University of Coimbra
Departament of Engineering Informatics
Communications and Telematics Laboratory

D4.0: Global Business Layer Framework

Document

Title	Global Business Layer Framework
Type	Deliverable
Nature	Public
Creation Date	2007-04-09

Authors

Alexandre Veloso de Matos
Fernando Menezes Matos

Revision History

V 1.0	Launched at 2007-04-09
V 2.0	Launched at 2007-04-17
V 3.0	Launched at 2007-05-17
V 4.0	Launched at 2007-06-22

Abstract

Next Generation Network (NGN) encompasses a new usage of voice and data services which leads to service tending to be separated from transport functions. This feature allows both to be offered separately what is a first important step in order to service providers reach a global context.

This leads to an improvement in the amount of service available and in inevitable complexity. Because this, may occur situations where a service is too complex that one single provider cannot provide it.

IPsphere is a framework that is a subject of standardization and this deliverable is about an implementation of IPphsere proposal that supports end-to-end service provisioning. In order to discuss its features we based our tests on an inter-domain VPN provisioning problem.

Keywords

Business, End-to-end service, architecture, IPsphere, service offer, SLA, SLS, SOA.

Table of Contents

1 Introduction 6
2 Service Oriented Architecture 6
3 Service Oriented Computing 7
3.1 Enterprise Service Bus 7
4 Next Generation Networks 8
4.1 Framework Objectives 9
5 Related work 9
5.1 Integration perspective 9
5.2 End-to-end service delivery 10
5.3 Service provisioning 10
5.3.1 Service Structuring Stratum 12
6 Global Business Architecture 13
6.1 Service Structuring Stratum 14
6.1.1 SSS Information Level 14
6.1.1.1 Publisher 15
6.1.1.1.1 PublishManager 15
6.1.1.1.2 RegistryServiceManager 15
6.1.1.1.3 UDDIManager 15
6.1.1.2 Order Management 16
6.1.1.2.1 ServiceRequestor 16
6.1.1.2.2 ServiceOfferAccess 16
6.1.1.3 SMS Admin 16
6.1.1.3.1 SMSAdminManager 16
6.1.1.3.2 ServiceBuilder 17
6.1.1.3.3 InterDomainVPNBuilder 17
6.1.1.3.4 ElementSort 17
6.1.1.4 SMS Parent 17
6.1.1.4.1 SMSParentManager 17
6.1.1.4.2 SMSParentNotification 17
6.1.1.5 SMS Child 17
6.1.1.5.1 SMSChildManager 18
6.1.1.5.2 SMSChildNotification 18
6.1.1.6 Logging 18
6.1.2 SSS Functional Level 18
6.1.2.1 Service start solicitation 18
6.1.2.2 Service and element publishing 19
6.1.2.3 Service order creation 19
6.1.2.4 Service order activation 20
6.1.2.5 Service configuration/activation 21
6.2 Service Presentation and Ordering 22
6.2.1 SPO Information Level 22
6.2.2 SPO Functional Level 24
6.2.2.1 Customer searching a service 24
6.2.2.2 Customer ordering a service 24
6.2.2.3 Customer starts a service scheduled 25
6.2.2.4 Customer verify order service status 26
6.2.2.5 Service Architect synchronizes service offers 26
6.2.2.6 Service Architect monitors services ongoing 27
6.3 Policy and Control Stratum 28
6.3.1 First challenge: management paradigm 28
6.3.2 Second challenge: information representation 29
6.3.3 Third challenge: communication model 29
6.3.3.1 PCS Component Diagram 30
6.3.3.2 AAA Server 31
6.3.3.3 Policy Management Agent 31
6.3.3.4 Policy Decision Point 32
6.3.4 PHS Functional Level 32
6.4 Packet Handling Stratum 33
6.4.1 PHS Information Level 34
6.4.1.1 PHS Component Diagram 34
6.4.1.2 Policy Enforcement Point 35
6.4.1.3 Resource Agent 35
6.4.2 PHS Functional Level 36
7 Conclusions 38
8 References 39

List of Figures

Figure 1 - IPsphere Architecture 11
Figure 2 - IPsphere SSS organization. 12
Figure 3 - Global Business Framework 14
Figure 4 - Business layer component diagram 14
Figure 5 - Publisher class diagram 15
Figure 6 - Order Management class diagram 16
Figure 7 - SMS Admin class diagram 16
Figure 8 - SMS Parent class diagram 17
Figure 9 - SMS Child class diagram 18
Figure 10 - Logging class diagram 18
Figure 11 – Service start solicitation 18
Figure 12 - Service and element publishing 19
Figure 13 - Service Order Creation 20
Figure 14 - Service Order Activation 21
Figure 15 - Service Configuration/Activation 21
Figure 16 - Components of Portal B2C 23
Figure 17 - AEE component classes 23
Figure 18 - SPO Use Cases 24
Figure 19 - Customer search service. 24
Figure 20 - Customer ordering a service. 25
Figure 21 - Customer starts a service scheduled by him 26
Figure 22 - Customer verifies order service status 26
Figure 23 - Service Architect synchronyzes service offers with public UDDI 27
Figure 24 - Service Architect monitoring services. 28
Figure 25 – PCS Overview 30
Figure 26 - PL Components 30
Figure 27 - Service Management classes 31
Figure 28 - Policy Management classes 32
Figure 29 - Policy Decision Point classes 32
Figure 30 - Scenarios for use of PCS 33
Figure 31 - PHS Architecture 34
Figure 32- IFL Components View 35
Figure 33 - PEP Informational View 35
Figure 34 - Resource Agent Classes 36
Figure 35- IFL Functional Overview 37

List of Tables

Table 1 - ESB capabilities for Global Business Framework. Adapted from [12] 8
Table 2 - Message description at Service start solicitation... 19
Table 3 - Messages description at service and element publishing 19
Table 4 - Messages description at service order creation 20
Table 5 - Messages description at service configuration/activation 21
Table 6 – Use case descriptions for PL architectural element 32
Table 7- IFL Scenarios 36

1 Introduction

Nowadays, we are seeing an increasing involvement in technology and this involvement is occurring rapidly. This leads to an improvement in the amount of service available and also at their complexity. Because this, may occur situations where a service is too complex that one single provider cannot provide it. One solution for this is to split this service in smaller portions which can be provided by different providers, thus the compound of these smaller portions will build the entire service.

But to do that, the providers which cooperate, need to establish private and secure connections between them to guarantee the service provisioning with its requirements. One of the most used solutions to achieve these constraints are the inter-domain VPNs. Nowadays to establish an inter-domain VPN it is necessary certain human intervention to: communicate resources location and configuration parameters; effectively configure resources and also to start service provisioning. All these constraints made inter-domain VPNs an eligible service to be handled at this deliverable. The framework here presented will be based at this case; however the conclusions can be, in the future, applied to other service provisioning.

We verified that Service Oriented Architecture (SOA) concepts would be very important to make the problem clarified, so this discussion will be presented at section 2. SOA is the most important approach to implement applications based on heterogeneous and distributed environments [1]. This SOA's feature guided the advent of the Service Oriented Computing paradigm (SOC) [2] [3], maybe the first attempt to make service integration a standard at all. SOA and SOC are two core concepts used at this framework and this importance is expressed at section 3. At section 4 we present the modern concept of service expressed at Next Generation Networks (NGN) e also the standardization efforts we considered here in order to present the problem statement. At section 5 we present the contribution of other three related works: Agave Project [4], Tequila Project [5] and IPsphere Forum initiative [6]. After this, at section 6 we present the organization of the specification part of this document, at this section we also present some terminologies and concepts used at the framework. Section 7 presents some conclusions from the ongoing implementation of this framework. Section 8 states some future contributions of this framework and finally section 9 presents the references.

2 Service Oriented Architecture

The first attempt to bring to Internet a business perspective occurred over the client/server model. At this moment, services could be seen as monolithic elements maintained by providers whose complexity was managed by only this provider. This strategy has a critical drawback [7]: service providers provide services with its own engines and generally in a limited reach, it means services could not reach all possible customers. From the point of view of business at all, this is a bad characteristic. However, nowadays, just as in the real world services can be a product from a chain of providers [8] that work together in order to reach any customer and also to improve the quality and availability.

Customers' requirements for quality and security became crucial. In fact, services that do not encompass these features probably do not keep working. These features can be reached trough a business model where providers can work together in order to reach a better service that cross geographical frontiers.

Our framework needs consider this behaviour. Services that act in a global perspective must consider that providers must be globally connected and customers can not be restricted by geographical locations. Other important feature to be considered is that this contribution between providers can lead services to be more qualified to customers.

Some can state that this situation is an integration matter [2] [9], it means it is necessary to conceive an architecture where providers share its engines and customers request services that transparently are provisioned based on providers contribution. SOA is an architecture where components are grouped to satisfy a business intention [10]. Components are resources owned by the provider that offer interfaces to be accessed to form services. Services, for his time, also need divulgate its interfaces. This is a key aspect of integration at SOA, it means that public service interfaces can be defined in order to other providers discover and form a service in a higher level. The incorporation of services occur trough three situations [11]: by contracts, by message exchange and by directory access. At this framework the three situations will be considered, at distinct situations.

SOA is an important approach in the development of distributed application and service provisioning, however there are other concerns that this framework must address. Particularly, it is necessary to solve other problems that arise from this global perspective:

- How customers access this framework in a transparent way ?
- How services arrive to endpoints same at domains with restrict access policies ?
- How providers trust each other ?
- How providers know how to communicate with its peers ?
- How end resources are reached in order to obtain a configuration according to requirements stated ?

These questions indicate that this framework must not only focus on architecture to integrate customers and providers, but it is necessary to contemplate: a customer interface in order to communicate parameters for requirements and service results and a policy and control interface in order to reach the end resources.

3 Service Oriented Computing

Service is a concept with a broad application spectrum. A service can mean since just a product being selling (like a computer or a tv) until network connectivity even another abstract product as the access to some resources directory. It means services also need to establish policies to be accessed and to fit in customer requirements. The SOA concept, at this way would not be enough to conceive our framework, it is important to follow a *modus operandi*, or identify a methodology to compose our framework.

The SOC is a paradigm based on this enlarged notion of service [2] [3]. As stated at [3], applications are the result of the discovery and invocation of services rather than building new applications or the invocation of applications that accomplish some task. SOC interests vary since adequate service description until methods to reach an on-demand operating environment. All this lead us to an important first conclusion to compose our framework: services must be autonomous and need of public interfaces that would be used to be part of an overall service.

This service grouping leads enterprises to face a new kind of integration. Business relationships can be established *ad hoc* and services can arise from this integration. External service providers can manage this integration and offer services compound from third parties. Technological resources like web services became crucial to achieve this. Another important conclusion to compose our framework: service integration must not be a static task. Services can be compound at a dynamic fashion, but for this, it is necessary that providers know and trust each partner and that services interfaces be public at some global accessible directory.

3.1 Enterprise Service Bus

In order to permit the desired integration between services, there is the ESB. [12] defines ESB as a middleware where tools and resources are used to collaborate in the integration of enterprise software components.

As heterogeneous environments would be found, it was necessary to conceive a medium where messages and events are intercepted, recognized and handled by the respective targets. In a SOA, providers must understand requisitions from other providers, understand instructions guided by hierarquical providers and also must allow its services to be understood and invoked. All these tasks require agreements which mean not only common protocols but also invocation and execution parameters.

Considering the functionalities above ESB becomes crucial to SOA applications and the architecture itself. To adopt this concept at our framework it is important to verify which capabilities are expected. According to [2] and [12] we can summarize these capabilities to the following: communication, service integration, integration, QoS, security, service level, message processing, management and autonomic, modelling and infrastructure intelligence. However, nor all SOA is obligated to follow these capabilities which means they would vary depending of implementation desired. In this case, [12] reduce this to the following minimum ESB capabilities expected: communication, integration and service interaction.

In order to meet our objectives at the framework our expected capabilities are:

Table 1 - ESB capabilities for Global Business Framework. Adapted from [12]

Capability	Tasks
Communication	- An administration capability to control service addressing and naming; - Support to inter-provider messaging; - Support to routing and addressing services.
Integration	- Support to at least one form of integration to service providers (such as Web Services)
Service Interaction	- An open and implementation-independent service messaging and interfacing model, that should isolate application code from specifics of routing services and transport protocol
Management	- Logging, monitoring and alerts; - Integration to systems management and administration tooling; - Self-monitoring and self-management.
Infrastructure Intelligence	- Policy driven behaviour, particularly for service level

4 Next Generation Networks

SOA and SOC brought important contributions to the design of distributed application based on service concept and with a broad (even global) action scope. These contributions are favouring many integration solutions to be conceived and also to improve the concept of service. Services even more require improvement on digital traffic leading, in a near future, the traditional PSTN model to a possible collapse.

Next Generation Networks (NGN) [13-15] is a concept that arose to take account these new services each time required in telecommunication world. This is a packet-based network able to provide services whose conception must be splited from transport-related technologies, allowing customers to reach any provider in order to acquire services. According [14, 15] this allows a general mobility that guides to a consistent an ubiquitous provision of services to users.

According [13, 14] a service based on NGN must be distributed at three layers: transport, control and services. The most important component at services layer is called IMS (IP Multimedia System). The project TISPAN (Telecoms and Internet converged Service and Protocols for Advanced Networks) from ETSI (European Telecommunication Standards Institute) is the standardization channel for IMS that looks offer to providers the enough infra-structure of IP services in order to provide new multimedia services together with telecommunication and data.

Despite other standardization bodies are involved at service management definitions like TMF, ITU-TMN and even ETSI and 3GPP, TISPAN has the focus in NGN, and as observed at [14], concerning aspects like: service aspects; network architecture and functional requirements; protocols and profile definitions; numbering, naming addressing and routing; QoS, resource control and network performance; conformance and operability testing; security aspects and telecommunications management.

This new service configuration has changed the focus of classical support systems. OSS and BSS now must encompass SOA features in order to attend services. It means the concept of NGOSS would change to NGNOSS.

The problem can be splited at three scenarios: the first is the core of the business transactions or the ESB for this framework, after there is the customer side and the last is about the management and control of providers' resources.

At the first scenario, the major concerns are: to allow providers to publish its service offers, to cooperate each other in an on-demand strategy and to compound quality offers to customers is the core preoccupation of this framework. On the customer side, we expect that customers can order services and monitor its ongoing status. In the last scenario, we need to allow particular providers to monitor its resources and manage policies in order to avoid security and performance disturbs.

Some state that these situations are associated with integration problems [7] [16], other indicate that this is a service provisioning question [17] [10] and still there are works who believe this is just the delivery of end-to-end services [9] [18]. Despite of the characterization of purposes, this framework aims to integrate service providers trough a friendly services network. It means, providers can work together with the guidance of one interested provider in order to offer customers more qualified services. Customers, in their time, would be able to research, request and monitor these services. At same time, our framework needs to support providers to manage and control its resources that were designed to be part of the framework at all.

4.1 Framework Objectives

To clarify problem statement and to guide our interests trough this document, we present which are the objectives to reach with Global Business Framework:

- Establish an inter-domain infra-structure to allow business based on SOC to be accomplished at the internet;
- Establish means to customers advertise their interests in a service offer and to order this;
- Establish means to providers monitor services ordered;
- Establish means to allow services to be dynamically linked in order to form a complete offer;
- Establish means to allow providers activate resources' configuration and manage this;
- Establish a complete support to all business chain troughs SOC since customer requirement until billing processes.

The problem obviously pours a lot of requisites that will be presented and discussed at sections 6 and 7. At the next section we present the related works.

5 Related work

To conceive this framework we researched three categories of solutions to the problem stated: integration perspective, end-to-end service delivery and service provisioning systems. At this section first we identify the characteristics of each category and after we present the works.

5.1 Integration perspective

Integration was a word of order in the transition of the Internet's third wave to fourth wave []. The third wave is known as information-based and received an improvement with the advent of SOA concept and web services technology, this already on service-based or fourth wave. It caused many changes at business world that modified strategies like the business processes and migration of legacy systems to adapt to this service-centric strategy.

Enterprise Application Integration received an important focus as a financial, organizational and business approach to providers. As we observe at [16] this approach promoted web services integration in order to make available different services between enterprises. Collaboration was one of the core objectives, strengthening partners' relationships. Other important feature of this approach is the preoccupation to handle heterogeneity with web services support.

Unfortunately this perspective still could not be applied to this framework if we consider the lack of a customer interface, even a management strategy to providers' resources. At this time, these integration ideas were an innovative approach leading specific problems to be handled with solutions based on this. As we will observe at next subsection, QoS became a common target to integration of providers.

5.2 End-to-end service delivery

Despite the very specific objective, the end-to-end service delivery plays an important role at the conception of our framework. Particularly we mention the Tequila (Traffic Engineering for Quality of Service in the Internet, at Large Scale) Project [5, 18] and the Agave (**A** li**G**htweight **A**pproach for **V**iable **E**nd-to-end IP-based QoS Services) Project [4, 9].

The problem studied at Tequila is about the guarantee of end-to-end QoS based on traffic engineering tools and service definitions. At this project one important component arose as crucial topic for our framework: agreements. In order to accomplish services with its specifications it was important to establish contracts (SLAs) to ensure service requisites. These contracts were converted at policies such that each domain belonging to service scope would handle these policies.

Agave preoccupation is associated with the infra-structure where services are based. According to [9], end-to-end IP service delivery was very based on heavy communication solutions, making the delivery a task of low performance. To lessen, even eliminate this overhead, service providers' networks are logically divided according to correspondence between connectivity requirements for each service. This division is accomplished by a new concept called Network Plan. The Network Planes forces traffic to be classified and adequately handled according to QoS requirements. Even being logical partitions, network planes allow providers to form the so called Parallel Internets.

We found five contributions of the work related to this topic to our framework:
- The importance to establish and manage contracts;
- The necessity to include the customer as an active actor, in order to interact with its services. Otherwise, the framework would not reach the objective to support customers at a broad range of service offers;
- The importance to evaluate alternatives to interconnect providers;
- The importance to establish a business infra-structure that do not only observe providers interests, but also services and customers interests;
- The importance to establish a business infra-structure with low impact at the performance at all.

5.3 Service provisioning

In order to reach our objectives it was necessary to incorporate ideas that transcend the establishment of means to transit service integration. It is important to think in means to integrate service, provider and customer in a business chain.

In [8] we face a user-centric approach at the service provisioning. User-centric means that the service is splited in two abstraction level. In the first level the service is recognized as a set of parameters that accomplish business requirements. At the second level, service means a set of instructions to handle underlying resources according with the service definition of the first level. The translation from the first to the second level occurs via contracts. The contracts established from Service Level Agreements guide intermediate providers to form relationships. These relationships are responsible to form dynamic provisioning of services that will be available to a mass of customers.

The former strategy, also called SPS (Service Provisioning System) focus on a dual management effort: the management of service provider and enterprise networks. It means that as on end-to-end delivery, service providers need to accomplish integration between all them to form a dedicated channel to follow services, even of an enterprise network.

However services also vary a lot in the payment, performance and kind. It means that services does not have the same profile for any customer and that it is not interesting to conceive fixed services to all customers. Each customer will have different possibilities, services and payments. The IPSphere [6, 17] is a new framework where service providers could create and expose their services without the limitations of the classical IP model.

According to [17] it would be possible that: providers distribute their services to costumers, sell (and distribute) services to other providers, classify the services offered to clients and to deal with distinct client policies.

The idea is to create a new layer (Business Layer) above the IP architecture: SSS (Service Structuring Stratum). This new layer supports all business negotiation necessary to start, operate, and terminate a service, as can be observed on figure 1.

There is no any interest in substitute or create new protocols but just map the classical layer architecture in a model that do not give to the providers the most hard part of business relationship negotiation.

The IPSphere reference architecture indicates some important behaviors:

- Providers can exchange business information via the SSS
- Providers are interconnected with customers trough Customer-Network Interfaces (CNIs)
- Providers are linked each other via Inter Carrier Interface (ICI)

These interfaces must keep a double channel of communication with SSS, which means that SSS must communicate with ICI and CNI and both interfaces also must establish communication. Besides, in a web scenario other elements must be addressed: the network management system of a specific domain (this entity is very important to definitively configure the resources on behalf of users and service itself) and routers (essential elements of interconnection). Thinking about these former entities, the IPsphere model still has:

- A Policy and Control Stratum (PCS) whose function is to intermediate policies negotiation and allocate management functions from/to SSS and the domain affected
- A Signaling Network Interface (SNI) whose function is to handle signaling messages originated from SSS e from the domain

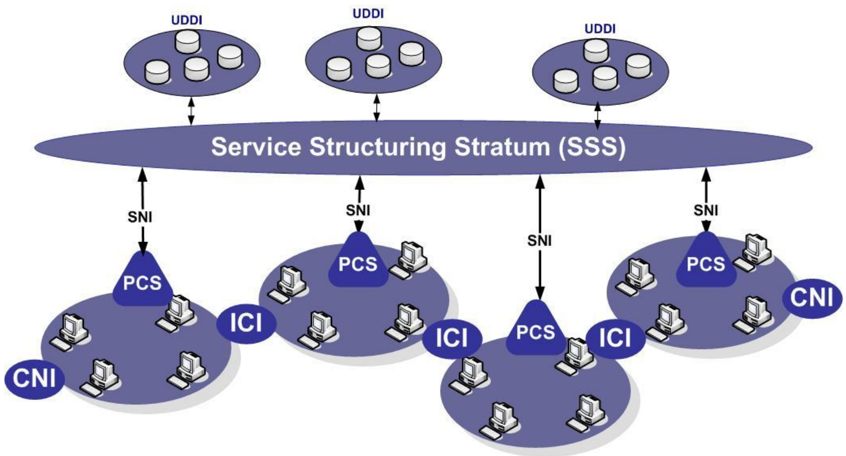

Figure 1 - IPsphere Architecture

The IPsphere initiative tries to follow standardization efforts like those coming from IETF [19] and TMF[20]. These standardization efforts impact in strategies like:

- The business process at all with application of the model eTOM [21] (enhanced Telecommunication Operations Map);

- The TMF Information Model [22] in order to guide in the conception of a common exchange information model between providers;
- The NGOSS (Next Generation Operation Support Systems) [22, 23] in order to improve customer relationship with the SSS.

The IPSphere reference architecture [24] indicates some expected behaviors: providers can exchange business information via the SSS; providers are interconnected with customers trough Customer-Network Interfaces (CNIs) and providers are linked each other via Inter Carrier Interface (ICI). These interfaces must keep a double channel of communication with SSS, which means that SSS must communicate with ICI and CNI and both interfaces also must establish communication.

5.3.1 Service Structuring Stratum

This is the core component of IPpsphere framework. Every business transactions must be encompassed at SSS. Particularly, the AO is responsible to capture customers' orders to services. Customers can send their requirements trough support systems like OSS. However, IPpshere Reference Architecture [24] keeps this interface opened to any other implementation.

AO must handle the order according Service Architect definitions during service design. It means there are pre-established agreements in order to serve the customer. As observed at figure 2, a customer does not have any knowledge of the existence of EO, Service Architect and even Element Architect. All transactions occurred at SSS are intermediated by AO who is the only entity the customer really knows.

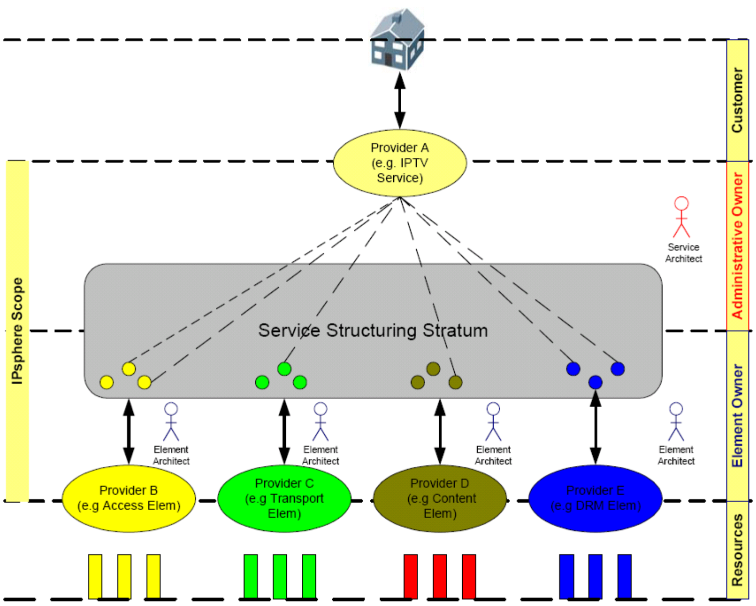

Figure 2 - IPsphere SSS organization.

To complement IPsphere proposal, other elements must be addressed: the network management system of a specific domain (in order to configure resources on behalf of users and service itself) and routers (essential elements of interconnection). Thinking about these former entities, the IPsphere model has a Network Policy and Control Stratum (PCS) whose function is to intermediate policies negotiation and allocate management functions from/to SSS and the domain affected.

However, policies must be translated to configuration instructions and for this reason we complement this framework with the Packet Handling Stratum (PHS) which operations are related to resources configuration and management.

6 Global Business Architecture

In this document we describe the system architecture proposed by the University of Coimbra. This architecture aims to intermediate customer requirements for end-to-end services, published as offers and that can be reached trough a search directory service. We do a brief description of the architecture that is based on IPsphere initiative.

In the proposed architecture, a customer search for the desired service, according to parameters indicated by search service. The customer choice indicates that he also has chosen a provider to intermediate its service. This provider, called Administrative Owner, must present to customer a SLS (Service Level Specification) that can be analyzed by the customer and whose missing information must be filled for him. This SLS is compound in a SLA (Service Level Agreement) that the customer will analyze again to decide if he really agrees with all constraints presented. With the customer agreement, the Administrative Owner needs prepare all elements that are part of the service at all to start the service requested.

After this Administrative Owner intervention, each individual provider (that is part of the service at all) needs to setup its equipments in order to reach the service bought by the customer. It is important to observe that agreements must be previously established between providers in order to publish a complete service offer.

To describe the system architecture two entities will be presented: architectural elements and components. Architectural elements represent pieces of the architecture responsible for critical functions that will be implemented at the components.

Figure 2 presents the general purpose of this architecture with five architectural elements: Service Presentation and Ordering (SPO), Service Structuring Stratum (SSS), Policy and Control Stratum (PCS) and Packet Handling Stratum (PHS). Figure 2 states a single flow that happens during a service establishment. An ISP customer requires to an AO (Administrative Owner) a service provisioning. The AO searches for contracted providers which can provide the service requirements. Once they were found, the SMS Parent of the AO and the SMS Child of the providers, exchange necessary information for the service configuration and establishment.

In the architecture, the SSS (Service Structuring Stratum) plays an important role in the service provisioning. This is the layer where all business transactions must flow, guaranteeing: inter-connection between providers, privacy for customer service requirement and allowing contracts to flow safely.

We propose handle SSS as a protocol where SOAP messages would traffic in a secure fashion and the elements which can access the SSS, would send messages in the pre-established format with the security permissions.

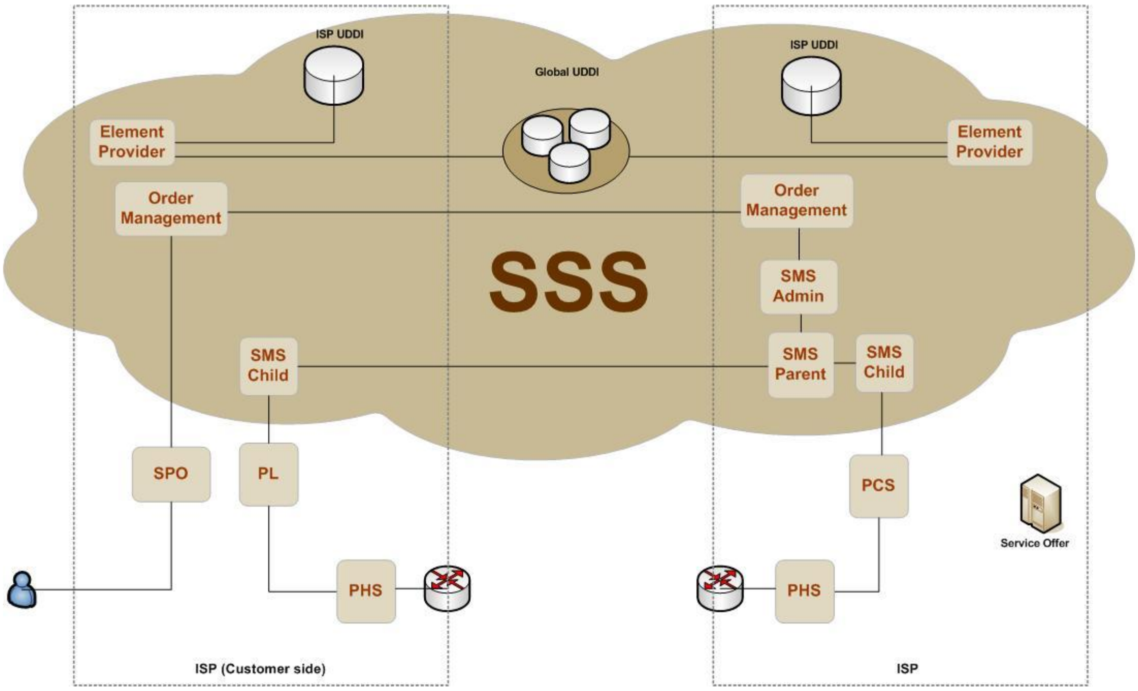

Figure 3 - Global Business Framework

At the next sub-section will discuss two architectural elements from this framework: SSS and SPO.

6.1 Service Structuring Stratum

The Business Layer of the architecture is responsible to deal with:

- Management of the services and service elements at the service directories;
- Request of the services;
- Management of the negotiation between providers in order to provide the services.

6.1.1 SSS Information Level

It is composed by the following components.

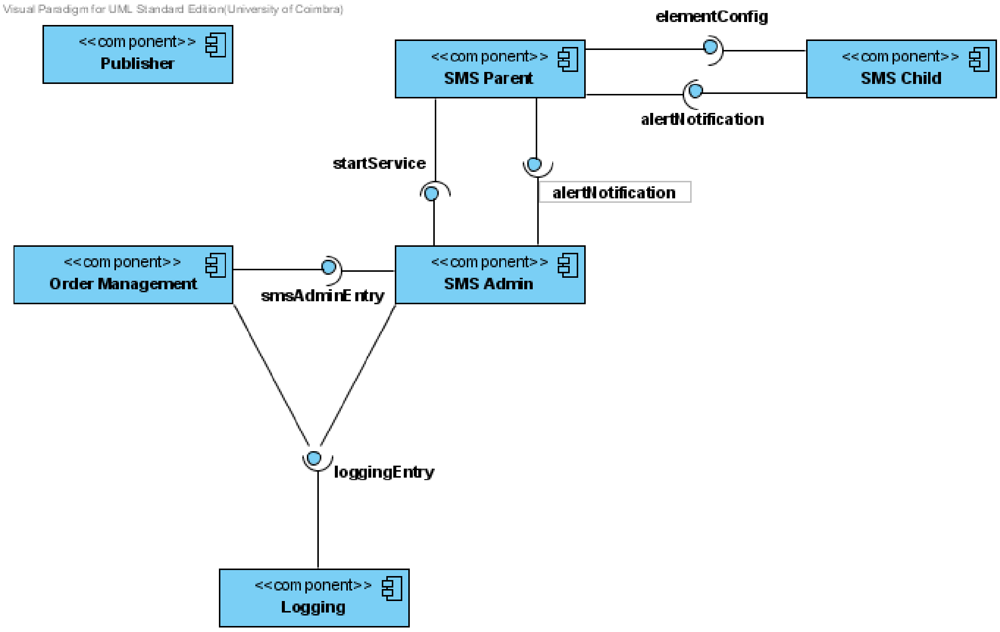

Figure 4 - Business layer component diagram

6.1.1.1 Publisher

This component allows a Service Provider to make its capabilities available. It is possible to identify two kinds of capabilities for a Service Provider.

- Element Service Offer – this is possibility of a Service Provider to publish its offer for an isolated element to compound a Service Offer. This strategy allows Administrative Owners to offer services to customers at a dynamic and cheap way. An Element Service Offer is published at public directories, however private directories must be kept in order to guarantee privacy for some information about this offer and to keep service information actualized;
- Service Offer – a service is compound of elements, and a provider can publish its offer of a particular service. It means, an Administrative Owner, even a customer can search at public directories in order to reach the service required.

This component has many functions as: publish, update and withdraw offers at directories. There can be many types of directories, each of them working in some way. In our case, we use the UDDI directory service.

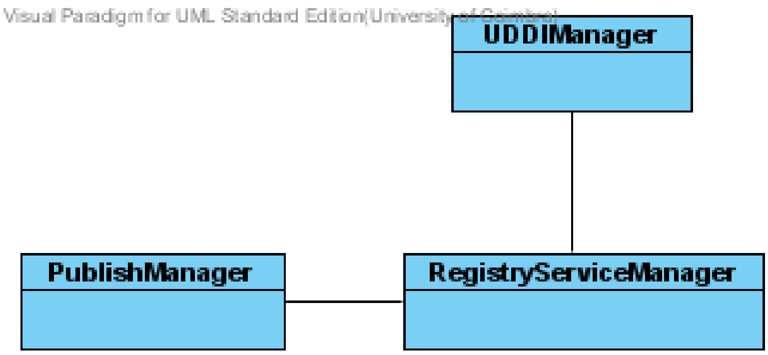

Figure 5 - Publisher class diagram

6.1.1.1.1 PublishManager

This class is responsible to receive the requisitions to publish, update and/or withdraw the service and element service offers. It may receive requisitions from an external entity, which can be a user or another system. This class is an interface for the external agents to manage the service and element offers.

6.1.1.1.2 RegistryServiceManager

This class acts as a proxy in the Publisher component. It captures the requisition from the PublishMagager class and transforms them into requisitions for the appropriate directory service (in this case, UDDI).

6.1.1.1.3 UDDIManager

This is the class that really access the service directory. It receives the requisitions from the RegistryServiceManager class and executes the appropriate action into the UDDI. This approach allows us to change the service directory without change the entire component. In case the service directory be changed, it is sufficient to create another class that access this new one.

6.1.1.2 Order Management

This component is responsible to receive the requisitions from the client when he wants to start a new service. It searches for the services that the provider can offer to show to the client. It also receives the service parameters the client informs, creates a service order instance and completes this order instance with provider information. This order instance is then used to start the service configuration and the service provisioning.

Figure 6 - Order Management class diagram

6.1.1.2.1 ServiceRequestor

This class creates the service order instance and stores it. It also receives the client requisition to start the service configuration and the service provisioning, obtaining the service order instance previous stored and sending it to the SMS Admin component.

6.1.1.2.2 ServiceOfferAccess

This is a class to intermediate searches that come from CLI architectural element, specifically from AEE component.

6.1.1.3 SMS Admin

This component is responsible to orchestrate the service to be provisioned by the providers to the customer. After it receives the requisition to start the service, it searches for possible element to provide it. Once these elements are found, it selects among them which ones will be used, taking into account the provider policies. An example of policies to select the elements is the combination of them which results in the best (lower) price.

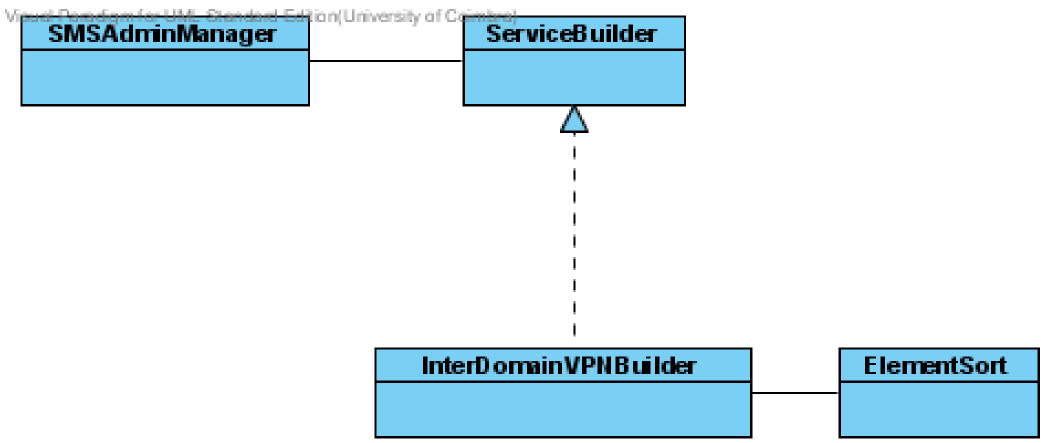

Figure 7 - SMS Admin class diagram

6.1.1.3.1 SMSAdminManager

This class receives the requisition to start the service provisioning and obtains the possible elements to compound the service. After the elements selection, it sends the built service script to the SMSParent. Meanwhile, it also updates the order service instance information stored, to reflect the elements selection used to configure the service.

6.1.1.3.2 ServiceBuilder

It is an interface used to build the service script according the desired service type.

6.1.1.3.3 InterDomainVPNBuilder

It is the class which implements the ServiceBuilder interface. This class selects which elements will be used to compound the service taking into account the provider policies. It reads the policies, which are stored in XML documents, and extracted the appropriate rule for each type of element to compound the service. Once it discovers these rules, they are applied to build the service script in order to start the service configuration.

6.1.1.3.4 ElementSort

It is a utility class used to help the element ordering.

6.1.1.4 SMS Parent

This component receives the service script from the SMS Admin component and executes it, communicating with the selected element providers in order to start the service configuration and service provisioning. Is also receives possible failure notifications during the service configuration or service provisioning and informs the SMS Admin component to take the appropriate actions.

Figure 8 - SMS Parent class diagram

6.1.1.4.1 SMSParentManager

This class executes the service script received from the SMS Admin component. It calls the web services in the selected element providers to configure and start the service provisioning.

6.1.1.4.2 SMSParentNotification

This class can receive alert messages from the SMS Child to indicate some problem or disagreement occurred in the service negotiation, configuration or provisioning. These messages are sent to the SMS Admin component to be used in the recovery actions. They are also sent to the Logging component for billing and auditing issues.

6.1.1.5 SMS Child

This component is responsible for receive the parameters for negotiation (SLA), configuration and provisioning of the service and to forward these parameters for network equipment configuration.

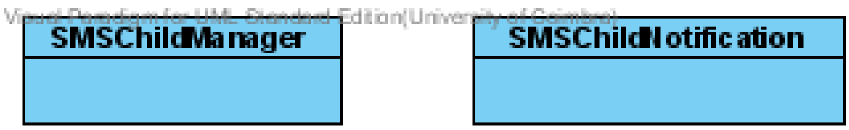

Figure 9 - SMS Child class diagram

6.1.1.5.1 SMSChildManager

It receives messages from the SMS Parent to configure, start or terminate the service provisioning. It transforms the messages received from the SMS Parent in a template and sends them to the PCS for network equipment configuration.

6.1.1.5.2 SMSChildNotification

This component can receive alert messages from the PCS to indicate some problem or disagreement occurred in the service negotiation, configuration or provisioning. These messages are sent to the SMS Parent.

6.1.1.6 Logging

This component stores information about all transactions performed in the SSS. This is used for billing and auditing matters.

Figure 10 - Logging class diagram

6.1.2 SSS Functional Level

The functions of the Business Layer can be divided into four main functions, described below, with the respective sequence diagrams and tables with messages description for each diagram.

6.1.2.1 Service start solicitation

For some services, it is necessary the endpoints to accept the service initiation. For these cases, the Figure 10 illustrates the solicitation made to the remote endpoints (users) to start the service.

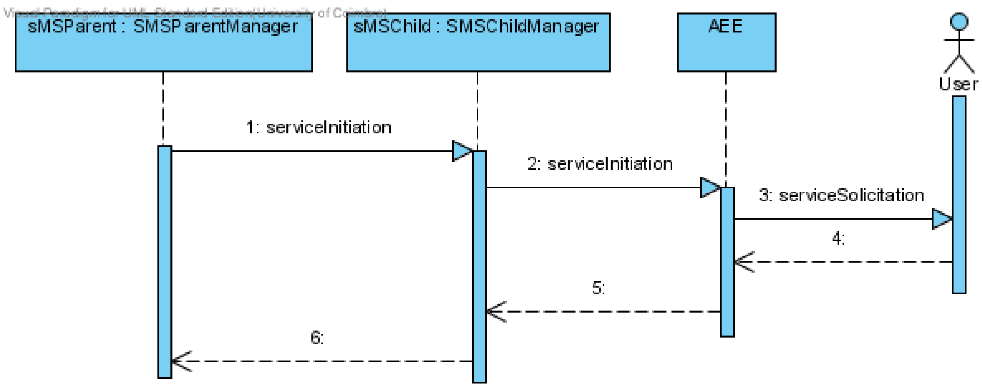

Figure 11 – Service start solicitation

Table 2 - Message description at Service start solicitation

Messages	function
serviceSolicitation	This message is triggered by the SMSParent component, after it receives the order to initiates the service, to solicit the remote endpoint permission to start the service

6.1.2.2 Service and element publishing

In this activity the service and element offers are published at the directory service. The publisher component intercepts the requisition for publishing from an external entity and translates it in commands to specific directory service (UDDI) access.

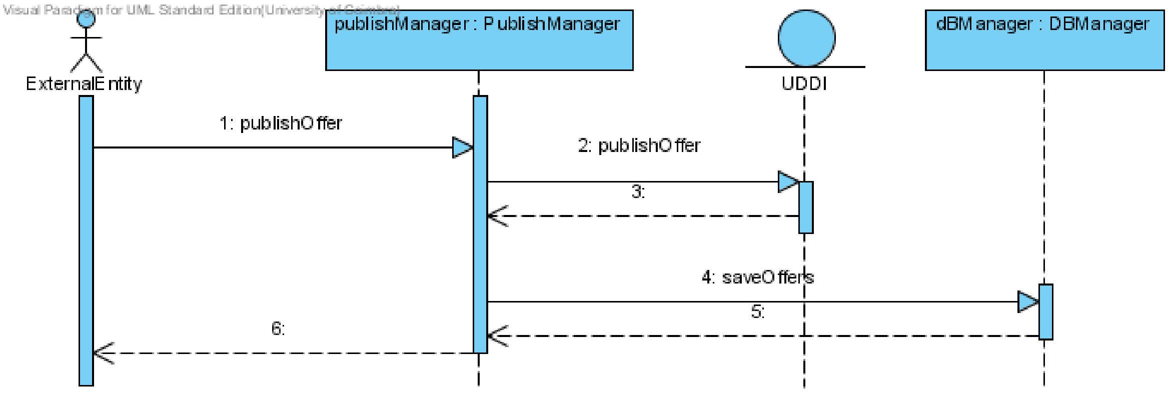

Figure 12 - Service and element publishing

Table 3 - Messages description at service and element publishing

Messages	function
publishOffer	This message is triggered by some external entity, like a system or a user (provider), in order to publish a service offer. This offer is published in the UDDI.
saveOffers	This is message is triggered by the Publisher component to save the service offers locally at the provider.

6.1.2.3 Service order creation

In the service order creation, a customer requests for some specific service, informing some service requirements. The Business Layer accesses the directory service to discover which services are available and who can provide them, based on those requirements. Once the customer chooses a service from a specific provider, the template of that service is requested. The customer then fills the template with the service parameters. This template is sent for the Business Layer to complete with provider specific and technical information. In the end, an order service instance is created, based on the template information, and is stored for further activation.

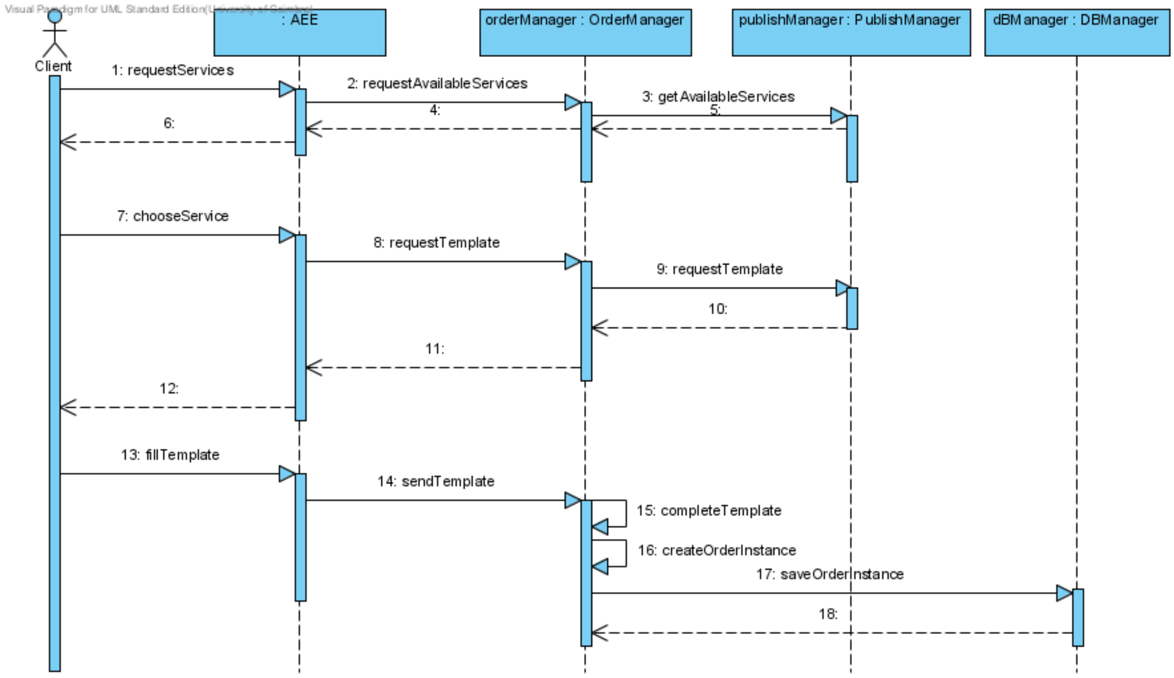

Figure 13 - Service Order Creation

Table 4 - Messages description at service order creation

Messages	function
requestServices	This message is triggered by the client to request the services the client interface (AEE) can offer to him.
requestAvailableServices	This is message is triggered by AEE component to request the available services at each provider.
getAvailableServices	This message is triggered by the OrderManagement component to get the available services.
chooseService	This message represents that the client choose a specific service.
requestTemplate	In this message, the AEE component requests the template of the service chosen by the client.
fillTemplate	This message contains the service requirements, provided by the client, to configure and provide the service.
sendTemplate	The AEE component sends the template to the OrderManagement component
completeTemplate	The OrderManagement component completes the template with provider specific and technical information. This information is non-public.
createOrderInstance	An instance of the service order is created.
saveOrderInstance	The created service order instance is saved for future activation.

6.1.2.4 Service order activation

When the Business Layer receives the customer approval to start the service, it retrieves the previous stored service order instance, gets the local policies provider and searches for possible elements to compound the service. After that, a selection on those elements is performed taking into account the service requirements and the local policies, resulting in a service script to start the service configuration and service execution.

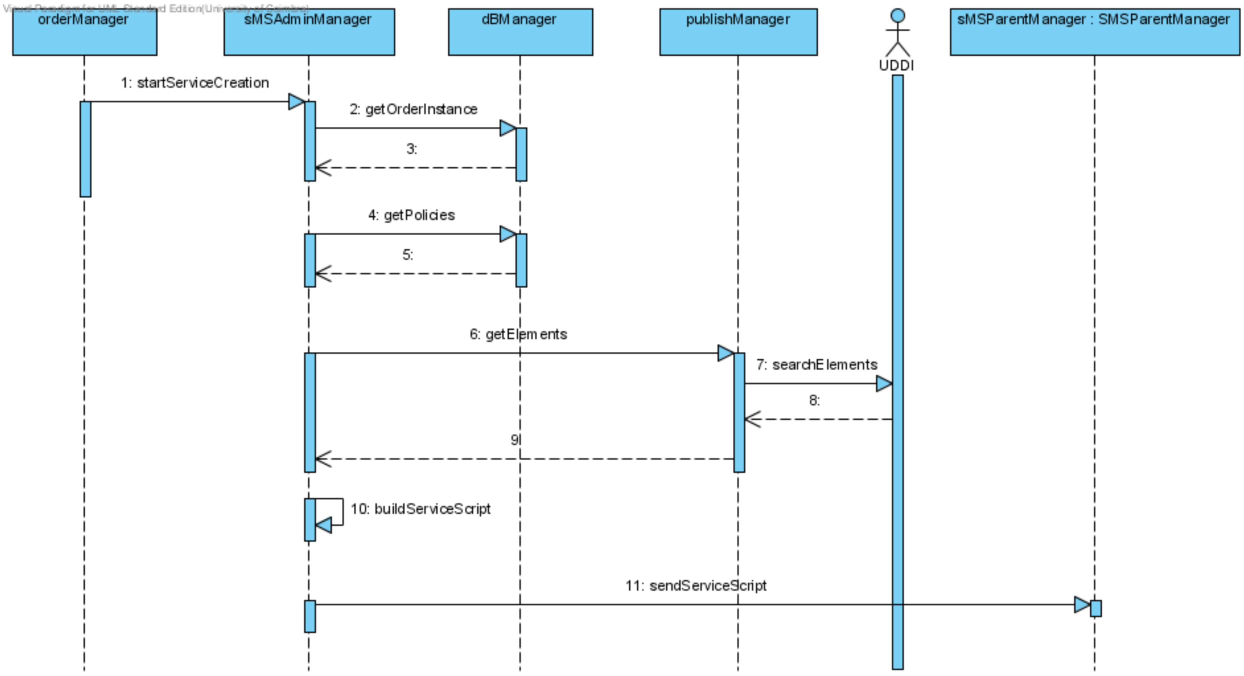

Figure 14 - Service Order Activation

6.1.2.5 Service configuration/activation

Finally, with the service script, the SMS Parent component contacts the SMS Child components from each chosen element to start the service configuration and the service execution.

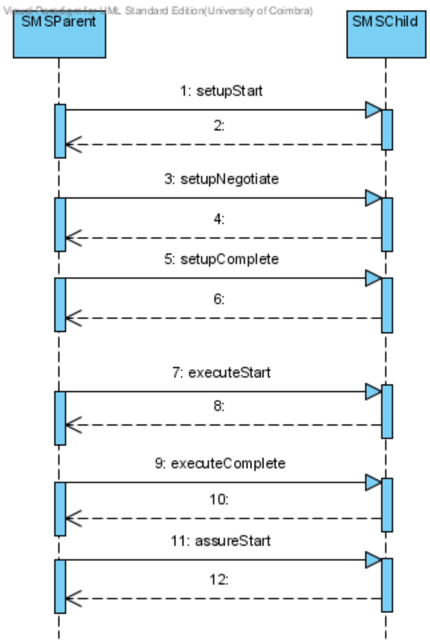

Figure 15 - Service Configuration/Activation

Table 5 - Messages description at service configuration/activation

Messages	function
startServiceCreation	The OrderManagement component, after the client approval, initiates the service execution process.
getOrderInstance	The SMSAdmin component gets the service order instance previous saved. It uses the service order ID provided by the client.
getPolicies	The SMSAdmin component gets the local policies to choose and assembly the

	elements that will compound the service.
getElements	The SMSAdmin component solicits to the Publisher component the possible elements to compound the service.
searchElements	The Publisher component searches for elements in the UDDI, based the criteria from the service order and provider local policies.
builsServiceScript	The SMSAdmin component builds the service script to be used by the SMSParent component to instantiate the service. This script is built based on local policies and service requirements.
sendServiceScript	SMSAdmin component sends the service script to the SMSParent component

6.2 Service Presentation and Ordering

Customers need an interface to access framework services. This layer is responsible to present service details to a customer and starts an ordering process. Customers can research for desired services trough a services directory called Portal B2C. Service offers from public and partners UDDIs are stored at Portal B2C directory. These offers must obey a template format. The SPO is responsible to deal with the following activities:

- Provide to customer a clear front-end;
- Intermediate customer service orders to business layer;
- Intermediate service architect monitoring of services;
- Intermediate service architect synchronization of UDDI information about offers.

6.2.1 SPO Information Level

The Business to Customer Portal (shortly Portal B2C) is a set of pages and scripts around xml databases. This is a search engine to provide customer enough information about services he wants to buy.

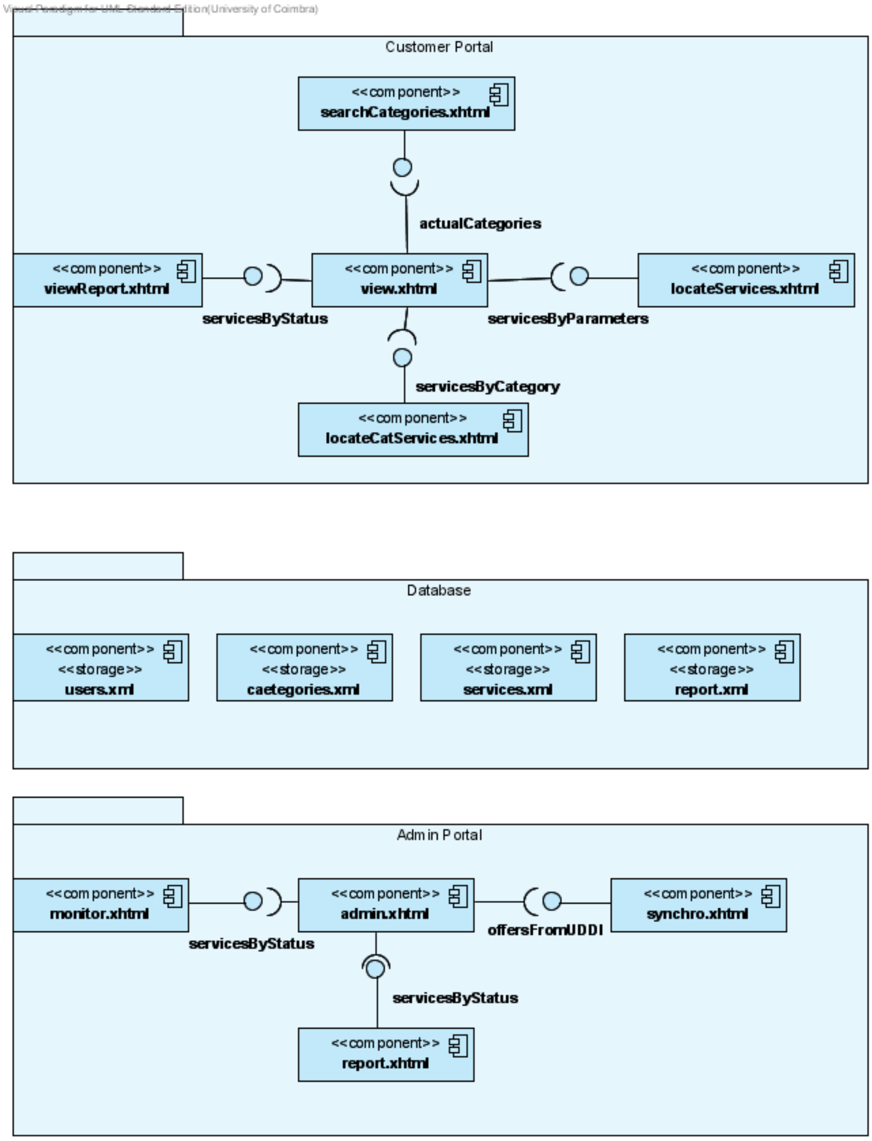

Figure 16 - Components of Portal B2C

In other hand, the SPO is a component where customers and Portal B2C can reach the business layer. This is where services are located, initiated, interrupted and negotiated. To describe how information is organized and manipulated at this component, we splited the component at classes, as observed at figure 17.

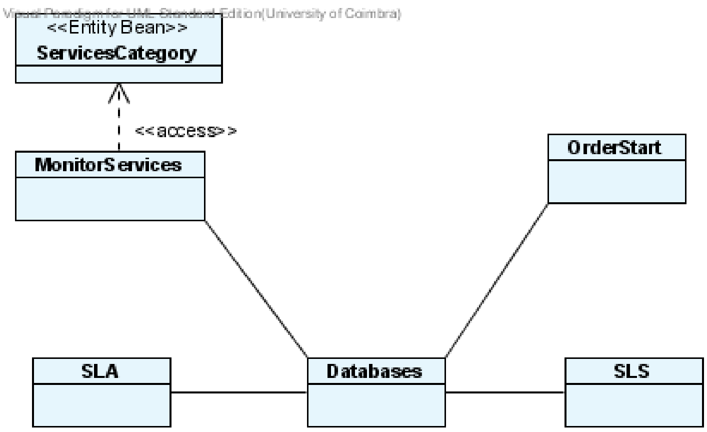

Figure 17 - AEE component classes

6.2.2 SPO Functional Level

SPO has six distinct use scenarios that will guide the description of this architectural element. The figure 18 presents these use cases.

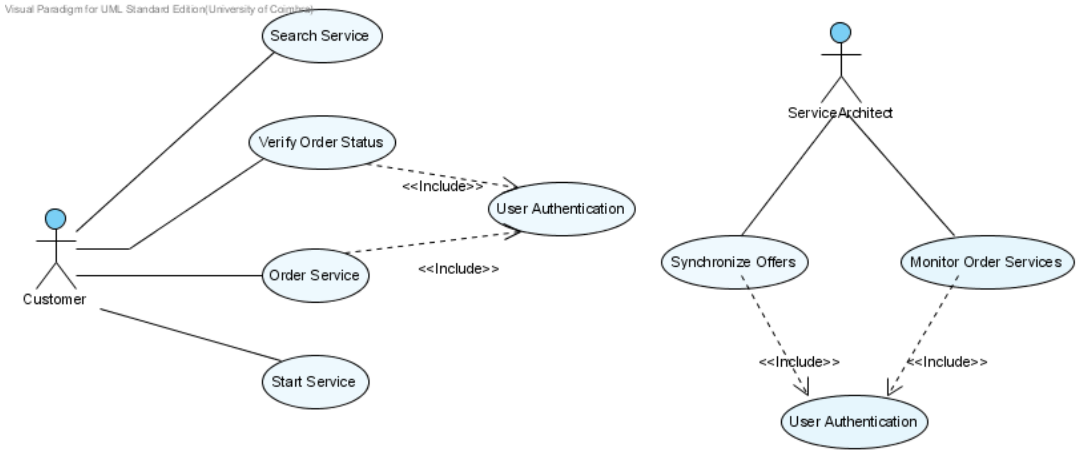

Figure 18 - SPO Use Cases

The description of this architectural element will be based at activities diagrams, considering that this tool would be more adequate to explain all transactions that occur at the front-end level. To guide this description, each use case will be detailed as that.

6.2.2.1 Customer searching a service

To make it easier to a customer finds the desired service a search engine will provide means to reach the right service. To search a service, a customer only needs to choose the category of the service and after these fill parameters for this like: dates, location and a specific name for the service desired.

As observed at figure 19, all the interactions are executed by the Portal B2C which means a database must be present and actualized. This operation will be detailed at other scenario.

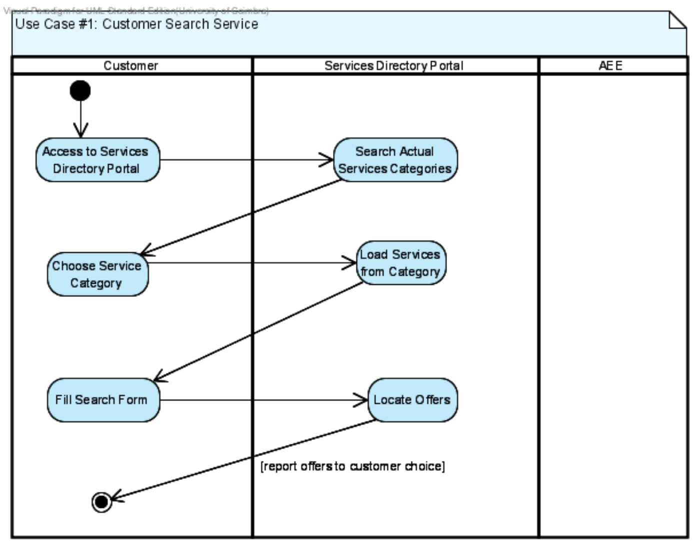

Figure 19 - Customer search service.

6.2.2.2 Customer ordering a service

The Portal B2C allows a customer to find the desired service, however the ordering will be transferred to the SPO component. As could be observed at figure 20, the portal will only format SLS and SLA documents in order to make it easier the transport. To allow customers to access globally (it means by any portal) its order service access, the business layer generates an GUidD (General Unique Id) for the service which must be authenticated every time the customer accesses this service.

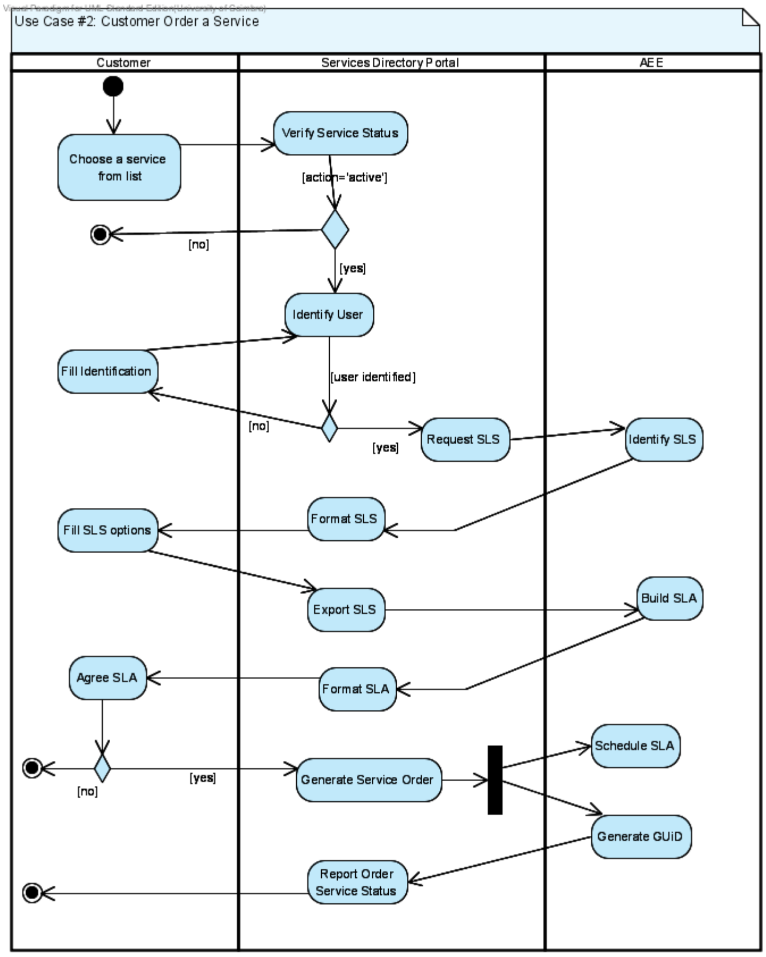

Figure 20 - Customer ordering a service.

6.2.2.3 Customer starts a service scheduled

Services don't need to be started immediately. Customers can buy and use that after. These behavior impacts of three ways: SLAs must determine clearly the possibility of a service not be available, component SPO needs to be monitor services in order to capture their real status and the service architect needs a way to be reported of these status.

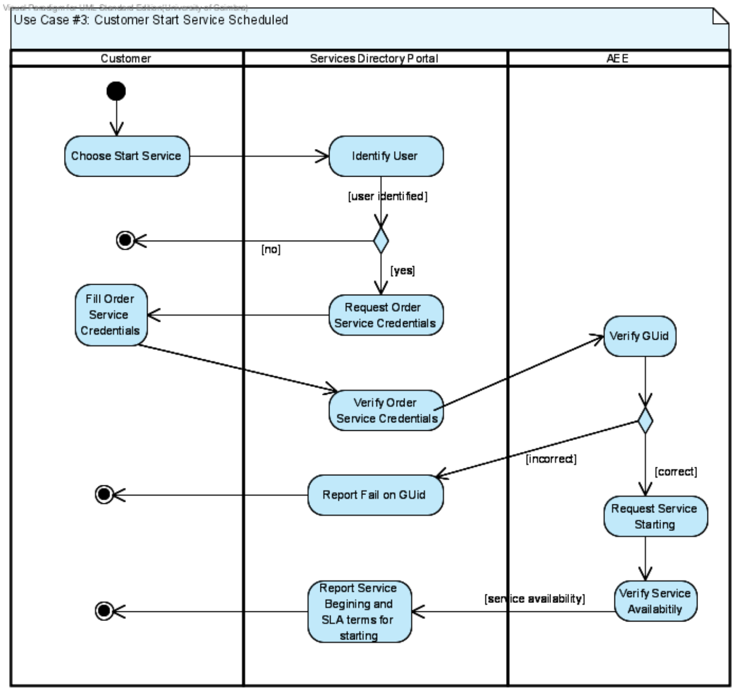

Figure 21 - Customer starts a service scheduled by him

6.2.2.4 Customer verify order service status

Customers that dispatch long term services need a way to verify how its service is. This option also allows customers to identify reasons for services interrupted or with critical performance behavior.

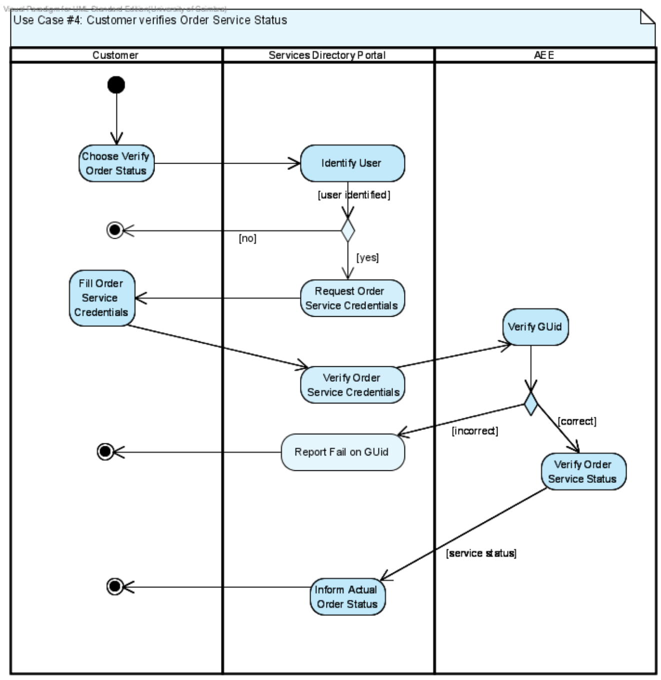

Figure 22 - Customer verifies order service status

6.2.2.5 Service Architect synchronizes service offers

The service offers are published at public UDDIs. It would be an expensive and delay operation to deal all customer searches directly at these UDDIs. One important reason to the conception of Portal B2C comes from that fact. It means that the portal must act like an engine that synchronizes its information with the UDDIs.

At a first time, the portal will act like an independent search engine, based on any UDDI and do not worrying about contracts between providers or any kind of individualization.

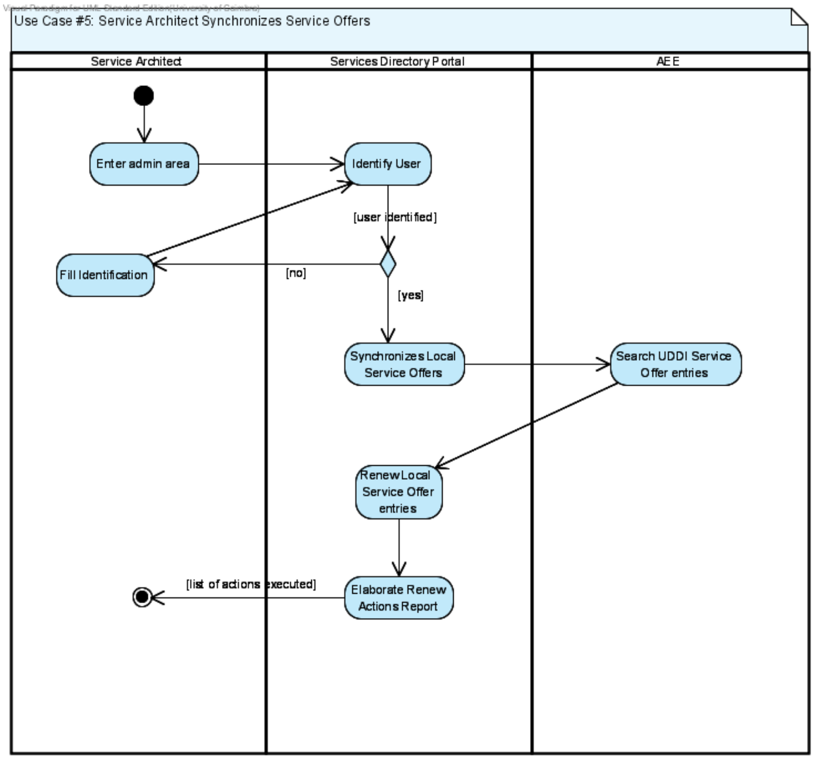

Figure 23 - Service Architect synchronyzes service offers with public UDDI

6.2.2.6 Service Architect monitors services ongoing

Services can be interrupted for technical reasons identified by each provider. Or services can reach the time interval contracted at SLA. In order to inform correctly all customers about problems and ongoing status of its services, it is necessary to monitor ongoing services.

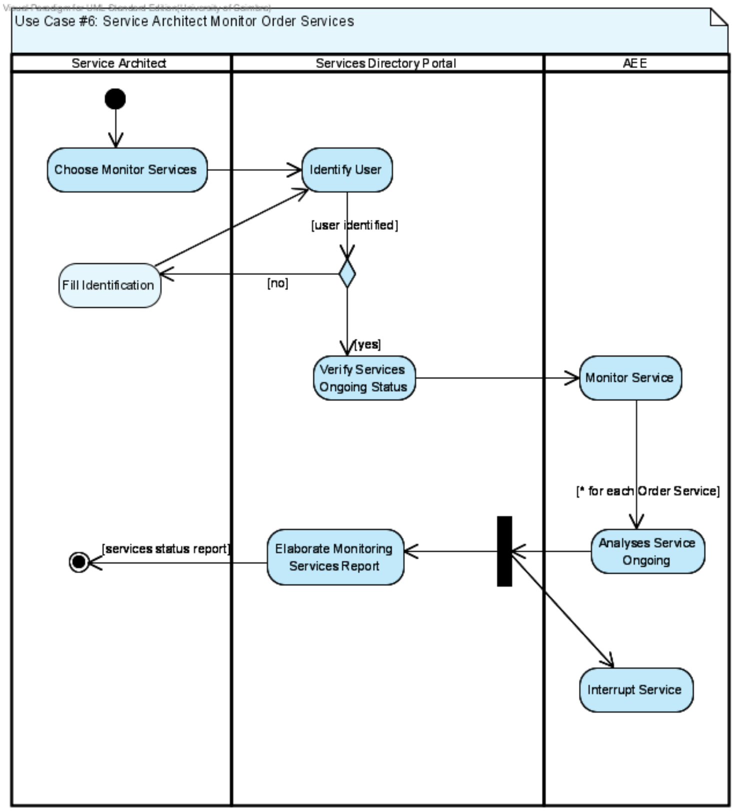

Figure 24 - Service Architect monitoring services.

6.3 Policy and Control Stratum

One of the major contributions of this framework is the flexibility to handle customer service requirements. When a customer order a service a SLA is conceived at SPO and the SSS is responsible to follow this to the appropriate providers. However each provider has its own set of rules, already established in order to allow local resources to work safely. Facing the necessity to conceive low layers to handle these agreements and in order to achieve resources configuration and management it was necessary to conceive a Policy Layer. Instead of running scripts and overcharge network administrators with configuration management from every part, we conceived this architectural element based on Policy-Based Network Management (PBNM) [25, 26] and prepared to interact with high level definitions for policies coming from SPO and SSS.

This architectural element presents some challenges that guided us at its conception. The first challenge is about the management paradigm to follow, after this the information representation and at last the communication model.

6.3.1 First challenge: management paradigm

To obtain the global business perspective, this framework is organized at layers. Each layer would operate over Internet protocols and each domain would maintain its normal services despite this new approach. Each service request is compound of a set of agreements that must be recorded and analyzed for future provisioning.

Management solutions generally are based at a centralized approach and in the manager-agent paradigm. Some alternatives arose with the introduction of XML [27, 28] and flexible information representation. Two major factors helped us to adopt PBNM at this framework: the flexibility that policies provide when services are ordered, allowing the system to decide and control about resources and service provisioning before effective execution and also because policies allow a loose coupling between higher and low layers. Low layers do not need to understand complex high definitions of SLA and providers are free to analyze the impact of a service provisioning with an anticipate knowledge of users' intentions.

This layer receives SLA statements that must be checked against local policies. These high level agreements must be translated to configuration instructions; however this task must be properly authorized by Infra-Structure Layer.

Two type of information must be managed by PCS: at a Policy Repository we must find user and resources information that must handled by Policy Management Agent and at the PIB (Policy Information Base) there are the policies.

6.3.2 Second challenge: information representation

Traditional PBNM solutions are based at policy description languages that allow network managers to introduce and manage policies for its domain. However, at this framework one major objective is to reduce the human intervention. Facing this we decided to adopt an information representation that would not require so much human intervention and that would integrate other layers with the minimum of translations required.

XML is this choice. Agreements and specifications are represented in XML at almost all entities of the ordering flow and as will be observed later, the PHS also expects XML statements for the resources configuration.

6.3.3 Third challenge: communication model

It is also important to decide which protocols would be applied in order for the appropriate execution of tasks and for the establishment of a communication model.

We decided to adopt an IETF standard: COPS (Common Open Policy Service). However, this decision has an impact at the information representation because XML is not a standard at COPS domain. To bypass this and to keep with the same objectives of the second challenge, we decided to follow a specific COPS implementation as observed at figure 25. This decision was motivated for two works [27, 29] whose intention is also to adopt XML in a PBNM solution.

A last preoccupation is about authentication and authorization tasks. In order to establish trust relationships between higher and lower layers, we decided that it is important to incorporate an AAA Server based on Diameter protocol that supplies credentials for SMS Child requests flow to PCS and also to allow PCS operates on PHS.

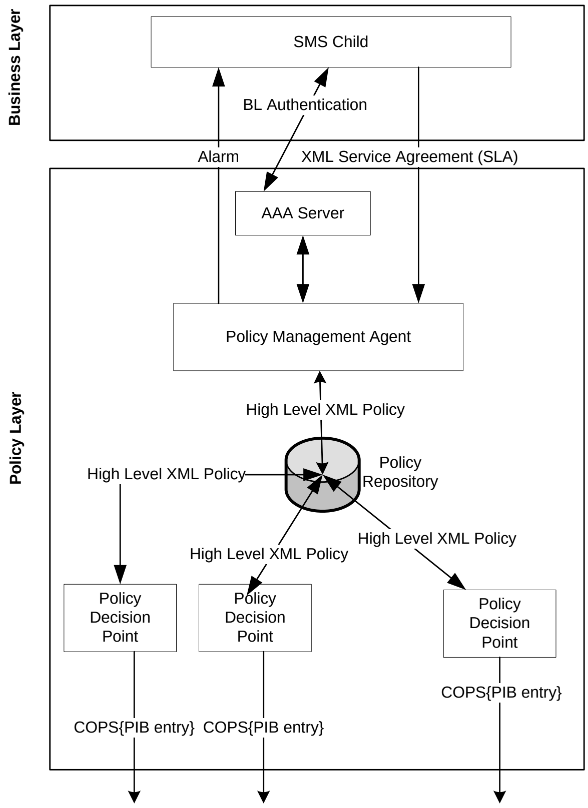

Figure 25 – PCS Overview

As observed at figure 25 PCS is compound of three components AAA Server, Policy Management Agent and Policy Decision Point. At the next sections we will present discussions about the information gathered, generated and transported by these components.

6.3.3.1 PCS Component Diagram

PCS has an interface with SSS and must generate entries to Infra-Structure Layer. From SSS, two entries are expected: higher element identification and the SLAs. For the PHS are produced low level policies in a format that is possible to be recorded at PHS's PIBs, it means there is a translation process before this. Alarms are generated by Policy Management Agent that can be gathered by SSS or PHS. It is important to observe that AAA Server, based on Diameter, allows credentials to be delivered both for Policy Decision Point and Policy Management Agent. In former case this is important for PHS transactions and the other for the appropriate job of PCS.

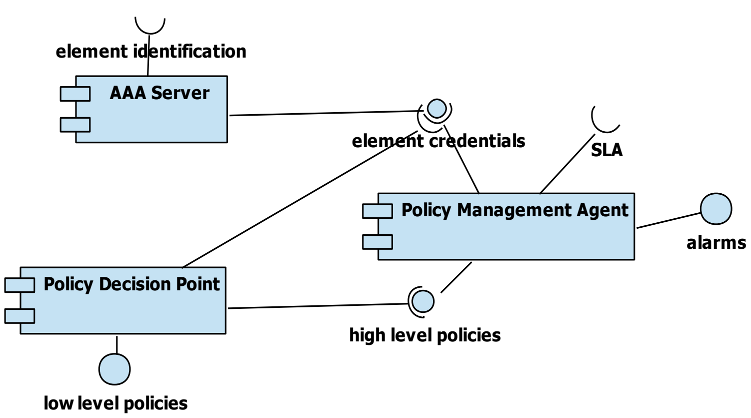

Figure 26 - PL Components

In order to complete our informational discussion, we will present classes that compound each component.

6.3.3.2 AAA Server

Authentication, authorization and accounting tasks are critical for systems associated with configuration and this is a core feature at PBNM solutions [30]. At this framework our choice was the Diameter protocol [31, 32], considering that authentication information must traffic between layers and that the AAA Server is a critical component.

The higher layers must supply identification tokens that are analyzed by the class Authenticator. Based on this token a diameter node must verify if there is local user (an user for this layer) associated with this identification. If that is true, credentials are recovered and presented to Policy Management Agent.

Each not successful access attempt would generate an alarm and this alarm would form a logging storage for future accounting or debug interests. At figure 27 we present the classes responsible for the tasks former discussed.

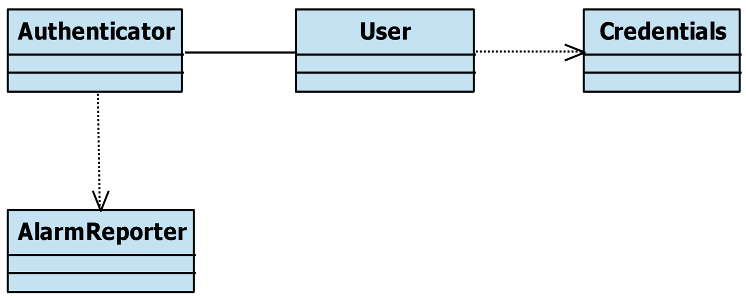

Figure 27 - Service Management classes

6.3.3.3 Policy Management Agent

The Policy Layer makes the interface with Business Layer and it is especially important to Administrative Owner. It allows decisions to be done in order to delivery control to each service provider.

This component plays the interface with SMS Child component from SSS in order to receive agreements and to dispatch alerts. Each policy management when invoked would have knowledge of its domain resources. The higher level policies received from SMS Child must be followed to Management Console in order to filter this according to user credentials and local resources policies. It means that another class, the Dispatcher will receive instructions to deliver policies to specific Policy Decision Points, according with the resources involved at the SLA.

At figure 28 we can observe that the Policy Manager must maintain a list of policies that would be applied at the domain. In case of any conflict the Analyzer must be invoked in order to send an alert to SSS.

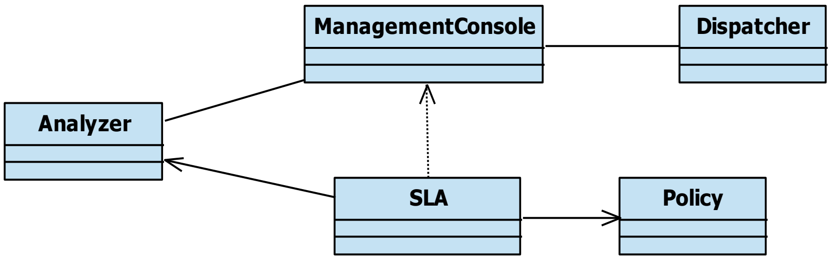

Figure 28 - Policy Management classes

6.3.3.4 Policy Decision Point

According to three level PBNM model from IETF [30], it is important that a third component (so called Policy Enforcement Point) receive instructions for their associated resources to be configured. However, in order to keeps the pattern of XML traffic between layers, which contributes to a loose coupling between layers and to diminish human intervention, the Policy Decision Point must knows which storage format the Policy Enforcement Point at IFL uses.

With this knowledge, the PolicyEnconder class translates higher level policies (already filtered by Policy Decision Point) to the target PIB (Policy Information Base). After this the PolicyClassifier groups all policies according to a respective Policy Enforcement Point in order to PolicyDispatcher delivers this.

The figure 29 presents the static association between these classes, it is important to emphasize that the PolicyClassifier creates a group of policies (PolicyList) to be delivered to the low layer.

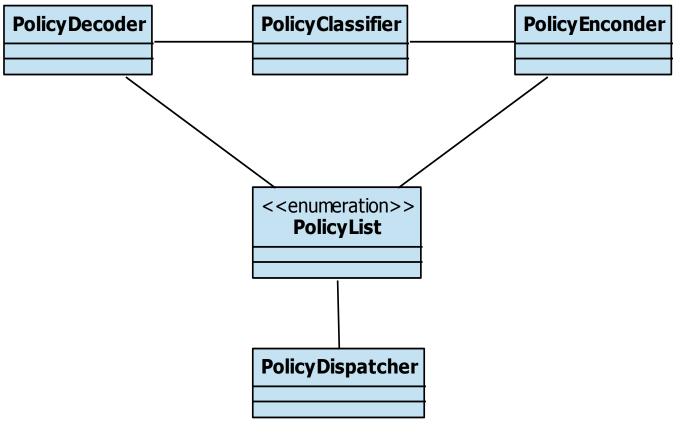

Figure 29 - Policy Decision Point classes

6.3.4 PHS Functional Level

In order to complete description of this architectural element we identify its uses' scenarios. Our interest is to provide a view of how informational layer can be applied in all practical situations.

In order to complete the description the table 6 describes each use case.

Table 6 – Use case descriptions for PL architectural element

Use Case	Description
Manage Services	1. AO receives from Business Layer a requirement to apply policies for a service; 2. AO verifies GuiD in order to testify this is not an ongoing service; 3. AO actualizes service status; 4. AO splits respective requirements for each Service Provider and, via web services, dispatch them.
Manage Incoming	1. The local policy manager receives policies to be applied from BL;

Policies	2. A list of available resources of the domain is loaded; 3. Policies are classified according the target; 4. Policies are classified according the resources involved; 5. Policy Manager follows policies to each Policy Decision Point.
Translate Policy	1. The policies are received by Policy Decision Point; 2. If Policy Decision does not verify conflicts, the policy is followed to translation; 3. Each policy is translated to the target PIB format; 4. Policy Decision follows this to Policy Enforcement.

At figure 30 we present two actors that drive former scenarios, the BL actor comes from BL, specifically acting as the SMS Child, while the Policy Manager is the own PolicyManagerAgent.

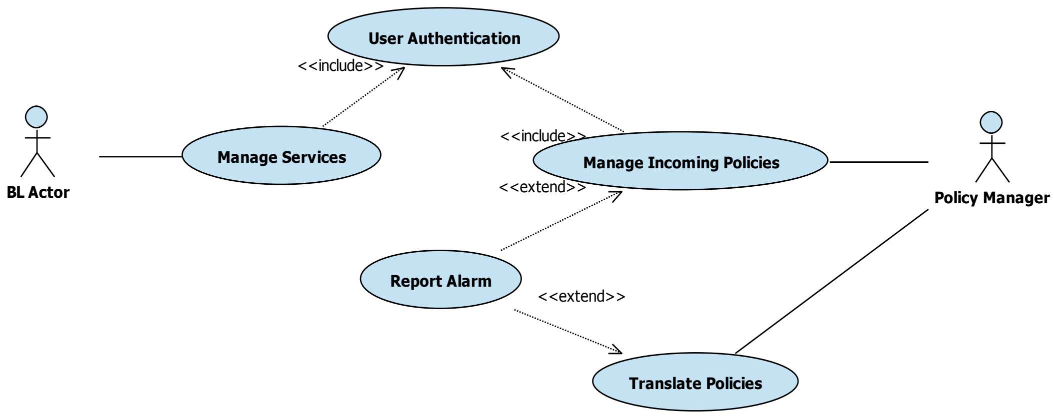

Figure 30 - Scenarios for use of PCS

6.4 Packet Handling Stratum

According [33], IETF defines two architecture models for PBNM: the two-tier model and the three-tier model. Both models pay attention to a resource policy point or a point where the resources are definitely configured. This point is called the Policy Enforcement Point and this is the entry point for the Infra-Structure Layer.

Two distinct strategies could be adopted to implement this layer: COPS-PR [26] or Netconf [27]. The first is a standard and its presence is totally related to PCS architecture. The second is an ongoing effort of IETF in order to bring XML to configuration management.

Despite there are some works where the union between COPS-PR and Netconf are addressed [33-35], we decided not to adopt Netconf. So it would be important to adopt a parser that translates XML SLA statements to policies and this is done by Policy Decision Point (PDP).

So, to implement this layer, it was necessary to idealize a solution that would involve XML and PBNM. In [25], we find a strategy that conceives a distributed network management model for the next generation internet that is based on XML and PBNM. This work explores JDOM API from Java and implements XML Parsers and maps this to MIBs (Management Information Base) however unfortunately do not present any incorporation to standard protocols like COPS and Diameter. Another implementation can also be found at [29]. At this work COPS and COPS-PR are considered and XML data is stored at XML policy databases with high level description of policies. At low level, policies are mapped to PIBs trough XER (XML Encoded Rules) that is a derivation of ASN.1 representation of PIB data. Other interesting innovation of this work is related at [29] that discusses AAA support for this framework and the application of Diameter as support for that.

The impressions of the work at [25, 29] collaborate to the idealization of this framework, specifically of this layer. So, the PHS, as presented at figure 6 is compound of two major components: Policy Enforcement Point and Resource Agent. Also there are the target configuration elements: resources, which vary from routers to firewalls or web servers.

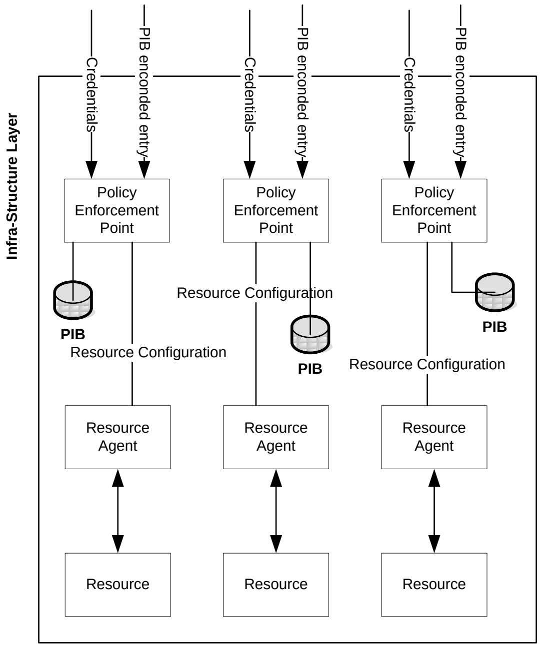

Figure 31 - PHS Architecture

6.4.1 PHS Information Level

As with the other components we supply a description of the information organization trough components and classes diagrams. The IFL has one element that can not be described here – the resource. Resources are any service instrument that is crucial in the service deployment and that receives some configuration established at SLAs. While component diagrams allow us to observe interfaces and dependencies between software components, classes' diagrams are an important tool to visualize static information.

6.4.1.1 PHS Component Diagram

Following higher layers, this layer also operates on authentication based on Diameter. This is one of the entries expected by the PolicyEnforcementPoint component. The other entries are the XML Encoded Policies that would be translated to target configuration instructions. Alarms are tasks managed by PolicyEnforcementPoint that decides if it is important advertise PL or just to log some alarm. The Resource Agent is a resource representative and acts receiving ResourceTargetConfiguration. It is important to observe that each policy that arrives to this layer already was verified, it means that conflicts are handled at PL that do not allow this flow to arrive to IFL. Other important feature of ResourceAgent is your support to the resource or to a set of resources; it means that ResourceAgent must incorporate some knowledge of its resources. The decision if a ResourceAgent is associated to one or more resources is of the Provider Manager that directly informs this to ResourceAgent.

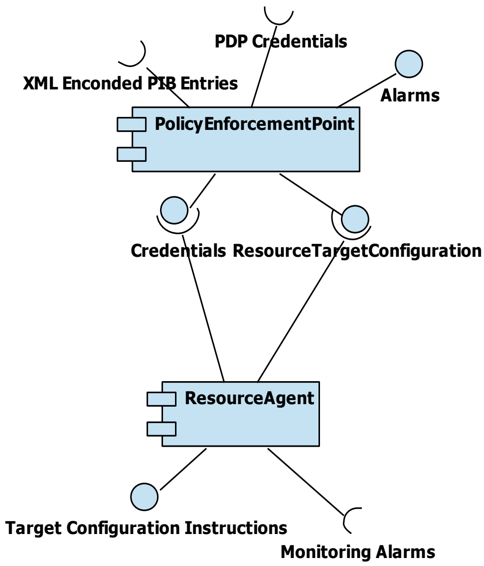

Figure 32- IFL Components View

6.4.1.2 Policy Enforcement Point

This component comprises of four classes: Authenticator, Alarm, PolicyTranslator and PolicyDispatcher. The Authenticator is a Diameter node that receives together with XML Encoded Policies from PCS also credentials to allow a configuration flow. Any not accepted credential is reported to Alarm that dispatches this to PCS or just logs this, based on rules defined by Provider Manager.

Before the configuration arrives to target resource, the PolicyTranslator classify and convert this to the target resource. The classification only is important if the PolicyTranslator identifies that this PEP is responsible for more than one Resource Agent and it means that a new policy direction must be supplied to the right resource. After the translation, the PolicyDispatcher deliver the configuration statements to the resource.

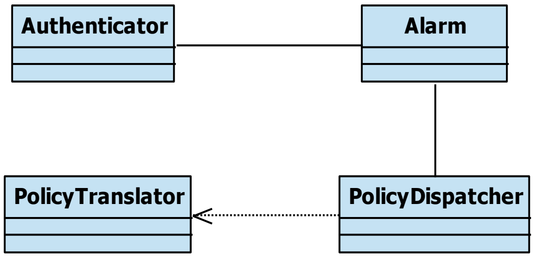

Figure 33 - PEP Informational View

6.4.1.3 Resource Agent

In order to configure the target resources, this component has three main tasks: to line up requests for configuration, deliver the configurations and expect for any kind of feedback from resource. At figure 34 we present our strategy to handle information for these tasks.

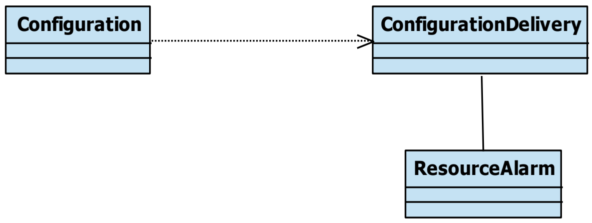

Figure 34 - Resource Agent Classes

The Configuration class is responsible to format configuration instructions and to dispatch them to the Configuration Delivery. While this, the ConfigurationDelivery just hold these configurations expecting for a feedback from resources in order to continue the delivering of other configurations. To avoid that ConfigurationDelivery waits during an indefinite time for resources' feedback, before each attempt of configuration to a target resource must be established a session with a time stamp and that allow ConfigurationDelivery to acts correctly in case of resource connection loss or reconnection.

6.4.2 PHS Functional Level

In order to clarify how above information strategy would be operated, we provide a discussion of use cases for this layer. It is important to observe that IFL is the destiny of any service ordered; it means that all the use cases consider that a service or a management task is ongoing. The table 7 illustrates these use cases and describes them.

Table 7- IFL Scenarios

Use Case	Description
Manage Encoded Policies	1. PEP receives encoded policies and verifies authentication provided by PDP; 2. PEP verifies its Resource Agent Number and if this number is larger than one, a classification process must start; 3. Every policy classified is translated and dispatched to its Resource Agent.
Dispatch Instruction	1. The Resource Agent receives the first configuration instruction for one Resource; 2. Resource Agent starts a session with the Resource, informing a time stamp; 3. Instructions that arrive are stored until Resource Agent receives a confirmation or a alarm from Resource; 4. If the time stamp finishes the Resource Manager or tries a new connection if is there one or more instructions yet or send an alarm, depending of Resource feedback.
Manage Local Resource	1. This is the only human intervention necessary. It starts with a manager authentication; 2. If the authentication is successful, provider configures each Resource with entries that or recover a resource from a fail or apply new configuration for that. It is important to emphasize that this scenario is not associated with the domain of this framework.

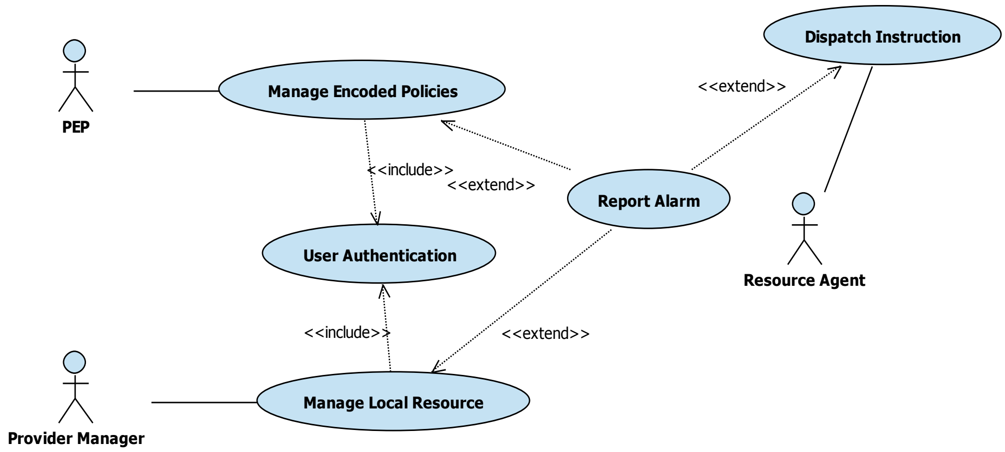

Figure 35- IFL Functional Overview

7 Conclusions

The conception of this framework is showing a good solution to the establishment of end-to-end service provisioning. The study of service management tasks is of a crucial importance in order to guarantee in the future security and QoS studies of this proposal.

Despite the essential contribution of IPsphere reference framework, some additional features were required: a strategy to handle SLA statements; a strategy to coordinate SLA translation; an approach to manage policies of an inter-domain reach and meanings for service discovery by customers.

This is important to mention too which contributions we expect for the future: analyze security and performance questions of the framework service provisioning; evaluate scalability by the injection of new simple and complex service offers and discuss the security and liability questions around the conception of federated UDDIs for services directory.

8 References

1. Mathew, C.L., K.; McCabe, F.; Brown, P.; Metz, R. *OASIS Reference Model for Service Oriented Architecture 1.0*. 2007 [cited; Available from: http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.html.

2. Michael P. Papazoglou, P.T., Schahram Dustdar and Frank Leymann. *Service-Oriented Computing Research Roadmap*. [cited; Available from: ftp://ftp.cordis.europa.eu/pub/ist/docs/directorate_d/st-ds/services-research-roadmap_en.pdf.

3. Huhns, M.N.S., M.P. , *Service-oriented computing: key concepts and principles*. IEEE Internet Computing. **Volume 9**(Issue 1): p. Page(s):75 – 81.

4. *Agave Project*. [cited; Available from: http://www.ist-agave.org/.

5. *Tequila Project*. . [cited; Available from: http://www.ist-tequila.org/.

6. *IPsphere FORUM*. [cited; Available from: http://www.ipsphereforum.org/.

7. Kontogiannis, K.S., D.; O'Brien, L. *On the role of services in enterprise application integration*. in *10th International Workshop on Software Technology and Engineering Practice*. 2002.

8. Rodosek, G.D.a.L., L. . *Dynamic Service Provisioning: A User-Centric Approach*. [cited; Available from: www.loria.fr/~festor/DSOM2001/proceedings/S2-1.pdf.

9. Quoitin., D.G.J.S.J.G.N.W.M.H.G.P.S.G.a.B. *D1.1-final Parallel Internets Framework*. [cited; Available from: http://www.ist-agave.org/results/D1.1-final-public.pdf.

10. Anand, S.P., S.; Ganesh, J. *Perspectives on service oriented architecture*. in *IEEE International Conference on Services Computing*. 2005.

11. Pallos, M. *Service-Oriented Architecture: A Primer*. [cited; Available from: http://www.eaijournal.com/PDF/SOAPallos.pdf.

12. Robinson, R. *Understand Enterprise Service Bus scenarios and solutions in Service-Oriented Architecture, Part 1*. [cited; Available from: http://www-128.ibm.com/developerworks/webservices/library/ws-esbscen/.

13. Danfeng, Y. and Y. Fangchun. *A Universal Service Layer Management Model in NGN*. in *Advanced Communication Technology, The 9th International Conference on*. 2007.

14. Grida, I., et al. *Next/New Generation Networks Services and Management*. in *Networking and Services, 2006. ICNS '06. International conference on*. 2006.

15. Yin, Z.M., F.C. Yang, and Y.Z. Liu. *Service management architecture and information model for next generation network with dynamic service level agreement management*. in *Networks, 2004. (ICON 2004). Proceedings. 12th IEEE International Conference on*. 2004.

16. Lin, H.G.X. *Research and design on inter-enterprise application integration model*. . in *Ninth International Conference on Computer Supported Cooperative Work in Design*. 2005.

17. Nolle, T. *A New Business Layer For IP Networks*. . [cited; Available from: http://www.ipsphereforum.org/newsevents/07nollereprint.pdf.

18. D., G. *TEQUILA - Traffic Engineering Quality of service in the Internet at Large scale*. . [cited; Available from: http://www.ist-tequila.org/deliverables/D0-2.pdf.

19. *IETF - The Internet Engineering Task Force*. . [cited; Available from: http://www.ietf.org.

20. *TMF – Telemenagement Forum*. [cited; Available from: http://www.tmforum.org.

21. Daho, Z.B.S., N.; Chevanne, M.; Betge-Brezetz, S. *An information model for service and network management integration: from needs towards solutions*. in *IEEE/IFIP Network Operations and Management Symposium - NOMS 2004*. 2004.

22. Gallen, C.R.a.R., J. S., *Using Open Source to realise a NGOSS Proof of Concept*, in *10th IEEE/IFIP Network Operations and Management Symposium - NOMS 2006*. 2006.

23. Gupta, B.a.S., M. *Business Integration Architecture for Next Generation OSS (NGOSS)*. 2007 [cited; Available from: http://www.infosys.com/industries/communication/white-papers/Business-integration.pdf.

24. Alateras, J. *IPsphere Framework Technical Specification - Release 1*. 2007 [cited; Available from: http://www.ipsphereforum.org/R1Spec.html.

25. Meijuan, W., X. Debao, and G. Jintao. *A Distributed Network Management Model for Next Generation Internet Based on XML and Policy*. 2006.

26. Carvalho, G.T.a.P., *CORDENA – Um framework para gestão de redes baseada em políticas*, in *8ª Conf sobre Redes de Computadores, Workshop CRC-Ensino (CRC2005)* 2005, FCCN: Portalegre, Portugal,.

27. Franco, T.F., et al. *Substituting COPS-PR: an evaluation of NETCONF and SOAP for policy provisioning*. in *Seventh IEEE International Workshop on Policies for Distributed Systems and Networks, 2006. Policy 2006*. . 2006.

28. Mi-Jung, C., et al., *XML-based configuration management for IP network devices.* Communications Magazine, IEEE, 2004. **Volume 42**(Issue 7): p. Pages: 84-91.

29. Clemente, F.J.G., G.M. Pérez, and A.F.G. Skarmeta, *An XML-Seamless Policy Based Management Framework*, in *Computer Network Security*. 2005. p. 418-423.

30. Rajan, R.V., D.; Kamat, S.; Felstaine, E.; Herzog, S., *A policy framework for integrated and differentiated services in the Internet.* IEEE Networking, 1999. **Volume 13**(Issue 5): p. Pages:36 - 41

31. Jeffrey Liu, S.J., Hicks Lin. *Introduction to Diameter*. [cited; Available from: http://www-128.ibm.com/developerworks/library/wi-diameter/index.html.

32. Lopez, G.G., A.F.; Marin, R.; Canovas, O. *A network access control approach based on the AAA architecture and authorization attributes*. in *Proceedings of 19th IEEE International Parallel and Distributed Processing Symposium*. 2005. Denver, Colorado - USA.

33. Franco, T.F., et al. *Substituting COPS-PR: an evaluation of NETCONF and SOAP for policy provisioning*. 2006.

34. Meijuan, W., X. Debao, and G. Jintao. *A Distributed Network Management Model for Next Generation Internet Based on XML and Policy*. 2006.

35. Mi-Jung, C., et al., *XML-based configuration management for IP network devices.* Communications Magazine, IEEE, 2004. **42**(7): p. 84- 91-84- 91.