FOURTH QUARTER 2003, VOLUME 5, NO. 2

IEEE COMMUNICATIONS SURVEYS
The Electronic Magazine of Original Peer-Reviewed Survey Articles

www.comsoc.org/pubs/surveys

SERVICE ARCHITECTURES IN H.323 AND SIP: A COMPARISON

JOSEF GLASMANN, MUNICH UNIVERSITY OF TECHNOLOGY (TUM)
WOLFGANG KELLERER, DOCOMO COMMUNICATIONS LABORATORIES EUROPE GMBH
HARALD MÜLLER, SIEMENS AG

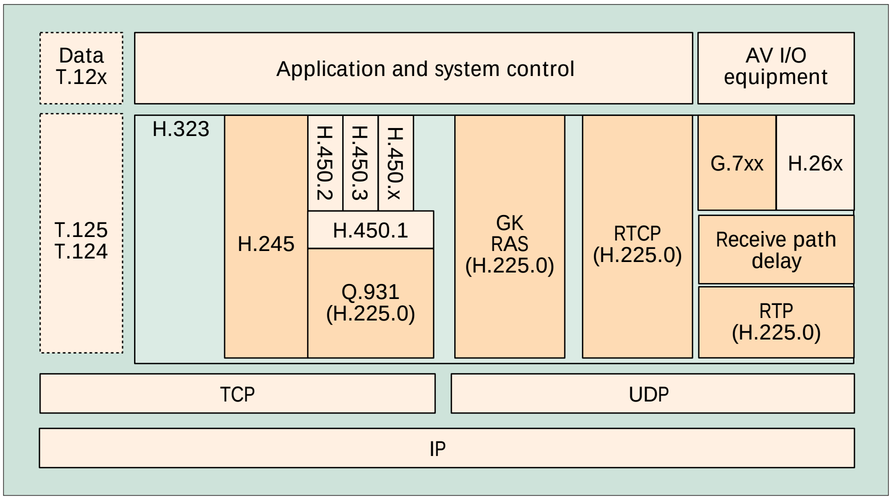
ABSTRACT

One of the major challenges for next-generation IP networks is to provide new, attractive multimedia services. This includes traditional telephony (voice over IP) and the interworking with legacy telephony systems. In addition to the general problems regarding the support of real-time services in the IP network, e.g., quality of service, voice over IP focuses on the control of advanced features such as supplementary services well known from telephony and on the mechanisms for their fast and efficient development and deployment. The two most promising approaches in the area of multimedia over IP are the protocol suites H.323 (ITU-T) and SIP (IETF). Several comparisons of these two protocols have already been published, but comparisons of their service architectures have been rarely addressed. This tutorial describes and compares the service architectures of H.323 and SIP. The basic protocol architectures are explained, followed by an in depth evaluation of the service implementation mechanisms. The analyses focus mainly on the control of telephony supplementary services in H.323 and SIP and are backed up by detailed examples. Although the two protocol architectures are quite similar, it is shown that there are considerable differences regarding their supplementary service architectures. H.323 (together with H.450) has been especially focused on supplementary services, smooth interworking with the PSTN, and interoperability between different implementations. In this respect, it has clear advantages for IP telephony applications. SIP has been designed with a broader scope, providing more generic syntax and semantics regarding feature definition and session description. Since the SIP standards do not describe details of possible application and service features, this bears the danger of interoperability problems, e.g, for supplementary services. SIP offers advantages for non voice over IP services and applications. A coexistence of both protocols can be foreseen, stressing the importance of interworking between them.

Fast and efficient development and deployment of new services are important drivers for advanced multimedia business. In the Internet world, ITU-T (H.323) and IETF (SIP) standards are important for advanced telephony services. Although using terms such as "voice over IP" (VoIP) these architectures provide far more services than just setting up voice calls. The expectations and requirements for these architectures are that they will provide those services that are well known from traditional telephony and that they will offer mechanisms to support the implementation and integration of new features. The comparison of the two standards in this article focuses on their service implementation concept and, in particular, on supplementary services.

Some studies have already been published comparing H.323 and SIP, for example, [1]. They mostly refer to the Basic Call architectures and focus on issues such as complexity and scalability. In [2] the focus is on service aspects and investigates the usage of the two standards for an overall service architecture according to criteria derived from the Telecommunications Information Networking Architecture (TINA). Until now, concrete service implementation issues have not been explained and discussed in detail.

In this tutorial H.323 and SIP are compared according to the following criteria: standardization philosophy, standardization status, supported services, supplementary service architecture, proprietary extension and negotiation mechanisms, interoperability of services and features, interworking with Public Switched Telephone Networks (PSTN), and service creation issues. Basic call control features such as call setup and session modification are distinguished from supplementary services. Here, supplementary services are referred to as user-perceived features that enhance the call with specific



■ **FIGURE 1.** *H.323 protocol suite.*

functionality that is only to be used with a Basic Call feature such as Call Hold or Call Forwarding, which are well known from traditional telephony. Supplementary service support is essential for the interworking with the PSTN. The term 'feature' is used in this tutorial as a more general term for characteristics of all types of user perceived applications including supplementary services. For analysis of the standards it is important to understand that extensibility does not only mean being open to all kinds of extensions; it also means implementation and interoperability issues. Quality of service issues, network services such as conferencing and addressing, and factors relating to the complexity and performance of the two protocols, are out of the scope of this tutorial.

The remainder of this tutorial is structured as follows. In the next section we present a brief overview of the protocols' architecture and their standardization status, concluded by a discussion of the general similarities and differences of the two approaches. Following that, the principle methods for service implementation in H.323 and SIP are explained in detail. A comparison highlights the significant differences that exist. The comparative statements are further illustrated by means of concrete service examples. Finally, a summary of the results concludes this tutorial.

THE BASIC PROTOCOL ARCHITECTURES

H.323 BASIC PROTOCOL

The ITU-T started work on defining VoIP signaling protocols in May 1995. In December 1996, Study Group 16 passed the H.323 v.1, referred to as a "standard for real-time videoconferencing over non-guaranteed quality of service LANs." In the meantime, the fourth version of the H.323 basic standard has been released [3]. This recommendation describes terminals and other entities (gatekeepers, gateways, multipoint control units) that provide multimedia communication over packet-based networks. Support for audio is mandatory, while data and video are optional. A rapid standardization process and straightforward interworking with the PSTN have been the main goals from the very beginning. Some existing protocols could be reused directly (Real Time Protocol (RTP) and Real Time Control Protocol (RTCP)). Others, such as H.225.0-CC [4] and H.245 [5], have been derived from the ITU-T H.320 [6] protocol suite H.221 [7] and H.242 [8]. The Registration, Admission and Status (RAS) protocol finally had to be designed from scratch. H.323 v.4 defines the Basic Call control and signaling for setting up multipoint multimedia conferences. The Basic Call procedure comprises RAS signaling functions and call signaling functions. RAS signaling functions are required for endpoint registration, admission control, and address resolution. Call-signaling functions include connection setup, capability exchange, and open logical channel procedures. Conferences in H.323 are normally tightly coupled and can be established out of a point-to-point connection via a multipoint controller which performs conference and floor control. To provide scalability, an extension for loosely coupled conferences without a central controller has been defined in H.332 [9].

Among other enhancements, the later versions of H.323 enable enhanced services on top of H.323. ITU-T Study Group 16 evolved the H.450 series recommendations in order to support supplementary services over IP networks. The scope of H.323 v.4 is depicted in Fig. 1 in the area shaded light green. H.323 v.4 contains optional components (light tan) as well as mandatory components (dark tan). H.450.1 defines a generic functional protocol on top of H.225.0-CC for all supplementary services [10]. It also defines the control procedures for the terminal equipment involved in handling the protocol messages. The most important supplementary services have been standardized already and new supplementary services are being added in an ongoing process. Table 1 shows the list of currently standardized supplementary services.

In recent years the ITU-T has standardized several multimedia service architectures for different types of networks (ISDN, ATM, IP). Currently, the work in the ITU-T regarding these multimedia (MM) services and systems is taking place in Study Group 16, which is organized into four Working Parties (WP). In the context of H.323, WP2 and WP4 are the most relevant. Table 2 gives an overview of the most important Questions within these two Working Parties, those directly related to H.323.

Question B/16 in WP4 defines an architectural framework as a common basis for the different MM systems (e.g., H.323, H.320, H.324) and tries to identify synergies between them. Question C/16 identifies and describes multimedia services and applications on top of the MM systems. WP2 deals more specifically with the architectures and protocols of the different MM systems. The H.323 umbrella standard and the core protocols of H.323 are dealt with in Question 2/16. Two Questions (D/16, 3/16) deal with interoperability topics, including PSTN and supplementary services interworking. Several other Questions in WP2 are working out how specific cross-sectional topics (e.g., QoS, mobility, security) are solved for the MM systems. This comprises the integration and use of existing protocols and methods as well as the definition of new ones.

SIP BASIC PROTOCOL

The IETF Multiparty Multimedia Session Control Working Group (MMUSIC WG) develops protocols to support Internet teleconferencing and multimedia communications. One main component is the Session Initiation Protocol (SIP). SIP is a session-layer transaction protocol that provides advanced signaling and control functionality for a large range of multimedia communications. It was specified by the MMUSIC WG as a proposed standard in 1999 (IETF RFC 2543) and was updated by the SIP WG in 2002 (IETF RFC 3261 [11]). The

Standard	Supplementary service	Subfunctions
H.450.1	Generic functions for supplementary services in H.323	
H.450.2	Call transfer for H.323	Single-step transfer Transfer with consultation
H.450.3	Call diversion for H.323	Call forwarding unconditional Call forwarding on busy Call forwarding on no reply Call deflection
H.450.4	Call hold for H.323	Near end hold Remote end hold
H.450.5	Call park and call pickup in H.323	Directed park and pickup Group park and pickup Pickup of alerting call
H.450.6	Call waiting in H.323	
H.450.7	Message waiting indication for H.323	Unified messaging system Message waiting call back
H.450.8	Name identification for H.323	
H.450.9	Call completion for H.323	Call completion on busy Call completion on no reply
H.450.10	Call offering for H.323	
H.450.11	Call intrusion for H.323	Conference type of connection Held type of connection Silent monitoring Forced release wait on Busy
H.450.12	Common information for H.323	

Table 1. *List of currently standardized features in H.450*

main functions are: location of resources/parties, invitation to service sessions, and negotiation of session parameters.

To fulfill this functionality, SIP provides a small number of text-based messages to be exchanged in separate transactions between the SIP peer entities (SIP user agent in a user terminal). In this way, the Basic Call control functionality is provided by one signaling transaction using the INVITE request message, whereas SIP is independent from the session it establishes. Other transactions complement the Basic Call, e.g., explicit call release. Network entities such as proxy servers or redirect servers that can be traversed by the messages, and can be used for support, e.g., for address resolution. It is fundamental to the SIP architecture that the signaling path is independent from the data path.

The session itself is described at two levels. The SIP protocol contains the parties' addresses and protocol processing features; the description of the media streams that are exchanged between the parties of a multimedia session are defined by another protocol. Therefore, the IETF suggests the Session Description Protocol (SDP, IETF RFC 2327, [12]). SDP is, in fact, not a protocol, but a structured, text-based media-description format that can be carried in the SIP message body. Since the message body is transparent to SIP any session description can be transferred, including a Web link. SIP sessions are not restricted to telephony calls or conference capabilities, but can include information retrieval or broadcast sessions, depending on the session description. SDP also allows the scheduling of session start and stop times or to describe recurring sessions. This tutorial refers to SDP in the context of SIP unless stated otherwise.

In addition to *baseline SIP* according to RFC 3261, several RFCs and a large number of Internet-Drafts complete or enhance the architecture regarding SIP applications, supplementary services, feature programming, conference, routing, preference management, and interworking issues. In this way, the IETF SIP IP-telephony standardization is still a "work in progress" and has not yet reached a final state. The baseline RFC and other RFCs and the Internet-Drafts are under continuous refinement by several WGs.

Upon observation of the drafts and at the WGs in particular ([13, 14]) one can note that the current approach for SIP standardization comprises three levels. The base level deals with the maintenance of the baseline SIP protocol, which is done by the SIP WG. There are several application fields of SIP. Supplementary services for call control are one application field. In order to support a broad range of supplementary services a call control framework was drafted in July 2001 [15]. The development of SIP extensions to support multiple applications forms the second level of SIP standardization. Here, a number of recently issued RFCs extend the baseline SIP RFC 3261, e.g., event notification (RFC 3265 [16]), session update (RFC 3311 [17]), provisional responses (RFC 3262 [18]), or resource management (RFC 3312 [19]). The third level will be concrete features or supplementary services such as Call Transfer. So far, there are no RFCs defining supplementary services for SIP. There are some standardized SIP applications such as the use of SIP in support of deaf, hard of hearing, and speech impaired individuals (RFC 3351).

This approach is illustrated in Table 3, which summarizes the IETF Working Groups and their role in the ongoing SIP standardization process. The RFCs are included to show the current state of standardization.

The MMUSIC WG is dealing with the revision of SDP and has transferred the responsibility for SIP to the SIP WG, which has produced the current version of the SIP standard (RFC 3261). All types of possible SIP applications and usage scenarios are investigated in the Session Initiation Proposal Investigation (SIPPING) WG. SIPPING acts as a filtering function in front of the SIP WG. It describes the requirements for any extension to SIP determined to be needed. The SIP WG finally decides and includes extensions in SIP or issues separate RFCs [20].

For example, conferences in SIP are normally lightweight multicast conferences, to which a user can be invited. Some SIP extensions for the management of distributed multipoint conferences have been drafted within the SIPPING multiparty task. However, advanced conference control as floor control or the support of roles is not in the scope of SIP. Other extensions discussed to set up a call control framework are described later.

Beneath call control there are other application fields for SIP that are handled by separate Working Groups. The following Working Groups have been founded: SIP for Instant Messaging and Presence Leveraging Extensions (SIMPLE); Service in the PSTN/IN Requesting InTernet Service (SPIRITS); PSTN to Internet Integration (PINT); and IPTEL. The IPTEL WG, for example, proposes several possibilities for the programming of services either for administrators or for the users themselves [21]. This will be discussed later.

Figure 2 depicts the overall IETF SIP protocol suite. Work

Question	Title	Focus tasks	Standards (examples)
		Working Party 2/16: Multimedia Platform and Interworking	
D/16	Interoperability of Multimedia Systems and Services	Interoperability of services (e.g., supplementary services); interoperability of multimedia systems with each other and with the legacy telecommunications network; measures to enhance interoperability of different implementations.	H.246
F/16	Quality of Service (QoS) and End-to-End Performance in Multimedia Systems	QoS needs of MM systems; QoS signaling methods; common API to different QoS signaling methods; aspects of end-to-end performance as perceived by the user.	Contributions to standards in other questions
G/16	Security of Multimedia Systems and Services	Threat analysis of MM systems and services; definition of a security framework; contribution to MM architecture to incorporate security .	H.235
1/16	Multimedia Systems, Terminals and Data Conferencing	Improvements and enhancements of audiovisual communication systems over fixed, mobile, and B-ISDN networks; data sharing; use of enhanced audio and video coding	H.310, H.320, H.321, H.324, T.120
2/16	Multimedia over Packet Networks using H.323 Systems	Ongoing work in basic H.323 protocols, currently focused on mobility; interactions with SIS/IN; stimulus-based call signaling in H.323 combined with network control of terminating call services.	H.323, H.225, H.450, H.332
3/16	Infrastructure and Interoperability for Multimedia over Packet Network Systems	H.323 gateways and interworking; PSTN interworking; SS7/IN interworking; gateway decomposition; MCUs; management of H.323 systems; H.323 MIB; updates to connection control signaling (H.245).	H.245, H.246 Annexes, H.248, H.341
4/16	Video and Data Conferencing using Internet-Supported Services	Architecture and protocols to integrate video and data conferencing functions with Internet-supported service functions; mechanism of synchronization between audiovisual and other service presentations; multipoint aspects.	None so far
5/16	Mobility for Multimedia Systems and Services	Further develop mobility for H.323 and H.324; consider protocol support for MM mobility for both users; terminal and service mobility.	H.501, H.510, contributions to standards in other questions
		Working Party 4/16: Multimedia Framework	
B/16	Multimedia Architecture	Common architectural framework for multimedia projects; consistency among different MM systems; support common protocols and architecture elements (e.g., H.245).	Multimedia architectural framework
C/16	Multimedia Applications and Services	Identify MM services and applications; provide service descriptions (service examples: retrieval services, distribution services, messaging services, collection services, emergency multimedia services, e-commerce services or applications, telemedicine applications)	F-Series

■ **Table 2.** *The most important H.323-related activiteis in the ITU-T SG16.*

in progress extensions are included, and the independence of SIP session signaling from audio/video media processing is shown.

Comparing the Basic Architectures

It is quite important to understand the different basic philosophies in H.323 and SIP standardization. The ITU-T follows a top-down approach. The H.323 standards are more specified in the sense that they describe a complete framework and detailed protocols, state machines, and message flows for multimedia communication. This includes specific solutions for cross-sectional factors such as QoS, security, and mobility. SIP, on the other hand, follows a more bottom-up approach, according to the IETF philosophy, where systems and applications are formed by combining generic modules. Every protocol standardized by the IETF should be independent of a specific application. Therefore, the SIP specification does not include issues such as QoS or mobility.

There are also differences in the focus of H.323 and SIP in the past and at present. In the beginning, H.323 was concentrating on basic multimedia functionality, supplementary services, and interworking (e.g., PSTN). Now that there exist sufficient solutions in those areas, one can observe a shift in focus to solution topics such as security, mobility, and QoS. SIP took another route, starting with the definition of a generic protocol to set up service sessions. Currently, SIP tends to put more focus on topics for specific applications, including supplementary services and interworking with legacy networks (e.g., PSTN).

Technically, SIP and H.323 are based on similar concepts. Tables 4 and 5 show the components and protocols of the two approaches. H.323 and SIP comprise many analogies with respect to function split and service location. In SIP as well as in H.323, Basic Call and feature control are performed mainly in the terminals. Features requiring network support, such as servers (gatekeeper resp. proxy server, …), are provided in the networks.

When looking at the current activities in the ITU and IETF, it can be assumed that H.323 and SIP will further converge in the near future. When looking at the standardization of supplementary services, notable differences can be seen. H.323 is

Working Group	Description	Focus tasks	Standards (RFCs)
MMUSIC	• Revision of SDP (SDPng) • Maintenance and revision of RTSP Revision is based on implementation experience and additional demands from other WGs (SIP, SIPPING, MEGACO, AVT).	Baseline SDP evolution	SIP (RFC 2543) — obsolete SDP (RFC 2327) SAP (RFC 2974) RTSP (RFC 2326) SDP for ATM (RFC 3108)
SIP	Specification and maintenance of the basic model and architecture defined by SIP and its extensions ("chartered to be the owner of SIP"). Responding to general-purpose requirements for changes to SIP provided by other WGs (SIPPING, SIMPLE, IPTEL).	Baseline SIP and extensions	SIP (RFC 3261) INFO method (RFC 2976) PRACK (RFC 3262) Locating SIP servers (RFC 3263) Event notification (RFC 3265) Reason (RFC 3326) UPDATE (RFC 3311) Res. Mgmt (RFC 3312) MIME-Types (RFC 3204)
SIPPING	Documenting the usage of SIP to solve real problems that need to be solved in a standardized manner. Describing the requirements for any extension determined to be needed and handling them to SIP WG ("filter function"). Looking for commonalities among tasks. Close cooperation with WGs (SIP, IPTEL, PINT, SPIRITS, SIMPLE, AAA, MMUSIC) and 3GPP, 3GPP2, DCS.	Investigate application areas: • Telephony (PSTN, 3G) • Messaging • Multi-party • Media servers	Hearing impaired (RFC 3351) SIP-T requirements (RFC 3372) SIP-ISUP (RFC 3398)
IPTEL	IPTEL is on problems related to propagation of routing information for VoIP protocols, (Gateway Location Protocol, CPL-Framework, TRIP).	Call processing and routing	CPL (RFC 2824) GW-location (RFC 2871) TRIP (RFC 3219)
SIMPLE	The focus lies on the application of the Session Initiation Protocol (SIP, RFC 2543) to the suite of services collectively known as instant messaging and presence (IMP).	Instant messaging	No RFC
SPIRITS	Concerns architecture and protocols for secure transport of IN trigger information from PSTN/IN to the IP network, and optional responses from the IP network back to the PSTN/IN. Collaborate with other WGs (IPTEL, MMUSIC, PINT, SIP) and other relevant standards bodies (ITU-T SG11).	PSTN/IN and IP interworking	SPIRITS-Architecture (RFC 3136) Pre-SPIRITS (RFC 2995) Protocol req. (RFC 3298)
PINT (concluded)	Addresses connection arrangements through which Internet applications can request and enrich PSTN (Public Switched Telephone Network) telephony services, e.g., "click-to-dial." Specification of a service support transfer protocol between Internet and PSTN.	IP-initiated PSTN/IN calls	Pre-PINT (RFC 2458) PINT-ext.-to-SIP (RFC 2848) PINT-MIB (RFC 3055)

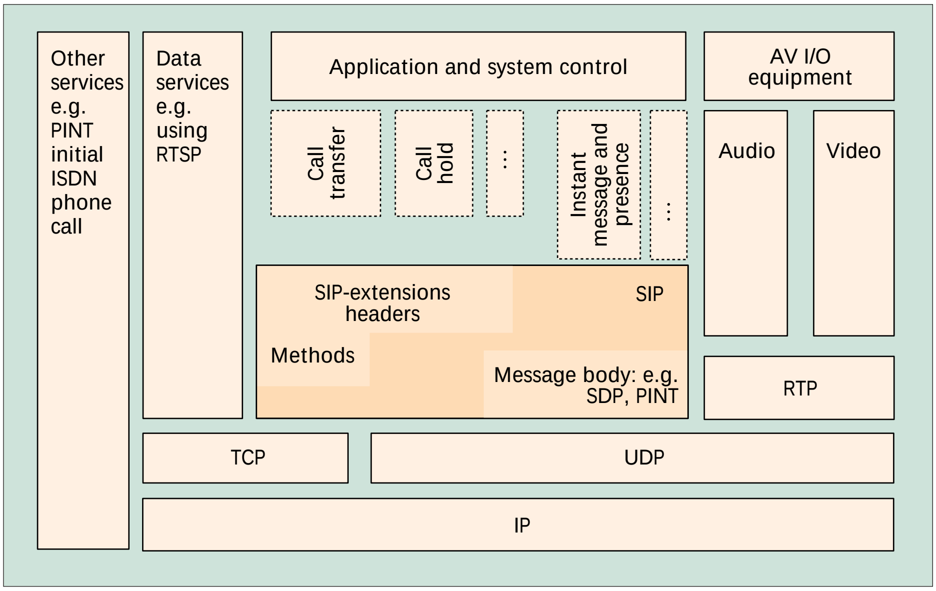
Table 3. *SIP-related activities in the IETF.*

standardizing supplementary services, whereas SIP is standardizing protocol transactions as extensions to baseline SIP that could be used for supplementary services. This makes it difficult to compare SIP with H.323. However, it can be expected that SIP will take a similar direction as H.323. Both define a general framework for call control features or are working toward it, respectively. The framework defines a standardization process and rules for the implementation of new features.

When looking at the H.323 call setup procedure, the ITU-T has additionally introduced the optional fast connect procedure and signaling via UDP. The fast connect procedure is similar to the lightweight SIP session setup and combines the opening of a call control channel, the capability exchange, and the open logical channel procedure in one single signaling transaction. This means H.323 becomes more lightweight, as is SIP.

The substantial difference between the two protocols lies in their targeted range of applications. SIP has been designed as a general transaction protocol for setup and tear down of generic sessions. Voice and multimedia are only possible example applications of SIP. When not supporting voice or multimedia, a core SIP user agent can be very thin (e.g., only containing the SIP protocol and a generic session description). H.323, on the other hand, has been designed as a control protocol suite with the focus on multimedia applications, including telephony. Naturally, because the scope is more restricted as compared to SIP, the range of applications for H.323 is not as wide as for SIP. A couple of simple endpoint types (SETs), comprising only a well defined subset of the full H.323 functionality, have already been specified. But even the SETs are more complex as compared to SIP user agents. On the other hand, H.323 provides a more precise and detailed specification of voice and multimedia functionality.

The main focus of this tutorial is on describing and comparing the service architectures of H.323 and SIP. The following sections provide a deeper look into the service implementation process and compare the two approaches, with a focus on supplementary services. This is backed up by explicit service examples.



■ **Figure 2.** *IETF SIP protocol suite.*

The Service Architectures

H.323 Service Architecture

In this section the three main models for supplementary service control currently existing in H.323 are described. They are: distributed feature control (H.450), stimulus feature control (H.323 Annex L), and application-layer feature control (H.323 Annex K).

Distributed Feature Control Using H.450

Classification of Features in H.323 — Features in H.323 can be categorized into three classes: local features, network-based features, and supplementary services.

Local features can be implemented in the endpoints without requiring specific signaling to other network entities. Examples of local features are: repeat a call, call history and call lists, local address book, speed dialing, privacy functions such as do not disturb and mute, and so on.

The second class of features are those that require centralized control. These network-based features are implemented in a centralized fashion in the gatekeeper (GK) or as a backend service behind the GK. Examples are authorization, address resolution, call admission, call detail recording, name/number suppression, and so on.

The third class of features is the set of supplementary services (H.450). These are features that require special signaling between the corresponding entities. Examples of supplementary services are Call Forwarding, Call Transfer, Call Completion, and Call Hold.

H.450 Design Philosophy — The H.450 supplementary service architecture uses a decentralized function split as far as possible. Thus the peer entities (servers, clients, MCUs, GWs) communicate directly using H.450 signaling without involving centralized network control. For those features that require centralized control, a feature server is used, which is a special form of an H.323/H.450 endpoint. Examples where a feature server is required are: user not available proxy (acts on behalf of unavailable endpoints), messaging server, automatic call distribution (ACD) server, or group-server for group features (e.g., call park and pickup).

Besides a fully distributed feature control, H.450 also describes a model where parts of the H.450 functionality can be carried out in H.450 proxies on behalf of the endpoints. The H.450 proxy can, for example, be collocated with the gatekeeper (GK).

One of the most important requirements for the design of H.450 was to simplify feature interworking with switched private (QSIG, standardized networking protocol for PBX interconnection) and public networks (ISDN). Furthermore, H.450 has been designed to be a highly extensible protocol as described in the next subsection. This also includes several mechanisms to ensure interoperability between endpoints with differing feature sets, which is a precondition for multi-vendor interoperability and smooth deployment of new features. The basic interoperability mechanisms provided by the H.450 framework are: feature identification (H.450.1); declaration of

	Client	Servers in the network		
SIP (IETF)	Terminal	Proxy server, registrar	Conference server[1]	Gateway[1,2]
H.323 (ITU-T)	Terminal	Gatekeeper	MCU	Gateway

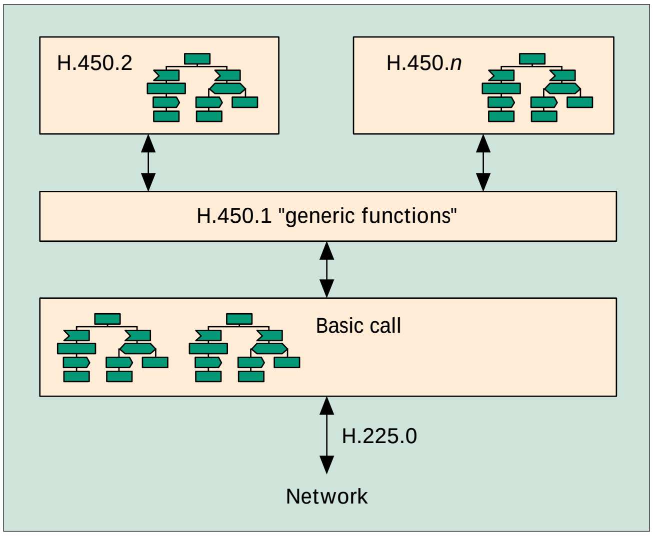
[1] Not standardized up to now. [2] Addressed in several drafts, e.g., SIP-T
MCU: Multipoint control unit. Gateway: maintains transition to traditional telephony.

■ **Table 4.** *SIP and H.323 components.*

	Real-time data transmission	Call control	Feature Control			Signaling procedure variants	
			Framework	Extensions	Supplem. services		
SIP (IETF)	RTP/RTCP	SIP (SDP)	Call control framework*	REFER, INFO SUBSCRIBE, NOTIFY, UPDATE	Transfer* Call hold	—	SIP-INVITE transaction
H.323 (ITU-T)	RTP/RTCP	H.225.0, H.245	H.450.1	H.450.1	H.450.2–H.450.12	Basic call setup	Fast connect

*Not explicitly standardized as a supplementary service.

■ **Table 5.** *Protocols running on the terminal.*



■ **FIGURE 3.** *H.323/H.450 architecture in the endpoints.*

default behavior for features that are not supported (H.450.1); a feature look-ahead mechanism (H.450.12); and architectural provisions to separate feature states from call states. This provides a good basis for dealing with interoperability at design time and at runtime.

Extension of H.450 — H.450 also provides several mechanisms that allow easy extension of this feature set. This subsection gives an overview of the H.450 architecture and its extension mechanisms.

H.450 supplementary service information is sent in H.450 application protocol data units (APDUs) that may be contained in any H.225.0-CC message. H.450 APDUs are exchanged between supplementary service entities and do not influence the underlying H.225.0 call state. Among other information, H.450 APDUs contain remote operations service (ROS) operations that define the semantics of the supplementary services. As with the other H.323 protocol components, H.450 APDUs are specified and coded using ASN.1 notation.

The H.450 APDUs can be extended by manufacturer-specific information (NonStandardData). This can either be additional information elements or even new operations. Using this extension mechanism, new supplementary services can easily be defined.

The standard H.450.1 ("Generic Functions" (GF) [10]) provides generic services for feature control that are common for all standardized and manufacturer-specific supplementary services. H.450.1 provides call-related and call-independent transport of H.450 APDUs. Further, H.450.1 defines in a generic way how to proceed with H.450 APDUs that are not supported. This enables interoperability between endpoints with differing feature sets and a stepwise deployment of new supplementary services without having to support them in all endpoints at the same time.

Another standard that facilitates interoperability between heterogeneous endpoints is the emerging Recommendation H.450.12 ("Common Information," [22]). H.450.12 can be used to exchange the endpoints' feature capabilities. This can be used to react in advance on these capabilities. One application for this is to not present to the user the opportunity to make a transfer if the other endpoint does not support it.

As shown in Fig. 3, the architecture of H.323/H.450 enables a separation between Basic Call and the supplementary services. H.225.0 messages are routed to the Basic Call entity, where they trigger the respective actions and state changes. H.450 information is passed on to H.450.1. The generic services are carried out and the ROS operations are passed on to the respective supplementary service entity. The GF is also the place where simultaneously activated features can be coordinated or even blocked in case of feature interactions. Each supplementary service entity has its own state machine that defines the semantics of the supplementary service. The supplementary service state machine is only invoked when a supplementary service is requested. The H.450.x standards specify these state machines using ITU's Specification and Description Language (SDL) diagrams.

This modular architecture is based on an object-oriented approach and provides scalability with respect to features as it enables easy addition of new supplementary services. When creating a new feature, the new state machine is defined without the need to change the state machines of the Basic Call and of other supplementary services. This even allows concepts for dynamic introduction of new features into running systems ("feature pluggability"). In contrast, having only a single-state machine including the supplementary services can grow very complex if many features are added.

Building Feature Combinations for 1st Party Applications — One of the basic ideas of the H.450 features is to define them in such a way that they can be used together with the Basic Call as building blocks. By combining the atomic feature blocks, more complex features and services can be built. Thus, a huge set of features can be created by using only a small set of carefully designed building blocks. Applications and further features are built by using telephony APIs on the local machine (e.g., TAPI, JTAPI). Some examples of such combined features are:

- Consultation transfer = Call Hold + Basic Call + Single Step Call Transfer.
- Messaging = Basic Call + Call Transfer to Announcement Server + DTMF Control (Basic Call).
- Attendant console = Basic Call (multiple line) + Call Hold + Call Transfer.

Building Feature Combinations for 3rd Party Applications — By introducing a CTI interface (computer telephony integration) that connects to the H.323/H.450 functionality in a remote endpoint, the H.323/H.450 building blocks may be remote controlled, thus making possible the creation of 3rd party call control applications. The functionality provided via the CTI interface may also include functions such as monitoring endpoints (e.g., detect when a user is available, …). The CTI interface is accessible via an API; a common protocol that may be used is Computer Supported Telephony Applications (CSTA). An example of a 3rd party application is automatic call distribution (ACD). It can be built up by combining Basic Call, Call Transfer to a music/video server, monitoring an ACD agent using the CTI interface, and CTI-initiated Call Transfer to an agent. Note that the CTI interface and protocol is an add on and is not specified in H.323.

Stimulus Feature Control using H.323 Annex L — Stimulus feature control is the opposite approach as compared to the decentralized H.450 architecture. A centralized feature server in the network is used to control the features in the endpoints. This approach is defined in H.323 Annex L. Endpoints conforming to H.323 Annex L still use functional signaling (H.225.0) for controlling the Basic Call, which yields Basic Call interoperability with fully functional H.323/H.450 endpoints. For centralized feature control, a so called feature key management protocol is defined. It contains basic procedures to control user interface functions such as key presses, display messages, and feature indicators (e.g., LEDs). There is little intelligence in the endpoints; the feature logic and the semantic procedures are defined in the centralized feature server. This allows an easier deployment of features: only the feature server has to be updated; the endpoints do not have to be changed. On the other hand, there are no standardized semantics for the features. The semantics are implementation-dependent. Applications that want to use the feature functionality for 1st party call control need a functional interface (API), which cannot easily be provided using the stimulus approach. A further downside of stimulus feature control is the scalability problem due to feature processing in centralized network components.

Application-Layer Feature Control using H.323 Annex K — H.323 Annex K defines an optional way to control general services or applications beyond supplementary services in H.323. It allows the development and deployment of new services without updating the H.323 protocol and endpoints. Annex K introduces a service plane above the H.323 call control plane. A service control session can be established after exchanging the relevant information (e.g., sessionID, URL for service control, ...) in RAS or H.225-CC messages. This mechanism allows for call-related and call-independent service control. Service control sessions can be maintained between endpoints or between endpoints and the network (e.g., GK).

The HTTP protocol is used in the service control channel to actually offer, select and activate the services. The service logic is described in HTML pages, scripts, etc. that are transferred via the HTTP protocol. Thus, features can be controlled from any device running a conventional Web browser (including, e.g., PDAs).

Example applications may include transferring XML pages, possibly including Java code or scripts, downloading of tones and announcements, or uploading call processing scripts from a client to the GK. A concrete example scenario is explained later.

SIP SERVICE ARCHITECTURE

Distributed Feature Control in SIP — Similar to H.323/H.450, SIP feature control is based on a distributed feature control model. As SIP relies on intelligent terminals, stimulus or transaction protocol-based remote control is not in the scope. There are other complementing standards for this architecture, as described below. For SIP the same feature categories can be applied as in H.323: local features, network-based features such as authorization and address resolution in an outbound SIP proxy, and supplementary services.

SIP Protocol Design Philosophy and Standardization Process — As previously mentioned, SIP's supplementary services have not been standardized. Currently, the working groups are evaluating application scenarios and deriving requirements for SIP protocol extensions that can be used to support multiple features, including supplementary services. The standardization process itself follows the IETF approach to agree on a rough consensus and to proove results by running code prototypes. The SIP design process concentrates on the SIP protocol rather than on specific supplementary services and their interworking. A key requirement for the evolution of the SIP protocol is modularity to achieve flexibility regarding all types of features, including supplementary services. SIP is intended to be as general as possible, which means extensions should be carefully selected so that they serve multiple application purposes and features.

SIP must remain a generic transaction protocol defining different protocol messages for different high-level behaviors. The transactions are independent from each other and independent from the semantic of the session they are controlling. SIP transactions rely on network-layer services (e.g., QoS) that are developed outside SIP. The messages may traverse different networks or autonomous systems than the user data.

Considering this philosophy SIP is going to evolve step by step to provide a rich but general set of mechanisms for a plethora of applications encompassing many features, including supplementary services (Fig. 4). Features can make use of baseline SIP mechanisms or require protocol extensions. The standardization of these extensions is still in progress.

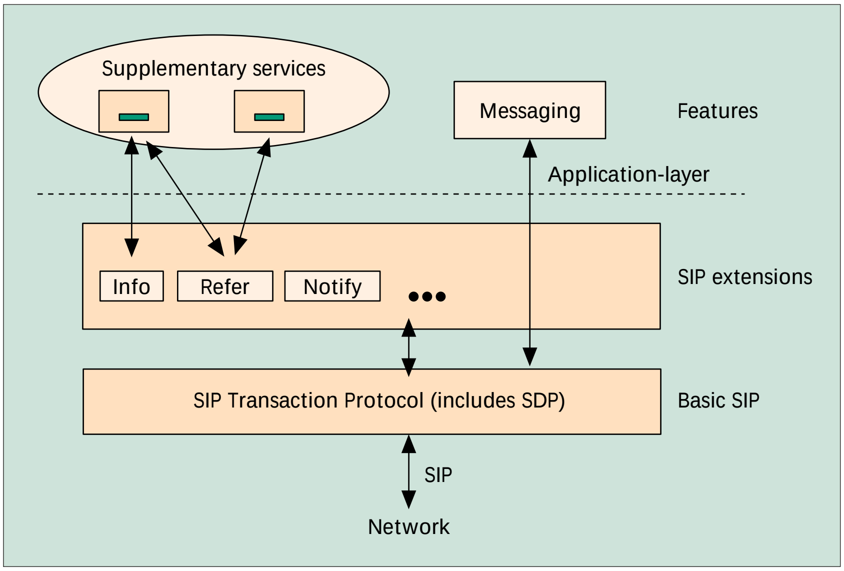
Protocol extensions describe new SIP methods, new headers, or new response codes. Methods are SIP request messages such as INVITE that define a SIP transaction. Headers are identifiers for message parameters such as receiver (To:) or route discriminators (Route:). The server's answer on a request message is similar to the HTTP protocol coded in a three-digit response code, i.e., 200 for "OK."

Supplementary Services by Baseline SIP Mechanisms — Referring to the proposed standard (RFC 3261) there are no explicitly standardized supplementary services in SIP. There are drafts provided by the SIPPING WG showing sample sequence diagrams for certain features [23]. Some of these supplementary services can be realized by the baseline SIP protocol functionalities, i.e., by SIP requests and the transported session description. There is no explicit signaling of supplementary services in SIP as it is in H.323, which is due to the different SIP philosophy.

Definition of SIP Extensions — For other features, including supplementary services, the definition of new headers and new methods has been done in several RFCs or draft proposals. Regarding call control-related features, a draft proposes a special framework for call control extensions regarding supplementary services such as Call Transfer, conferences, call park/pickup, and call monitoring [15]. Here, the authors want to make sure that supplementary services are modularly defined and are separated from each other, which allows standardized support of call control supplementary service negotiation. At the time this article was written, the SIP WG had defined the following extensions to SIP in order to support such a generic call control framework (the names of the new methods are given in parenthises): modification of conversation space (REFER, REPLACE); non state-changing information (INFO); events (SUBSCRIBE/NOTIFY); alter session (UPDATE); diversion (JOIN/FORK); messaging (MESSAGE); session keep-alive timer. As an example, the Call Transfer supplementary service [24] has been drafted. This might be an indication that SIP tends to evolve in the direction of H.323/H.450-like supplementary service definitions.

Table 6 gives an overview of several selected RFCs and current working documents for the standardization of SIP supplementary services.

Feature Negotiation — SIP provides a well-defined specification for feature negotiation. When a header is not known by a SIP entity it is ignored without affecting the rest of the request. SIP provides the Require header that could be used by a SIP client to make sure in advance that a desired behavior, e.g., an extension involving one or more new SIP headers, is known by the peer server. An error indicating the nature of the problem is returned if the behavior is not supported. Headers are referenced by their name, which has to be registered with the Internet Assigned Numbers Authority (IANA) to guarantee correct functionality. As an alternative, headers could be referenced by their reverse name of location (e.g.,



■ **FIGURE 4.** *SIP architecture.*

de.company.office.server). For the negotiation of features based on new methods the Allow header can be inserted in a SIP message. It shows the receiver which methods are understood by the sender of the message. The Supported header enumerates all the extensions that allow specific behavior and are supported by a SIP user agent. The negotiation of extensions (Require header, Supported header) is based on so called option tags that are unique identifiers used to designate new options (extensions). These option tags are to be defined in standard-track RFCs to support multi-vendor interoperability. This makes it possible, for example, to explicitly specify supplementary services. The correct interworking between different SIP implementations is tested in so called SIPIT events that are organized regularly (http://www.cs.columbia.edu/sip/sipit/).

Alternative Session Descriptions — To transport a message body other than SDP may also be regarded as an extension. For example, the message body may include a Web page showing a map with the current location of the callee. Therefore, SIP can also be used for other session signaling tasks, e.g., the PINT protocol [25] uses SIP to set up calls between two PSTN phones via an ISDN third-party call setup gateway. Since these calls are normally terminated in the PSTN, the SIP BYE message only ends the signaling relationship between the PINT entities. In [25] the authors provide rules for the application of BYE in this special case.

Stimulus Feature Control Based on Megaco — SIP standardization does not include stimulus control. Complementing the SIP architecture in this respect, the Media Gateway Control Protocol (Megaco) standard defines a protocol for the interaction between a media gateway and a media gateway controller [26]. The goal of the Megaco architecture is to couple PSTN networks and their services (based on non-intelligent terminals) with the Internet. Media gateways at the borderline are for the control of Internet services, and they are thus able to obtain and transfer the stimulus information from non-intelligent endpoints via the media gateway controller to the session control.

Application-Layer Feature Control by Service Programming Languages — In general, service creation in SIP can be done using any programming language. By service creation is mean the implementation of a service logic that either controls a specific message flow (cf. the feature combinations approach of H.450) or that reacts on a message request. Targeted for the implementation in SIP proxy servers, the IETF has drafted several programming languages that can be used for the implementation of services started by message requests [21].

The IETF generally distinguishes between trusted and untrusted users. For untrusted users, i.e., the end-users, the Call Processing Language (CPL) provides the means to handle SIP INVITE transactions (deciding whether an incoming request should be rejected, forwarded, or proxied). Addressing inexperienced users, the CPL-functions are very restrictive to avoid security and performance problems (e.g., loops).

Two approaches for service creation are proposed for trusted users, e.g., server administrators: SIP-CGI scripts and SIP servlets. SIP-CGI scripts are derived from HTTP-CGI scripts, but have a number of enhancements for supplementary service control such as the generation of multiple responses or the ability to handle additional requests. Unlike HTTP-CGI scripts, SIP-CGI scripts can be re-invoked in order to manage a complete SIP transaction. Whereas SIP-CGI scripts are independent from any programming language, SIP servlets require a Java environment. SIP servlets can be triggered by incoming requests and instruct SIP servers how to handle those requests.

In this way SIP provides a number of mechanisms for service programming that can all be easily applied since they are based on well known programming methods. This is a great advantage compared to the proprietary programming methods of the traditional telephony service creation environments, e.g., intelligent networks. For all service implementations on SIP proxy servers, it must be ensured that the request traverses the responsible SIP proxy. Since IP networks are not route-aware, SIP provides the *Route* header to determine the path a message must take.

In addition to the programming methods described above that are aimed at program services with dedicated languages on a dedicated SIP server, a new approach suggests the use of Java applets to be used in SIP requests [21].

COMPARING THE SERVICE ARCHITECTURES

The comparison of the implementation methods between the SIP and the H.323 service architecture in this tutorial is based on the following criteria: architecture, protocol extensions, message coding, and service programming.

The key characteristic of the H.323 service architecture is its explicit definition of separate state machines for each supplementary service, independent of the Basic Call state machine. From the signaling point of view, the function split of feature control into framework and extensions is a consequence of this separation. The ITU-T has already standardized the feature state machines of the most important telephony supplementary services, with further supplementary services being added in the future. The syntax is specified using ASN.1 and the semantics are described using SDL diagrams.

This elaborated and object-oriented approach is based on a long period of experience with the implementation and maintenance of telephony supplementary services, mainly in the more feature-rich private networks (QSIG). As a consequence, important challenges such as the subsequent integration of new features into a running system, the interoperability with heterogeneous endpoints, and the often neglected but very important field of feature interaction can be dealt with. By reusing the PSTN protocols, ease of interworking with already existing telephony systems (ISDN, QSIG) has been taken into consideration also. Altogether, this approach allows short product cycles.

The baseline SIP RFC provides only rudimentary implementation instructions. According to the SIP baseline standard [11], SIP features are not explicitly signaled. At the moment the path taken by IETF for the standardization of supplementary services is quite different than the path taken by the ITU. In contrast to the ITU and following the SIP design philosophy, released SIP extensions should be independent of a single feature and comprise only a single protocol transaction (method) or a parameter set (header). A feature may consist of one protocol transaction or a sequence of protocol transactions, whereas the service logic itself is not part of the standardization process. This implies that the syntax of a single feature is not standardized explicitly and the semantic is left to the application. This may not be a problem for the initiation of applications following a request-response scheme such as Web services, for example. Here an application programmer can use SIP transactions as feature building blocks. For features that require an explicit signaling to invoke a standard behavior, such as in telephony supplementary services, the problem arises of how to exactly identify a feature. The use of the identical method may activate different behavior in different implementations, because SIP is standardized on the transaction level and not on the supplementary service level. This is especially important for the interworking with legacy telephony system with their vast range of supplementary services. IP-based VoIP systems should be able to provide the same functionality the customer is used to. The extension negotiation procedure between user agent client (UAC) and user agent server (UAS) basically addresses these concerns, but so far standardized supplementary services are not known.

Working Group	Document	RFC#
SIP	SIP REFER method	Draft
	SIP MESSAGE method (for use in instant messaging)	3428
	SIP INFO method	2976
	Event notification (SUBSCRIBE/NOTIFY)	3265
	SIP UPDATE method	3311
	SIP REASON header	3326
	SIP change process	3427
SIPPING	SIP for telephones (SIP-T) (interworking issues with PSTN)	3372
	SIP-ISUP	3398
	ISDN/ISUP to SIP mapping	3398
	SIP call control — transfer (supplementary service requirements)	Draft
	Message waiting (supplementary service requirements)	Draft
	SIP support of deaf, hard of hearing, and speech-impaired individuals	3351

Table 6. *Working documents essential for the implementation of supplementary services in SIP.*

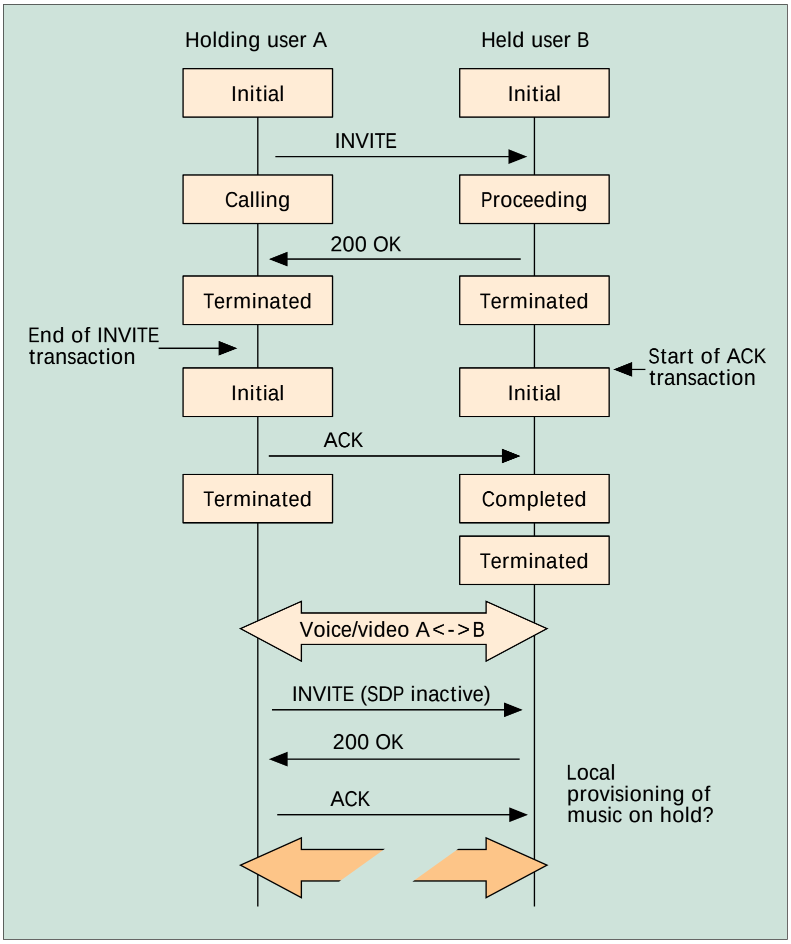
The problem arising from feature interaction is well known from traditional circuit switched telephone networks. As also described in [27], this problem is still more complicated in IP telephony for several reasons. Some of these reasons are shared between H.323 and SIP, mainly the fact that the states are kept in the endpoints and that features can be programmed by different parties and executed in different locations, for SIP even on the application level. Other reasons are caused by specific SIP properties, such as request expiration. In [27] the authors provide good examples of feature interactions that may arise out of this. One of the major remedies to cope with the feature interaction problem for supplementary services is explicitness, i.e., to clearly express via signaling the context of the supplementary service and/or the required behavior. Although attempts in this direction were started [28], standardization is still in progress. Approaches that suggest more extensive verification testing will also require explicitness if undesired interactions are detected and must be resolved. It remains to be seen whether the ongoing standardization efforts in the IETF are sufficient to meet the challenges of interoperability and feature interaction. As of today, it seems that the ITU is ahead of the IETF regarding supplementary services.

To support the definition of features in SIP, a standardization process has been postulated [14, 20]. In [20] the authors describe an IETF process to ensure the appropriateness and consistency of SIP extensions. Furthermore, it is pointed out that extensions must solve problems in a generic way rather than for a specific use. Related drafts, such as [23], are of informational nature to illustrate the potential usage of an extension in an example message flow. Technical guidelines are provided in [14]. However, these guidelines do not go as far as H.323 to require specific state machine descriptions.

For the negotiation of proprietary extensions, H.450 allows the transmission of individual rules inline with the request messages. These rules instruct the receiver what to do if the extension is unknown. SIP has a very robust general negotiation mechanism. Actions on errors due to not-supported extensions cannot be sent in the same message, but could be provided in the following message. In order to identify features, H.450 defines a hierarchical name space using vendor-specific extensions. Thus, no central authority is required for changes as soon as the vendor has an official vendor ID. This is quite different with SIP, where features are not signaled explicitly and SIP's negotiation procedure concerns only the extensions. SIP extensions must be registered with IANA to avoid interworking problems on the transaction level.

Several programming languages are defined in the context of SIP for the programming of SIP servers. Although missing in H.323, they could also be applied for it. Since these programming languages are derived from the HTTP context they are more easily applicable for SIP, as SIP is based on HTTP.

In [1] the use of ASN.1 syntax for coding H.323 messages is criticized as an overhead in complexity contrary to the text-based SIP approach. Text-based coding has advantages in rapid prototyping of individual solutions. A second point is the analysis of signaling messages. For example, a network administrator can interpret the content of a message without an interpreter. On the other hand, when using ASN.1 there is



■ **FIGURE 5.** *SIP: far end hold.*

a systematic support of the software development process. Tools for syntax checking and automatic code generation speed up and ease the implementation of message processing functions. Further, the packed encoding rules (PER) used in H.323 compress ASN.1 signaling messages very effectively. SIP also provides an efficient header compression for links with low data rate that makes the transferred number of bytes comparable to H.323.

DETAILED SERVICE EXAMPLES

In the previous sections we have presented an explanation and discussion of the characteristics and the functionality of the two IP telephony standards. In the following sections we will present detailed examples to illustrate significant differences between H.323 and SIP. Two typical supplementary service examples (Call Hold and Call Transfer) and one non-VoIP service scenario (click to fax using PINT) have been chosen.

CALL HOLD

The supplementary service Call Hold allows the holding party A to interrupt the communication to party B during an active call. The signaling association between the two parties is not terminated. There are two basic variants of Call Hold: near end hold and Remote End Hold (in SIP: Far End Hold). With near end hold, the bearer channels remain open, but they no longer carry voice/video data from user A. Instead, they can be used to transmit media on hold (MoH), which is, for example, an announcement, a melody, or a video clip. The second form is Remote End Hold, where the communication channels are idle during the hold condition. Endpoint B may play MoH to user B locally. Remote end hold is described in the example scenario. Using call retrieve procedures, the communication channels can be activated again and the call returns to the active condition.

SIP — The following example shows the SIP realization of a Far End Hold scenario. SIP-compliant client/server implementations are assumed in the participating endpoints. Since proxy servers do not play a role in this example, they are not shown in the scenario. Figure 5 shows the Far End Hold scenario. For completeness, the Basic Call setup has been included in the message flows. SIP only keeps state for one transaction, e.g., for INVITE. Even ACK is considered to be a separate transaction. Timers are not shown in Fig. 5.

There is no explicit message (SIP method or header) defined for the request of a Call Hold supplementary service. Therefore, one has to rely on the basic SIP transaction mechanism. In order to put user B on hold, user A re-invites user B (second INVITE message) according to RFC 3264 [29], offering an updated session description. The media stream in the session description (SDP) is set inactive. This indicates to endpoint B that A stops sending media streams. Another re-INVITE from user A containing the original address parameter lets user B send again. This is basically the result expected from Call Hold. However, if we think of the supplementary Call Hold, endpoint B must detect the request of the Call Hold supplementary service from within the SDP carried as SIP payload, which could be ambiguous.

This ambiguous supplementary service activation may cause severe implementation problems in systems, which must be aware of supplementary services due to interoperability and user expectations. It remains to the developers of the SIP user agents to recognize the "setting of some media streams inactive" as a Call Hold request. There is no explicit feature invocation since the method name (INVITE) and the headers are the same as for a Basic Call. Thus, the same message with identical parameters may cause different reactions by the receiver depending on the actual context of the session.

The problem of supplementary service awareness is illustrated as follows. The status of user B being a held party and user A having put somebody on hold must be remembered to insure robust call processing. It remains to the application to remember in which features it is involved. The application may be the (extended) SIP state machine or the user himself (or a higher-layer application). Another open issues is to detect for the application when to play music on hold. The author in [23] only show an example where the holding party explicitly invites a music server to play music to the held party.

The problem of feature (non-) awareness is even worse in combination with other features or in a more complex context, where there might be situations where Call Hold should not be carried out or at least should be processed carefully. Imagine being involved in a tightly coupled conference, controlled by a conference server. The correct function of the Call Hold service will depend on the feature awareness of the conference server. Since this is not signaled explicitly, the whole conference group may be set idle or even worse in case

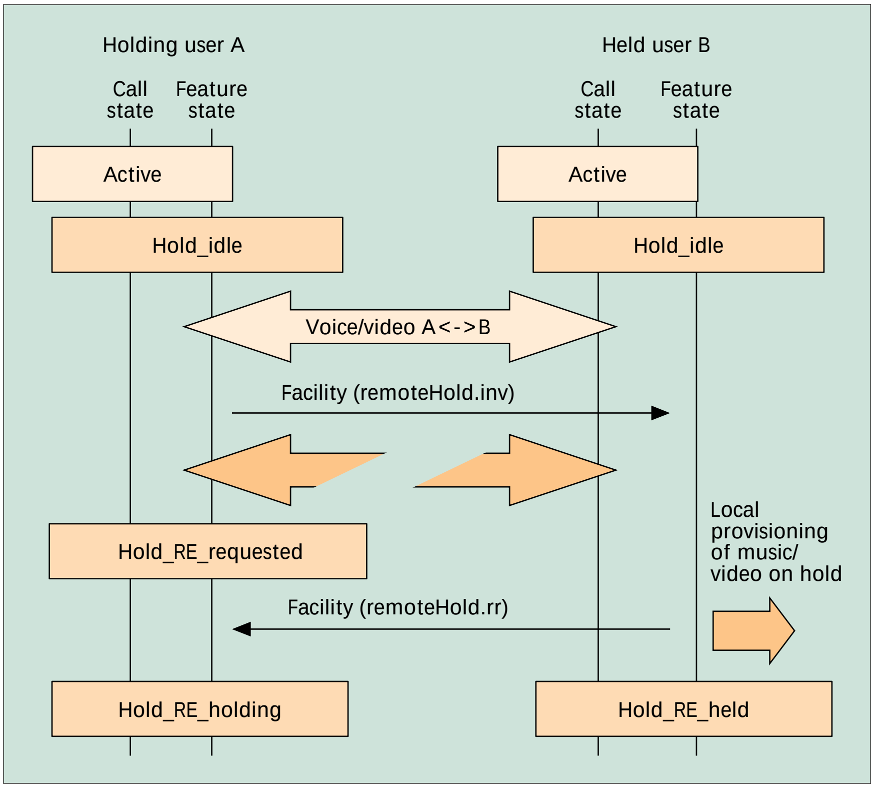


FIGURE 6. *H.323/H.450.4: remote end hold.*

of near end hold music being played to all parties. A typical feature interaction situation occurs. The technical functionality of each entity is correct, but the user expectations for the result of a supplementary service are different. This is especially true when interworking with legacy telephone systems.

H.323/H.450 — The following scenario describes the supplementary service Call Hold in H.323/H.450 as standardized in H.450.4 [30]. The examples assume the fully distributed model with H.450 implementation in the participating endpoints. The GK is transparent in this case and is therefore not shown in the diagrams.

Figure 6 shows the Remote End Hold scenario in H.323/H.450. The two vertical lines represent the state machines for Basic Call and the feature state machine of each endpoint for the Call Hold feature.

In the beginning the served user A has an active call with user B. User A pushes, for example, a hold button on his endpoint, which results in a FACILITY message containing a remoteHold.inv operation. This clearly identifies a Remote End Hold. At the same time endpoint A interrupts the existing media (voice, video, …) from/to B. No bandwidth is consumed any longer. The feature state of A changes to Hold_RE_Requested to remember locally that a remote hold has been initiated. The Basic Call state machine does not have to be changed when introducing the supplementary service, since the feature state machine defines the states and actions. Further, feature interaction is facilitated since all involved endpoints can determine, by looking at their feature states, that they are in a hold condition and whether new events and actions lead to unwanted interactions with other features.

Upon reception of the remoteHold.inv operation, B checks whether he can support Remote End Hold. In the positive case, endpoint B interrupts the media channels also and provides local media on hold to the user. The confirmation of this action is signaled to A in a FACILITY message.

If B did not support the operation Remote End Hold, the rejection would be signaled to A. Note that B needs no support of H.450.4 to reject the request. The Generic Functions H.450.1 contain a generic mechanism that indicates the required actions when a requested operation is not supported (e.g., ignore, reject). Even if Remote End Hold is supported by endpoint B, there might be situations where the hold operation should be rejected (e.g., if B is in a conference). This can be signaled including the respective reject cause.

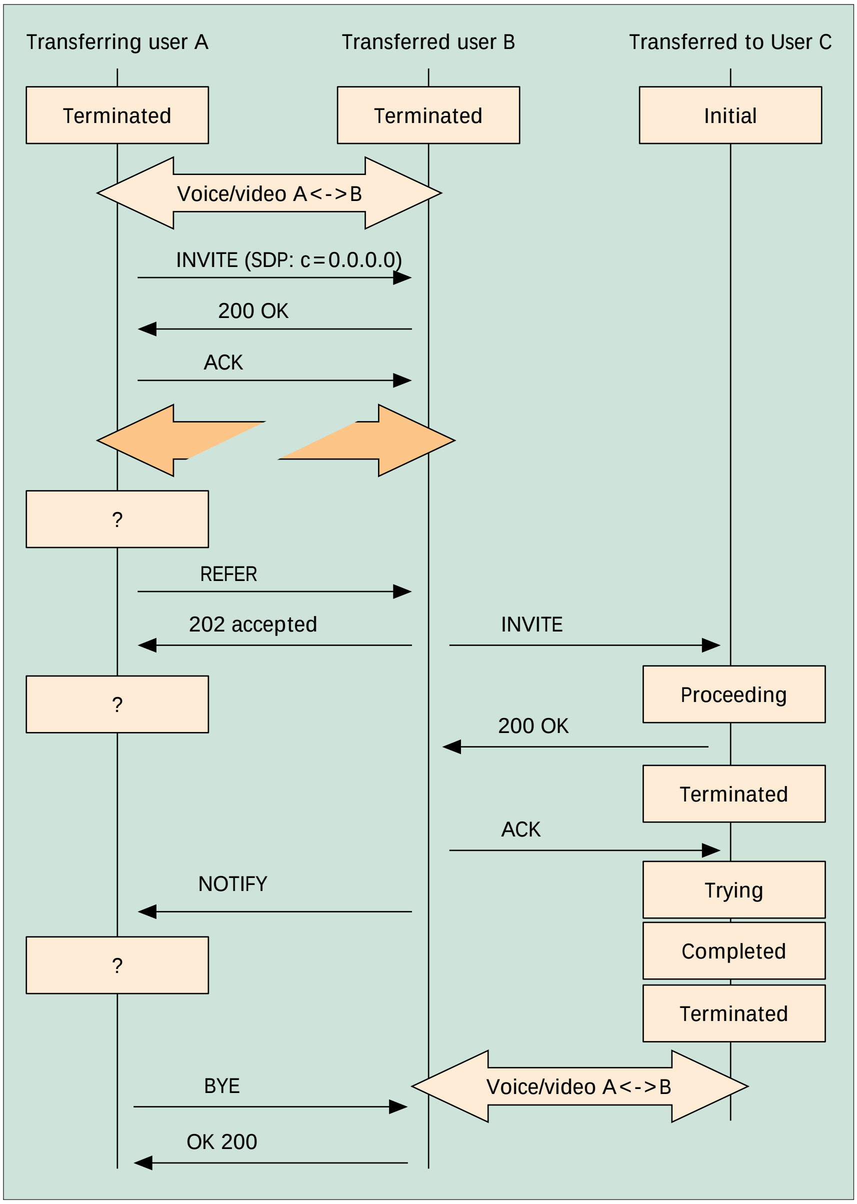
CALL TRANSFER

The supplementary service Call Transfer allows a transferring user A to transfer an active call with transferred user B to a transferred-to user C. The outcome of the actions is that the previous call between A and B is cleared and a new call between B and C is in the active state. There are two basic forms of Call Transfer: Single Step Transfer (or Unattended Transfer) and Multi Step Transfer (or Consultation Transfer). The Single Step Transfer is described in the example.

SIP — The method REFER [11] is suggested to be used in SIP for Call Transfer. Figure 7 shows a sample message flow for an unattended (single-step) transfer starting from an active call between A and B. A initiates the Call Transfer by putting the transferred user B on hold. When B accepts Call Hold, A initiates the transfer procedure by sending a REFER request with C's address to B. This indicates that B should invite C and issue a success response ('200 OK') to the originator, in this case user A. On reception of a success response, A terminates its signaling relationship with B issuing a BYE request. A SIP entity simply returns an error message ('501 not implemented') to the initiator if the REFER method is not supported.

Similar to the previous example and consistent with the SIP standardization philosophy, the use of REFER does not determine a Call Transfer supplementary service call. It could be used, therefore, as shown in the example flow, but for a supplementary service call that is interoperable with traditional telephony supplementary services important information for the implementation of this feature is missing. In Fig. 7 those points are marked where call states must change in order to provide appropriate message handling. In our opinion, the absence of a finite state machine in the standards might lead to interoperability problems with different implementations. However, even if these new states are added to the Basic Call state machine, the Basic Call would become more complex by adding more features and scalability problems might result.

RFC3261 states the functionality of the basic SIP protocol as a simple transaction protocol rather than a stateful call control protocol. All transactions (e.g., INVITE, BYE, and even ACK) of one session are performed independently without keeping context between them. The task of state-keeping is thus clearly left to the application program. As already discussed, this could cause problems for interworking between supplementary services.



■ **FIGURE 7.** *SIP: single-step transfer.*

H.323/H.450 — Figure 8 shows a Single Step Transfer according to H.450.2. Starting from an active call with endpoint B, endpoint A initiates the transfer by sending a FACILITY message with the ctInitiate.Invoke operation. This operation contains the transferred-to address of party C. In the case of a multistep transfer, it would contain information about the call identity of the consultation call, which is required to associate the two calls at endpoint C.

Upon receiving the ctInitiate.Invoke operation, endpoint B starts a new call with party C using a SETUP message with a ctSetup.Invoke operation. This new call may inherit the media capabilities of the call with A or negotiate the media capabilities from scratch. The ctSetup.Invoke operation contains information about the transferring user A (transferringNumber), which can be displayed to the user or examined by other features and applications. It may also be required for call admission and/or billing purposes. The H.450.1 GF would indicate to ignore the ctSetup.Invoke operation if it is not supported by endpoint C. This would result in a normal call setup taking place even if endpoint C did not support H.450.2.

After an ALERTING message containing a ctSetup.rr operation is received, endpoint B disconnects the first call with endpoint A. The user B will hear a ringing tone until user C goes off hook. Up to that point the intermediate feature states in A, B, and C make it possible to rollback the Call Transfer and restore the original call between A and B if anything fails. After the CONNECT message is received from endpoint C the communication between B and C is established.

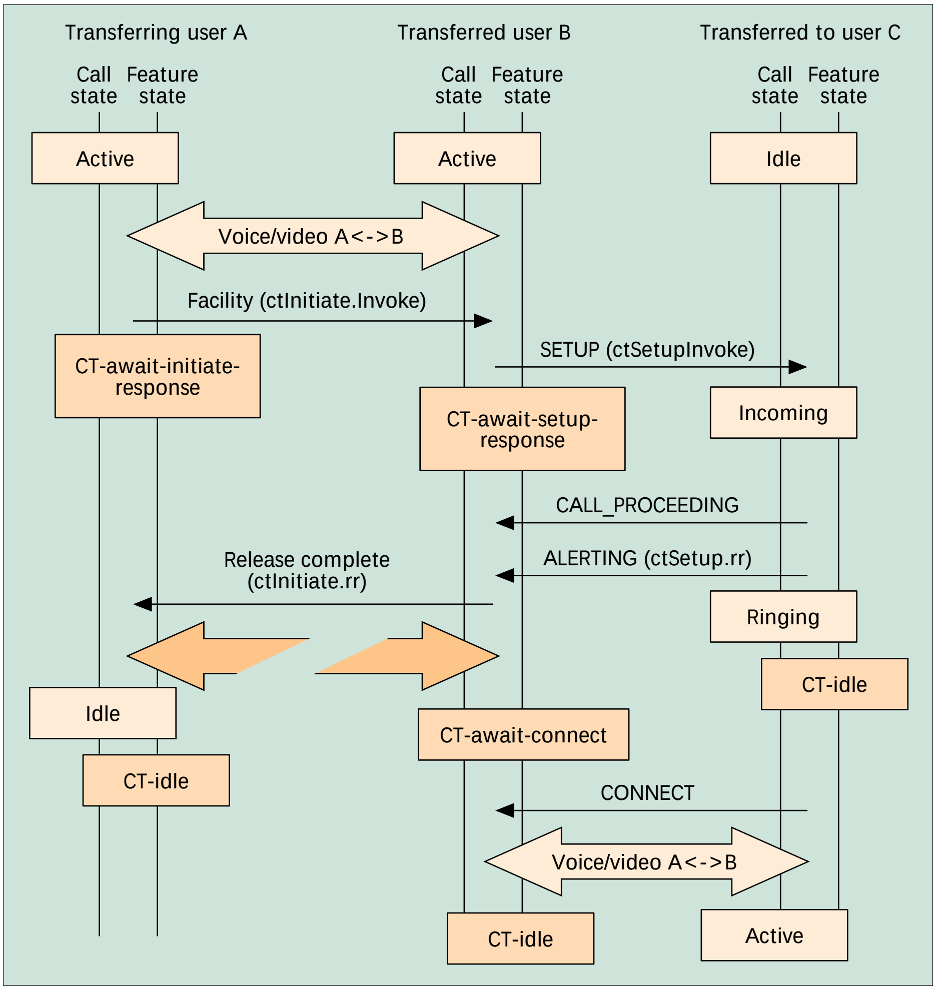
NON-VOIP FEATURE EXAMPLE

SIP — In addition to the invitation of parties to participate in a multimedia session, SIP makes it possible to initiate sessions that go beyond VoIP and are not bound to a specific media. Even non IP-based sessions, e.g., a PSTN-call, may be invoked by SIP. To illustrate this open character of SIP regarding session initiation, the PINT service protocol is explained in the following paragraph. Other applications are discussed and standardized in the SIMPLE, SPIRITS, and SIPPING working groups.

The PINT protocol (PSTN/Internet Interworking, RFC 2848 [29]) uses SIP and SDP for the invocation of telephony services in the GSTN from an IP network. For this purpose, a PINT server will be set up in the telephony network. All SIP extensions specified in PINT are in line with the SIP baseline behavior. The SIP INVITE message is used as a transport container for the assured exchange of service control information (e.g., a GSTN service description) between a PINT user (SIP client) and a PINT gateway (SIP server). The PINT gateway relays the request to a specific GSTN network control component and the latter performs the requested GSTN telephony service. Examples of service scenarios are “click to dial” or “click to fax back.”

Whereas the PINT user applies SIP to invite a remote PINT server into a session, the particular description of the telephone network session is carried as a SDP payload in the INVITE message body. SDP has been enhanced with additional parameters for the support of new network types (e.g., ISDN, GSM), new media types (e.g., fax, image), and format-specific attribute tags. The SDP session description is transparent for the SIP INVITE transaction and only the PINT gateway knows how to process. Figure 9 shows an example message flow for a “request to fax content” service.

H.323 — Although H.323 has put its focus on multimedia and voice services, it can also provide non-VoIP services. In the following, an approach using H.323 Annex K is sketched that provides functionality similar to the SIP/PINT example. With Annex K it is possible to construct very simple endpoints for service control, containing only parts of the RAS protocol and an HTTP client (e.g., Web browser). Upon RAS registration with the GK, the endpoint receives a URL of the Faxback GW to contact for the service control session. Using HTTP, a selection of service options is presented to the user. When the user requests an action (e.g., Click to Faxback) this is again



■ **FIGURE 8.** *H.323/H.450: single-step transfer.*

transferred to the Faxback GW using HTTP. The Faxback GW then carries out the respective actions toward the PSTN network (e.g., by using IN call control, as in the PINT approach).

Compared to the SIP/PINT example, it becomes clear that the function split is different. Whereas SIP provides a transaction protocol to transmit a session description, in H.323 Annex K the transaction handling must be programmed in the HTML (or other means) description of the service (Fig. 10).

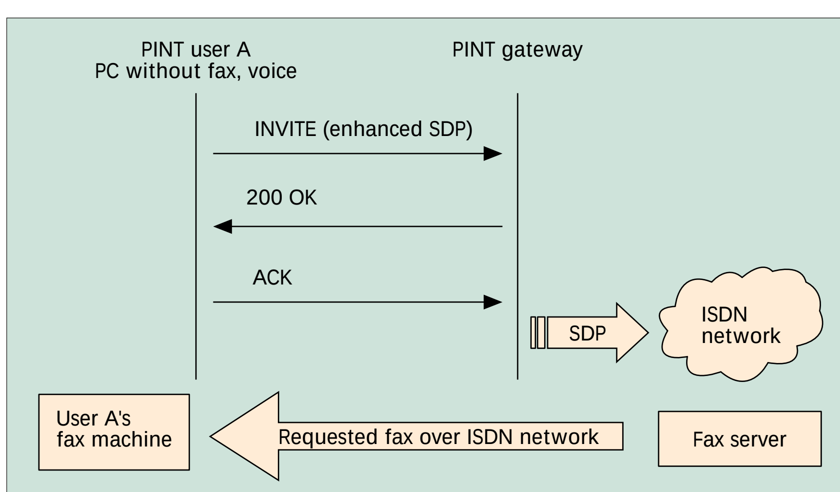
CONCLUSION

This tutorial has presented an overview of the two VoIP standards, H.323 and SIP, focusing on their service architectures and the mechanisms to develop and deploy services. Although both protocols may be used for VoIP applications, their original focus is very different. While SIP has been designed as a generic transaction protocol for session initiation not bound to any specific media such as audio or video, the focus of H.323 has been to handle voice and multimedia calls, including supplementary services.

Regarding standardization of call control and especially supplementary services, SIP still shows some shortcomings regarding the standardization of supplementary services. Such standardization is of importance for interoperability with traditional telephone systems. Meanwhile, the IETF has identified need in their scope regarding telephony-related requirements and services such as SIP-H.323 interworking, SIP-PSTN/IN interworking, or supplementary services. As a consequence, SIP is going to be extended in order to keep up with the functionality of H.323.

Comparing the service architectures and the resulting consequences for feature implementations, H.323 describes and enables an object-oriented approach based on QSIG, separating supplementary services from Basic Call control. Current standardization activities around SIP reveal that the IETF extends SIP's Basic Call to control a variety of applications, including supplementary services, but avoiding the introduction of feature-specific syntax or semantics. In the case of supplementary services, the absence of an explicit labeling of features may lead to migration, interoperability, and feature interaction problems. In conclusion, H.323 provides better functionality, interoperability, and interworking (PSTN and PBX) with respect to supplementary services.

On the other hand, SIP's broader scope forms the basis for a wider range of possible applications. H.323 was not targeted to support non-VoIP services in the beginning. However, there are some extensions proposed to the standard that go in that direction, such as H.323 Annex K, providing some limited support. To summarize, SIP provides more effective mechanisms for controlling non-VoIP services than H.323. In addition, SIP has advantages with respect to the design of low cost non-voice terminals due to its modular and flexible protocol



■ **FIGURE 9.** *SIP: click to fax with PINT.*

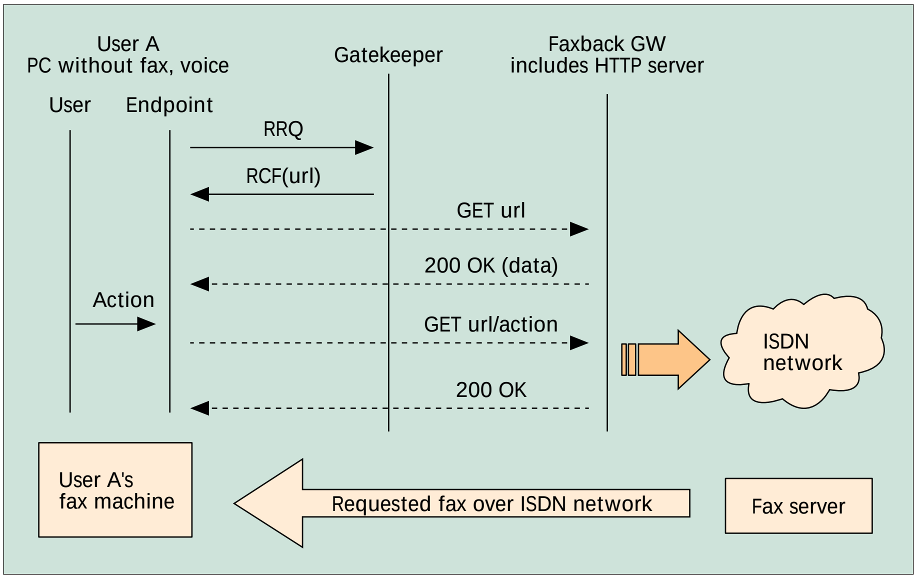


FIGURE 10. *H.323: click to fax using H.323 annex K.*

design. A SIP client, for example, does not have to support media processing capabilities, protocols such as SDP, or procedures such as capability exchange, if it is not required by the application. Therefore, very lightweight SIP clients can be built to control non-VoIP services. SIP has strengths for lightweight and easily implemented solutions with a focus on flexible session initiation.

With the focus set on voice over IP with supplementary services that are largely interoperable, H.323 has its advantages. These include replacement scenarios for legacy PBXs, but is especially true when IP telephony supplements and coexists with legacy telephone systems. Although this may sound like a typical application for enterprise scenarios, the trend of outsourcing applications, e.g., application service provider (ASP) solutions, can also be observed for IP telephony. Thus, supplementary services and H.323 become more important for carrier implementations.

Although the two standards are approaching each other, their focus and applicability is still different. It is not expected that one of the two protocols will dominate over the other. They will probably coexist in different environments and implementations over a longer time, which will also place a strong requirement on interworking between them.

REFERENCES

The most important readings are [3, 7] for H.323 and [9, 11, 31] for SIP.

[1] H. Schulzrinne and J. Rosenberg, "A Comparison of SIP and H.323 for Internet Telephony," *Proc. of NOSSDAV*, Cambridge, U.K., Jul 1998.
[2] R. Glitho, "Advanced Service Architectures for Internet Telephony: A Critical Overview," *IEEE Network Mag.*, Jul/Aug 2000, pp. 38–44.
[3] ITU-T Recommendation H.323v4, "Packet-Based Multimedia Communications Systems," 2000.
[4] ITU-T Recommendation H.225v4, "Packet-Based Multimedia Communications Systems," 2000.
[5] ITU-T Recommendation H.245v8, "Packet-Based Multimedia Communications Systems," 2001.
[6] ITU-T Recommendation H.320, "Narrowband Visual Telephone Systems and Terminal Equipment," 1993.
[7] ITU-T Recommendation H.221, "Frame Structure for a 64 to 1920 kb/s Channel in Audiovisual Teleservices," 1999.
[8] ITU-T Recommendation H.242, "System for Establishing Communication between Audiovisual Terminals Using Digital Channels up to 2 Mb/s," 1999.
[9] ITU-T Recommendation H.332, "H.323 Extended for Loosely Coupled Conferences," 1998.
[10] ITU-T Recommendation H.450.1, "Generic Functional Protocol for the Support of Supplementary Services in H.323," 1998.
[11] J. Rosenberg *et al.*, "SIP: Session Initiation Protocol," IETF Request For Comments RFC 3261, Jun 2002. Obsoletes RFC 2543.
[12] M. Handley and V. Jacobson, "SDP: Session Description Protocol," IETF Request for Comments RFC 2327, Apr 1998.
[13] The Internet Engineering Task Force (IETF): Working Groups; http://www.ietf.org/html.charters/.
[14] J. Rosenberg and H. Schulzrinne, Guidelines for Authors of SIP Extensions, IETF Internet Draft, draft-ietf-sip-guidelines-06.txt, Nov. 2002. Work in Progress.
[15] B. Campbell, "SIP Call Control Framework," IETF Internet Draft, draft-ietf-sip-cc-framework-00.txt, Jul 2001 — expired; work in progress.
[16] A. Roach, "Session Initiation Protocol (SIP) — Specific Event Notification," IETF Request For Comments RFC 3265.
[17] J. Rosenberg, "The Session Initiation Protocol (SIP) UPDATE Method," IETF Request For Comments RFC 3311.
[18] J. Rosenberg and H. Schulzrinne, "Reliability of Provisional Responses in the Session Initiation Protocol (SIP)," IETF Request For Comments RFC 3262.
[19] G. Camarillo, W. Marshall, and J. Rosenberg, "Integration of Resource Management and Session Initiation Protocol (SIP)," IETF Request For Comments RFC 3312.
[20] A. Mankin *et al.*, "Change Process for the Session Initiation Protocol (SIP)," IETF Request for Comments RFC 3427, Dec 2002.
[21] H. Schulzrinne and J. Rosenberg, "The Session Initiation Protocol: Internet-Centric Signaling," *IEEE Commun. Mag.*, Oct. 2000, pp. 134–41.
[22] ITU-T Recommendation H.450.12, "Common Information Additional Network Feature for H.323," 2001.
[23] A. Johnston *et al.*, "SIP Service Examples," Internet Draft draft-ietf-sip-service-examples-04.txt, Nov 2002; work in progress.
[24] R. Sparks and A. Johnston, "SIP Call Control — Transfer," IETF Internet Draft, draft-ietf-sipping-cc-transfer-00.txt, Oct 2002; work in progress.
[25] S. Petrack and L. Conroy, "The PINT Service Protocol: Extensions to SIP and SDP for IP Access to Telephone Call Service," IETF RFC 2848, Jun 2000.
[26] ITU Recommendation H.248, Megaco Protocol Version 1.0. IETF Request for Comments RFC 3015, Nov 2000.
[27] J. Lennox and H. Schulzrinne. "Feature Interaction in Internet Telephony," M. Calder and E. Magill (Eds.), Feature Interaction in Telecommunications and Software Systems VI, IOS Press, 2000.
[28] H. Schulzrinne and J. Rosenberg, "Session Initiation Prototcol (SIP) Caller Preferences and Callee Capabilities," IETF Internet Draft, draft-ietf-sip-callerprefs-07.txt, Nov 2002, work in progress.
[29] J. Rosenberg and H. Schulzrinne, "An Offer/Answer Model with the Session Description Protocol (SDP)," IETF Request For Comments RFC 3264, Jun 2002.
[30] ITU-T Recommendation H.450.4, "Call Hold Supplementary Service for H.323," 1999.
[31] M. Korpi and V. Kumar, "Supplementary Services in the H.323 IP Telephony Network," *IEEE Communications Magazine*, Jul 1999, p.118–25.

ADDITIONAL READING

[1] ITU-T Recommendation H.450.2, "Call Transfer Supplementary Service for H.323," 1998.
[2] ITU-T Recommendation H.450.3, "Call Diversion Supplementary Service for H.323," 1998.
[3] ITU-T Recommendation H.450.5, "Call Park and Call Pickup Supplementary Services for H.323," 1999.
[4] ITU-T Recommendation H.450.6, "Call Waiting Supplementary Services for H.323," 1999.
[5] ITU-T Recommendation H.450.7, "Message Waiting Indication Supplementary Service for H.323," 1999.
[6] ITU-T Recommendation H.450.8, "Name Identification Supplementary Service for H.323," 2000.
[7] ITU-T Recommendation H.450.9, "Call Completion Supplementary Services for H.323," 2000.
[8] ITU-T Recommendation H.450.10, "Call Offer Supplementary Service for H.323," 2001.
[9] ITU-T Recommendation H.450.11, "Call Intrusion Supplementary Service for H.323," 2001.
[10] R. Sparks, "The Refer Method," IETF Internet Draft, draft-ietf-sip-refer-03.txt, Jul 2002; work in progress.

BIOGRAPHIES

JOSEF GLASMANN (Glasmann@ei.tum.de) received his diploma degree (Dipl.-Ing. Univ) in electrical engineering and information technology from the Munich University of Technology (TUM), Germany, in 1996. Since October 1996 he has been a member of the research and teaching staff at the Institute of Communication Networks (Prof. J. Eberspächer) at TUM. His research topics are QoS in IP networks and multimedia service architectures. Currently he is working on his doctoral thesis, which is about the design and implementation of a resource management architecture for VoIP in corporate networks.

WOLFGANG KELLERER (Kellerer@docomolab-euro.com) heads the service platform and middleware technology group at NTT DoCoMo's European Research labs in Munich, Germany. He received a Dipl.-Ing. degree in electrical engineering and information technology and a Ph.D. from the Munich University of Technology (TUM) in 1995 and 2002, respectively. Before he joined DoCoMo he worked at the Institute of Communication Networks at TUM. In 2001 he was a visiting researcher at the Information System Laboratory of Stanford University. He is a member of the IEEE Communications Society and the IEEE Computer Society.

HARALD MUELLER (H.Mueller@siemens.com) received his diploma degree (Dipl.-Ing.) in 1990 from the Darmstadt Technical University, and his doctor's degree (Dr.-Ing.) from the Munich University of Technology, Germany. He is with Siemens AG, Germany, where he has been working in different positions within product definition for IP-based communication systems. Currently, he is responsible for the development of applications for real-time communication systems. His current areas of interest include signaling and applications for IP-based and converged communication systems.