X-domain QoS budget negotiation using Dynamic Programming*

Hélia Pouyllau
IRISA/INRIA, Campus de Beaulieu
F-35042 Rennes cedex, France
Helia.Pouyllau@irisa.fr

Laurent Ciarletta
LORIA, Campus Scientifique,
F-54506 Vandoeuvre-lès-Nancy Cedex, France
Laurent.Ciarletta@loria.fr

Armen Aghasaryan
Alcatel Research & Innovation
DIS - Delivery of Integrated Services
armen.aghasaryan@alcatel.fr

Stefan Haar
IRISA/INRIA, Campus de Beaulieu
F-35042 Rennes cedex, France
Stefan.Haar@irisa.fr

Abstract

Quality of Service (QoS) has been a major concern in the field of network management, even more so for emerging dynamic multimedia applications (Video on Demand, Telefony over IP etc.) that are becoming mainstream. This problem is particularly sensitive in the context of exchanges accross multiple independent and heterogeneous domains (X-domain), where global SLAs (Service Level Agreement) have to be satisfied accross domains. In this paper, we consider a typical scenario of X-domain provisioning of a video-conference session. The article addresses the problem of how to automatically negotiate QoS budgets between possible service providers (SP) that will meet the end-to-end requirements. We propose a QoS budget negotiation algorithm based on the dynamic programming principle. The negotiation is strictly distributed in the sense that all contracts are agreed on bilaterally between adjacent SPs, and each SP becomes responsible for the sub-budget between it and the end-domain.

1. Introduction

One of the main challenges of distributed applications concerns the Quality of Service (QoS). Deployed in a commercial context, the ability of distributed systems to guarantee QoS-requirements in contracts (SLA, Service Level Agreement) is a major topic for Service Providers (SPs). However, since applications tend to involve several heterogeneous domains in one service, ensuring global QoS becomes hard since no global dedicated service will in general be available, neither for computing composite QoS nor for ensuring it; this problem has been raised, for instance, in the ETSI TIPHON project reports [11] from which we will borrow the term of *QoS budget* below. Other recent works underline the importance of QoS ([7], [9]) and study the inter-domain QoS specification problem ([5], [6]). ([5], [6]) propose an end-to-end distributed verification of SLSs (Service Level Specification) with regards to domain service classes. Consider a request for a X-domain video conference. Each domain is supervised by local components, some of which are dedicated to the Quality of Service management; no single instance, however, can monitor or ensure the global QoS across the different domains. Our QoS budget negotiation process involves agreements between immediately adjacent Service Providers operating on different domains. Unlike the method in [1] where "negotiation [...] may require manual intervention", our negotiation process is automated by a distributed algorithm based on the dynamic programming paradigm.

We will start by explaining the context of this problem. Secondly, we will describe the QoS budget negotiation chain between SPs and particularly the data structures which intervene in this process. A third Section will give the description of the negotiation algorithm using interacting web services. Finally, we will discuss the open questions and future work.

2. Preliminaries and Context

QoS contracts, or rather QoS parts of SLAs, are com-

*This work is funded by the french RNRT Swan project.

posed of QoS objectives (SLOs, Service Level Objectives). SLOs can be represented by numerical components. A SLO is characterized by a name, a type (inequality, multiplicative etc.) and a value to be satisfied (e.g. "delay $\leq$ 50ms", "availability = 95%" etc.). Thus, the QoS part of an SLA corresponds to an N-dimensional numerical vector whose components correspond to thresholds in the SLOs. We treat all parameters involved as additive; in fact, multiplicative (e.g. availability) or statistical (e.g. 95% requests treated) constraints can be transformed taking logs or complementaries into an equivalent additive constraint.

Interdomain context. The Figure 1 represents communication links between two domains. Each domain is managed by a domain manager which supervises the internal nodes. The domain manager is also informed about the adjacency links which are inter-domain connexions. It regularly exchanges information about their adjacencies links so that the X-domain routing tables can be maintained. The Service Provider of a domain allows the end-users to access services. Consider the process of establishing a video conference the end-user u_0 who wants to begin a conference with another end-user u_t contacts the Service Provider of his domain. The Service Provider SP_0 of u_0 will ask its domain manager about the routes to follow to reach the final service provider SP_t of u_t. Then a chain of QoS contracts has to be set up between all Service Providers belonging to the route from SP_0 to SP_t. This negotiation for fixed route is the topic of the present paper; future work will include the choice of route in the negotiation process.

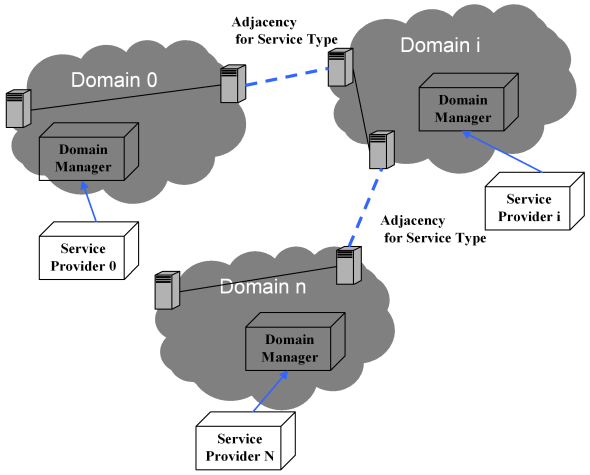

Figure 1. X-domain provisioning.

Intra domain components. Consider Figure 2. In each domain, the domain manager is in charge of supervision. It interfaces with a legacy network manager for local ressource control purposes. It also manages data exchanges with a Service Manager about the applicative services provided by the domain. This communication includes resource allocation requests, routing requests, alarms etc. The Service Manager receives requests from end-users invoking services. It can also receive service alarms from the corresponding monitoring devices. The Figure 2 illustrates internal and external communication between these components, all performed using web services.

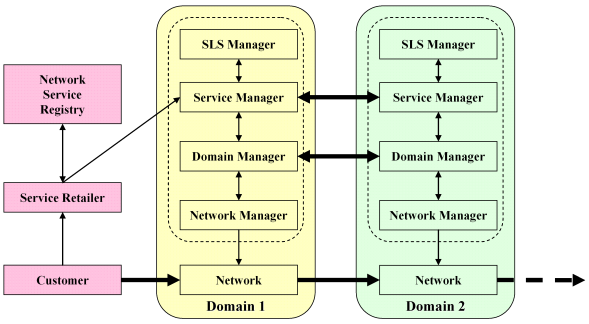

Figure 2. Components in charge of the management of a domain.

The deployed services correspond to Service type specifications handled by an operator through the SLS (Service Level Specification) Manager. A Qos contract negotiation occurs for a type (e.g. video-conference, VoIP) of provided services. With respect to those types, different QoS classes will be defined by the domain.

3. Ingredients of the Negotiation Process

As we have explained, the purpose of the negotiation process is to provide a QoS contract chain from a Service Provider SP_0, to the service provider of the target domain, SP_t. This process is activated with an overall QoS budget to be satisfied by the end-to-end contract chain. Thus, we construct a pool of contract chains in which the target Service Provider has to select the optimal one. Based on this description of a QoS budget negotiation scenario (illustrated by Figure 4), we propose a nested contractualization. For n Service Providers, SP_2 agrees on a contract with SP_1. SP_2 is thus responsible for finding and guaranteeing the QoS in the service

provided by its own domain *and* those on the remaining path to SP_n; the optimal path for n domains is obtained from the optimal path of those $n-1$, and so forth. We have applied the dynamic algorithmic paradigm (see [4]), which reduces a problem into subproblems sharing constraints; here, break the global QoS budget negotiation problem into neighbor-to-neighbor contract negotiations. Note that nested contractualization is coherent with the principle of autonomy of domains: no knowledge or control needs to be centralized, and no information on contracts or performance of a given domain needs to be communicated beyond its immediate contract partners (in fact, SP_i needs to "know" SP_{i-1} and SP_{i+1}, but neither SP_{i-2} nor SP_{i+2} as shown by Figure 3).

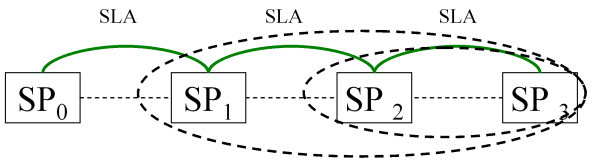

Figure 3. Nested contractualization.

3.1. Service provider

A Service Provider SP_i is associated to a domain d_i. It is the system unit in charge of providing services and contractualizing QoS objectives which are embedded in SLAs with other domains. A Service Provider computes two kinds of data: its set of admissible contracts w.r.t. the previous QoS budget received (vectors $\mathbb{Q}_{\mathbf{i-1}}$), ($\mathbb{C}_{\mathbf{i}}$ for SP_i) and the new set of requests (QoS budget vectors $\mathbb{Q}_{\mathbf{i}}$ for SP_i) to be satisfied by SP_{i+1} as a result of each possible local choice of contract.

3.2. QoS budget negotiation scenario

Recall that the QoS budget negotiation process between Service Providers is initiated by a Service Provider SP_0 seeking the optimal contract chain to a Service Provider SP_t subject to the criteria defined in an SLA $\mathcal{Q}_0$. SP_0 transmits its request to its neighbor SP_1 which computes the set $\mathbb{C}_\mathbf{1}$ of local admissible contracts. Thus, the request is transmitted to the neighbor of SP_1 and so forth until it reaches the destination SP_t. To transfer data between SPs, we define a data structure, $\mathbb{Q}$, which is a set of QoS budget vectors. The correspondence between a QoS vector and an admissible contract is saved locally by a SP. When a QoS budget is available for several contracts, only the "cheapest" contract is kept. When SP_t receives the request, it selects the cost-optimal contract chain among all admissible paths. At this point, the negotiation proper is terminated; next, the reservation process is launched by SP_t following the route of the elected optimal contract chain until it reaches SP_0. For this subsequent phase of reservation, a RSVP-type procedure [10] can be used : when a Service Provider SP_i receives a reservation request, it allocates temporarily the resources involved in its selected contract $\mathcal{C}_i^*$ of the optimal contract chain. The reservation request is constituted by the selected output QoS budget issued from the negotiation, $\mathcal{Q}_i^*$. Recall that SP_i stores the correspondences between selected admissible contracts $\mathbb{C}_\mathbf{i}$ and QoS budget output $\mathbb{Q}_\mathbf{i}$. SP_i forwards the reservation request with $\mathcal{Q}_{i-1}^*$ (based on the SLOs of $\mathcal{Q}_i^*$ and $\mathcal{C}_i^*$) as parameter to SP_{i-1}. When SP_0 receives the reservation request (note that the parameter is equivalent to $\mathcal{Q}_0$), it starts a validation process. The reservation becomes effective as soon as the validation process is terminated.

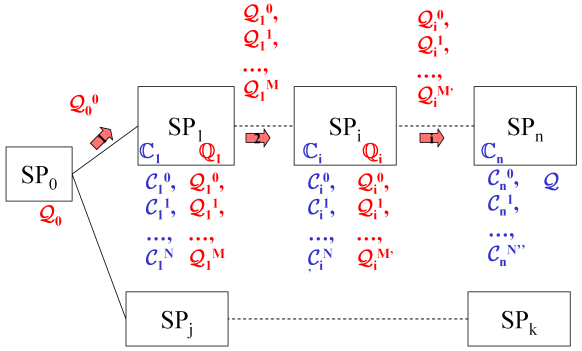

Figure 4. Negotiation process along the SP_0-SP_n path.

3.3. Routing data

In our prototype, we assume no dynamic routing problems have to be solved, that is, each Service Provider has a routing table. An entry of the routing table is composed of: the destination (the identifier of a Service Provider), the gateway (the identifier of the Service Provider SP_{i+1} which will permit SP_i to reach its destination). Accordingly, the routing table (handled at the domain level by the domain manager) will be used to know the identity of the next Service Provider which has to continue the negotiation process.

3.4. Local resources

The state of local resources available in a domain varies over time, with traffic changes, component failures, and so forth. As a result, the set of contracts proposed by an SP is variable and will not be the same in general for different requests; in fact, determining the QoS from the state of the domain is in itself a nontrivial problem. This question, however, is independent of the negotiation process but related to the QoS classe definitions and how to automate their monitoring. Moreover, we assume that during the (short) interval in which the contract negotiation that we describe here is carried out, no modification of the contract offers occurs. Here, we will not model the resources, and instead use a local predefined set of contracts $\mathbb{C}_\mathbf{i}^\mathbf{c}$ to simplify computations of the local set of admissible contracts $\mathbb{C}_\mathbf{i}$.

3.5. Qos budgets and contracts

The diagram 4 represents the negotiation process, in particular the data flow between SP_0 and a target Service Provider SP_t. The admissible private contracts are colored blue and the public QoS budgets colored red.

QoS budget and admissible contract. QoS budgets and admissible contracts are both SLA defined by numerical components (SLOs) and a cost. The main difference is that the cost and the SLOs of a QoS budget are computed whereas for a contract they are static.

Set of admissible contracts. $\mathbb{C}_\mathbf{i}$ is the set of admissible contracts computed by Service Provider SP_i. It consists of the $\mathcal{C}_i^j$ with $j = 0, ..., N_i$ such that vector $\mathcal{C}_i^j$ satisfies QoS budget vector $\mathcal{Q}_{i-1}^k \in \mathbb{Q}_{\mathbf{i-1}}$ with $k = 0, ..., N$the set of QoS budgets requested by SP_{i-1}. The term "satisfies" means the SLO characteristics of $\mathcal{C}_i^j$ are included in the $\mathcal{Q}_{i-1}^k$ SLO thresholds.

Set of QoS budgets. $\mathbb{Q}_\mathbf{i}$ is the set of $\mathcal{Q}_i^k \triangleq (\mathcal{S}_\mathbf{i}^\mathbf{k}, \mathbf{f})$ such that:

- $\mathcal{S}_\mathbf{i}^\mathbf{k}$ is an N-dimensional numerical vector whose components correspond to thresholds in the SLOs.
- $\mathbf{f}$ is the cumulative cost for $\mathcal{S}_\mathbf{i}^\mathbf{k}$, obtained as the sum of the local contract costs.

3.6. The algorithm, optimization with constraints

Each SP_i is in charge of composing an SLA and proposing it to its left neighbor. The contract contains obligations for $SP_i, \ldots, SP_t$, and is guaranteed by SP_i. Our optimization tends to minimize the size of the qos budgets set transmitted accross the SP chain. Contractualization propagates by backward nesting (see Figure 3).

Input

1. ordered set $d_1, \ldots, d_t$ of domains with associated SP's SP_i to be crossed; request received at d_1, target in d_t.
2. requested QoS budget $\mathcal{Q}_0 \in \mathbb{Q}^\mathbf{N}$ of SLA.

Algorithm

1. *Compute admissible paths "left-to-right"*
   (a) $\mathbb{Q}_\mathbf{0} \leftarrow \mathbb{Q}_\mathbf{0} \uplus \{\mathcal{Q}_\mathbf{1}^\mathbf{0}\}$ where $\mathcal{Q}_1^0 = (\perp, \mathcal{Q}_0, 0)$
   (b) Propagation For $i = 1$ to $N - 1$ do
      i. $\mathbb{Q}_\mathbf{i} \triangleq \mathsf{nil}$
      ii. Recursive computation of updated path list:
      While $\mathbb{Q}_{\mathbf{i-1}} \neq \mathsf{nil}$ do
      $\mathcal{Q}_{i-1}^j \triangleq (\mathcal{S}_{\mathbf{i-1}}^\mathbf{j}, \mathbf{f}) \triangleq \mathsf{FIRST}(\mathbb{Q}_{\mathbf{i-1}})$.
      Select admissible contracts
      $\mathbb{C}_\mathbf{i}^\mathbf{a} \leftarrow \{\mathcal{C}_\mathbf{i}^\mathbf{k} \mid \mathcal{C}_\mathbf{i}^\mathbf{k} \triangleq (\mathcal{S}_\mathbf{i}^\mathbf{k}, \mathbf{f})$
      s.t. $\mathcal{S}_\mathbf{i}^\mathbf{k}$ satisfies $\mathcal{S}_{\mathbf{i-1}}^\mathbf{j}\}$
      For each $\mathcal{C}_i^k$
      Set $\mathcal{Q} \triangleq (\mathcal{S}_{\mathbf{i-1}}^\mathbf{j} - \mathcal{S}_\mathbf{i}^\mathbf{k}, \mathbf{f})$
      if $\exists \tilde{\mathcal{Q}} \in \mathbb{Q}_\mathbf{i}$
      s.t. $\tilde{\mathcal{Q}} \equiv \mathcal{Q}$ or $\tilde{\mathcal{Q}} \leq \mathcal{Q}$
      if $\mathbf{f}(\mathcal{Q}) < \mathbf{f}(\tilde{\mathcal{Q}})$
      $\mathbb{Q}_\mathbf{i} \leftarrow (\mathbb{Q}_\mathbf{i} \uplus \{\mathcal{Q}\}) \backslash \{\tilde{\mathcal{Q}}\}$
      $\mathbb{C}_\mathbf{i} \leftarrow (\mathbb{C}_\mathbf{i} \uplus \{\mathcal{C}_\mathbf{i}^\mathbf{k}\}) \backslash \{\tilde{\mathcal{C}}\}$
      else
      $\mathbb{Q}_\mathbf{i} \leftarrow \mathbb{Q}_\mathbf{i} \uplus \{\mathcal{Q}\}$
      $\mathbb{C}_\mathbf{i} \leftarrow \mathbb{C}_\mathbf{i} \uplus \{\mathcal{C}_\mathbf{i}^\mathbf{k}\}$
      $\mathbb{Q}_{\mathbf{i-1}} \triangleq \mathbf{TAIL}(\mathbb{Q}_{\mathbf{i-1}})$.
      iii. Transmission to next SP.
2. *Select and propagate optimal path $\mathcal{Q}$.*
   (a) Let $\mathcal{Q}^* \triangleq (\mathcal{S}^*, \mathbf{f}^*)$ be the optimal QoS budget in $\mathbb{Q}_\mathbf{t}$ with $\mathbf{f}^* \triangleq \mathbf{f}^*(\mathcal{Q}^*$ minimal among the QoS budgets in $\mathbb{Q}_\mathbf{t}$.
   (b) For $i = N - 1$ to 1 do
      i. Reserve $\mathcal{C}_i^* = (\mathcal{S}_\mathbf{i}^*, \mathbf{f}^*)$ matching $\mathcal{Q}^*$
      ii. $\mathcal{Q}^* \leftarrow (\mathcal{S}^* - \mathcal{S}_\mathbf{i}^*, \mathbf{f}^*)$.

The Figure 5 exemplifies the first part of the algorithm execution between two Service Providers, SP_0 and SP_3. The initial QoS budget is composed of two SLOs (e.g. delay and jitter). The diagram must be read

from left to right: a new QoS budget SLO is the difference between a former QoS budget SLO and a local contract SLO. The paths which do not satisfy the QoS budget (e.g. $\mathcal{C}_1^1\mathcal{C}_2^2\mathcal{C}_3^1$) are discarded. The other paths are valid and SP_3 will select the optimal one.

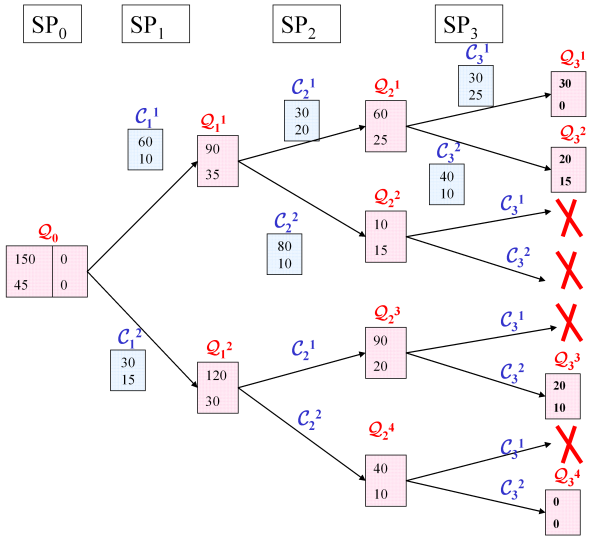

Figure 5. Example of execution.

4. Distributed negotiation process using Web Services

The QoS routing problem has already raised interest ([3], [2], [8]) but those works present centralized solutions which require knowledge of the entire system. Our approach is strictly distributed and implemented using web services.

4.1. Local functionalities

We explain here local functionalities used by the provided web services involved in the negotiation process.

getRoute(SP_t). This functionality searches in the local routing table the SP which permits to reach the target SP, SP_t. The routing is considered fixed.

getAdmissibleCs($\mathcal{Q}_i^k$, $\mathbb{C}_\mathbf{i}^\mathbf{c}$). In the first version of our prototype, this functionality searches in the local set of admissible contracts $\mathbb{C}_\mathbf{i}^\mathbf{c}$, the ones satisfying the QoS budget request $\mathcal{Q}_i^k$. It then returns a set of admissible contracts. A contract $\mathcal{C}_i^j = (\mathcal{S}_\mathbf{i}^\mathbf{j}, \mathbf{f})$ is admissible for a budget $\mathcal{Q}_i^k = (\mathcal{S}_\mathbf{i}^\mathbf{k}, \mathbf{f})$ if $\mathcal{S}_\mathbf{i}^\mathbf{k}$ satisfies $\mathcal{S}_{\mathbf{i-1}}^\mathbf{j}$.

saveContractQosbudgetSet($\mathbb{C}_\mathbf{i}$, $\mathbb{Q}_\mathbf{i}$). This functionality saves the set of admissible contracts, $\mathbb{C}_\mathbf{i}$. Only the *contract identifiers* are transmitted in the path. Thus, the SLOs of the proposed contracts stay private.

buildOutputData($\mathbb{C}_\mathbf{i}$, $\mathcal{Q}_{i-1}^k$, $\mathbb{Q}_\mathbf{i}$). This functionality builds and updates a set of ouput data vectors $\mathbb{Q}_\mathbf{i}$ from a set of contracts $\mathbb{C}_\mathbf{i}$ and from a QoS budget $\mathcal{Q}_{i-1}^k$.

Input:
  $\mathbb{C}_\mathbf{i}$: set of admissible contracts
  $\mathcal{Q}_{i-1}^k$: a qos budget
  $\mathbb{Q}_\mathbf{i}$: set of QoS budgets
Algorithm:
  For each $\mathcal{C}_i^j \in \mathbb{C}_\mathbf{i}$
    $\mathcal{Q}_{new} = (\mathcal{S}_\mathbf{new}, \mathbf{f}) \leftarrow$ computeQoSBudget($\mathcal{Q}_\mathbf{i-1}^\mathbf{k}, \mathcal{C}_\mathbf{i}^\mathbf{j}$)
    If $\mathcal{Q}_{new} \in \mathbb{Q}_\mathbf{i}$ Then
      $\mathcal{Q}_{old} = (\mathcal{S}_\mathbf{old}, \mathbf{f}) \leftarrow \mathcal{Q}_\mathbf{i}^\mathbf{k}$
        where $\mathcal{Q}_i^k \equiv \mathcal{Q}_{new}$ or $\mathcal{Q}_i^k \leq \mathcal{Q}_{new}$
      If $\mathbf{f}(\mathcal{Q}_{new}) < \mathbf{f}(\mathcal{Q}_{old})$ Then
        $\mathbb{Q}_\mathbf{i} \leftarrow \mathbb{Q}_\mathbf{i} \backslash \{\mathcal{Q}_\mathbf{old}\}$
        $\mathbb{Q}_\mathbf{i} \leftarrow \mathbb{Q}_\mathbf{i} \uplus \{\mathcal{Q}_\mathbf{new}\}$
      EndIf
    Else
      $\mathbb{Q}_\mathbf{i} \leftarrow \mathbb{Q}_\mathbf{i} \uplus \{\mathcal{Q}_\mathbf{new}\}$
    EndIf
  EndFor

computeOptimalPath($\mathbb{Q}_\mathbf{i}$). This functionality is performed by the target SP to select the cost optimal contract chain.

4.2. Provided global services

Negotiate web service. As each SP uses the same algorithm to negotiate the QoS budget, the algorithm starts with a check of the target Service Provider. If the current SP is the target, it computes the optimal path and initiates the reservation process. When the SP is an intermediary, it first computes its admissible contracts and builds the set of qos budgets. The associations between an admissible contract and a budget are saved, to keep private the admissible contract parameters and provide to the following SP only the set of budgets. Finally, the SP calls the next intermediary SP invoking the negotiate web service.

Input
  SP_t: the targeted Service Provider
  $\mathbb{Q}_\mathbf{i-1}$: the set of qos budgets
Algorithm
  If SP_i is $\neg SP_t$ then

$\mathbb{C}_{\mathbf{i}}^{\mathbf{c}} \leftarrow$ getPredefinedContract()
$\mathbb{Q}_{\mathbf{i}} \triangleq \mathsf{nil}$
For each $\mathcal{Q}_{i-1}^{k} \triangleq (\mathcal{S}_{\mathbf{i-1}}^{\mathbf{k}}, \mathbf{f}) \in \mathbb{Q}_{\mathbf{i-1}}$
$\mathbb{C}_{\mathbf{i}} \leftarrow \mathbb{C}_{\mathbf{i}} \uplus \{\text{getAdmissibleCs } (\mathcal{Q}_{\mathbf{i-1}}^{\mathbf{k}}, \mathbb{C}_{\mathbf{i}}^{\mathbf{c}})\}$
$\mathbb{Q}_{\mathbf{i}} \leftarrow \mathbb{Q}_{\mathbf{i}} \uplus \{\text{buildOutputData } (\mathbb{C}_{\mathbf{i}}, \mathcal{Q}_{\mathbf{i-1}}^{\mathbf{k}}, \mathbb{Q}_{\mathbf{i}})\}$
EndFor
saveContractQosbudgetSet($\mathbb{C}_{\mathbf{i}}, \mathbb{Q}_{\mathbf{i}}$)
$SP_{i+1} \leftarrow$ getRoute(SP_t)
ServiceCall(SP_{i+1}, negotiate(SP_t, $\mathbb{Q}_{\mathbf{i}}$))
Else
$\mathcal{P} \leftarrow$ computeOptimalPath($\mathbb{Q}_{\mathbf{i}}$)
ServiceCall(SP_{i-1}, Reserve ($\mathcal{P}$))
EndIf

Reserve web service. As our first prototype does not deal completely with the resource allocation, it reserves only predefined contracts. It performs a call to the reserve service of the previous SP. When it reaches the initiator, SP_0, the validation process is launched.

Validate Reservation web service. It activates the resource allocation reservation. When it is executed by the target SP, it sends a message to the initiator SP to inform it about the termination of the negotiation process.

5. Discussion

In this paper, we considered the context of X-domain QoS management. We presented a distributed algorithm for X-domain QoS budget negotiation. The dynamic programming principle our algorithm is based on, comes within both the architecture we consider (distributed web services) and the contract signatories requirements. Additionally, we mentioned some questions which remain unanswered. Our future work will focus on the following topics.

Dynamic renegotiation. This re-negotiation should not disrupt (or at least at possible) the former global QoS budget negotiation result. As a consequence, negotiating again a contract between two partners appears to be a problem of optimization with global constraints.

Resource allocation. Service Providers will not have a list of predefined QoS contracts $\mathbb{C}_{\mathbf{i}}^{\mathbf{c}}$ anymore. But they will propose some admissible contracts regarding its resource availabilities.

Dynamic routing. The negotiation algorithm can be extended to search for both routing path and optimal contract chain. We plan to use existing routing tables (provided by the domain manager) and routing criteria (e.g. domain banishment etc.) to curb routing table growth. Those criteria will permit to set up a preference policy. For example, a Service Provider often mentioned in optimal contract chains could be set as a privileged partner.

References

[1] A. Keller, G. Kar, H. Ludwig, A. Dan, and J. Hellerstein. Managing dynamic services: A contract-based approach to a conceptual architecture, 2002.

[2] M. Kodialam and T. V. Lakshman. Dynamic routing of locally restorable bandwidth guaranteed tunnels using aggregated link usage information. In *INFOCOM*, pages 376–385, 2001.

[3] M. Kodialam and T. V. Lakshman. Restorable dynamic quality of service routing. *IEEE Communications Magazine*, 2002.

[4] B. Korte and J. Vygen. *Combinatorial Optimization - Theory and Algorithms*. Springer, 2002.

[5] S. Lima, P. Carvalho, and V. Freitas. Self-adaptive distributed management of QoS and SLSs in multiservice networks. In *9th IEEE/IFIP International Symposium on Integrated Network Management*, 2005.

[6] S. Lima, A. Santos P. Carvalho, and V. Freitas. A distributed admission control model for CoS networks using QoS and SLS monitoring. In *IEEE ICC*, 2003.

[7] H. Ludwig. Web services QoS: External slas and internal policie or: How do we deliver what we promise? In *First Web Service Quality Workshop*, 2003.

[8] S. Nelakuditi and Z.-L. Zhang. Localized adaptive proportioning approach to QoS routing. *IEEE Communications Magazine*, 2002.

[9] Shuping Ran. A framework for discovering web services with desired quality of services attributes. In *ICWS*, pages 208–213, 2003.

[10] R.Bradan, L. Zhang, S. Berson, S. Herzog, and S. Jamin. Rsvp rfc 2205, resource reservation protocol, 1997.

[11] Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON). *Part3: Signalling and Control of end-to-end Quality Of Service (QoS)*. ETSI, 2002.