Towards service awareness and autonomic features in a SIP-enabled network

Guillaume Delaire
Laurent Walter Goix
Giuseppe Valetto

Telecom Italia Lab

Outline

- Context and Approach
- Introducing Service Awareness in NGN: the Service Bus
- Towards AC scenarios
- Open issues and future work

The Context

- Telecom Italia is deploying an NGN service platform based on SIP protocol
  - SIP-based platform developed in TILAB
  - Already used for Alice Mia service

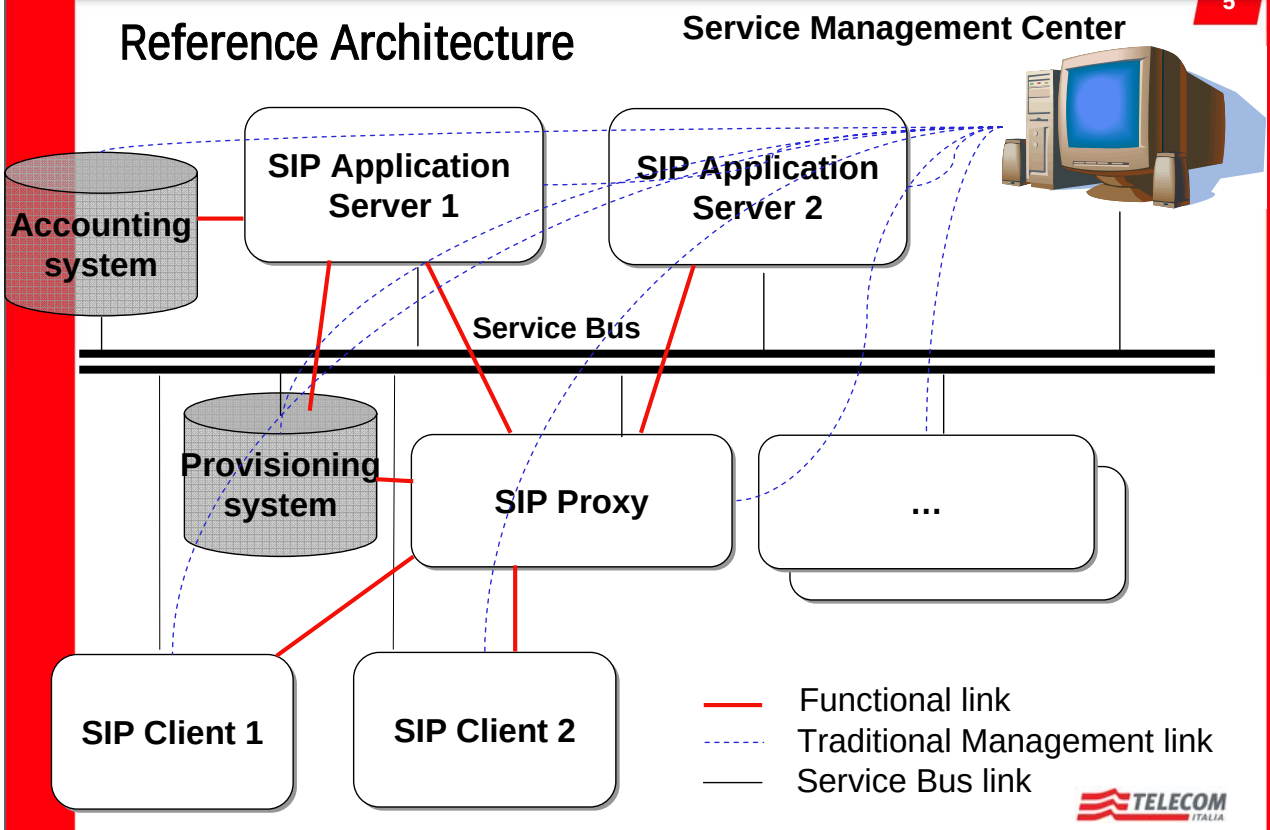
- Research has been focusing on Adaptive Service Execution, Delivery and Management issues over such a network
  - Service Delivery and Management is nowadays very complex in the operator's network
    - Various 'service items' to be deployed and managed: service logic, triggers, accounting, provisioning, etc
    - Heterogeneous resources such as Application Servers, proxies, etc.
  - NGN supposes numerous short-lived services and high dynamicity in providing and using them
  - Service Execution also need to adapt (routing, composition, binding, etc)

The Approach

- Achieve an environment that adapts and optimizes services (and the network) based on users' demand
  - Fast Service Delivery and Management mechanisms over heterogeneous resources
  - Automation of Service Lifecycle to avoid human maintenance!
  - Self-configuration of the control plane and the execution environment

- Introduce awareness at service level in the network
  - Starting from human-based management and basic optimizations…
  - …Evolving towards Autonomic Communication!

Reference Architecture



Introducing Service Awareness: the Service Bus

- The Service Bus has been designed to share service-level information within a SIP-based network.
  - Provides a fast service delivery environment over distributed, yet heterogeneous network resources
  - Provides a real-time support for monitoring of services throughout all resources either for management or functional actions (e.g. routing, service logic, etc)
  - Notifies end-users in real-time about service availability together with configuration information to properly access them
- The Service Bus implementation supports
  - Deployment of any service-related information (service logic, trigger, configurations, accounting, provisioning, etc)
  - Monitoring of services (and resources)
  - Advertising of service information to end-users or other services for spontaneous aggregation

The Service Bus is already used by TI's SIP-based platform…for service management

Service Bus: main concepts

- It is based on the SIP Event Framework as
  - Event-based middleware based on publish/subscribe/notify
  - NGN already relies upon SIP and most resources have a native SIP stack
    - Minimum impact on the network (native to control plane)
    - No need to introduce ad-hoc protocols
  - Encompasses end-user devices
- Data exchanged as XML to be extensible
- Each network resource has
  - One SIP URL to be uniquely identified
  - One 'type', which also defines a set of XML extensions and type-specific information
  - Automatically subscribes to all its deployment events at bootstrap
- Deployment supports distributed workflows (dependencies) and feedbacks
  - Collaboration between peers
- Monitoring allows for service/network discovery
  - Service data aggregation/composition

TELECOM ITALIA

Towards Autonomic Communication

- Investigate use cases of Autonomic Communication in the context of a SIP-based network providing Value-Added Services.
  - Adaptation and optimization of services across nodes based on user's demand
  - Self-management of the NGN Platform to handle faults and traffic peaks and save time and money…
- At Network level
  - The network is self-adjusting, based on internal and/or external stimuli (e.g. the service usage), adding or removing services from network elements
- At Control level
  - Each network element has a set of embedded autonomic behaviours (intra-node)
  - Application Servers and Proxies collaborate to achieve autonomic behaviours on the network (inter-nodes)
    - It may require additional dedicated network elements (supervisors)
- At Service level
  - Services are "passive": service logic is influenced by autonomic behaviours (e.g. dynamic binding, configuration/deployment)
  - Service are "active" autonomic elements that trigger some feature in the network (e.g. self-aggregation, dynamic routing)

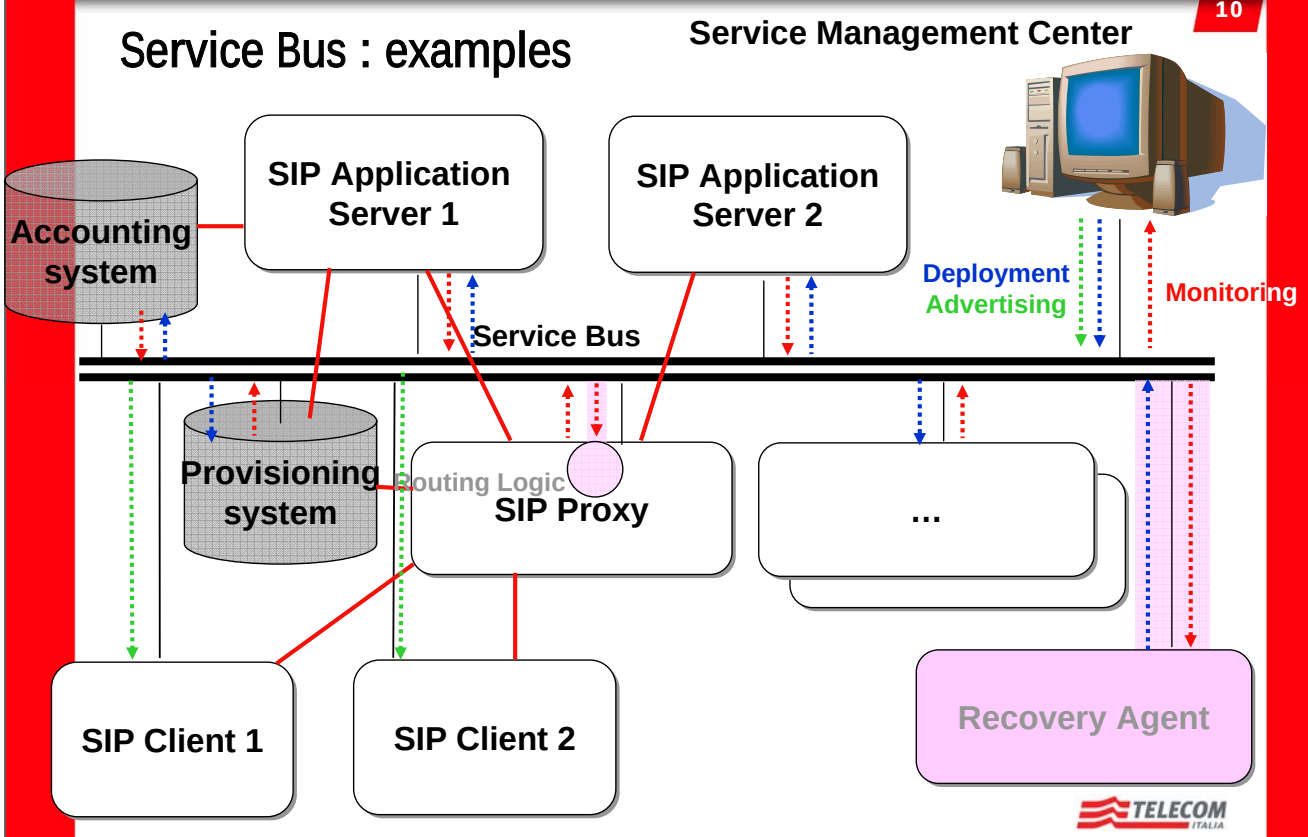
TELECOM ITALIA

Service Bus : some examples of enabled behaviours

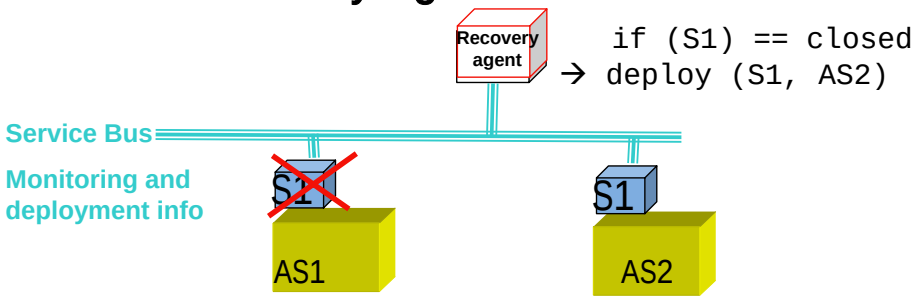
- Fault recovery and fault avoidance (self-healing/self-protection):
  - Through the Monitoring event, SIP Application Servers and Proxies publish various types of service- or network-level problems, which can be triggered for any reason (CPU overload, traffic load, spam, DOS attack, etc)
  - Any entity embedding a specific logic can subscribe to such events and use the Deployment module to act back on the network, for example to deploy the same service on a different Application Server or restart the service on the failing node

- Dynamic load balancing/Network 'breathing' (self-optimization)
  - Whenever a traffic peak for a given service happens on an Application Server that it cannot handle, any entity can publish a workflow to the network to deploy the same service on other Application Servers, and have the triggers on proxies reconfigured accordingly. Whenever a service is not much used, it can be removed from some Application Server.

Service Bus : examples



Scenario 1: External Recovery Agent



- The Recovery Agent is a preconfigured logic that monitors the status of a service across multiple AS nodes and can act on them to optimize load or repair faults.
  - Eventually this element can be embedded in any node, i.e. AS o proxy

- Whenever service S1 is not available on AS1 anymore (node overload, bug, etc.), the recovery agent re-deploys and activates S1 on AS2, which is serving other services
  - Monitor: RA checking for service status across the network (Monitoring)
  - Sensor: service state on AS1 (Monitoring)
  - Analyze: AS1 is last node to offer the service -> need to deploy service on other node -> AS2
  - Plan: deployment/activation of S1 on AS2
  - Execute: AS2 install service logic and activate it

TELECOM ITALIA

Scenario 2: Dynamic Load Balancing

- In NGNs, proxies route SIP messages to AS nodes based on some static information.

- They can do 'static' load balancing while routing towards various AS nodes

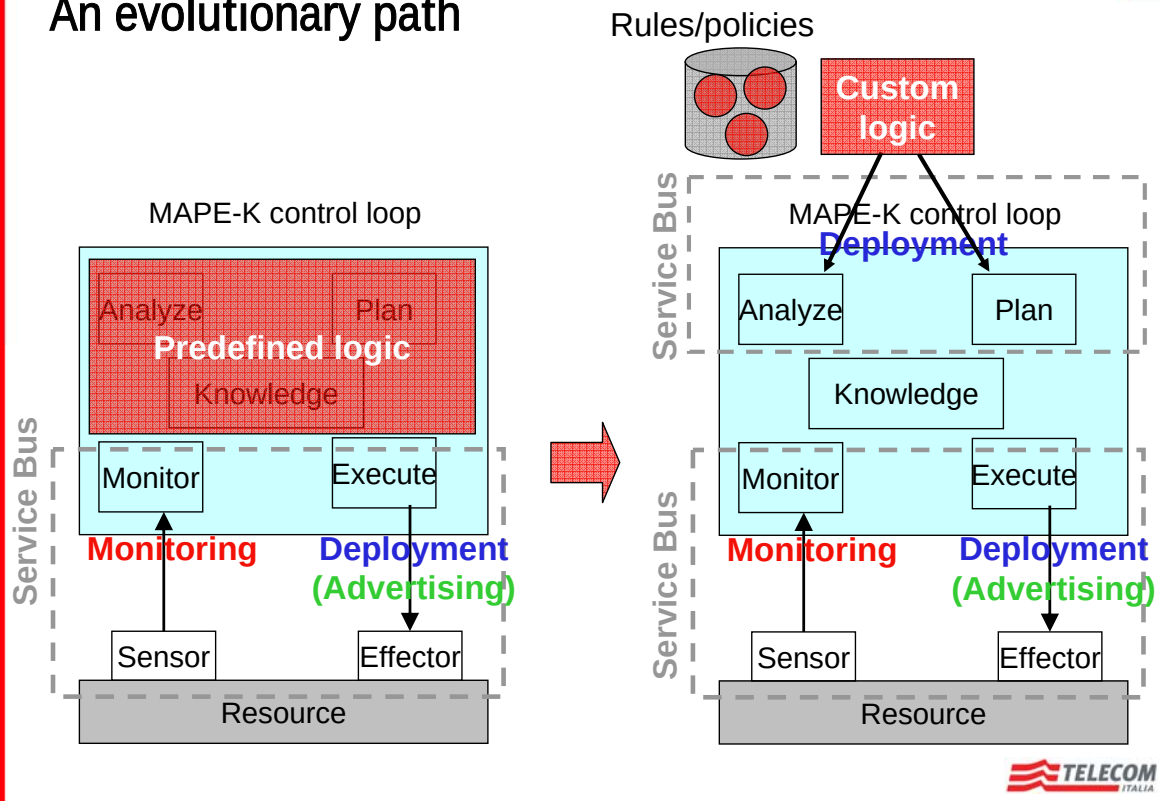
- Using monitoring module we can monitor the status of services on specific AS nodes to optimize load balancing algorithms

- Furthermore we can dynamically provision the set of AS nodes to be contacted by proxies (proxy subscribe to AS nodes serving a specific service)

- Similar techniques can be applied between proxies and end-users devices through the advertising module

TELECOM ITALIA

An evolutionary path



Open issues and future work

Enhance the autonomic communication

-Enrich the awareness on sensors
- Service-related information (statistics, load, etc) and NE-information (CPU, memory, etc.)

-Improve the dynamicity of behaviours
- Rule engine to run autonomic logic as policies that can be deployed dynamically
- Aspect-Oriented Programming: aspects to be used/deployed to modify behaviour of nodes in sensing or effecting.

-Investigate communication paradigms between nodes
- p2p using SIP
- Improve involvement of end-user devices
- Improve support for spontaneous service aggregation and composition

Add scenarios and use cases

-Work on embedded autonomous logic/scenarios useful for NGN services management

-Build a library of predefined policies, deployment actions and aspects

-Investigate a hierarchical procedure for AC scenarios: internal, collaborative, supervisor