Chapter 2

Internet Telephony: The Evolution to a Service-Oriented Architecture

The field of telecommunications has evolved to resemble the distributed computing domain, where general-purpose computers communicate over a common network. This evolution has culminated in the Third-Generation Internet Multimedia Subsystem (3G IMS) Architecture. In the computing domain, the Web Services Architecture or, in general, the Service-Oriented Architecture (SOA) is the modern trend in distributed computing today. Both the IMS and the Web Services Architecture provide services built on common, standardized, and well-known protocols. In this chapter, we present our views on the continuing evolution of these two architectures as the boundary between telecommunications and computing services continues to blur. We provide a high-level architectural overview of both the telecommunications and Internet networks to provide a context for the requirements we derive for a telecommunications SOA. Detailed architectures of both the networks will be provided in Chapter 3.

2.1 Introduction

Until recently, the line between the telecommunications network and the Internet was well demarcated. The former was a special circuit-switched network, tuned to transporting one media: voice. Over the years, it had also evolved to provide voice-related services to its users — colloquially known as subscribers — such as call forwarding, call waiting, and other services. In the telecommunications network, intelligence was concentrated in the core of the network, with the edges (phones) being very simplistic. The Internet, on the other hand, resided at the opposite spectrum from the telecommunications network. It was designed as a packet-switched network that would transport any type of media — voice, video, gaming, text — in a packet. The core of the Internet was relatively simple and stateless — it only performed the routing of packets; the intelligence resided at the edges of the network in the form of powerful general-purpose computers [KEM04].

More recently, the established lines between the Internet and the telecommunications network have started to blur. Today's 3G cellular phone is capable of providing many Internet services: e-mail, presence, Web browsing, and instant messaging, to name a few. On the other hand, transporting voice, which was once thought to be the domain of the traditional telephone network, is now done by the Internet. Two recent advances in technology have aided in this shift. The first advance is related to the form factor of what constitutes a computing device. Moore's law and other advances have continued to shrink hardware components to the point that a fairly sophisticated computer can be embedded in a handheld telephone. Second, the advances in the field of networking have made the Internet faster and more pervasive than ever. These two advances have caused each of the networks to steadily encroach on the principles held dear by the other.

In the telecommunications network, we note that the intelligence is being pushed to the edges. Witness the rise of the cellular network with much more sophisticated endpoints than the traditional telephone network. For its part, the Internet has started to adopt certain services in the core of the network, especially those pertaining to federated computing and quality of service. Two good examples of this are grid computing and Internet telephony. According to one definition [FOS], grid computing is comprised of systems that "use open, general purpose protocols to federate distributed resources and to deliver better-than-best-effort quality of service." To do this, some intelligence in the core network is necessary. Internet telephony also depends on some intelligence in the network to maintain the quality of service of a voice or video session. A "flow control" label maintained in a packet aids the Internet routers in associating incoming packets with an existing stream and providing the guaranteed quality of service to that stream. Admittedly, this is “soft state” [KEM04] in the network when compared to the state of a call maintained by a traditional telephony switch, but it is state nonetheless.

The telecommunications and Internet architectures continue to evolve toward a single network, on both the physical and transport layers, as well as the service layer. The first step in the evolution was the merging of the physical and transport layers. (Voice is packetized and transported on the Internet. In an ironic twist, the telephone network is able to provide a Digital Subscriber Loop (DSL) broadband connection that can be used to transport voice packets, thus bypassing the telephone network for transporting voice. But the larger point is that at the physical and transport layers, the Internet may have usurped the circuit-switched telecommunications network.)

The second and probably more important step is the evolution of the services layer. Both the telecommunications network and the Internet have their own service architecture. As these networks merge, one model of services — the computing style of services or the telephony style — will prevail. Which service architecture is better suited for the future? Although we do not have an answer to this question, we do provide an analysis of the service architectures of telecommunications and the Internet and draw parallels between them to extrapolate some attributes that may be present in an overall architecture in the future.

2.2 Service Architecture for Traditional Telephone Network

The service architecture for the traditional telephone network (wireline and wireless) is defined around the Intelligent Network (IN) [FAY96]. IN is a conceptual architecture that separates the call control from the service execution. Figure 2.1 shows a simplified IN architecture. (In reality, there are more IN entities than depicted in Figure 2.1, but for our discussion, the ones depicted in the figure suffice.) Subscribers use telephonic devices that are connected to a telephone switch called the Service Switching Point (SSP). An SSP, in turn, is connected to yet other IN entities via a packet network called Signaling System 7 (SS7). The most important IN entity is the Service Control Point (SCP), which is added to the call by the SSP. An SCP is a general-purpose computer that hosts and executes the service logic for a subscriber. The service logic can invite other IN devices into the call; for instance, if a service requires the caller to interact with a voice response system, an Intelligent Peripheral is dynamically added to the call.

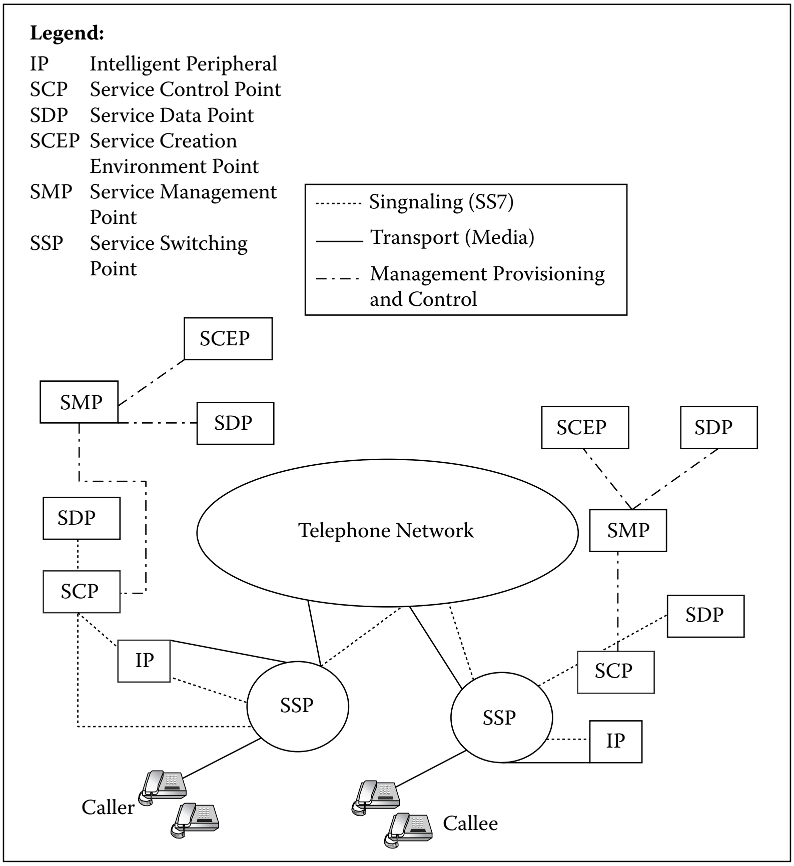

Figure 2.1 Traditional telephone network architecture.

The service logic can also access data pertinent to each subscriber stored in a specialized database called the Service Data Point (SDP).

When a call is originated on the telephone network, the caller's SSP arranges for the SCP to be brought into the call. The SCP then executes the service logic, depending on the services subscribed to by the caller. This process is repeated on the callee's side as well; the callee's SSP on receiving a call setup request arranges for an SCP to be brought into the call, and so on.

Services themselves are created in a general-purpose computer called the Service Creation Environment Point (SCEP). A service in IN is created by chaining reusable components called Service Independent Building Blocks (SIBs); many well-known SIBs exist, such as number analysis or adding new devices into a call. Service logic programmers employ a SIB palette to drag and drop individual SIBs to compose a service. Once a service is thus created, it is deployed at the SCP using the Service Management Point (SMP), which is yet another general-purpose computer through which service management and provisioning are performed.

The cellular telephone network also uses IN to create and deploy services. The process of executing a service in the cellular network is similar to its wireline equivalent; the crucial difference is that in a cellular network, there are more entities involved in providing a service. A set of databases — home location register and visitor location register — track the subscriber and store the services associated with the subscriber. Authentication servers authenticate and authorize a cellular endpoint, and other infrastructure (base stations, mobile switching centers) provide radio access networks and the capability to connect to other cellular subscribers or to the wireline telephone network.

More information on IN is provided in [FAY96].

2.3 Internet Services Architecture

At the onset, the Internet supported predominantly client–server type of service architectures, wherein a client made a request of a server and the server provided the service. The service could be as simple as a one-time generated piece of information (for instance, ldap, ntp, etc.), or it could be more complex, like a ftp or a telnet session. The next step to the client–server architectures was the move toward distributed objects.

Distributed object frameworks — like Distributed Component Object Model (DCOM), Common Object Request Broker Architecture (CORBA), and Remote Method Invocation (RMI) — hid the complexities of client–server interactions in well-defined application programming interfaces (APIs). Programmers no longer needed to worry about Internet addresses and port numbers to access servers, or even how to structure the protocol data unit for a service. Precompilers read an interface definition file and generated the corresponding server skeletons and client stubs. The application programmer had to only fill in the application-specific logic in the generated code. Over the years, distributed object frameworks have evolved to provide message reliability and transactional guarantees.

The next step beyond distributed objects is service-oriented computing (SOC) and service-oriented architectures (SOAs), as exemplified by Web services. Papazoglou and Georgakapoulous [PAP03a] describe SOC best as a “computing paradigm that utilizes services as fundamental elements for developing applications.” They go on to provide the following description of SOA: “Basic services, their description, and basic operations (publication, discovery, selection, and binding) that produce or utilize such descriptions constitute the SOA foundation.”

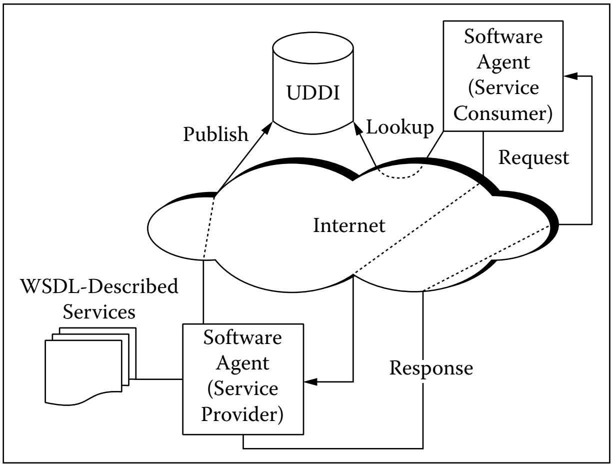

Figure 2.2 Web Services Architecture.

SOAs are characterized by loose coupling between the interacting software agents. The software agents exchange well-defined messages to execute services. The format and semantics of these messages are collectively defined by Web services. Web services transport eXtensible Markup Language (XML) documents between software agents using standard Internet protocols. Software agents that send and receive these documents implement the semantics of a particular service.

Web services are described using the Web Services Definition Language (WSDL), which is an XML format for describing network services as a set of endpoints operating on messages [CHR01]. WSDL describes all information pertinent to a Web service: its name, transport binding (using Simple Object Access Protocol (SOAP), Hypertext Transfer Protocol (HTTP), or Multipurpose Internet Mail Extension (MIME)), and a set of operations supported by the service. Once described as such, services are stored in a Universal Description, Discovery, and Integration (UDDI) registry. Clients can query the registry to get a listing of all matching services. Figure 2.2 contains an overview of this process. A software agent (service producer) describes the services it supports through WSDL and registers them with a UDDI registry. At some later point, another software agent (service consumer) looks up the service in the UDDI and contacts the service producer directly to execute the service.

2.4 Requirements of a Telecommunications SOA

Unarguably, the intelligence in the telecommunications networks continues to move out to the edges. Early analog phone systems were completely centralized. Call processing and service execution were often intertwined and occurred on centralized platforms owned by the telephone company, and also often used networks owned by the same telephone company. Digital switching systems coincided with the separation of call signaling from service execution (the IN concept). The intelligence was more distributed here and, in fact, it was not strictly necessary that the service execution platform be owned by the same provider that owned the switch (although this was, by far, the most common deployment scenario).

The advent of the mobile cellular network furthered the move of intelligence to the edges. The first-generation (1G) cellular network was an analog circuit-switched system. Mobile handsets were bulky, voice quality was poor, and security was nonexistent. 2G networks improved on the disadvantages and provided additional data services, like Short Message Service (SMS). 2.5G is an intermediate step toward 3G, utilizing Internet protocols and packet switching in portions of the cellular network. 4G networks are a step beyond 3G, providing data transmission speed equivalent to a local area network and more personalized services for its subscribers. As can be observed, in successive generations starting from 2.5G, the Internet Protocol plays an increasingly bigger role, and the mantra of the Internet Protocol is that intelligence resides at the edges [KEM04]. Thus, if the underlying telecommunication technology is moving toward the Internet Protocol, there appears to be some promise in the argument that the telecommunications service architecture will also evolve toward a computing-based one. Initial stages of this progression are characterized by the crossover services model, whereby both networks are cooperating to execute the services [GUR04a, GUR05a]. The final stages will witness the wholesale move of telecommunication services to the Internet, thus establishing a telecommunications SOA.

The as yet unanswered question is: What shape will this new architecture take? Let us first examine the similarities and differences between the service models of the two networks, and then attempt to tease the requirements of a telecommunications SOA.

The biggest similarity between the service models is that they both use message passing as an underlying mechanism for service execution. In the telecommunications network, an SSP sends a well-formatted message — a request — to the SCP, which in turn sends a well-formatted response back. This interaction matches the Web services description of exchanging structured information in a distributed environment. Thus, at a very high level, both networks contain primitives to name resources and route messages between the resources.

Closely related to message passing is the desire to use standardized protocols to transport these messages. Within their respective domains, both Web services and telecommunications services use standardized protocols for message passing. Web services use XML, SOAP, UDDI, and WSDL, whereas telecommunication services use protocols such as SS7 [RUS02] and Intelligent Network Application Part (INAP) [FAY96].

A second similarity lies in the manner services are defined within the computing discipline. In computing, a service can be described as an autonomous, platform-independent computational element. This is also true of telecommunication services. The autonomous, platform-independent computing element in telecommunications services is called a SIB (introduced in Section 2.2). Application programmers in telecommunications typically use a palette of well-known SIBs; creating a service is as simple as dragging a SIB and connecting it to other SIBs through edges that represent the actions and events (of course, the service has to be compiled and tested before being widely deployed). A SIB, much like a computing service interface, absolves the programmer from knowing the internal protocol details [WIL97].

A third similarity is the need for orchestration. An example will illustrate: Consider a stock price Web service that returns the stock price of a certain company. In and of itself, it is a useful service, but when it is coupled with another service — call this the stock-purchasing service — that purchases the stock when it falls to a certain price range, the utility of the stock price service is increased. The need for orchestration arises because some intelligence is required to trigger the stock-purchasing service when the price falls below a certain watermark. The business community, lead by IBM, Microsoft, and others, is coalescing around an orchestration language called Business Process Execution Language (BPEL). Other domain-specific orchestration languages and coordination strategies will undoubtedly emerge as the Web Services Architecture matures.

In telecommunication services, orchestration is already present in the form of a call controller. A call controller can be thought of as a service that is instantiated when a caller initiates a call, or it can be instantiated based on other service logic (i.e., at 3:00 P.M., start a conference call between the following people…). The call controller is the intelligent centralized entity that controls all aspects of the call: it knows the number of parties in a call, the duration of the call, billing information of the call, and so on. As an advanced example of coordination in telecommunication services, imagine that there are two discrete services: a presence service that determines where a subscriber is present (home, in transit, work) and an instant message service that can communicate with the subscriber. The call controller can use the presence service to locate the called party, and it could use the instant message service to first send an IM to the called party and explicitly ask its disposition before allowing the called party's phone to ring.

Having examined the similarities, we now enunciate the differences. The first and most apparent difference between the two service architectures lies in the realm of discovery, selection, and binding of services. The Web Service Architecture includes WSDL and UDDI to aid in this effort; an equivalent mechanism in telecommunications services is missing. One possible reason for this is that in the telecommunication domain, the service vendor and the service provider typically have preexisting business relationships, thus alleviating the need for service discovery and binding. This is not the case for Web services. A Web service written by an arbitrary service vendor should, at least in spirit, work across the platforms of multiple service providers. This process automatically results in the best-of-breed services percolating to the top of the list. There is no equivalent for discovery, selection, and binding in the telecommunications network. Preexisting business relationships between the service provider and the network vendor have alleviated the need for this (traditionally, the service providers have been the equipment vendors, who create and sell services directly to the network providers, who in turn bill their subscribers for these services). *It thus follows that some manner of a discovery, selection, and binding mechanism based on open and standardized protocols should be a requirement of a telecommunications SOA.*

A second difference lies in the real-time nature of telecommunication services versus Web services. The latter are mostly data driven, i.e., move information from point A to point B. These services can withstand delay and are generally more forgiving in the face of a best-effort network like the Internet. Web services are also characterized by a single protocol for communications (SOAP) and data representation (XML). Telecommunication services, on the other hand, are both data driven and media driven (e.g., voice, video, gaming, and presence). They do not suffer delay in a graceful manner. This is true at both the media layer (delay in terms of lost packets introduces jitter and an inferior audio/visual experience) and the services layer (delay in accessing a database may prohibit the productive execution of a service). They also tend to be driven by a multiplicity of protocols: Session Initiation Protocol (SIP), SDP, Real-Time Transport Protocol (RTP), and Real Time Control Protocol (RTCP), to name a few. *Thus, a second requirement of the telecommunications SOA would be support for real-time delivery of information across a multiplicity of protocols.*

Another difference is in the security infrastructure of the networks. In the telecommunications network, security is addressed at two different layers. At the physical layer, the telephone lines and equipment (switches, base stations) and related databases are located in secure facilities. It is hard, although not impossible, to tap into a telecommunication line to eavesdrop on a call. If this itself was not a deterrent, the legal system has evolved to protect the conversation occurring across a phone line and levy fines for malicious access and disruption to the telephone service. The Internet, on the other hand, is a wide-open system by design. Tapping into it is no more work than enabling an Internet Protocol (IP) transceiver to go into promiscuous mode and sniff all the packets crossing the transport. The same level of legal scrutiny afforded to the telephone system has not yet made its way to the Internet. Another rather subtle reason for more security in the telecommunications system has been how its protocols have evolved. Telephone networks have traditionally been environments where the inner workings of the protocols and services, although not entirely secret, were not subject to as much public access and scrutiny as Internet protocols have been. Consequently, the number of people attempting to understand and implement these protocols, and indeed to mount malicious attacks, has been much less than the Internet equivalent. *Thus, we derive a third requirement of the telecommunications SOA: to provide security in a manner conducive to the Internet.*

Closely related to security is authentication. When a telephone subscriber makes a call, the network authoritatively knows the identity of the subscriber (because in the telecommunications network, the service provider owns the network and the subscriber data). Consequently, subscribers of the telecommunications network have learned to trust the content that flows over the telephone network. Services provided by the telephone network — call forwarding, caller ID, etc. — need not be further authorized or authenticated. The telephone service provider is the final arbiter on the authenticity of the service on that particular network. This is not the case on the Internet, where the provider of the transport (the network) is distinct from the provider of the content. The authentication problem on the Internet is made worse by the ease of mining new identities and the lack of a central trusted arbiter that can be used to rendezvous two previously unknown users. *Some means of authoritatively authenticating the communicating user agents is a final requirement for a telecommunications SOA.*

2.5 Conclusion

Our vision of a telecommunications SOA is that it will be an open, federated, secure architecture that allows the subscribers to choose the best-of-breed services from competing service providers and have these services work cooperatively.

The telecommunications SOA has to be federated. Services on it will require aggregation across autonomous boundaries. Take, for instance, using the presence service as a prerequisite to establishing a voice or video session between two participants. The presence state of the participants may be distributed across multiple devices (cellular phones, laptop computers, personal digital assistants, desktop computers) and service providers (Yahoo!, AOL, MSN, enterprise networks). Intelligence will be required to aggregate the presence and availability of a subscriber across states maintained in different autonomous systems. We note that grid computing, with its notion of virtual organizations, is well suited to solve portions of this puzzle. Further research is needed on the application of grid computing to telecommunications SOA.

The telecommunications SOA has to be built on open protocols. Not only will this aid in communications between federations, but also it will allow subscribers to describe their preferences and devices to enunciate their capabilities and store all of these in a secure repository accessible by trusted services. Open protocols will also allow devices to negotiate and prioritize services among themselves. In short, a robust discovery, selection, and binding mechanism for services in the telecommunications SOA will be another area of active research.

The telecommunications SOA has to be built with pervasive security. Devices must not be allowed to update a repository unless they can prove their authenticity. A subscriber must not be allowed to set up a session with another unless he can prove his veracity. Authentication and data integrity are that much more important in a telecommunications SOA because of the need for privacy that we accord to our communication needs.

And finally, the telecommunications SOA must support a multiplicity of service providers. In the Internet domain, innovation is played out as various providers — AOL, Google, Yahoo! — attempt to make themselves the preferred portal for a Web surfer by providing attractive services. Google, in particular, has played this card well. A Web surfer can use Google not only for browsing realty listings, but also to contact the listing realtor using Google Talk, take a virtual tour of the area in which the property is located using Google Earth, and then find directions to the property using Google Maps. Further innovation in the telecommunications domain will occur when multiple service providers vie for the attention of a subscriber. Providing extensible SOA frameworks for telecommunication services continues to be another area of fruitful research.

The evolution of a telecommunications SOA has just begun. The rest of this book provides an early glimpse into this evolution.