A Service Platform for Internet-Telecom Services using SIP

S. Bessler[1], A.V. Nisanyan[2], K. Peterbauer[1], R. Pailer[3], J. Stadler[4]

[1] *Kapsch AG, Pottendorferstrasse 19, A – 1121 Vienna, Austria*
{Bessler,Peterbau}@kapsch.net

[3] *SIEMENS AG Österreich,Gudrunstrasse 11, A – 1100 Vienna Austria*
agop-vart.nisanyan@siemens.at

[3] *Ericsson Austria AG, Pottendorferstrasse 25 – 27, A – 1121 Vienna, Austria*
Rudolf.Pailer@sea.ericsson.se

[4] *Vienna University of Technology, Institute of Communication Networks, Favoritenstrasse 9 / 388, A – 1040 Vienna, Austria Johannes.Stadler@tuwien.ac.at*

Keywords: Service Platforms, SIP, Multimedia Services, Convergence in Heterogeneous Networks, PARLAY

Abstract: This paper proposes the introduction of a Service Platform for the creation, execution and management of multimedia services in heterogeneous networks. Examining the business-roles, the actors in the Service Platform are identified. Furthermore several common building blocks for developing services for the Internet are described and a brief overview of some modern technologies for an object-oriented, component-based, distributed platform for multimedia services is given. The Session Initiation Protocol (SIP) has been identified as very useful to implement all functions according to the multimedia part of the Platform. The usage of the PARLAY API as an open interface between the Platform Services and SIP, the PSTN or the mobile network provides a lot of additional advantages. The interworking between SIP and PARLAY is shown in a call-routing example. Furthermore the call-setup for a multimedia (e.g. video) conference is explained. This should demonstrate the usefulness and the ability of this protocol for introducing a session concept. Finally an outlook of open research topics regarding this concept is given as well as a short overview of related work.

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35522-1_37

H. R. van As (ed.), *Telecommunication Network Intelligence*

1. INTRODUCTION

The FTW research project 'Service Platform and Interoperability' proposes the introduction of a Service Platform (SP) for the creation, operation and management of real-time multimedia services in heterogeneous networks.

The SP is a middleware, which provides a distributed framework for services. The managed services shall be accessible from different types of terminals, carried over different types of networks and using different protocols.

During analysis of the requirements for the SP, the need for a multimedia session concept has been identified. Sessions are responsible for keeping track of the various data streams associated with one multimedia communication (e.g. data, voice, video streams). This paper proposes the usage of the Session Initiation Protocol (SIP [1]).

The paper is organized in six chapters. After this introduction, the second chapter introduces the concept of a Service Platform and describes the requirements it must fulfill. As an important technology for the architecture of SIP as an object-oriented, component-based system CORBA is identified.

The third chapter shows our architecture of the Service Platform. All necessary components are discussed and explained.

In the fourth chapter we answer the question why we use an API between the Platform and the underlying call control. We discuss the advantages of such an approach, and give an account of why using PARLAY [2]. An example of a call-routing process should prove this theory. Also the setup of a multimedia conference using SIP over Parlay is shown.

In the fifth chapter we provide an overview of open research topics regarding the SP concept and we give an outlook on the possibilities of implementing the SIP session control as part of the SP.

The sixth chapter presents a brief summary of related work.

2. THE CONCEPT OF THE SERVICE PLATFORM

2.1 The Business Case

The business roles model is defined in accordance with the suggestions and discussions of TINA-C, TSAS (OMG) and Open Marketplace (OMG). We

identify the business roles end user, subscriber, service provider, 3rd party service provider and network provider.

The service provider runs and maintains the service platform. He maintains the data store with all the users and service logic as well as the contracts and configuration of the subscribers. Furthermore he creates new services, which are deployed on the Platform and offered to the subscribers. Also he provides connectivity and service agreement to network providers.

The SP has to provide mechanisms for registration of subscriber resources such as content and web-servers, as well as enterprise resources needed for the implementation of a certain service.

A subscriber is a user or a company that pays for a service offered by the service provider. Subscribers do neither host services nor have to maintain the service platform.

End users are registered on the behalf of the subscriber. The representatives of the subscriber (operators) have a subscription side management application to configure and manage the service of their users.

Depending on the service, a registered user can configure an application by himself, such as call forwarding, or the task can be done via subscriber side management, e.g. call center scenarios.

A 3rd party service provider provides content, which is offered by the Service Platform. A service provider could be 3rd party service provider too.

2.2 Basic Services

Several common building blocks for developing services for the Internet can be identified, including

- User and subscription management.
- Security management, enforcement of access control, management and distribution of certificates.
- Generic session management.
- Usage tracking, accounting, billing.

These common capabilities can be modeled as *basic* or *horizontal services,* which are always present at any host of the SP (billing and certificate management are outside the scope of this project).

Beside these functional requirements, several other basic services are valuable:

- *Registry and Lookup Service* used to register and locate services running at different hosts of the platform.
- *Transaction Service*, supporting two-phase-commit transactions.
- *Scheduling Service*, available to all services on the platform rather than just processes on a single host. Tasks can be scheduled and executed on the local or a remote host.
- *Event Service* as a publish/subscribe system decoupling the service, which is the source of an event from other services acting as "listeners".
- *Log Service* as a centralized facility, supporting internationalized log messages e.g. with timestamps or complete stack traces.

Logically these services can be seen as single instances in the whole platform. Transparent replication might be implemented solely for the purpose of increased availability.

2.3 Service Accessibility

Access can be provided to users for a variety of terminal types, e.g.

- PC with a standard Web browser for rendering HTML/XML/XSL documents, displaying multimedia data with plug-ins, and executing downloadable code such as Java applets.
- Multimedia PC running dedicated applications e.g. a fully featured SIP client.
- WAP/WTA-enabled terminals (Wireless Application Protocol/Wireless Telephony Application), such as mobile phones and personal digital assistants. These terminals might also be aware of their current location, e.g. by using the GPS satellite system.
- Black telephones for simple voice communication or interaction using DTMF signaling or voice recognition.
- Fax machines.

Some services will be dedicated to certain terminal types, but many services will be accessed from more than one terminal type with a varying degree of functionality. These services can share basic infrastructure provided by the SP, shielding the services from the details of the underlying network protocols.

2.4 Backend Systems

The SP encapsulates the services and provides controlled access to various backend systems, including relational and object databases, directory services, application software packages and other legacy systems (billing, customer care systems etc.).

We will focus on the access to those legacy systems, which are governed by open standards, e.g. relational databases.

2.5 Service Life Cycle

The life cycle of a service involves numerous steps, including design, implementation, testing, deployment, operation, management, maintenance, and removal.

The service creation process is beyond the scope of our project. Our major goal is to define an appropriate programming model for services, which is prepared for an integrated service creation environment.

2.6 MIDDLEWARE Technologies

There are many modern technologies for an object-oriented, component-based, distributed platform for multimedia services, for instance CORBA or Enterprise Java Beans. These technologies fulfill many of the requirements of the SP with respect to the identified basic services and may serve as 'glue' for the SP. For our architecture we identified CORBA as the best suiting possibility for our requirements.

CORBA is the most widely used and deployed distributed object technology. It is an open industry standard, providing an object-oriented middleware spanning over heterogeneous platforms, (packet-) network protocols and programming languages. A large number of today's modern application servers are based on CORBA as backbone of an enterprise's information service, used in the middle tier between legacy systems at the backend and typically conventional Web servers at the front-end.

The core of CORBA is the Internet Inter-ORB Protocol (IIOP), which enables different applications to communicate via application-specific interfaces.

CORBA specifies a set of horizontal and vertical services. Among the *horizontal services* are

- a *Naming* and *Trading Service* for service registry and discovery,

- an *Event Service* for pulling and pushing typed and untyped events,
- and a *Transaction Service* for coordinating distributed transactions across multiple databases and legacy systems.

Recently the OMG specified a component-based object model, defining interfaces between a server-side component and the server application in which it is contained, e.g. with respect to transactional behaviour and persistence.

Most notably among the *vertical services*, the *TSAS* (Telecommunication Services Access and Subscription) specification defines standardized interfaces enabling end-users to access and configure telecommunication services, as they prefer and to allow value-added service providers to offer their services.

The Parlay API from the Parlay consortium specifies a high level CORBA-API that allows service developers to access resources in PSTN, mobile and IP-based networks. Consideration of a Parlay interface to the service platform unifies the model and defines the responsibility of service provider and network provider. Since Parlay operates only at the enterprise level, the mapping to user granularity has to be made, possibly as proposed by TSAS.

3. SERVICE PLATFORM APPROACH

Fig. 1 illustrates our approach for the realization of a service platform, which integrates different (PSTN, PLMN, IP) networks and services.

Our Service Platform Solution is based on following principles:

- The customer (end user side) access works via HTTP and WAP. Thus both, the wirelined (LAN, WAN) and wireless mobile (GSM, GPRS and UMTS) terminals are supported.
- The requirements on the end-user-terminals are minimum. They need only a usual webbrowser (HTTP)/WAP and a SIP User Agent.
- The service provider who owns the service platform and who has contracts with 3rd party service provider is the SP operator. He holds all the subscriber data, which is the most valuable asset in the Internet world. This service provider, together with all other contracted 3rd party providers, build a virtual service domain.

- A central idea is one point authentication in this whole virtual service domain. Ways to ensure this must be still investigated. A possible (temporary) solution can be the use of certificates.
- The application / service logic is distributed among the 3rd party service providers. Every 3rd party service provider can provide services independent of other service providers if needed. He needs only a contract with the SP operator if he wants to use services of Service Platform as authentication (one-point-shopping), call control (connection provisioning), media streaming or location services.
- For the implementation of application / service logic servlets are intended.
- Every contacted 3rd Party Service Provider can access the service platform (usually via servlets) over CORBA.
- The Service Platform acts as controlling and switching center for cross domain (PSTN, PLMN, IP) network connections and intelligent communications services as call forwarding.
- The service platform uses PARLAY as the controlling interface to the underlying networks.
- The IP-signaling will be realized over SIP.
- In accordance to the decision to use PARLAY we will implement on the SIP-Proxy a PARLAY interface and check the effects.
- Only the control data passes the service platform. The user data is transferred transparently between end-points that means without any Service Platform interaction. An exception to this point is the IP-PSTN-Gateway, which will be part of the service platform because of the lack of such gateways in the used network.
- The SIP will be used not only for voice-communication, but for all types of multimedia communication as e.g. video streaming.
- It is intended to use JAIN as the PARLAY (Client) wrapper if available in time for the implementation.
- For the realization it is intended to use JAVA, JAVA Media Framework, CORBA, SIP, CPL [3] and the servlet API.

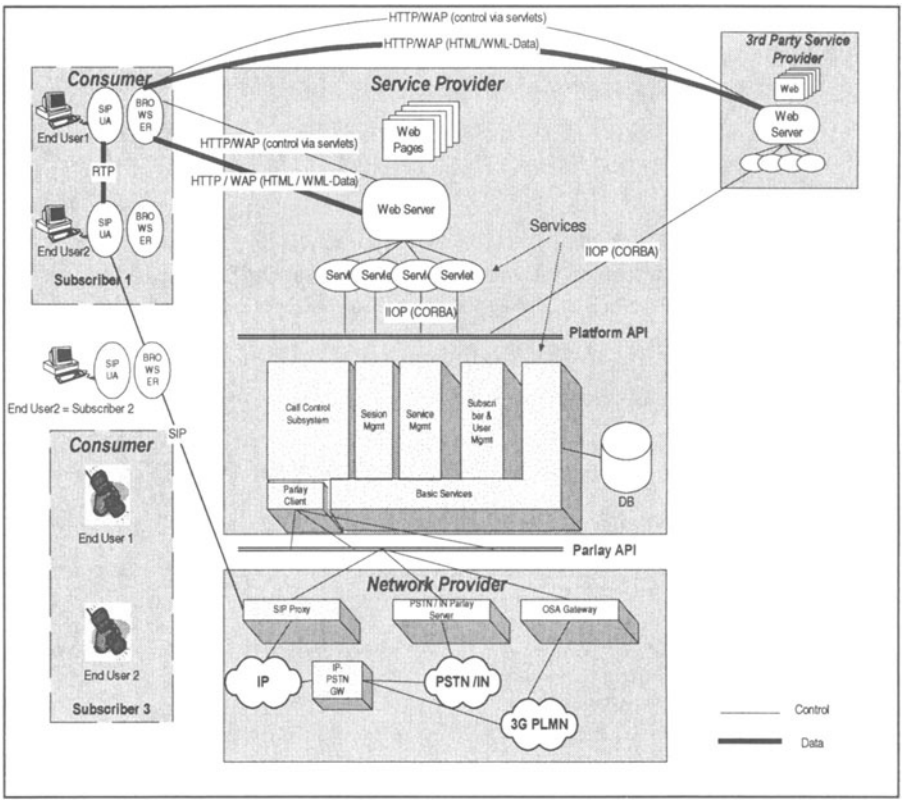

Fig. 1. Service Platform Approach

4. SIP AS BUILDING BLOCK OF MULTIMEDIA SERVICES

The IETF SIP protocol has been identified to be one of the most promising ways for a large-scale deployment of multimedia communications over today's data networks.

There are different entities defined in SIP. At first there are the SIP User Agents (UA), which could be separated in UA Servers and UA clients. UA clients initiate a call, while UA servers only react to calls. Both are statefull devices, because in the end terminal the state of a call has to be known. The second important entity in SIP is the SIP network server, which could act as proxy or as redirect server. Because of scalability the network server in the core net should be stateless, while the edge servers have to be statefull to provide all required functionality. Location server and register server complete this enumeration.

The session concept of SIP offers a generic way to establish, control and tear down multimedia sessions between communication end-points. Publications about SIP mostly concentrate on the usability of SIP to implement Voice over IP (VoIP) services. It is one of our research goals to show that SIP sessions can be used to establish multimedia streaming sessions as well as interactive multimedia communications.

In this chapter we show the way, how the interworking between SIP and Parlay could be performed, as well as some scenarios for multimedia functionality.

4.1 Third Party Call Control with SIP

The main task of the SP can be seen as a platform for the creation, management and execution of 'third party services'. In this paper a third party service is understood to be some kind of service logic that is located and runs outside of the trusted domain of the network resources that this service controls.

Third party control interfaces shall have the following important properties:

- · The third party control interface offers an abstract view of the underlying network resource so that the features and functionality of the controlled resource can be used without detailed knowledge about the specific realization or implementation.
- · The third party control interface offers security management, so that service logic running in an untrusted domain outside of the network resource can be authenticated and authorized to perform service specific tasks.

The PARLAY API 2.0 specification has been identified by our group as a good solution for above requirements. PARLAY is an industry initiative to create a CORBA based interface from an enterprise application (client) to different networks (PSTN, mobile, IP) and their resources e.g. voice- or mailboxes. Especially for the control of call processing in advanced mobile switches (e.g. UMTS), the PARLAY API seems to develop into an industry wide standard.

The Service Platform will control call processing in the SIP Domain Controller via a PARLAY interface in order to realize third party call control services. A call control service running on the Service Platform will therefore act as a PARALAY client, while the SIP Domain Controller has to implement the PARLAY server interfaces. To show the interworking between SIP and PARLAY the Use Case 'Routing of a SIP call' is described in detail.

4.2 Use Case 'Routing of a SIP call'

The router logic (e.g. a CPL script) can be configured by the users. The Service Platform only provides a simple interface that offers a method to notify the router logic of a new incoming call. A second method is used to inform the router logic of the result of a requested routing task. The platform service classes ServiceRouteNewCall_Manager and ServieRouteNewCall_Call have to implement the PARALY IpAppCallControlManager interface and the IpAppCall interface respectively. These central classes are responsible for the specialization of the general PARLAY functionality towards a well-defined feature, namely the routing of new incoming calls towards a new destination.

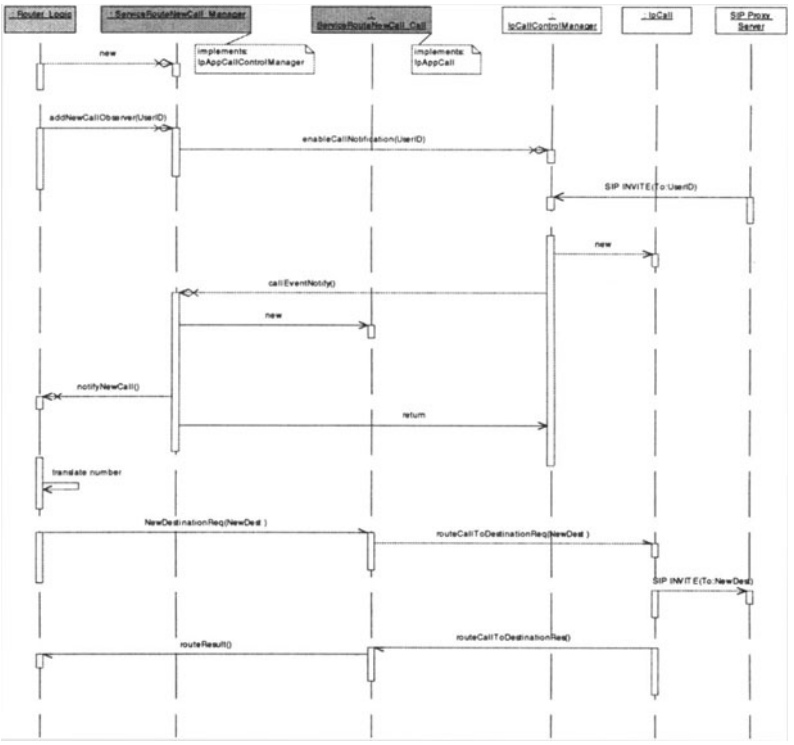

Fig. 2. Routing of a SIP call

The sequence of events is as follows:

1) The router logic registers to receive notification in case of a new incoming call.

2) The service manager adds the router logic to the listed observers and requests the notification from the PARLAY server. The PARLAY server interfaces are implemented by a SIP proxy server.
3) A new INVITE request is received and the PARLAY server notifies the service manager.
4) The service manager creates a service call object and notifies the router logic.
5) The router logic computes a new destination for the call.
6) The router logic has got together with the notification a reference to the service call interface, where it requests now the new destination for the call.
7) The service call object forwards this request to the SIP server, which proxies the INVITE to the new destination.
8) The PARLAY server responds with the result of the routing operation.
9) This result is sent to the routing logic.

4.3 Further Use Cases

The service platform covers following requirements:

- The user is able to receive multimedia streams, such as videoclips or audioclips, from a content server, located at the 3rd party provider premises and registered at the service platform. An additional goal is to easily adapt the existing content servers as well as the user terminals to support streaming.
- The user is able to build up a VoIP or videoconferencing connection to a 3rd party provider or another user. In case the 3rd party provider implements a SIP UAS too, a pure IP-based connection is set up. In case the 3rd party provider only offers a connection over the PSTN, a VoIP connection should be set up only with a media gateway. The connection to the PSTN is done using a media gateway controller, which transforms the SIP INVITE request to PSTN signaling. The data is transferred over the media gateway.

The cases above could and should be implemented as a 3rd party call to give the Platform the control over the call.

- The user is able to build up a VoIP connection to other users currently logged in.

- The user is able to allow the subscriber to send web pages directly to his desktop.
- A user that changes location or availability is able to define a redirect service to the new location, web page or messaging box.
- The subscriber is able to manage his user base himself.

4.4 Media Streaming

Representative for most of the use cases described above we want to show the sequence of a media streaming session (see Fig. 3). Maybe the user wants to stream down a video clip or wants to set up a VoIP connection to a 3rd party service provider. The iPApp in the following diagram represents the PARLAY client. The user clicks to a link on a website. This causes a HTTP-request. Via a servlet the request comes to the PARLAY Client, which generates a routeToOrigination-Request. The call setup works in this case like a 3rd party call.

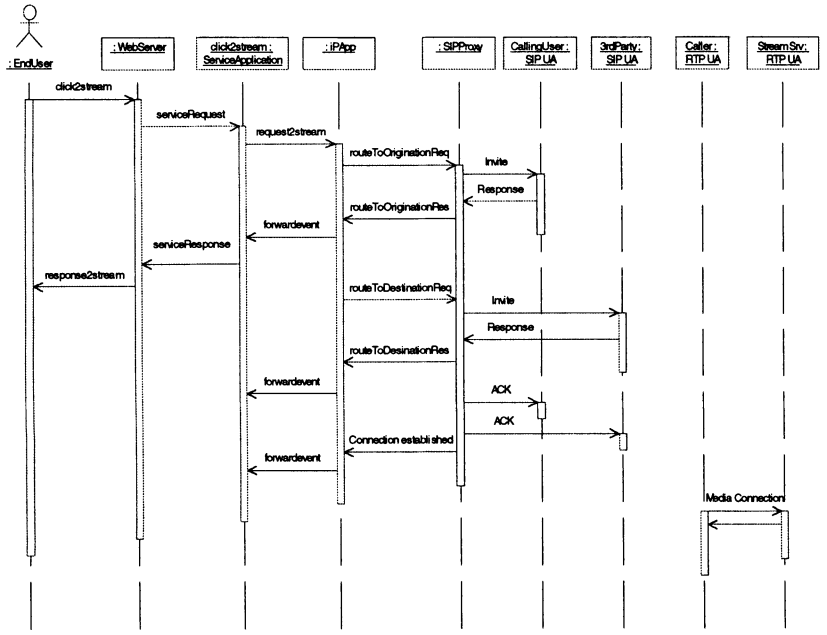

Fig. 3. Media Streaming

An INVITE request (without IP and Port of the second UA) is sent to the originating UA. After the response (with IP and Port of the originating UA as well as supported media types) the answer to the HTTP request is per-

formed. If the originating UA would not be reachable, an error page could be sent back here. After this a routeToDestination-Request is sent to the SIP proxy, which generates an SIP INVITE with IP and Port of the originator. In the response the destination UA sends back its IP and Port, which reach the originator in the ACK message. After all ACK's, the SP is notified about the established connection and the media channel is set up. The transport of the media should be done using RTP [4]. We decided to use JMF, which provides good possibilities for synchronizing different media streams. JMF 2.0 includes already classes for the RTP transport.

5. OPEN TOPICS

In the following we list open research topics regarding the SP concept.

- Evaluation of the described technologies as middleware for the SP.
- Service creation and service interactions.
- QoS aspects.
- SIP enabled content server (e.g. video streaming).
- Definition and implementation of a generic session management API.
- Design and implementation of SIP enabled multimedia client software.

6. RELATED WORK

The idea of a service platform is not new at all. The most popular service architecture for the PSTN is the Intelligent Network (IN). The IN is provided with a Service Creation Environment (SCE) in which a service is created from a set of standardized service-independent building blocks (SIBs).

Gbaguidi [5] proposes a service platform for so called hybrid services, which is based on a Service Creation Environment (SCE) and a Service Infrastructure (SI). In the SCE a service logic is put together by the service creator using a set of service components as building blocks. The service logic is then sent to a service factory, which creates a new instance of service. The service instance is organized into service subsystems that are further uploaded into the system infrastructure, more specifically in the controllers of the system elements.

The Service Infrastructure is composed of a collection of controllers and the underlying system elements of the Intelligent Network, such as terminals, gateways, network nodes, etc. An event interface enables the data flow between the system elements and the service logic.

Manione and Renditore [6] describe a service platform based on a TINA architecture that has been simplified to cope with the requirements of internet-telecom service development: simplified session modeling, identity of the service provider business role with the retailer, enhanced subscriber management, logical separation between service logic (service environment) and generic platform services.

The WebCentric web call center application provides added value by offering VoIP call together with pushed WebPages to the caller, or other media combinations such as chat or email. The paper describes clearly the technology used for the different terminal types, without addressing generic interfaces to IN, or to go into the aspects of service creation and deployment [7].

REFERENCES

1 M. Handley, H. Schulzrinne, E. Schooler, J. Rosenberg: SIP: Session Initiation Protocol. Internet Engineering Task Force RFC 2543, March 1999
2 Parlay API Specification 1.2, User Guide. http://www.parlay.org, Sept. 1999
3 J. Lennox, H. Schulzrinne: CPL: A Language for User Control of Internet Telephony Services, Internet Draft, Internet Engineering Task Force, March 2000.
4 H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson: RTP: a transport protocol for real-time applications. Internet Engineering Task Force RFC 1889, Jan. 1996
5 C. Gbaguidi, et al, Integration of Internet and telecommunications: An Architecture for Hybrid Services. IEEE JSAC vol.17, no.9, Sept. 1999
6 R. Manione, P. Renditore: A TINA Light Service Architecture for the Internet-Telecom Scenario, TINA 99 Conference, Hawaii, USA, April 1999
7 R. Manione, M. Festa, P. Renditore, D. Sereno, M. Spaziani:. Service Issues in Web Call Centers, Telecom 99 Forum, Geneva, 1999