Matos F, Matos A, Simões P *et al.* Provisioning of inter-domain QoS-aware services. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 30(2): 404–420 Mar. 2015. DOI 10.1007/s11390-015-1532-3

Provisioning of Inter-Domain QoS-Aware Services

Fernando Matos [1], Alexandre Matos [2], Paulo Simões [3,4], *Member, IEEE,* and Edmundo Monteiro [3,4], *Member, ACM, IEEE*

[1] *Department of Computer Systems, Federal University of Paraíba, João Pessoa 58051-900, Brazil*

[2] *Department of Information Systems, State University of Santa Catarina, São Bento do Sul 89283-081, Brazil*

[3] *Department of Informatics Engineering, University of Coimbra, Coimbra 3030-290, Portugal*

[4] *Centre for Informatics and Systems, University of Coimbra, Coimbra 3030-290, Portugal*

E-mail: fernando@ci.ufpb.br; alexandre.matos@udesc.br; {psimoes, edmundo}@dei.uc.pt

Received February 21, 2014; revised September 15, 2014.

Abstract Cooperation among service providers, network providers, and access providers in the Internet allows the creation of new services to offer to customers that are in other domains, thus increasing revenue. However, the Internet heterogeneous environment, where each provider has its own policies, infrastructure and business goals, hinders the deployment of more advanced communication services. This paper presents a Quality of Service (QoS) for Inter-Domain Services (QIDS) model that allows inter-domain QoS-aware services to be defined, configured, and adapted in a dynamic and on-demand fashion, among service providers. This is accomplished by: 1) the use of a common communication channel (business layer) where service providers publish and search for services, and interact with each other to contract and manage these services; 2) the templates to specify the business and technical characteristics of the services; 3) the automatic composition of services using service elements (smaller services) according to performance and service-specific QoS parameters; and 4) the creation and enforcement of configuration rules for the underlying infrastructure. A prototype was implemented to validate QIDS and performance tests were conducted on an inter-domain Border Gateway Protocol (BGP)/Multiprotocol Label Switching (MPLS) Virtual Private Network (VPN) scenario.

Keywords service provisioning, inter-domain, QoS, service composition

1 Introduction

Providing end-to-end quality of service (QoS) over the Internet is still a complex task. The heterogeneity of policies, equipments and business goals hinders providers (i.e., content providers, network providers) from cooperating with each other to offer services beyond best-effort quality levels across their domains. Nowadays, to provide a QoS-aware service that crosses multiple domains, the involved providers use human-based interactions (e.g., faxes, e-mails) to establish contracts and exchange configuration parameters. Moreover, they also need to manually configure the necessary equipment. This status-quo is disadvantageous for both the customers and the providers. If a customer needs a QoS-aware service other than those services of the access provider portfolio, it is unlikely that this service will be provisioned in an on-demand fashion. In this case, the customer must adjust his/her needs to the terms and guarantees that the access provider offers, which conflicts with one of the service convergence premises, that is, entities should cooperate to meet the customer needs[1].

In providers' perspective, the lack of advanced cooperation mechanisms to offer end-to-end QoS-aware service restrains the opportunity to increase their market share. Some content provider may own a service that fulfills the customer requirements. However, if the customer resides outside the content provider's domain and the access provider does not have a previously established contract with the content provider, the service may be not satisfactorily provided, since the content provider cannot guarantee the customer requirements.

Regular Paper

This work was partially funded by FCT under Grant Nos. SFRH/BD/16261/2004 and SFRH/BD/29103/2006.

One can argue that providers do not want to cooperate, thus protecting its customer base so that they do not have to share the profit. However, Ma *et al.*[2] have shown that if providers use global (inter-domain) strategies to provide services, they can maximize the aggregate profit of the aggregated system. Moreover, the presence of distinct players (e.g., content provider, access provider) stimulates competition and compels providers to look for new business models in order to increase their revenue by cooperating with each other[3].

Steps towards effective interaction between providers to achieve a more advanced cooperation have been proposed in the industry and scientific communities. The service delivery platform (SDP) emerged as a set of standards that support the service life-cycle[4], thus facilitating the interaction between providers to manage these services. Some SDP-like solutions have been developed, such as IPsphere① and global business framework (GBF)[5]. Other solutions tackle the issue of providers' cooperation in an operational support system (OSS) level, such as Multi-Technology Operations Systems Interface (MTOSI)② and [6]. Although these studies handle with the problem of inter-domain service provisioning, there are still no effective mechanisms for supporting QoS guarantees in the inter-provider scenario.

Another aspect concerning QoS-aware service provisioning is how to support services other than connectivity. Nowadays, providers want to diversify their service portfolios. They do not want to rely only on performance parameters (e.g., delay, jitter) to differentiate their services. To expand their offers, providers need to consider the specific QoS parameters of each service. For instance, frame rate and encoding type are specific QoS parameters of video services, while storage capacity is a specific QoS parameter of backup services. By doing this, providers can increase their market share, since they have more service options to offer.

The main proposal of this paper is to present a QoS model to support automatic and on-demand end-to-end service establishment. This model called QoS model for Inter-Domain Services (QIDS) employs a three-phase composition process that uses performance and service-specific QoS parameters to compose the service and it also deals with QoS adaptation. It facilitates the cooperation between providers by employing the following mechanisms: 1) a common communication channel (business layer) that providers use to publish and search for services, and to interact with each other to contract and manage these services; 2) the definition of templates to specify the business and technical characteristics of the services and those used as documents to exchange the service information; 3) the automatic composition of services considering performance and service-specific QoS parameters; and 4) the creation and enforcement of configuration scripts on underlying equipment. This work rests on the assumption that a federation of providers will be formed[7-8], which means that all providers agree to a standard representation of templates' information, such as employing the same representation for QoS parameter values. Moreover, it is assumed that a service level agreement (SLA) is established between the parties, and thus the service requisites are well defined. Finally, it is also assumed that providers employ monitoring nodes to check the contracted traffic as stated in the SLA.

The use case scenario chosen to validate QIDS involved establishing an inter-domain Border Gateway Protocol (BGP)/Multiprotocol Label Switching (MPLS) Virtual Private Network (VPN)[9] to support video streaming with QoS requirements. This scenario was emulated and it was mainly chosen for three reasons: 1) inter-domain VPNs are services that are often offered by providers to their customers; 2) the establishment of a VPN is an awkward task that involves human intervention and manual configuration; this leads to long-term VPNs, which makes it difficult to offer applications that require more granular, "immediately available" and short-term VPNs; and 3) VPNs are a convenient means of supporting QoS in inter-domain scenarios, since they are able to separate the transmitted traffic. This paper extends the preliminary work presented in previous papers[10-11] by modeling the entire service composition process, depicting service composition and service adaptation operations, and presenting the automated equipment configuration mechanism. Moreover, it also presents performance results from service composition and adaptation test scenarios.

The rest of this paper is organized as follows. Section 2 discusses some related studies. Section 3 presents the mechanisms used by QIDS to provide inter-domain QoS-aware services. In Section 4, experimental tests and results from an inter-domain BGP/MPLS VPN

① IPsphere framework technical architecture (release 2.0), July 2010. http://www.tmforum.org/TechnicalReports/TR158IPsphere-Technical/42835/article.html, Sept. 2014.

② MTOSI release 2.0, April 2009. http://www.tmforum.org/MTOSI/2319/home.html, Sept. 2014.

scenario running over a prototype are presented and discussed. Finally, Section 5 concludes the paper and presents future work.

2 Related Work

The main barrier to guarantee QoS levels in inter-domain services lies on the heterogeneity and the weak cooperation among providers. Some standardization groups have already made recommendations to achieve this objective.

The Software Enabled Services (SES) Management Solution (previously known as Service Delivery Framework (SDF))③, proposed by the TeleManagement Forum (TMF), is a management structure that enables the delivery of next generation services. It defines a set of standards to manage the service life-cycle (concept, design, deploy, operate, and retire), regardless of the technologies used to implement the services or to build the network infrastructure. The 3rd Generation Partnership Project (3GPP) standardization body set up an architectural framework, called IP Multimedia Subsystem (IMS)④, which was designed to deliver multimedia and voice services for mobile users through Universal Mobile Telecommunications System (UMTS) technology. In the context of fixed networks, the Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN) group of the European Telecommunications Standards Institute (ETSI) set out the TISPAN Next Generation Network (NGN)⑤. It is a subsystem-based architecture, where new subsystems are added over time to address the needs of new service classes and meet new demands. Like ETSI, the Telecommunication Standardization Sector (ITU-T) of the International Telecommunication Union (ITU) made its own NGN architecture recommendations that are currently being maintained by the NGN Global Standards Initiative (NGN-GSI)⑥. The objective of this architecture is to support the provisioning of content delivery and multimedia services. The TMF proposed a set of best practices and standards, called Frameworkx (formerly known as New Generation Operations Systems and Software (NGOSS))⑦, to implement a new generation of operational support system (OSS) and business support system (BSS). Its objective was to enable providers to develop and deploy new OSS/BSS solutions, thus they can be given support in their service management operations. The Multi-Technology Operations System Interface (MTOSI) standard, which is supported by TMF, is an open interface used to achieve interoperability between providers through their OSSs⑧.

Although these recommendations are well-specified, and widely accepted by the research community, together with the fact that some of them already have a reference implementation, they still need to incorporate other features that are needed to leverage the service provisioning process. These features include the following: the mediation of business relationships between the involved parties, the negotiation and delivery of end-to-end QoS in inter-domain scenarios in a dynamic and on-demand fashion, automatic service composition with the aid of smaller services, and the QoS adaptation of services.

Several researchers also have studied the provisioning of inter-domain QoS-aware services, thus resulting in different approaches to resolve this issue.

Mathieu *et al.*[12] proposed a cooperation between providers achieved by an open collaboration interface between network and applications (CINA). Each application requested by an user forms an overlay application that may consist of nodes provided by different service providers, thus forming the entire service. The overlay aims to satisfy the requirements of the service and also allow service adaptation. Landa *et al.*[13] developed a model to build overlay networks taking into account asymmetric costs between pairs of ISPs. Information gathered from ISPs is used to create a consolidated preference list that guides the assembly of the overlay network, thus resulting in a path to forward the traffic. Hakiri *et al.*[14] proposed a policy-based framework called Velox that applies an analytical performance model to deliver end-to-end paths with QoS performance guarantees. This framework propagates QoS-based agreements along the path to ensure the negotiated requirements. Jesus *et al.*[15] presented a frame-

③Software enabled services management, 2009. http://www.tmforum.org/ServiceDeliveryFramework/4664/home.html, Sept. 2014.

④IMS: 3GPP TS 23.228 release 11, June 2012. http://www.3gpp.org/DynaReport/23228.htm, Sept. 2014.

⑤ETSI: TISPAN. NGN functional architecture, Mar. 2010. http://www.etsi.org/tispan/, Sept. 2014.

⑥ITU-T: NGN-GSI release I, 2005. http://www.itu.int/en/ITU-T/gsi/ngn/Pages/default.aspx, Sept. 2014.

⑦TMForum: Frameworkx. http://www.tmforum.org/TMForumFrameworx/1911/home.html, Sept. 2014.

⑧MTOSI release 2.0, April 2009. http://www.tmforum.org/MTOSI/2319/home.html, Sept. 2014.

work that sets values to services based on utility functions, and thus these services can be selected. Another solution to provide QoS paths is proposed by Obreja and Borcoci[16]. In this work, logical QoS paths are established between known content servers (CSs) and predicted regions of content consumers (CCs), forming a whole connection graph. This graph crosses IP domains and contains all possible paths. Virtual Topology (VT) advertisement is the strategy proposed by Freitas *et al.*[17] VT is an abstraction of the network status of a domain. This strategy enables domains to advertise different VTs for different purposes. When the domain in charge of handling the customer request receives all requested VTs, it calculates a QoS path to provide the services. Wang *et al.*[18] introduced the concepts of Network Planes (NP) and Parallel Internets. NP is an abstraction within an Internet network provider (INP) that is used to differentiate IP traffic flow that crosses the INP domain and is forwarded accordingly. A parallel Internet is an aggregation of several NPs, with the same meta-QoS-classes, from several INPs to support a service request with QoS requirements. Burakowski *et al.*[19] described the framework proposed by the End-to-End Quality of Service over Heterogeneous Networks (EuQoS) European Project[20] system to ensure end-to-end QoS on heterogeneous multi-domain networks. It assumes that users can be attached on different access networks and it relies on the Enhanced QoS Border Gateway Protocol (EQ-BGP) also proposed in the EuQoS context to advertise the QoS information[21].

Despite the contributions of these studies, some of them do not cope with relevant aspects, such as on-demand service establishment and path building[16], results from test scenarios[18], and the use of service parameters and not only performance parameters[12-15]. Other work increases the overhead due to additional messages[17] or relies on BGP extensions[19-21], which can compromise the routing scalability.

To overcome the limitations of the above related studies, a QoS model (QIDS) to support end-to-end service establishment is presented in Section 3. The main contribution of QIDS is to support inter-domain QoS-aware service provisioning in a dynamic, on-demand and automatic fashion. Moreover, QIDS allows providers to compose end-to-end service paths that satisfy both the performance and the service-specific parameters. Additionally, it also handles adaptation requests in the case of contract violations. Finally, QIDS supports the enforcement of configuration rules in providers' equipment.

3 QoS for Inter-Domain Services

The goal of QIDS is to alleviate the complexities in providing inter-domain QoS-aware services by employing mechanisms to automate providers' cooperation. QIDS also enables providers to adapt the inter-domain services in the case they do not fulfill the agreed QoS requirements or one of the parties that constitute the service requests for the adaptation.

By using the business layer of the global business framework (GBF), QIDS supports providers in exchanging information about their services and service elements (smaller services). This information is stored in templates (XML documents) and comprises the service and service element characteristics used by QIDS to perform the service path composition. Once the path is created, contracts are established among the providers that compose the service path. These contracts state the providers' obligations and the service requirements. After the contract establishment, providers configure their equipments according to these service requirements.

In the following subsections, the inter-domain QoS-aware service provisioning is presented in detail.

3.1 Business Layer

QIDS was integrated on the GBF developed in our previous work[5], which was inspired by the IPsphere[⑨] and the Software Enabled Services (SES) Management Solution[⑩] (previously known as Service Delivery Framework (SDF)). In the same way as IPsphere and SES Management Solution, GBF aims to provide an infrastructure that supports providers in managing end-to-end services. However, although IPsphere and SES Management Solution are important studies concerning inter-domain service management, they are still very generic and do not provide details of crucial aspects of inter-domain management, such as the combination of elements needed to create an end-to-end path. They also fail to specify how to deal with service classes and do not provide details on how to handle service adaptations. On the other hand, the integration of GBF

[⑨] IPsphere framework technical architecture (release 2.0), July 2010. http://www.tmforum.org/TechnicalReports/TR158IPsphereTechnical/42835/article.html, Sept. 2014.

[⑩] Software enabled services management. http://www.tmforum.org/ServiceDeliveryFramework/4664/home.html, Sept. 2014.

along with QIDS creates an infrastructure that aims to fill these gaps. GBF is based on a business layer (BL) concept that acts as a communication channel between business entities located inside each operator. By using the BL as a communication channel, QIDS takes advantage of a collaborative infrastructure that clearly defines the business roles of the parties involved in the service provisioning process. Fig.1 presents the GBF with the QoS manager component that handles the QoS features on behalf of providers.

In the GBF, providers can play the role of either the service owner (SO), or the element owner (EO). The EOs are responsible to publish service elements at the Universal Description, Discovery, and Integration (UDDI), which acts as a service discovery repository for all providers. Service elements are smaller services that can be combined to create more advanced services. There are three types of service elements: 1) connection elements (CoE) — service elements that provide connection and/or transport services between two providers; 2) access elements (AcE) — service elements that provide access services for customers; and 3) application elements (ApE) — service elements that provide any type of application service apart from the two aforementioned service elements (e.g., e-mail, video server, FTP server, billing services, file storing, banking).

In the GBF, the SO is responsible to handle a customer service request by searching and combining service elements in order to provide a service that satisfies the customer requirements. A customer can request for services through its OSS, a front-end application, or by a third party web portal (e.g., a movie rental service). The operation calls between SO and EOs are performed by the means of Web services technology through the BL.

QIDS also supports service adaptation requests. A service adaptation process may be triggered by several reasons, such as customer dissatisfaction, financial compensation or technical problems, and can be requested by the customer or by any provider along the service provisioning path. Terms specifying the penalties for adapting the service must be described in the SLAs signed during the service configuration process. The difference between the service adaptation and the service configuration processes is that the former needs neither to search for available service elements nor to compose the service path, since the service elements that compose the service remain the same. If the SO detects that the same path is not enough to fulfill the adaptation request, it has to search for new service elements and calculate a new service path, which is a reconfiguration process. However, the adaptation process is preferred over the reconfiguration process, since the latter is time consuming and may demand a high resource restructuring on providers[22].

It is worth noting that in both the configuration and adaptation processes, the SO deals directly with each EO that is chosen to compose the service. EOs do not communicate with each other, which protects sen-

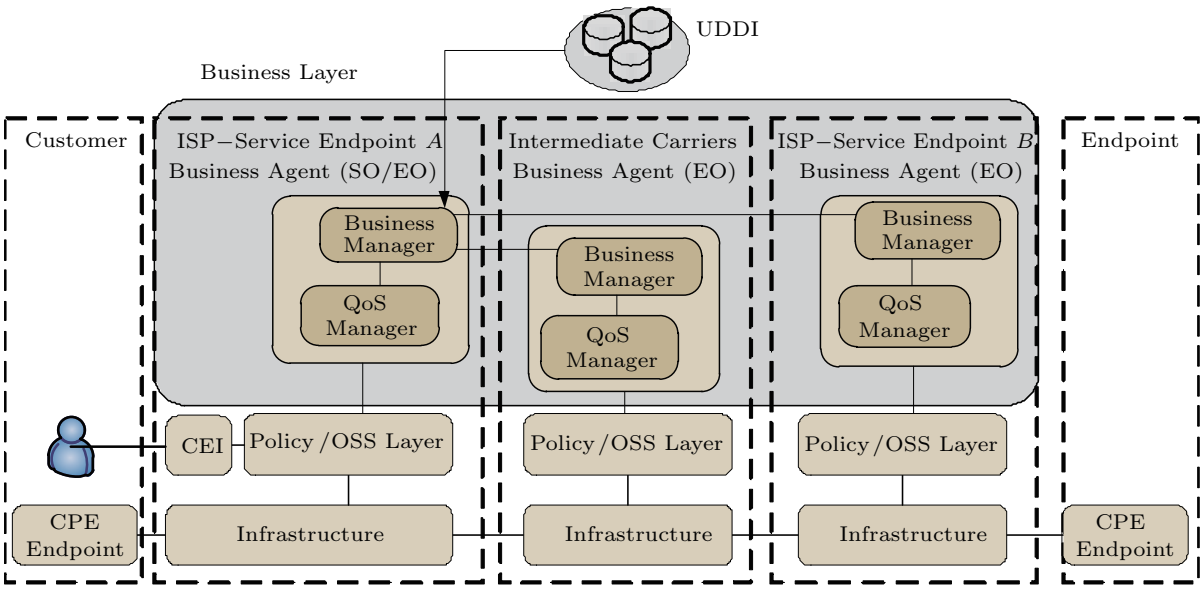

Fig.1. QIDS integrated on GBF architecture.

sitive business information. Each provider only knows about the service element it provides. Moreover, it may also prevent abusive behaviour from a neighbouring domain[23], since an EO could request from a neighbouring EO more resources than necessary to provide a service.

3.2 Service Templates

Services and service elements are described by means of service specification templates (SST) and element specification templates (EST) respectively. These templates are XML documents (inspired by TEQUILA European Project[11] and [24-26]) that contain business and technical information about the service/service element. They characterize the services/service elements according to their types and QoS classes. Some of the service/service element information that can be found in the templates includes: identification, type and class, owner identification and contact, QoS parameter values, reachability, monetary values, guarantees, monitoring aspects, etc. Among these, the QoS parameter values can be considered as the most important information. They define the parameter values supported by the service/service element for a specific class and can contain performance parameters and service specific parameters. Performance parameters are the usual connectivity parameters, such as delay and bandwidth. Service specific parameters are parameters that are specific to a particular service type, such as frame rate and audio channel parameters from video streaming services. By comparing the parameter values from different service elements, the SO is able to decide if they can cooperate to create a service path that fulfills the customer service requirements (see Subsection 3.3).

Fig.2 presents an EST fragment (with annotations) of a CoE, which is published by the EO at the UDDI to advertise the service element characteristics, such as the QoS parameter values supported by the CoE and the domains the CoE can connect. It is worth noting that other types of information can be used to represent the connections offered by CoEs, such as autonomous system (AS) numbers or IP address of edge routers. As stated before, there is more information included on templates; however, they are not presented for the sake of simplicity.

Templates of other service element types and/or services slightly differ from each other. For instance, ESTs of ApEs and AcEs do not give details about the domains they can connect. Instead, they have information about the service specific parameters the ApE offers and the QoS parameter values of its access link, respectively.

3.3 Service Composition

The service composition process aims to create an end-to-end service path that satisfies the requirements of the service requested by the customer. However, handling solely connectivity QoS requirements (e.g., bandwidth, delay and packet loss) is not sufficient. It is also necessary to cope with service-specific QoS requirements to achieve the customer satisfaction[27]. For instance, for a video streaming service, parameters such as frame rate and encoding must also be considered. In this work, a QoS parameter can be classified into four categories:

- *Additive*: the final value of the QoS parameter is the sum of the individual values along the provisioning path;
- *Multiplicative*: the final value of the QoS parameter is the product of the individual values along the path;
- *Concave*: the final value of the QoS parameter is defined by the minimum (or maximum) value in the path;
- *Independent*: the QoS parameter value does not need to be compared with other values (the service element itself guarantees that value or not at all).

The first three categories are a well-know classification for QoS parameters[28] and are usually applied for traffic performance parameters, while the fourth category was defined in the context of this work and is usually applied for service-specific parameters. This classification is used by the SO to exclude service elements in distinct moments of the service composition process.

Fig.3 illustrates the service composition process, where the three key phases of the composition are highlighted: *service elements filtering*, *graph building*, and *paths sorting and filtering*.

The algorithms of these three key phases are depicted next. In these algorithms, the *continue* command forces another interaction before reaching the end of the current loop and the *break* command forces the immediate termination of the current loop.

[11] Tequila - D1.1: Functional architecture definition and top level design, 2000. http://www.ist-tequila.org/deliverables/D1-1.pdf, Jan. 2015.

```
<ServiceElementTemplate>
 <ServiceElementOwner>                                  <!--
  <OwnerId>0351</OwnerId>                               <!-- Owner identification
  <OwnerName>Portugal Telecom</OwnerName>               <!--
 </ServiceElementOwner>
 <ServiceElementDescription>
  <ElementId>2351</ElementId>                           <!--
  <ElementName>ConnectionElement</ElementName>          <!-- Element identification,
  <ElementType>CoE</ElementType>                        <!-- type,class and publication date
  <ElementClass>silver</ElementClass>                   <!--
   <PublicationDate>2013-10-11</PublicationDate>
 </ServiceElementDescription>
 <SLS>
  <CarrierId>pt.pt</CarrierId>                          <!-- Identification and domain of the
  <CarrierDomain>pt</CarrierDomain>                     <!-- carrier that provides the CoE
  <ReachableCarriers>
   <ReachableCarrier>                                   <!--
    <ReachCarrierId>ft.fr</ReachCarrierId>              <!--
    <ReachCarrierDomain>fr</ReachCarrierDomain>         <!-- Identifications and domains of the
   </ReachableCarrier>                                  <!-- carriers that can be connected
   <ReachableCarrier>                                   <!-- to this CoE
    <ReachCarrierId>bt.uk</ReachCarrierId>              <!--
    <ReachCarrierDomain>uk</ReachCarrierDomain>         <!--
   </ReachableCarrier>
   </ReachableCarriers>
   <QoS>
    <Name>silver</Name>
    <Parameters>
     <PerformanceParameters>                            <!-- CoE traffic performance
      <Delay>                                           <!-- parameters
       <Qualitative>medium</Qualitative>                <!--
       <QuantitativeMax>200</QuantitativeMax>           <!-- Qualitative value, quantitative
       <QuantitativeMin>101</QuantitativeMin>           <!-- thresholds and unit of
       <Unit>ms</Unit>                                  <!-- the delay parameter
      </Delay>                                          <!--
      <Bandwidth>
       <Qualitative>medium</Qualitative>                <!--
       <QuantitativeMax>2500</QuantitativeMax>          <!-- Qualitative value, quantitative
       <QuantitativeMin>1000</QuantitativeMin>          <!-- thresholds and unit of
       <Unit>kbps</Unit>                                <!-- the bandwidth parameter
      </Bandwidth>                                      <!--
     </PerformanceParameters>
    </Parameters>
   </QoS>
  </SLS>
</ServiceElementTemplate>
```

Fig.2. EST connection element.

Initially, the SO must fit the customer requirements into one of its service policies (*policy matching*). This is a simple operation, since the SO determines the service level that is suited to the customer requirements by comparing these requirements with the parameter values laid down in the policies. By complying with the service policy rules and using the information from the ESTs, the SO starts the first phase of the composition (*service elements filtering*), which is depicted in the algorithm of Fig.4. In this phase, the SO checks if the domains of the ApEs and AcEs found at the UDDI, can reach the endpoints informed by the customer (line 4). If any of the service elements cannot reach an endpoint, it is discarded (lines 6∼8). The SO also compares the parameter threshold values from the service policy with the concave and independent parameters of each service element found at the UDDI. In the event that any parameter of a service element fails to comply with the corresponding threshold imposed by the service policy, the SO eliminates that service element (lines 9∼15). The output of this first phase consists of a list of service elements that comply with the parameters (concave and independent) from the service policy and can reach the endpoint domains specified by the customer (line 16).

During the second phase of the service composition process (*graph building*), depicted in the algorithm of Fig.5, the SO builds a graph representation of the connections between the service elements that were not discarded. To determine how they can be connected, the

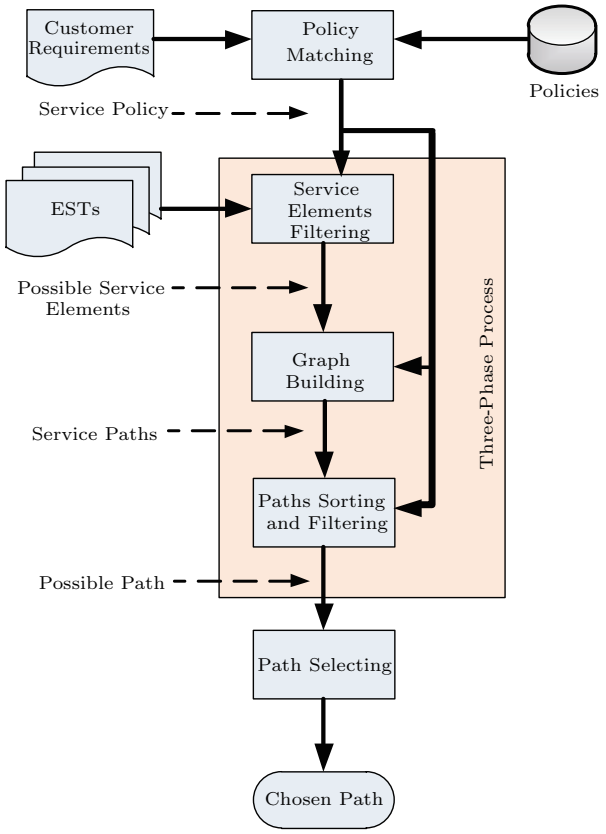

Fig.3. Service composition flowchart.

```
Input: seList, policyParamList, endpointList
Output: seList
// seList: List of service elements
// policyParamList: List of service policy parameters
// endpointList: List of endpoints
1 foreach (element ∈ seList) do
2   discard ← true;
3   foreach (endpoint ∈ endpointList) do
4     if (element = ApE or AcE) and ((ApE.domain =
        endpoint.domain) or (AcE.domain = endpoint.domain))
        then
5       discard ← false;
6   if discard then
7     eliminateFromList(element);
8     continue;
9   seParamList ← element.params;
10  foreach (policyParam ∈ policyParamList) do
11    if paramPolicy.type = CONCAVE or INDEPENDENT then
12      foreach (seParam ∈ seParamList) do
13        if (seParam.name = policyParam.name) and
            (seParam.value does not satisfy
            policyParam.threshold) then
14          eliminateFromList(element);
15          continue;
16 return seList;
```

Fig.4. Service elements filtering.

SO uses the reachable domain information contained in the templates of the CoEs and the domain information contained in the templates of the ApEs and AcEs. In this algorithm, the domains of the ApEs, AcEs and CoEs are defined as nodes, and are included in the graph (lines 4∼12). The algorithm also defines each

```
Input: AcEList, ApEList, CoEList
Output: G
// AcEList, ApEList and CoEList: List of service elements
// G: Graph containing the paths between the endpoints
1  Vertex V, Vr;
2  Edge E;
3  Graph G;
4  foreach AcE ∈ AcEList do
5    V ← defineAsVertex(AcE.domain);
6    G ← graphAddVertex(V);
7  foreach ApE ∈ ApEList do
8    V ← defineAsVertex(ApE.domain);
9    G ← graphAddVertex(V);
10 foreach CoE ∈ CoEList do
11   V ← defineAsVertex(CoE.domain);
12   G ← graphAddVertex(V);
13   foreach reachDomain of CoE do
14     Vr ← defineAsVertex(reachDomain);
15     if Vr ∉ G then
16       G ← graphAddVertex(Vr);
17     E ← defineAsEdge(CoE.domain, reachDomain);
18     if E ∉ G then
19       G ← graphAddEdge(E);
20 return G;
```

Fig.5. Graph building.

domain that is reachable by every CoE, as a node (lines 13∼14). A test is performed to find out if this node was already included in the graph. If it was not, the node is then included in the graph (lines 15∼16). The association between the domain of a CoE and the domains that are reachable by this CoE creates the edges of the graph (line 17). A test to avoid the inclusion of duplicate edges in the graph is also performed (lines 18∼19). If one of the reachable domains of a CoE matches the domain of an ApE (or AcE), this is proof that an edge exists between this CoE and the ApE (or AcE), since all the ApE and AcE domains were already included in the graph as nodes. The output of this phase is a graph that contains all of the possible paths between the endpoints (line 20).

During the last phase (*paths calculating and filtering*), it is necessary to calculate the cost of each path. To do this, the SO uses the rules regarding the QoS parameter categories (additive, multiplicative, concave, and independent) to compare the parameter values of every service element in each path. In this phase, another filtering process must be carried out so that only those paths that can guarantee the service requirements are selected. The algorithm of Fig.6 depicts this phase.

In this last phase, the SO calculates the k shortest paths using p as the path cost, where k is the number of paths that will be checked and p is any parameter, usually additive or multiplicative (e.g., price) (line 3). Both k and p are defined by SO internal policies. An implementation of the Bellman-Ford algorithm was used to calculate the shortest paths. Once the paths have

been calculated, the SO then sorts these k paths according to p (line 4). After the paths are sorted, the SO picks the first path in the sequence and compares the additive and the multiplicative parameter values of the path with the service policy requirements. If this path satisfies all the requirements, it is selected for the customer as a feasible service provisioning path. Otherwise, the SO repeats this process with the next path in the sequence until no path remains (lines 5∼15). It is worth remembering that both the concave and the independent parameters of the service elements were already compared in the first phase, which means that a graph where all the paths fulfill the requirements of these parameters has been produced.

```
Input: G, policyParamList
// G: Graph containing the paths between the endpoints
// policyParamList: List of parameters defined in the service
   policy
1  Graph G;
2  BellmanFord BlF;
3  pathList ← BlF.calculateKPaths(G, k, p);
4  sortPaths(pathList, p);
5  foreach path ∈ pathList do
6  |satisfy ← true;
7  |foreach policyParam ∈ policyParamList do
8  | |if policyParam.type = ADDITIVE or MULTIPLICATIVE then
9  | | |pathParam ← path.getPathParam(policyParam.name);
10 | | |if pathParam does not satisfy policyParam then
11 | | | |satisfy ← false;
12 | | | |break; // Check next path
13 |if satisfy then
14 | | a path was found;
15 | | break;
16 if not satisfy then
17 |  no path was found
```

Fig.6. Paths calculating and filtering.

To illustrate the service composition process, suppose that a customer requested a video streaming service and the SO found out the service policy of a service class that matches the customer requirements. Table 1 presents the performance (e.g., maximum delay, minimum bandwidth), the service-specific (e.g., encoding, maximum frame rate) and the business (e.g., maximum number of hops, maximum price the customer is willing to pay) parameter values the service must guarantee, according to this service policy. Table 2 presents the parameter values offered by each service element found by the SO at the UDDI. In the first phase, the SO compares the parameter threshold values from the service policy (Table 1) with the concave and the independent parameters of each service element found at the UDDI (Table 2). In the case that any parameter of a service element does not comply with the threshold from the service policy, the SO eliminates that service element. By comparing the values from both tables, it is possible to see that ApE2 and CoE3 do not satisfy the frame rate and the bandwidth requirements, respectively. Thus, these service elements are removed from the subsequent service composition phases.

Table 1. SO Service Policy (Video Streaming)

Parameter	Type	Value
Delay (ms)	Additive	$\leqslant 100$
Jitter (ms)	Additive	$\leqslant 90$
Bandwidth (Kb/s)	Concave	$\geqslant 512$
Encoding	Independent	MPEG-4
Frame rate (fps)	Independent	$\geqslant 20$
Audio channels	Independent	$\geqslant 4$
Number of hops	Additive	$\leqslant 3$
Price ($)	Additive	$\leqslant 200$

Note: we consider the number of hops as the number of intermediary domains (CoEs).

During the second phase of the service composition process, the SO builds a graph representation of the connections between the remaining service elements. These connections are established in accordance with the reachability information comprised on the CoE ESTs, which may result in several e2e service paths. Fig.7 shows the graph resulting from the connections between the remaining service elements of Table 2 and the values of the additive and multiplicative parameters of each path. As can be seen from the figure, there are three paths from the ApE1 (EO that hosts the video server) to the AcEs (EOs that can provide access service to the customer). However, not all paths will be able to deliver the service, which leads us to the third phase of the service composition process.

During the last phase, the SO calculates the shortest paths using some parameter as the path cost. Considering that the SO uses the price parameter to calculate and to sort the paths (illustrated in Fig.7), then it is easy to verify that path 2 will be chosen, since path 3, which is the cheapest, does not guarantee all QoS requirements.

3.4 Resource Configuration

In order to configure the service elements, the SO must incorporate the configuration information necessary to create the end-to-end service provisioning path into the SLAs and send them to the EOs. If any EO does not receive the configuration information before a timeout, the resource is released (marked as free). Otherwise, the EO creates a configuration script for

Table 2. Service Element Parameter Values

QoS Parameter	AcE1	AcE2	CoE1	CoE2	CoE3	CoE4	ApE1	ApE2
Delay (ms)	40	40	30	40	30	30		
Jitter (ms)	40	40	20	35	30	30		
Bandwidth (Kb/s)	512	512	1 000	1 000	384	512		
Encoding							MPEG-4	MPEG-4
Frame rate (fps)							30	15
Audio channels							4	2
Price ($)	50	35	40	35	25	30	40	30

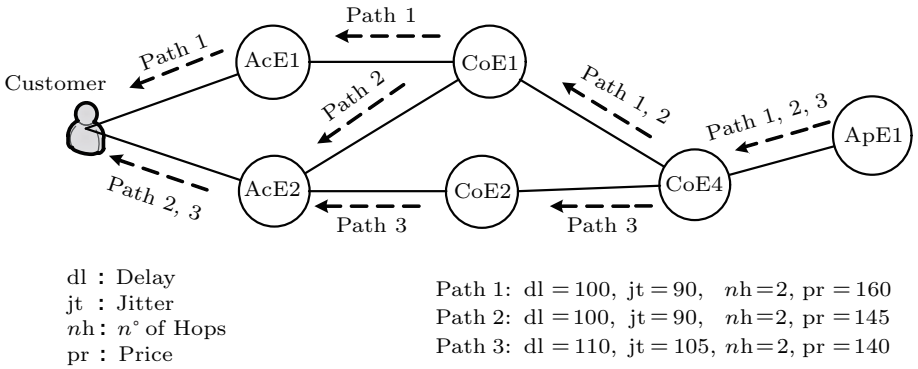

Fig.7. Connection graph and path costs.

the resource previously marked as booked. In the latter case, the EO may also update its service element offer at the UDDI, since its free resource capacity was reduced. The frequency of the update process is dictated by internal policies of each EO. If resources are booked and one of the providers or the customer gives up the service establishment, a penalty stated in the SLA can be applied. This is performed to compensate for eventual missed business opportunities.

The generation of the configuration script (illustrated in Fig.8) is a process that involves the combination of information from several sources. First of all, the EO gets the service element identification from the SLA so that it can fetch the correspondent EST. The EST then identifies the appropriate configuration policy, which has high level instructions. These high level instructions are in a pre-defined order and specify how the service element must be configured. They are mapped into the configuration commands of the vendor's equipment owned by the EO. There are basically three types of basic commands: independent commands that do not depend on external values, for instance, the command to enter in the configuration mode of the router (i.e., *config terminal*); EO dependent commands that depend on local information, such as IP addresses and port numbers of EO's routers; and SO dependent commands that depend on remote information received from the SO, such as Virtual Routing and Forwarding (VRF) name and IP addresses of neighbor edge routers. The EO and the SO dependent commands have a fixed part and a variable part. The variable part has arguments that receive values from the EO or the SO. Thus, when the EO receives the configuration request, it determines the set of basic commands associated to this configuration and substitutes each value, received from the SO or obtained from the EO itself, in the appropriate command argument. At the end of this process, the configuration script is created.

As can be noted from the aforementioned process, the generation of the configuration script is performed in accordance with the policies of each EO, thus guaranteeing provider autonomy. This autonomy assures three main advantages.

• *Security.* Other providers do not influence the configuration of one provider's equipment. The only external information the EO uses when configuring its equipment is the service parameter values previously accorded with the SO.

• *QoS Model Autonomy.* The EOs are free to use the QoS model (DiffServ, IntServ, MPLS) configuration they prefer inside their core networks. The only obligation is to deliver the traffic with the QoS requirements,

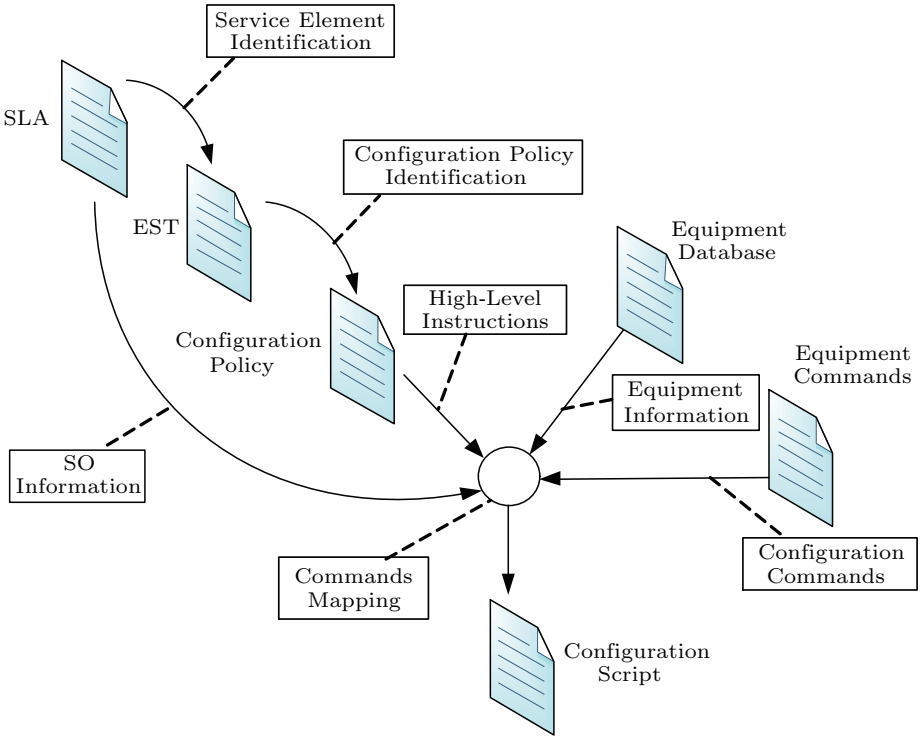

Fig.8. Configuration script generation.

agreed on the SLAs, to their connecting domains on the path.

• *Vendor Technology Autonomy.* Each EO configures its resources according to its vendor equipment technology (e.g., programming languages, operating systems).

After the script creation, the EO connects to the equipment and executes the script to prepare the resource for the service provisioning.

4 Inter-Domain Validation Scenario

In this section, a prototype of QIDS is presented. By using this prototype, experimental tests were performed in order to validate and evaluate QIDS.

4.1 Prototype Development

To validate QIDS, a prototype was implemented, using the Java programming language, and an inter-domain test scenario was developed. This scenario consists of an inter-domain BGP/MPLS VPN between two endpoints to support video streaming with QoS requirements. The topology of the testbed (Fig.9) consists of two transit domains (CoEs) and two endpoint domains (AcE and ApE) and each domain has an identification (bt.uk, ft.fr, pt.pt, and dt.de). According to RFC 4364[9] terminology, each transit domain has one

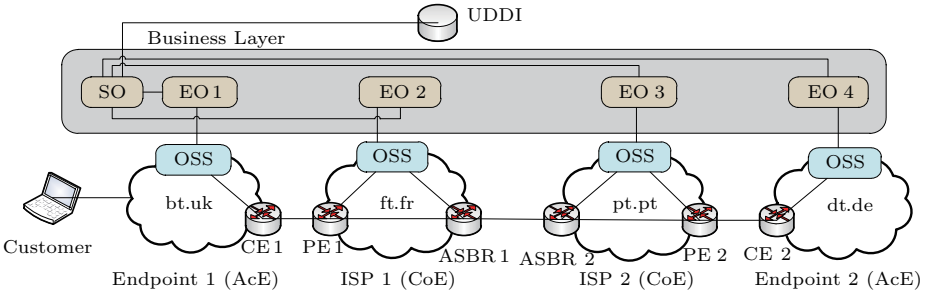

Fig.9. Inter-domain topology.

provider edge (PE) router and one autonomous system border router (ASBR) router. Each PE is connected to one customer edge (CE) router and the ASBRs connect to each other. The GNS3[12] router emulator (formerly known as Dynamips) and the Dynagen[13] front-end were used to emulate Cisco 7200 routers, which support BGP/MPLS VPNs configuration. This topology illustrates a real scenario, since it was verified that most of the inter-domain traffic (80% $\sim$ 90%) exchanged by an ISP travels only a few AS hops away (two to four hops)[29].

To establish a BGP/MPLS VPN, it is necessary to configure VRF[9] tables in each PE router. By using a VRF, it is possible to assign a virtual unique value for a customer's VPN. Therefore, users at a specific VPN cannot inspect packet traffics outside this VPN. To configure a BGP/MPLS VPN, the following information is necessary:

• VRF name: unique name that represents a VRF, and every PE router in the VPN must use the same VRF name;

• Route distinguisher (RD): a 16-bit or 32-bit number that helps to identify the VPN, and every PE router in the VPN must use the same RD;

• Interface or sub-interface IP address and autonomous system (AS) number of each endpoint: used to forward the traffic to the correct interface (or sub-interface) address on PE-CE connections; and

• Interface or sub-interface IP address and AS number of each ASBR router: used to forward the traffic to the correct interface (or sub-interface) address on ASBR-ASBR connections.

VRF name and RD are automatically generated by the SO, given that they must be unique for every PE on the VPN. Interface (or sub-interface) IP address and AS number of the CE, PE and ASBR routers are acquired from the templates of each service element. It is worth mentioning that service providers may still hide internal details from competing partners, since this exchanged technical information is mostly focused on border routers rather than on internal topology. For instance, those IP addresses are from edge routers (public addresses), which are not sensitive information that providers would be reluctant to share.

The configuration of the BGP/MPLS VPN is performed only on CoEs. Fig.10 presents a fragment of the template SO sends to each CoE (in this case, the ft.fr domain of the topology). In addition to the VRF name, RD, interface IP address, and AS number of each edge router, it also contains a unique identification for the service, the identification of the service element in the scope of the SO, the neighbor domain, and the alert URL, which is used to call the alert Web service on the SO. This information is necessary in the case the EO needs to send alert messages to the SO due to some service element problem.

When the EO receives the configuration request from SO, it creates the configuration script and connects to the equipment to enforce the configuration. Table 3 presents the commands executed by the EO to configure the BGP/MPLS VPN on a PE router (using IOS Cisco software syntax) according to the information presented on Fig.10.

Table 3. BGP/MPLS VPN Configuration

Cisco IOS Command	Description
*ip vrf vpnred*223	Enable the VRF configuration, defining a VPN instance with a VRF name (*vpnred*223)
rd 100: 100223	Create a routing and forwarding table for the specified VRF
route-target both 100: 100223	Create lists of import and export route-target extended communities for the specified VRF
*interface Serial*1/1.120 *multipoint*	Enter the sub-interface configuration mode
*ip vrf forwarding vpnred*223	Associate the VRF instance with the sub-interface
ip address 10.10.2.2 255.255.255.0	The sub-interface IP address is reconfigured (due to the VRF association, the sub-interface loses its IP address)
router bgp 1	Enter the BGP configuration mode
*address-family ipv*4 *vrf vpnred*223	Enter the address family configuration mode to configure routing sessions between PE and CE
neighbor 10.1.2.1 *remote-as* 65200	Add an entry to the BGP table
neighbor 10.1.2.1 *activate*	Enable the exchange of information with a BGP neighboring router
neighbor 10.1.2.1 *as-override*	This command is used to identify the site where a route originated, preventing routing loops between routers within the VPN

[12] GNS3 graphic network simulator. http://www.gns3.net/dynamips/, Sept. 2014.

[13] Dynagen. http://dynagen.org/, Sept. 2014.

```
<ElementParameters >
  <ServiceId>280</ServiceId>
  <ElementSOId>119</ElementSOId>
  <VRF>vpnred223</VRF>
  <RouterDistinguisher>100:100223</RouterDistinguisher>
  <CE-ASNumber>65200</CE-ASNumber>
  <CE-Domain>bt.uk</CE-Domain>
  <CE-IP>10.1.2.1</CE-IP>
  <ASBR-ASNumber>2</ASBR-ASNumber>
  <ASBR-Domain>pt.pt</ASBR-Domain>
  <ASBR-IP>10.4.4.10</ASBR-IP>
  <AlertURL>urn:alertservice;alert_service;AlertServiceService;
            receiveAlert;http://10.3.1.81:8080/axis/services/
            alert_service?wsdl
  </AlertURL>
</ElementParameters >
```

Fig.10. Service element configuration information.

The generated script also contains the required QoS configuration. In the sample scenario, let us assume a simple DiffServ-based configuration. Basically, to forward the traffic according to the QoS requirements, an access list is created to perform packet filtering and then the traffic from a specific source (interface or sub-interface) is associated to this access list. After that, a traffic class is created and associated to the traffic filtered by the access list. A QoS policy is created, which can associate a traffic class to one or more QoS actions. Finally, the QoS policy is associated to the interface (or sub-interface). Table 4 shows the QoS configuration enforced on a PE router. This configuration guarantees that all traffic that comes from the CE address is forwarded with a bandwidth of 512 Kb/s.

The configurations required for AcE2 and ApE1 are much simpler. In ApE1, the EO must send the video traffic with the appropriate QoS characteristics to the interface (or sub-interface) connected to PE2, while at the AcE2, the EO must redirect the traffic received from PE1 to the interface (or sub-interface) connected to the customer that supports the QoS characteristics. At the end of the configuration process of the service elements, the service can be provisioned.

4.2 Experimental Results

Experimental evaluation, covering service configuration and service adaptation, was undertaken using the aforementioned topology (Fig.9). These experiments aimed to validate QIDS and show that it can handle service establishment and adaptation requests using performance and service-specific parameters in a feasible time. Three PCs were used to perform the tests. One PC hosts the UDDI and the SO. Another PC "hosts" AcE2 and CoE1, while the last PC "hosts" ApE1 and CoE4. Each test was performed ten times and the values were averaged.

Table 5 presents the total time to handle 1, 10, 50, and 100 simultaneous configuration requests. The presented time seems to be very satisfactory, especially when compared with today's approach, in which providers use human-based interactions to exchange information and establish contracts, and configure their equipments in a manual fashion (which may take hours). Moreover, in the customer's point-of-view, considering that he/she requested for the QoS-aware service through a web portal, 3.3 seconds is fairly short for waiting and in most cases, is faster than the time to wait for the web page to load.

It is important to mention that the time presented here is concerned with the architecture performance to handle requests for service configuration (from customer request to resource configuration). In the case of BGP/MPLS VPNs, the total establishment time is much longer (about 2.5 minutes) due to the BGP convergence time to advertise the new routes. However, if pre-established routes are considered, the total configuration time of the QoS-aware services is similar to the time presented here, since there is no need for the BGP to advertise the routes. Furthermore, in a real provider scenario with real equipments and carrier-level servers, the results are expected to be much better.

Table 4. QoS Configuration

Cisco IOS Command	Description
*ip access-list extended vpnacl*1	Create an access list to perform packet filtering
permit ip 10.1.2.1 0.0.0.0 *any*	Associate the traffic from source 10.1.2.1 with the access list
class-map match-any vpnbasic	Create traffic class
*match access-group name vpnacl*1	Associate the access list with the created class
policy-map QoS	Create a QoS policy
class vpnbasic	Associate the traffic class with the created QoS policy
priority 512	Guarantee the specified bandwidth (Kb/s) for the traffic class
*interface Serial*1/1.110 *multipoint*	Enter the sub-interface configuration mode
service-policy output QoS	Associate the QoS policy with the sub-interface

Table 5. Service Configuration Requisition Time

Number of Requests	Time (s)
1	3.3
10	12.3
50	45.8
100	82.7

The service adaptation request use case is quite similar to the service configuration request use case. Since the adaptation only requires a modification at some service element or service requirements, the EOs that compose the service remain the same. Thus, there is no need to search for service elements and to compose the service path. Table 6 presents the time to handle 1, 10, 50, and 100 simultaneous adaptation requests.

Table 6. Service Adaptation Requisition Time

Number of Requests	Time (s)
1	2.5
10	8.8
50	35.1
100	66.7

The values from Table 6 show a slight decrease in the time to perform the service adaptation compared to the service configuration. As stated before, this difference occurs because the SO has neither to search for available service elements nor to compose the service path. Considering that the service path is not affected by the adaptation process, which excludes BGP route advertisements, the required adaptations on the service are performed in a short time period.

Table 7 presents the percentage of time major operations take during a customer request for configuration and adaptation of the video streaming service with QoS guarantees. As previously mentioned, operations between SO and EOs are performed by the means of Web services (WS). Other major operations are performed inside the SO and the EOs.

Table 7. Percentage of Operation Time

Operation	Configuration (%)	Adaptation (%)
SO/EO	19	12
WS	81	88

It is apparent from the table that the overhead of operations executed locally in the SO/EO is much lower than the overhead of operations that contain Web service calls. The Web service processing time of configuration and adaptation requests is about 81% and 88%, respectively. Web service calls are bandwidth and time expensive, given that they are based on Simple Object Access Protocol (SOAP) messages, which are textual in nature. However, this drawback does not outweigh the benefit of permitting providers to freely design their services by using interface definitions. This is an important advantage, since it allows the loose coupling of services, which facilitates the service composition process (a crucial prerequisite in the inter-domain provisioning). Additionally, when comparing the use of these Web service calls with the today's manual approach used by providers, we find the advantage is obvious: automated interaction between providers.

To evaluate the efficiency of QIDS, intermediary domains (CoEs) were added to the topology presented on Fig.9 and tested with service configuration requests. Fig.11 presents the time to handle service configuration requests in scenarios with two, three, four, and five CoEs. Five-hundred requests were performed for each scenario and each request was triggered immediately after the previous request was handled by QIDS. Each point in the curves represents the mean time of 50 successive requests.

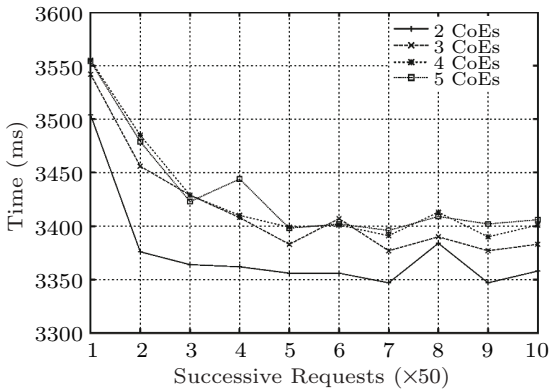

Fig.11. Mean time to handle service configuration requests.

As can be seen from the graph, the time to handle a service configuration request does not vary significantly as more CoEs are added to the configuration topology. The longer time at the beginning of each curve on the graph is due to the initialization of the Java classes. The mean time to handle a service request in the scenario with two CoEs is 3 376 ms, while the mean time to handle a service request in the scenario with five CoEs is 3 432 ms. In the case that more CoEs are added to the service path, it is expected that the time to handle a service request increases, since there are more domains

to negotiate and establish contracts. However, the difference between the mean time is very small, which shows the efficiency of QIDS in handling more complex scenarios (paths with more intermediary domains).

To evaluate the efficiency of QIDS in handling service adaptation requests, another test was conducted. In this test, there are five CoEs in the provisioning path. Fig.12 presents the time to handle service adaptation requests in four different scenarios (represented by the curves). In each scenario, the QIDS must adapt an increasing number of CoEs (from two to five). 500 requests were performed for each scenario and each request was triggered immediately after the previous request was handled by QIDS. Each point in the curves represents the mean time of 50 successive requests.

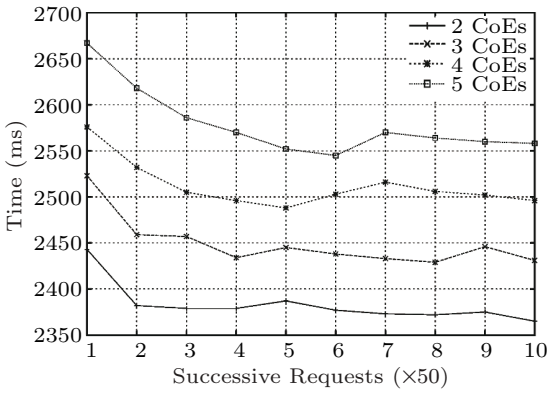

Fig.12. Mean time to handle service adaptation requests.

As the number of CoEs to adapt increases, the time QIDS takes to adapt the service also increases, which is an expected behavior. However, as also observed in the previous test, the increasing in the time is small. The mean time to handle a service adaptation request in the scenario with two CoEs is 2 383 ms, while the mean time to handle a service adaptation request in the scenario with five CoEs is 2 577 ms. This also shows the good efficiency of QIDS. Moreover, these results, along with the results from the previous test, demonstrate the advantage of QIDS in handling inter-domain QoS-aware services when compared to the manual process performed today.

5 Conclusions and Future Work

This paper presented a QoS for Inter-Domain Services (QIDS) model that employs mechanisms to enhance the interaction between providers, thus allowing them to achieve a more advanced cooperation to provide inter-domain QoS-aware services. By supporting these value-added services, QIDS benefits both the customer and the provider: the customer is not limited to his/her access provider service portfolio; and providers can increase their market share by offering services to customers that are outside their domains. Moreover, providers can exchange their current human-based interactions and manual configurations by an automated, on-demand, and dynamic process.

QIDS was integrated in the GBF architecture[5], using its business layer as a communication channel to facilitate the interaction between providers, which is achieved by the means of Web services. It can handle customer service requests with QoS requirements in an on-demand and dynamic fashion, by searching and composing services based on the templates of service elements published by EOs. To configure the underlying resources, configuration scripts are automatically generated and enforced on the equipments so that they can fulfill the customer requirements. QIDS also handles service adaptation. Any party can request for a service adaptation due to several reasons (technical problems, business non-conformances or financial downgrades).

An important advantage of QIDS is that it does not interfere with the providers' core network. In this sense, providers can use the QoS strategy they want according to their internal policies. Another plus of this proposal is to consider service-specific parameters together with connectivity parameters, which can increase the probability of customer satisfaction.

A testbed was developed to validate QIDS. Configuration and adaptation requests of inter-domain BGP/MPLS VPNs between two endpoints to support a video streaming with QoS requirements were the tested scenarios. The results showed that QIDS can handle customer configuration and adaptation requests in acceptable time, especially when considering the benefits in providing inter-domain QoS-aware services. The tests also showed that Web service calls are responsible for 81% of the time to handle a customer service request. However, this does not outweigh the advantage of Web services in allowing providers to freely design their services by using a loose coupled paradigm supported by interface definitions, which is essential to guarantee an automatic composition of services. Finally, the QIDS efficiency in handling more complex scenarios (more intermediary domains) was verified. The results showed only a slight increase in the time to: 1) handle customer configuration requests when more

CoEs are added in the service provisioning path; 2) handle customer adaptation requests when the number of CoEs to adapt increases.

As future work, studies to improve the performance of Web services calls will be conducted, for instance, reducing the size of SOAP messages, thus the serialization process can be improved. Moreover, studies on how to decrease the number of Web service calls will also be carried out. One possible approach is to use an expiry date of the offers in the templates. Thus, a provider would not need to confirm the offer if the service requests were performed before the expiry date. Another future work is to improve the service element search process by employing UDDI enhancements; thus the search process could be refined at the UDDI, for instance, to acquire all service elements at the UDDI that support certain bandwidth and delay.

References

[1] Huang C, Lee G M, Crespi N. A semantic enhanced service exposure model for a converged service environment. *IEEE Communications Magazine*, 2012, 50(3): 32–40.

[2] Ma R T B, Chiu D M, Lui J C S, Misra V, Rubenstein D. On cooperative settlement between content, transit, and eyeball Internet service providers. *IEEE/ACM Transactions on Networking*, 2011, 19(3): 802–815.

[3] Blum N, Boldea I, Magedanz T, Margaria T. Service-oriented access to next generation networks from service creation to execution. *Mobile Networks and Applications*, 2010, 15(3): 356–365.

[4] Yoon C, Lee H W. Service delivery platform for convergence service creation and management. In *Proc. the 12th International Conference on Advanced Communication Technology* (*ICACT*), February 2010, pp.1335–1338.

[5] Matos A, Matos F, Simoes P, Monteiro E. A framework for the establishment of inter-domain, on-demand VPNs. In *Proc. IEEE Network Operations and Management Symposium*, April 2008, pp.232–239.

[6] Ok K, Hong D W, Chung B. Design service for OSS based on the SOA and TMForum NGOSS. In *Proc. the 11th International Conference on Advanced Communication Technology*, Volume 1, February 2009, pp.423-427.

[7] Agiatzidou E, Courcoubetis C, Dugeon O, Johansen F T, Stamoulis G D. Inter-domain coordination models. In *Proc. International IFIP TC6 Workshops*, May 2012, pp.113–120.

[8] Pouyllau H, Douville R. End-to-end QoS negotiation in network federations. In *Proc. IEEE/IFIP Network Operations and Management Symposium Workshops* (*NOMS*), April 2010, pp.173–176.

[9] Rosen E, Rekhter Y. BGP/MPLS IP virtual private networks (VPNs), 2006. http://tools.ietf.org/html/rfc4361, Nov. 2014.

[10] Matos F, Matos A, Simões P, Monteiro E. A service composition approach for inter-domain provisioning. In *Proc. the 6th International Conference on Network and Services Management* (*CNSM*), October 2010, pp.487–492.

[11] Matos F, Matos A, Simões P, Monteiro E. QoS adaptation in inter-domain services. In *Proc. IFIP/IEEE International Symposium on Integrated Network Management* (*IM*), May 2011, pp.257–264.

[12] Mathieu B, Ellouze S, Schwan N, Griffin D, Mykoniati E, Ahmed T, Prats O R. Improving end-to-end QoE via close cooperation between applications and ISPs. *IEEE Communications Magazine*, 2011, 49(3): 136–143.

[13] Landa R, Mykoniati E, Griffin D, Rio M, Schwan N, Rimac I. Overlay consolidation of ISP-provided preferences. In *Proc. the 10th IEEE International Symposium on Parallel and Distributed Processing with Applications* (*ISPA*), July 2012, pp.506–511.

[14] Hakiri A, Berthou P, Gokhale A, Schmidt D C, Gayraud T. Supporting end-to-end quality of service properties in OMG data distribution service publish/subscribe middleware over wide area networks. *Journal of Systems and Software*, 2013, 86(10): 2574–2593.

[15] Jesus V, Aguiar R L, Steenkiste P. Inter-domain service alignment using consensus. In *Proc. International Conference on Computing, Networking and Communications* (*ICNC*), Jan. 30–Feb. 2, 2012, pp.421–427.

[16] Obreja S G, Borcoci E. Overlay topology based inter-domain QoS paths building. In *Proc. the 4th Advanced International Conference on Telecommunications*, June 2008, pp.64–70.

[17] Freitas R B, de Paula L B, Madeira E, Verdi F L. Using virtual topologies to manage inter-domain QoS in next-generation networks. *International Journal of Network Management*, 2010, 20(3): 111–128.

[18] Wang N, Griffin D, Spencer J, Griem J, Sanchez J R, Boucadair M, Mykomati E, Quoitm B, Howarth M, Pavlou G, Elizondo A J, Osma M L G, Georgatsos P. A framework for lightweight QoS provisioning: Network planes and parallel Internets. In *Proc. the 10th IFIP/IEEE International Symposium on Integrated Network Management*, May 2007, pp.797–800.

[19] Burakowski W, Beben A, Tarasiuk H, Śliwiński J, Janowski R, Batalla J, Krawiec P. Provision of end-to-end QoS in heterogeneous multi-domain networks. *Annals of Telecommunications*, 2008, 63(11/12): 559–577.

[20] Dugeon O, Morris D, Monteiro E, Burakowski W, Diaz M. End to end quality of service over heterogeneous networks EuQoS. In *Proc. the 4th IFIP Int. Conf. Network Control and Engineering for QoS, Security and Mobility*, Nov. 2005, pp.87–101.

[21] Beben A. EQ-BGP: An efficient inter-domain QoS routing protocol. In *Proc. the 20th Int. Conf. Advanced Information Networking and Applications,* April 2006.

[22] Lin K, Zhang J, Zhai Y. An efficient approach for service process reconfiguration in SOA with end-to-end QoS constraints. In *Proc. the 2009 IEEE Conference on Commerce and Enterprise Computing*, July 2009, pp.146–153.

[23] Pouyllau H, Douville R, Djarallah N, Sauze N. Economic and technical propositions for inter-domain services. *Bell Labs Technical Journal*, 2009, 14(1): 185–202.

[24] Georgievski M, Sharda N. A taxonomy of QoS parameters and applications for multimedia communications. In *Proc. International Conference on Internet and Multimedia Systems and Applications*, August 2003.

[25] Bouras C, Sevasti A. Service level agreements for DiffServ-based services' provisioning. *Journal of Network and Computer Applications*, 2005, 28(4): 285–302.

[26] Dobson G, Sanchez-Macian A. Towards unified QoS/SLA ontologies. In *Proc. IEEE Services Computing Workshops*, Sept. 2006, pp.169–174.

[27] Ibarrola E, Liberal F, Ferro A, Xiao J. Quality of service management for ISPs: A model and implementation methodology based on the ITU-T recommendation e.802 framework. *IEEE Communications Magazine*, 2010, 48(2): 146–153.

[28] Curado M. Quality of service routing for class-based networks [Ph.D. Thesis]. University of Coimbra, October 2005.

[29] Gürsun G, Crovella M. On traffic matrix completion in the Internet. In *Proc. the 2012 ACM Conference on Internet Measurement Conference*, November 2012, pp.399–412.

Fernando Matos is an assistant professor at the Department of Computer Systems of the Federal University of Paraba. He graduated in computer science from Federal University of Par (Brazil), received his master degree from State University of Campinas (Brazil) and his Ph.D. degree in informatics engineering from University of Coimbra (Portugal) in 2012. His research interests include network and service management, future Internet and middleware.

Alexandre Matos graduated in computer science in Federal University from Vicosa and received his master degree in computer science in Federal University of Santa Catarina, both in Brazil. Currently he is an assistant professor and with the research group in automation and communication networks of State University of Santa Catarina.

Paulo Simões is an assistant professor at the Department of Informatics Engineering of the University of Coimbra, Portugal, where he obtained his Ph.D. degree in 2002. He is also senior researcher at the Centre for Informatics and Systems of the University of Coimbra and regularly collaborates with Instituto Pedro Nunes as a senior consultant, leading technology transfer projects for industry partners such as telecommunications operators and energy utilities. His research interests include future Internet, network and infrastructure management, security, critical infrastructure protection and virtualization of networking and computing resources. He has over 100 publications in refereed journals and conferences, and he regularly serves on program committees of international conferences of these areas. He is member of the IEEE Communications Society.

Edmundo Monteiro is a full professor at the Department of Informatics Engineering (DEI) of the University of Coimbra (UC), Portugal. He is also a senior member of the Centre for Informatics and Systems of the University of Coimbra (CISUC). He graduated in electrical engineering (informatics specialty) from the University of Coimbra in 1984, and received his Ph.D. degree in informatics engineering (computer communications) and the habilitation in informatics engineering from the same university in 1996 and 2007 respectively. He has near 30 years of research and industry experiences in the field of computer communications, wireless technologies, quality of service and experience, network and service management, and computer security. He participated in many Portuguese and European research projects and initiatives. His publication list includes 6 books (authored and edited) and over 200 publications in journals, book chapters, and international refereed conferences. He is also a co-author of 9 international patents. He is a member of the Editorial Board of Elsevier Computer Communication and Springer Wireless Networks journals, and involved in the organization of many national and international conferences and workshops. Edmundo Monteiro is a member of Ordem dos Engenheiros (the Portuguese Engineering Association), and senior member of IEEE Communication Society, and ACM Special Interest Group on Communications.