Customer Dynamic Service Subscription Architecture for Peer to Peer Services

D. Kagklis[1], D. Giannakopoulos[1], C. Charalampous[1], C. Tsakiris[1], N. Liampotis[1], G. Balis[2], and E. Sykas[1]

[1] Telecommunications Lab
Department of Electrical & Computer Engineering
National Technical University of Athens, 15773 Zografou, Athens, Greece
[2] Multimedia Laboratories, OTE Research
Hellenic Telecommunications Organization S.A.
15122 Marousi, Athens, Greece

Abstract. The pervasive influence of the internet as long as the insatiable desire for bandwidth have led the researchers to promote massive and disruptive changes within the Internet technology. The end goal is to enable the rapid development of a wide variety of services and support them in a unified and consistent manner. The existence of efficient algorithms providing differentiated high quality services is compulsory. Although researchers had made milestones as far as the core network is concerned developing architectures for efficient traffic engineering and providing assured quality of service, minor steps have been done regarding the customer side. We propose an open architecture for dynamic service creation and subscription based on a negotiation strategy. The ISP offers the customer an automated manner to specify the services and their characteristics he wishes to be subscribed. This model reduces the operational cost of an ISP and the time needed for a customer to request and access services.

1 Introduction

The IP (Internet Protocol) technology being ubiquitous, reliable and flexible has constituted the determining factor which led to the current development and wide spreading of the Internet.

Standard IP provides what is called "best effort" service. This allows the network to remain relatively simple. The small complexity of the network has not been a problem so far for the traditional network applications. However, the new breed of applications including video and audio streaming demands the ability of the network to provide appropriately differentiated Quality of Service (QoS) for network applications. QoS means providing consistent predictable data delivery service, or in other words satisfying customer application requirements. But "best effort" can make no guarantees about when data will deliver, or how much it can deliver. Increasing bandwidth, which is the obvious solution, is inevitable but the need for efficient protocols, providing differentiated high quality services, still remains.

So far two architectures have been proposed for providing quality of service the Integrated Service (IntServ) [1] and the Differentiated Service (DiffServ) architectures [2].

The IntServ architecture has as its main aim the distinction of services according to the applications demands. The distinction of the applications in two categories led the IntServ working group to specify two kind of services: a) the guaranteed services for the real time applications and b) the controlled load services for the non real time applications. The IntServ model provides strict QoS guarantees, but due to per flow management of traffic introduces severe scalability in the core network element i.e. router where the number of flows reaches up to millions. Also there is a need for a host-to-host signaling protocol. Finally, IntServ does not provide means for simple users with no technical knowledge to determine his QoS requests thoroughly.

The DiffServ model was introduced to avoid the problems of the IntServ model. Scalability is achieved by offering services on aggregate basis rather than per flow and by forcing as much as possible the per-flow states to the edge of the network.

The DiffServ domain distinct the network elements in two groups: a) the boundary routers, which mark, police, meter and classify the packets and b) the core routers, which classify the aggregated flows. The classification of the aggregated flows is performed according to the Service Level Agreement (SLA) signed between each customer and the Internet Service Provider (ISP) that supports the DiffServ model. SLA is a legal service contract between a customer and a service provider that specifies the forwarding service a customer should receive [3, 4]. This contract includes levels of availability, serviceability, performance, operation and other attributes of the service. A customer may be an organization or another service provider or a residential user. The customer includes a set of sites. With the term sites we define the set of users, which are located in the same network area.

In order to reach to an SLA, the ISP provides the customer with a standard form in which the supported services are described and the customer highlights his requests and returns the filled form to the ISP.

Until now, the communication between the customer and the ISP in order to result in a specific SLA, is carried out manually. This we call it *static subscription*, which is a method that is not reliable for the following reasons. A human-to-human communication method such as sending the form via a fax or e-mail makes frequent modification difficult and time-consuming. Also a customer may come with difficulties filling the form, which means extra time and frustration. There is always a time gap between the sign of a contract and the activation of a service. Additionally there is neither a standard SLA form nor a standard platform at the customer's side for service subscription. Finally the authentication and authorization procedures are performed manually.

In order to overcome the problems with the static subscription we propose a *dynamic service subscription method*. The dynamic subscription through the SLA technologies offers the customers with an automated manner to specify the services he wishes to be subscribed, offered by an ISP and their characteristics. We describe a method through which a customer can have access to one or more ISPs at the same time, can subscribe, unsubscribe to a variety of services and make new agreements as well as modify them *electronically* at any time. In the proposed architecture a negotiation algorithm gives the customer the ability to modify a number of parameters of the services provided according to his need and come to an agreement with the ISP according to the network utilization and supplied resources.

The scope of this paper is to propose architecture for the dynamic service subscription at the customer side based in the SLS technology. In Section 2 we describe the dynamic service subscription architecture illustrating the functional blocks it consists of. The algorithms and the procedure for this architecture are described in Section 3. In Section 4 the operation of the proposed model is analyzed. Furthermore, the implementation issues of the proposed method are presented. Finally, the summarized results and our concluded remarks are presented.

2 Architecture

In this section we describe the main functions that are performed, as well as the interfaces and the interoperability between them and the data structures that hold information needed for the subscription operation. Before specifying the functions of each block we make a brief reference to the SLS technology. Moreover, the network requirements and the assumptions needed to take into account are stated.

2.1 SLS-Terminology

The SLS is the dynamic part of the Service Level Agreement between the service provider and the customer. It holds the values of the configurable parameters of each service. These parameters specify the more technical oriented properties of the service [5]. In the DiffServ environment of our architecture, the SLS defines parameters for compliant network behavior [6]. The parameters that are needed for our model and comply with the Service Level Specification [7, 8] are:

Topological Info: It defines the Ingress and Egress site. In more detail, the boundaries of the supplied data of the specific service are configured.

FlowID: The values of source and destination user addresses are given. The application info and the DSCP are optionally supplied. Performance-Quality parameters: Values such as delay, loss, jitter and throughput are given in order to specify the quality of service. If not set the service is "best effort".

CustomerID: A unique id of each customer needed for the communication between ISPs and the customer.

2.2 Assumptions & Requirements

There are a number of assumptions and new terminology related to the dynamic service subscription model.

Firstly, both the Customer and the ISP have a pre-determined common filled-form document for their interaction, the *service template* [9], which contains a number of information entities. There are specific constraints regarding the data type of the information entities as well as their position and role in the document. Furthermore, it is assumed that the ISP always provides the service template.

Secondly, the ISP has an automated system for SLS manipulation [10]. It supports admission mechanisms for the request of a customer, who wishes to subscribe to a certain service. Additionally, in a common scenario where the ISP cannot accommodate

the customer's demands the first provides the latter with a list of alternatives, in order to avoid rejecting him from the first time. Moreover, a protocol that supports end-to-end service negotiation is presupposed. This protocol should allow customers and ISPs to negotiate for QoS agreements and pursue an agreement on the content of a specific service proposal, described within the content of the service template.

Thirdly, each time a customer connects to the ISP, he is submitted all the information for the available customer identifications sites and users that are available to have access to. Also it supports an automated procedure through which it informs each customer for the available services.

Fourthly, the algorithms and protocols for the activation of the agreed subscribed services are not of concern to the proposed model. Finally, it is assumed that the model uses the services of a reliable transport service.

2.3 Functional Model

The functional model of the dynamic service subscription method is presented in Figure 1. Below we outline the main functions that each block of the proposed architecture performs.

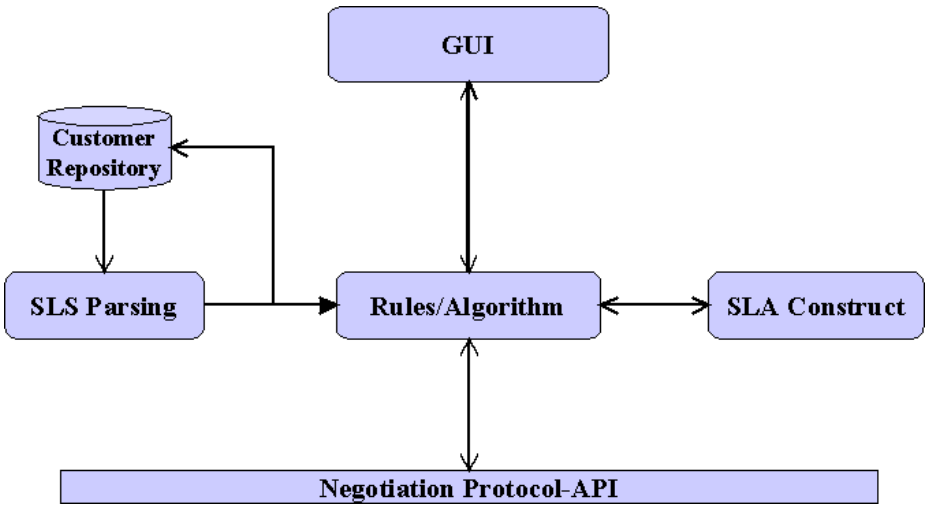

Fig. 1. Dynamic Service Subscription Architecture

Customer Repository: the Customer Repository is a data structure that holds information about all the users of the sites that comprise a customer and the unique customer's identity. Moreover, it holds a register with all the available services and servers that a specific ISP can supply. The information for the available services is described according to the pre-determined service template.

SLS parsing: SLS parsing manipulates the predetermine service template which is stored in the customer repository. Its main features are to provide a dynamic algo-

rithm for filling the SLS form and an efficient way for extracting the values and the information of the service template.

Rules - Algorithm: this function performs two roles. Firstly, it guides the customer to fill step by step, through the GUI, the service template and provides the user with all the information about available users and sites stored in Customer Repository. Secondly, after the completion of the service template it performs the subscription of the service to the ISP through the negotiation protocol API.

SLA Construct: Its main role is to create the contract according to the values given by the user and the rules specified by the SLS parsing module.

GUI: The GUI block provides the customer with a user-friendly graphical interface in order to fill the contract in an automated way and subscribe for a service of the corresponding platform of the ISP. The customer can specify his demands about a service in a high level logic, without having special technical knowledge.

3 Algorithms & Procedures

The proposed model consists of two main procedures, the SLA creation and the Negotiation procedure. During the SLA creation the contract specifying the requested service is composed. On the other hand, the Negotiation procedure describes the several operations needed in order to alternate specific values of information entities within the SLA through a dynamic reconfiguration mechanism.

3.1 SLA Creation

The performed operation of each functional block as well as the interaction and the data, which is exchanged between them, are described in this paragraph.

Primary, for the SLA creation the customer repository stores the information, such as the service template, the customer identification, the subscribed user and sites and the ISP's available servers, retrieved from the ISP. The main roles of the customer repository are to manipulate the stored data and provide other blocks with means in order to retrieve, modify and delete the stored data.

Then, this block feeds the SLS parsing block with the predetermined service template and the customer information. The functionality of the SLS parsing block is to process all of the above info and create a dynamic algorithm through which the customer is guided to fill the free attributes of the supplied service template. The dynamic algorithm, which comprises a set of rules, is created according to the structure of the service template. The set of rules specifies exactly the free fields of the service template that have to be completed mandatory or optionally and presupposes a concrete sequence of steps taken for a successful completion. Furthermore, the Rules and Algorithm block performs the mentioned sequence of steps and in collaboration with the GUI block it fills the fields for completion of the service template. After a successful verification of the produced service template by the Rules and Algorithm block the

given values of the filled template passed to the SLA construct block for the final step of the SLA creation procedure.

Finally, the SLA construct block uses the supplied values to create a service instance, which is conformed to the template and the rules of the dynamic algorithm. The created service instance is the contract, which will be negotiated with the ISP through the Negotiation API.

3.2 Negotiation

The Negotiation API is a software component that accepts requirements, offers input, negotiates with its peer and delivers a deal or conflict deal. This protocol should allow clients and servers in distributed object systems to negotiate for QoS agreements and pursue an agreement on the content of a specific service proposal. Several protocols have been presented that allow a client to attain from a distributed system a service with specific characteristics and guaranteed performance. (COPS SLS [11], HP [12], NAFUR [13], NRP [14], RNAP [18], DSNP [19], SrNP [8]). There is still no standard way for service negotiation, thus each ISP uses each own technique.

The negotiation API provides our architecture with the functionality for the intercommunication between customer and provider. Furthermore, there is an interaction between the negotiation protocol and the algorithm block in order to perform a service subscription after negotiation according to the dynamic algorithm's rules. The negotiation procedure can have three responses a) an immediate acceptance, b) a proposal of alternatives in case the ISP cannot accommodate the customer's demands and c) a rejection. As far as the proposal of alternatives is concerned, each time the customer receives an alternative; he creates a new service instance with the proposed alternatives or terminates the procedure as he has the last word for the agreement. Finally, in the scenario of an acceptance the Customer Repository stores the agreement-contract locally for further process.

4 Operational Phase

The following part of the paper describes how the functional model of the dynamic service subscription architecture may operate, based on the assumptions and the algorithms that were formerly presented. There will be explained how the different entities of this model interact and how the algorithms will be realized in the specific framework. This procedure can be divided into 4 parts:

4.1 Pre-Negotiation Phase

This is the registration and authentication phase for the customer. Firstly, the customer submits personal information (username, password and/or other personal details such as real name and address) to the ISP. If there exists an entry for this customer in the ISP's repository then the customer is identified and the operational phase proceeds to the 2nd stage. It should be mentioned that at the same time, the ISP provides the customer with significant information, which is then stored in the customer repository. Namely, these include:

- The names and a short description of the services, which are currently available by the ISP. If some services had been modified or a new service is being offered since the last time the customer logged on to the ISP, information about all the modifications is provided
- Information about the sites that the customer administrates (since the customer may be an administrator of many sites and not just a corporate end-user)
- Information about the servers that the ISP currently supplies such as video servers

4.2 Service Completion Phase

The main objective of this phase is the dynamic specification of the characteristics of the service that the customer wants to subscribe to. Practically, this consists of filling in forms, which have been produced by the SLS parsing function. The values, which are required to fill in these forms, are the configurable parameters of the service template. For instance, the customer has to choose which sites will be subscribed to this service, if connection to one of the ISP's servers is desired and also to supply values for the parameters that characterize the network in a DiffServ environment such as DiffServ code points (DSCPs) and entities such as bandwidth and delay, which define the way that the ISP will treat traffic.

The graphical environment (GUI) that was described it the architecture part of this paper, is employed for the visual realization of the described procedure. It provides the customer with a user-friendly and effective way to achieve the above purpose, i.e. by means of presenting the previously mentioned forms.

To specify the characteristics of the service, the customer merely has to fill in these forms (which are service-specific).

What is more, the graphical tool involves a step-by-step guidance in terms of completing them and avoiding and/or correcting any mistakes. This is based upon the dynamic rules and algorithms that are produced by the SLS parsing functions.

The whole procedure is very friendly towards the customer. There is no need to know very complicated information about networks. In addition, this allows the customer to easily modify the services' characteristics. A graphical representation of the topology of the network would make the above process even easier and more comprehensive.

After successfully filling these forms (that actually represent the customer's demands about the service), the values of the fields are converted dynamically through the SLA construct function to the pre-determined format and are submitted to the ISP.

4.3 Negotiation Phase

In this phase, a negotiation starts between the customer and the ISP in order to result to a specific SLA. The ISP examines the customer's requirements and decides if they can be accommodated (that decision may be based on factors such as the availability of network resources or some other policy). The ISP accepts or rejects the customer's demands. Also, if this specific service proposal cannot be accommodated, an alternative suggestion might be created and sent to the customer.

The customer has the ability to accept or reject the ISP's offer of alternatives and to create another service proposal according to the alternatives proposed values, which is then submitted to the ISP. The negotiation continues until an agreement is reached or one side decides to leave the procedure.

4.4 Agreement Phase

If the negotiation phase ends up in a mutual agreement, then the last service proposal becomes a contract between the customer and the ISP and both sides stores the agreement in their respective repositories. After that, the agreed service is available.

5 Implementations Issues

There are several issues that arise when implementing the proposed architecture for dynamic service subscription. A recommendation for accomplishing this purpose can be as follows:

Firstly, a way for modeling and describing the service template and the SLSs is required. As it has already been mentioned, a pre-determined common filled-form document is essential. This kind of document must define specific constraints regarding the data type of the information and their role in the whole process. These consist the input for the generation of the Rules/Algorithm functional block and are necessary in the SLA parsing and SLA construct functions.

XML language and XML-schemas [20] can be used in order to achieve the aforementioned goal. XML is appropriate in cases when there exists a need to describe entities of information without having any knowledge of the way they are implemented and administrated. It is designed to permit the exchange of formatted information over the World Wide Web and provides a standardized framework for the description of data. Moreover, many powerful tools (e.g. parsers), which are easy to use and to program, have been developed in every platform for converting different kinds of data to XML format and for extracting values out of XML files.

XML-Schema, which was released as a full W3C Recommendation in May 2001, gives the power to completely reflect the data types, constraints and structure of a data set using the XML syntax. It allows the handling and processing of the data model through XML tools. Complex data types can be defined from basic ones and additionally it provides a way to express inheritance. Furthermore, XML files can be validated according to a given schema.

Taking all these into consideration, in order to describe the SLS-template, an XML-schema with data types, which correspond to the SLS parameters, must be created. Every SLS will be an instance of this scheme, i.e. an XML file. So, the SLS parsing function will be substantially an XML parsing procedure and the SLA construct function, a filling of the configurable attributes of this XML file. Hence, the rules/algorithm functional block will be based on a set of instructions generated by such procedures.

Moreover, Java language is appropriate for implementing the XML managing as well as the GUI part of the architecture. There has been developed a great number of tools for handling XML files and schemas, which can be easily employed through standard

Java APIs. In addition, Java should be used for creating graphical environments, as it is platform independent and it contains a large assortment of fixed visual components.

As far as the negotiation API is concerned, a protocol that fulfills at least the following conditions is required:

- The client can define and request a service level
- The ISP can accept or reject a request submitted by the customer
- The ISP is also able to propose an alternative service level to the one requested
- The customer is capable of accepting or rejecting the ISP's offer and creating alternative requests

A protocol that meets the above requirements and is appropriate for the implementation of the proposed architecture is COPS-SLS [11]. This is an extension of the COPS protocol [21], which utilizes its distinctive features, in particular for the negotiation of service levels. Using COPS-SLS for the definition and the negotiation of SLAs, enables automatic and dynamic QoS provisioning. It is TCP-based and uses a client/server model. The PEP (Policy Enforcement Point) and the PDP (Policy Decision Point) entities of the protocol hold the place of the customer and the ISP respectively.

The PEP, i.e. the customer, requests a service level sending a REQ message to the PDP, i.e. the ISP, and the latter responds by sending a DEC message which denotes if the submitted request is accepted or not. Alternative proposals by either the customer or the ISP are sent including them in the same messages of the COPS-SLS protocol, REQ for the customer and DEC for the ISP.

However, the COPS-SLS protocol does not only implement the basic operations of a negotiation process but it is also defined by such attributes that suit the specific needs of the dynamic service subscription architecture. It encompasses two phases, the configuration one, where the PEP initiates the procedure and the PDP replies by providing information which is essential for its progress, and the main negotiation one, both of which directly correspond to the first two stages of the operational phase as they were previously described. Thus, in the former phase the ISP, or the PDP in the protocol's terms, can supply the information that we have defined as essential to be sent to the customer in the pre-negotiation stage of the operational phase, whereas the second phase, provides us with a manageable and effective mechanism for the realization of the negotiation stage of the operational phase.

The SLSs, which are converted to XML files and are exchanged between the customer and the ISP, are sent as a ClientSI [21] object of a COPS-SLS message, either Report (RPT) or Decision (DEC).

6 Conclusions

This paper describes work on customer dynamic service subscription architecture applied to a DiffServ network environment. This architecture is defined in the particular context of networks supporting mechanisms for SLS manipulation. We proposed a dynamic algorithm for the user to create an SLA and a negotiation mechanism needed both by ISPs and customers in order to reach to an agreement for the created SLA.

The proposed model supply with means for enabling automated service subscription, which for ISPs reduces the operational cost and contributes to an integrated and fully automated service provisioning process and for customers, reduces the time to request and access services.

In future work, we intend to continue development of the proposed architecture as well as define a signaling protocol needed for the activation of the subscribed services of our model.

References

1. Braden, R., Clark, D., S. Shenker, "Integrated Services in the Internet Architecture: an Overview", RFC 1633, (1994)
2. K. Nichols, V. Jacobson, L. Zhang, "A Two-bit Differentiated Services Architecture for the Internet", RFC 2638, (1999)
3. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, "An Architecture for Differentiated Services", RFC 2475, (1998)
4. A. Westerinen, J. Schnizlein, J. Strassner, M. Scherling, B. Quinn, S. Herzog, A. Huynh, M. Carlson, J. Perry, S. Waldbusser, "Terminology for Policy-Based Management", RFC 3198, (2001)
5. Ron A.M Sprenkels, Aiko Pras, Bert-Jan van Beijnum, Leo de Goede, "A Austomer Service Management Architecture for the Internet" *Proceedings 11th IFIP/IEEE Distributed Systems Operations and Management Conference (DSOM)*, (2000)
6. A. Prieto, M. Brunner, "SLS to DiffServ configuration mapping", *Proceedings 10th IFIP/IEEE Distributed Systems Operations and Management Conference (DSOM)*, (2001)
7. D. Goderis, Y. T'joens, C. Jacquenet, G. Memenios, G. Pavlou, R. Egan, D. Griffin, P. Georgatsos, L. Georgiadis, P. Van Heuven, "Service Level Specification Semantics, Parameters and Negotiation Requirements", Internet Draft, Internet Engineering Task Force, Work in progress, (2001)
8. Traffic Engineering for Quality of Service in the Internet, at Large Scale — `<http://www.ist-tequila.org>`
9. Creation and Deployment of End-UserServices in Premium IP Networkshomepage — `<http://www.cadenus.org>`
10. P. Trimintzios, I. Andrikopoulos, G. Pavlou, P. Flegkas, D. Griffin, P. Georgatsos, D. Goderis, Y. T'Joens, L. Georgiadis, C. Jacquenet, R. Egan, "A Management and Control Architecture for Providing IP Differentiated Services in MPLS-based Networks", *IEEE Communications Magazine*, 39(5), (2001)
11. T.M.T. Nguyen, N. Boukhatem, Y. Ghami Doudane, G. Pujolle, "COPS SLS: a Service Level Negotiation Protocol for the Internet", *IEEE Communications Magazines*, May (2002)
12. J. Koistinen, A. Seetharaman, "Worth-based Multi-category Quality-of-service Negotiation in Distributed Object Infrastructures", Hewlett-Packard Research Report HPL-98-51, (1998)
13. A. Hafid, G. v. Bochmann, R. Dssouli, "A Quality of Service Negotiation Approach with Future Reservations (NAFUR): a Detailed Study," *Computer Networks and ISDN Systems*, (1996)
14. G. Dermler, W. Fiederer, I. Barth, K. Rothermel, "A Negotiation and Resource Reservation Protocol (NRP) for Configurable Multimedia Applications", *Proceedings International Conference on Multimedia Computing and Systems (ICMCS)*, (1996)

15. A Hafid, G. V. Bochmann, R. Dassouli, "Quality of Service Negotiation with Present and Future Reservations", *Computer Networks and ISDN Systems*, (1996)
16. J. Rajahalme, T. Mota, F. Steegmans, P.F. Hansen, F. Fonseca, "Qos Negotiation in TINA", *Proceedings TINA Conference*, (1997)
17. G.D. Rodosek, L. Lewis, "Dynamic Service Provisioning: a User-centric Approach", *Proceedings 12th Annual IFIP/IEEE International Workshop on Distributed Systems: Operations & Management (DSOM)*, Nancy, France, October (2001) 37-48
18. X. Wang, H. Schulzrinne, "RNAP: a Resource Negotiation and Pricing Protocol", *Proceedings NOSSDAV Conference*, Basking Ridge, NJ, (1999)
19. J.-C. Chen, A. Mcauley, V. Sarangan, S. Baba, Yo. Ohba, "Dynamic Service Negotiation Protocol (DSNP)", Internet Draft, Internet Engineering Task Force, Work in progress, July (2001)
20. XML.org — `<http://www.w3.org>`
21. J. Boyle, R. Cohen, D. Durham, S. Herzog, R. Rajan, A. Sastry, "The COPS (Common Open Policy Service) Protocol", RFC 2748, (2000)