Logging and Accounting Service
GEMBus Infrastructure Services

GEMBus provides common (dynamically) configurable messaging infrastructure for Composable services communication