

Project Number: **4007 (RE)**

Project Title: **Multimedia Education and Conferencing Collaboration over ATM
Networks and Others (MECCANO)**

Deliverable Type: (PU/LI/RP)* **PU**

Deliverable Number: **D7.1**

Contractual Date of Delivery: **31 March 1999**

Title of Deliverable: **The Session Announcement and Management Facilities in
MECCANO Release 1.**

Work-Packages contributing to the Deliverable: **7**

Nature of the Deliverable: (PR/RE/SP/TO/OT)** **TO/RE**

Authors: **Colin Perkins, Jörg Ott & Dirk Kutscher**

Abstract:

This document describes the initial work by MECCANO on session announcement and management facilities. We describe the standards work undertaken by project members, the protocols under development, and the demonstrator applications we are delivering. The major protocol development with MECCANO WP7 is related to the Mbus, although we are also working on SDP, SIP and SAP. The demonstrators are SDR, RAT, FreePhone and AudioGate.

Keyword list:

SDP, SAP, SIP, Mbus, RAT, AudioGate.

* Type: PU - public, LI - limited, RP - restricted

** Nature: PR - Prototype, RE - Report, SP - Specification, TO - Tool, OT - Other

Contents

Introduction

Review of standards activities

- IETF
- ITU-T

Protocol Development

- Session Description Protocol
- Session Initiation Protocol
- Session Announcement Protocol
- Message Bus

Demonstrator Applications

References

Introduction

The MECCANO project calls for activity in three areas of conference control: setup of conferences, control of active conferences and setup & control of conference relays. We have produced a number of protocols and demonstrators which cover the full range of these activities, concluding the work begun in the MERCI [1] project and beginning the investigation of new areas.

In brief, since the start of the MECCANO project we have continued to be heavily involved in the standards process (both IETF and ITU-T) contributing to the development of SDP, SAP and SIP and proposing new protocols for local conference coordination (the Mbus). In addition, we have implemented a number of demonstrator applications and infrastructure components which we expect to use during the later stages of the project.

This deliverable is structured in three sections: a review of the standards activities undertaken by project members in the field of conference control, a description of the protocols we are adopting for our work and our involvement in their development and a description of the demonstrator applications we have produced.

This deliverable is a companion to MECCANO deliverable 6.1 and describes the background protocols used throughout MECCANO workpackages 6 and 7. It is expected that D6.1 will be read in conjunction with this deliverable.

Review of standards activities

IETF

There are a number of IETF working groups which conduct work related to multimedia conferencing and the MECCANO project. The audio/video transport working group, co-chaired by Colin Perkins of UCL, is responsible for media transport whilst the multiparty multimedia session control working group, co-chaired by Jörg Ott of the University of Bremen, handles conference control. In addition, the media gateway control working group is working on the decomposition of telephony gateways.

The multimedia conferencing framework adopted by the IETF is that of the Internet multimedia conferencing architecture [2]. This is a product of the multiparty multimedia session control working group, with major input from MECCANO partners.

Media transport is outside the scope of this deliverable, but it is important to note that both the IETF and ITU-T have adopted the real-time transport protocol, RTP [3], as their transport. The importance of this will become clear later when we discuss media gateways.

The multiparty multimedia session control working group has developed a number of protocols for the description, announcement and initiation of conferences. Of these, the session description [4] and initiation [5] protocols were initially developed under the previous MERCI project and have recently been published as proposed standards. The announcement protocol [6] was also initially developed under the MERCI project and is now under development by MECCANO partners. Recent work in this area has included a specification for authenticated announcements [7] and use with IPv6 [8].

The multiparty multimedia session control working group is also considering a proposal from the MECCANO project for a local conference coordination bus which provides an infrastructure for the decomposition of a conferencing endpoint.

The media gateway control working group is defining an architecture and protocol framework for the decomposition of gateways between telephone networks and voice over IP systems. Project members are not directly involved in this work at present, but are closely following it since it is of obvious relevance to the local coordination work being done in the multiparty multimedia session control working group.

ITU-T

The ITU-T Working Group of most relevance to multimedia communications and conferencing is ITU-T SG16, also referred to as Study Group for Multimedia.

The most active groups ("Questions") within SG16 with respect to multimedia communications are Questions 11-15: Q.11 deals with ISDN and PSTN-based multimedia communications, Q.12 with ATM, and Q.15 with advanced video coding schemes (particularly including error resilience schemes as in the 1998 revision of H.263 (also referred to as "H.263+"). Q.13 and Q.14 deal with the H.323 series of Recommendations. H.323 is currently considered the standard for controlling multimedia communications in small groups as well as for IP Telephony. The core Recommendations include H.323 (system and procedures), H.225.0 (message formats, encodings), and H.245 (capability/media descriptions, handling of media streams). H.235 covers

security mechanisms for H.323, the H.450.x standards address Supplementary Services for multimedia communications, and H.341 defines Management Information Bases. Other recommendations address remote device control and text conversation (transport and semantics in both cases). H.323 interfaces well to the loosely-coupled conferencing concept prevalent in the IETF through H.332 that defines how to set up H.323 conferences, map their parameters to SAP/SDP, and announce the conferences on the Mbone. Recent developments include the design of H.GCP -- related to and increasingly coordinated with the IETF MEGACO WG -- and H.225.0 Annex G as well as a variety of Annexes for more efficient communication procedures and simpler endpoints. The functional range of H.GCP which is likely to impact the design of the Mbus semantics layer for call signaling and control in gateways. H.225.0 Annex G covers inter-domain exchange of addressing information and (potentially) other characteristics of specific devices (such as gateways) -- which, again, may impact the design of the MECCANO gateway.

Finally, Question 3 in SG16 has developed the T.120 series of Recommendations used for data conferencing across arbitrary networks including the Internet. The T.120 infrastructure provides a platform that creates a multipoint communication environment from a set of point-to-point connections by ordering them in a tree structure and offers a rich set of conference control functions. Also, use of native multicast networks is supported. On top of this infrastructure, T.120 application protocols provide means for telecollaboration through shared whiteboards, file transfer, application sharing, and text chat. Recent developments include support for T.120-specific security features and the design of a semantic meeting room model (including different conferencing styles, roles, associated privileges, etc.). Most of today's (commercial) shared workspaces and particularly application sharing systems are based on the T.120 series of Recommendations: these include Microsoft NetMeeting and Intel ProShare among others.

Within MECCANO, possibilities are evaluated to take control of the Microsoft NetMeeting application engine through its well-defined API and offer an Mbus interface. However, this is still under study.

Protocol Development

Session Description Protocol

The session description protocol became an IETF proposed standard in April 1998. The protocol has been stable for the duration of the MECCANO project and partners have a number of implementations (for example within the session directory, delivered in D4.2). The limitations of SDP are beginning to become clear and it is possible that a successor will be developed during the lifetime of MECCANO. We plan to be involved in any such development.

Session Initiation Protocol

The session initiation protocol, SIP, became an IETF proposed standard in March 1999. As noted previously, the protocol was initially developed as part of the MERCI project, and MECCANO partners have kept a close interest in its further development.

We have two client implementations of SIP within the project: the Session Directory and the FreePhone audio tool. In addition, a SIP server is planned as an outgrowth of the message bus work being undertaken.

Session Announcement Protocol

Overview

The session announcement protocol has been under development for some considerable time. The basic mechanisms are well understood and widely implemented but need extension in a number of areas: authenticated and private announcements, support for IPv6 and integration with the world-wide-web. We have been working on these extensions for some time now, producing a number of internet-drafts and making presentations at most recent IETF meetings. We have recently taken over editorship of the SAP specification with a view to merging these extensions into the base document, leading to a combined specification.

Authenticated Announcements

It is necessary to provide authentication and integrity of a session announcements in order to ensure that only authorised persons can modify announcements and to ensure that announcements come from the person claimed. Since subsequent announcements will modify caches of future conferences, it is possible otherwise to spoof an original announcement, and thereby at least cause a denial of service attack.

In order to send authenticated announcements it is possible to use either the PGP or the PKCS#7 algorithms. For each format, the announcement originator calculates a message digest of the announcement. The originator's secret key is used to encrypt the message digest, together with an electronic timestamp, thus forming a digital signature. The originator sends the digital signature along with the message; the receiver receives the message and the digital signature, and recovers the original message digest from the digital signature by decrypting it with the sender's public key. The receiver computes a new message digest from the message, and checks to see if it matches the one recovered from the digital signature. If it matches, then this is considered adequate proof that the message was not altered, and that it came from the originator who owns the public key used to check the signature.

Each session announcement contains a message ID hash. The specifications for SAP announcements state that such announcements may be repeated frequently, but that if any change is made in the announcement, a different message ID must be used; as a result, a different message ID hash will be appended to the message. As a result, it is only necessary to authenticate an announcement the first time it is received.

To save space in the announcement message, only a public key identifier is generally included. It is then assumed that the public key itself has either been distributed previously or can be retrieved from a cache or directory. Optionally the public key itself can be included in the announcement in the form of a certificate removing the need for prior distribution. Consequently, providing that the public key is already available in a local cache or directory, or is distributed with the announcement, one can be sure that the same originator sent the announcement. Only if the full public key information, and a certificate authority Infrastructure, are accessible, can the originator be identified.

Our most recent specification for authenticated announcements following this procedure was presented at the 42nd IETF meeting in Chicago. It was well received, and the consensus of the group was that this should become part of the base SAP specification when that is next revised.

Support for authenticated announcements is included in the session directory tool.

Private Announcements

It has also been suggested that it may be desirable to allow for private session announcements. For example, if a session announcement contains encryption keys for the media streams it is clearly undesirable to send the announcement as clear-text. Alternatively, closed user groups may wish to announce sessions which only their members can receive.

Private announcements can be achieved by encrypting the body of an announcement. The body is encrypted using a symmetric algorithm, such as DES, and a privacy header is prepended to it. This privacy header contains the decryption key for the message body. The privacy header is itself encrypted using an asymmetric algorithm, for example PGP. The key to this asymmetric algorithm is distributed in a similar manner to that for authenticated announcements.

We presented a specification for this at the 42nd IETF meeting in Chicago, along with our authentication work, and it was also discussed at the 43rd IETF meeting in Orlando. Some concern was raised regarding the need to support private announcements (since they may be considered an inappropriate use of wide area bandwidth - an applicability statement will be needed) and regarding the use of PGP (it is unclear if the PGP system lends itself well to this role, even though the basic idea is sound).

Consensus was finally reached that private announcements were useful in some cases, and that symmetric encryption should be specified initially (whilst more study is made into the suitability of PGP). We will therefore be merging these features into the next revision of the base SAP specification.

Support for private announcements is included in the session directory tool.

Support for IPv6

In conjunction with ISI-East and Microsoft Research, we have produced a specification for IPv6 support in SAP. This was presented at the 43rd IETF meeting in Orlando, where it was accepted for inclusion into the next revision of the base SAP specification.

As part of this collaboration, a version of the session directory with IPv6 support was produced by Maryann P. Maher at ISI and made available with the Microsoft Research IPv6 stack [9]. We expect to merge these changes back into our release of the session directory.

Integration with the WWW

In certain cases, it is desirable to reference a SAP announcement. For example, if it is desired that a new participant join an existing session yet it is not known if that participant is within the scope of the announcement then an explicit reference to the announcement will enable them to determine if it can be received. Providing the session description contained within that announcement merely allows them to join the session, when they then notice that the media streams are not visible. Moreover, the addresses contained in a session description for one scope may not be valid outside that scope zone.

We have therefore defined a URL scheme for SAP announcements, allowing for simple and effective integration with the world wide web. The combination of msg-id-hash and originating-

source fields in the SAP header is sufficient to identify a particular announcement.

This scheme was presented at the 43rd IETF meeting in Orlando, along with the IPv6 support, as an experimental addition to SAP.

Message Bus

Goals

The Message Bus (Mbus) infrastructure has been designed to simplify development of complex communication systems intended to enable and augment interactive human-to-human (tele)cooperation. Such system may include (typically workstation-based) user terminals (as designed in WP4) as well as various types of interworking units (WP6) and other management entities (partially developed in the context of WP7 but to a large extent beyond the scope of the MECCANO project).

In our abstract system concept, the functionality of any conferencing system considered in MECCANO can be broken into a variety of components. These include but are not limited to media engines, user interfaces, conference control, and modules providing administrative mechanisms. Depending on the type of conference and the type of system, some or all these components may need to act in a closely coordinated fashion. Individual tools may be logically and "physically" separated into their respective engines and user interfaces which need to communicate to convey user actions and system responses back and forth. Systems may also have various media engines controlled by a single user interface and thus require mechanisms to simultaneously control these engines. Also, media engines may need to coordinate themselves e.g. to synchronize audio playback with presentation of video streams to achieve lip synchronization. Finally, in tightly coupled conferences or IP telephone calls, dedicated entities may perform conference control functions and may need to closely interact with other system components to make them adapt their behaviour according to the conference state. Altogether, local coordination is needed to achieve a coherent system behaviour in response to user actions as well as interaction with other conferencing systems.

While horizontal control protocols (such as SCCP) are intended to synchronize state between communicating systems (inter-system protocol), vertical control protocols serve intra-system coordination. The Message Bus shall provide an infrastructure for vertical coordination based upon mechanisms for inter-process communication (IPC). It is designed to satisfy the following needs:

- support a modular system design;
- simplify using building blocks from different sources through well-defined interfaces;
- enable efficient independent development and testing;
- allow for independence of programming languages;
- maximize re-usability of components in different systems as well as different system types;
- support separation of engines from user interface;

- enable easy system extensibility (at run time);
- not to prescribe system design or mandate certain components;
- allow for efficient and low-overhead communication;
- be robust against partial system failures; and
- support monitoring of system functions for debugging.

The Mbus infrastructure logically consists of two components: a transport infrastructure that provides message transfer, addressing, and basic bootstrap as well as awareness mechanisms and a semantic layer that is defined through the abstract services of the communicating modules. Within the MECCANO project, semantic layers is/will be defined for call and rudimentary conference control services, control of selected media engines and their user interfaces, and for simple user preferences/policies.

Systems to be built based upon the Mbus infrastructure include multimedia terminals and various types of (multimedia) gateways.

Mbus Transport Services

As basic service, the Message Bus provides local (intra-system) exchange of messages between components that attach to the Mbus (Mbus entities). A local system is typically expected to comprise exactly a single host, but a local system may also extend across a network link and include several hosts sharing the tasks; a local system must not extend beyond a single link. Message exchange takes place using UDP datagrams; i.e. the message size is limited to 64k including Mbus headers, but it is suggested to use only messages smaller than the link MTU. UDP datagrams are either sent via unicast to a single Mbus entity or are multicast using host-local or link-local scope (depending on how far the system reaches).

For point-to-point communications, message delivery is optionally performed reliably, i.e. acknowledgements and retransmissions take place at the Mbus transport layer. All multicast communication is performed unreliably. Point-to-point and multicast communication is not distinguished by the transmission mechanism employed at the IP layer but rather by the qualification of the Mbus destination address (briefly described below).

In addition to basic message transmission, the Mbus transport provides mechanisms for Mbus entities to automatically determine the availability and the address other Mbus entities as well as an entity's failure. Based upon these functions, a bootstrap procedure is defined that allows Mbus entities to determine whether all other entities they depend on are present (without bearing the risk of deadlocks).

It should be noted that Mbus entities are logical components. In particular, a process or a thread may represent an arbitrary number of Mbus entities (which may even communicate with one another via the Mbus).

A key concept of the Mbus is its flexible and extensible naming and addressing scheme. Mbus entities are identified by n-tuples. Each tuple component is represented by an (attribute:value) peer. Currently defined attributes include conference ("conf"), media ("media"), module type ("module"), application name ("app"), and application instance ("instance"). Individual or all

tuple components may be omitted or wildcarded ("*") to implement broadcast, multicast, and anycast services. Full qualification of an address (i.e. non-wildcarded presence of all defined components) denotes an Mbus unicast address.

Furthermore, the Mbus provides message authentication and encryption as inherent transport features. Message authentication is needed to prevent malicious entities from taking control or at least disturbing a user's system but also to prevent accidental interpretation of other users' message, e.g. in case multicast transport addresses of two users accidentally match and their transmission scopes overlap. As the Mbus may also be used to convey personal information or keying material (IVs, keys) between Mbus entities, the messages also need to be encrypted to prevent other entities from eavesdropping.

A set of user-specific resources (specified in a file, a registry, etc.) contains all the user-specific configuration information for all Mbus entities representing this user. The resource information includes user id, authentication/encryption keys, multicast address / port for message exchange among others.

These facilities combine to allow the Mbus to support multiple sessions of a user per host (including cross-session coordination) as well as any number of users on the same host or link (preventing accidental cross-user interaction).

Semantic Concept

As stated above, various component types that attach to the Mbus to form an integrated system can be identified. The aforementioned rather intuitive overview can be formalized to yield a finer granularity of component types that is sufficient to build all kinds of systems within the scope of the MECCANO project. The following (logical) component types are defined:

- (media) engines providing the functionality necessary for telecooperation (such as audio or video communications, shared workspace or editor, etc.);
- control (protocol) engines managing interactions with remote users (e.g. providing means for call/conference setup, floor control, mutual awareness in a conference, etc.);
- graphical user interfaces (GUIs) as suitable means for a human user to access the system functionality implemented by the (control and media) engines;
- policy modules that control automated system behaviour (e.g. based upon user preferences, administrative settings, call/conference processing scripts, etc.);
- applets that provide an abstraction from specific implementations of background/backend services (such as address resolution, certificate validation, user authentication, directory services, etc.); and
- an Mbus controller that may combine any number of the aforementioned modules and add specific interpretation/processing to create a particular integrated system type.

Particular instantiations of these (abstract) components are implemented within MECCANO to build the complex components to be delivered. Control protocols engines being implemented include an H.323 engine and a SIP engine as well as a module to communicate via ISDN lines. Media engines include at least the Robust Audio Tool (separate engine and user interface) with

an Mbus interface. Various user interfaces for integrated terminals are likely to be developed. Finally, Mbus controllers are designed to combine the aforementioned components to form two important deliverables:

- A future revision of AudioGate will consist of a media engine (from the Robust Audio Tool), a simple SDP parser, an ISDN call signaling module and some further applets all held together by a specific AudioGate Mbus controller.
- The multiway call signaling gateway StarGate (also *Gate) will be comprised of any number of call signaling protocol engines, policy and address resolution modules, and a specific StarGate Mbus controller to appropriately combine the functions of the other modules.

The following section provide a brief status report on the Mbus specifications and implementation themselves as well as on components based upon the Mbus.

Status

An Mbus architecture framework document is in progress and will be submitted as an internet-draft for discussion in the multiparty multimedia session control working group of the IETF.

The Mbus transport specification is essentially complete and has two implementations: at UCL and at Uni Bremen. Interoperability testing of these implementations is in progress. A previous version of this has been submitted as an internet-draft, we expect to do the same with the finished specification.

The basic Mbus operation and bootstrap procedures have been defined, together with specific commands for control of audio and RTP related functions. Basic call signaling has been defined, with specific commands for H.323, SIP, and ISDN lines to come. An internet draft is being written to document this.

Initial versions of these documents are provided as part of this deliverable:

- Requirements for a Coordination Infrastructure for Conferencing Systems
- A Message Bus for Conferencing Systems
- The Message Bus: Messages and Procedures

Please note that these are **draft** versions (PostScript) and are subject to change before submission as internet-drafts.

Future Work

We plan to build a number of demonstrator applications based on the Mbus architecture. An outline follows, although the details are subject to change based on experience:

- **Robust-Audio Tool** The current implementation of RAT uses the Mbus internally for communication between media engine and user interface. We plan to extend RAT with transcoder functionality, and to provide an Mbus interface to allow control of this.
- **ReLaTe integrated interface** This is an alternate interface to RAT which also

incorporates the video tool, vic. The Mbus will be used to provide coordination between the RAT media engine and the user interface, including lip-sync, voice-switched video and individual power meter information.

- **UCL Transcoding Gateway** The transcoder implementation in RAT forms part of the UCL transcoding gateway. This currently uses a number of ad-hoc control protocols to drive the individual components, and may be extended to use the Mbus in future.
- **AudioGate with Mbus** A future revision of AudioGate will consist of a media engine (from the Robust Audio Tool), a simple SDP parser, an ISDN call signaling module and some further applets all held together by a specific AudioGate Mbus controller.
- **StarGate** The multiway call signaling gateway will be comprised of any number of call signaling protocol engines, policy and address resolution modules, and a specific StarGate Mbus controller to appropriately combine the functions of the other modules.

This work will involve extension of the Mbus semantics to further cover basic call/gateway control, including protocol specific extensions, floor control, dynamic control of media session, etc.

We will also consider the work in the IETF MEGACO working group, ITU and TIPHON; and other conference coordination work by the MATES project and Henning Schulzrinne's Pattern Matching Multicast, for example. We are working on an expanded architecture document, to contrast our work with these other approaches.

Demonstrator Applications

In this section we describe the demonstrator applications produced as part of WP7, highlighting the features relevant to this workpackage. The tools fall into two classes:

- Demonstrators of the base session description, announcement and initiation protocols, such as the session directory (SDP, SIP client, SAP including private/authenticated announcements and IPv6 support) and FreePhone (SIP client).
- Demonstrators of the modular design and message bus protocol, such as the AudioGate and Robust-Audio Tool.

Since the latter design allows for considerably more flexibility in composition than the earlier monolithic approach we plan to focus our future work in WP7 on this architecture. This is not to say that the other tools will be abandoned, but we do expect to see a shift in emphasis over the remainder of the project.

We are delivering four applications (see Deliverable 4.2) along with this report:

- **Session Directory v2.6.1**

The session directory is designed to allow advertisement of and joining to multicast conferences using the session announcement protocol, SAP. This version supports authenticated and encrypted announcements using PGP, and the session initiation protocol, SIP. Sdr was initially developed at UCL, and has significant contributions from ISI, Xerox PARC and Lawrence Berkeley Labs.

- **FreePhone v3.7beta1.**

FreePhone is a voice-over-IP application, supporting a number of advanced features for resilience to adverse network conditions. The major contributions of FreePhone are to do with media transport, however this version also implements the session initiation protocol, SIP, and so is also included in this deliverable. FreePhone was developed at INRIA Sophia Antipolis.

- **Robust-Audio Tool v4.0.0**

RAT is a voice-over-IP application, supporting a number of advanced features for resilience to adverse network conditions. This version has a number of additional features compared to the previous release, and these are described in detail in D4.2.

Of interest to us in this deliverable is that this version of RAT contains an initial implementation of the Mbus. This is built as a standalone library, suitable for integration with other tools in the future.

This release of RAT comprises two Mbus entities: the media engine and the user interface which communicate entirely using the Mbus. We also have an alternative user interface in development, which also includes video with lip-sync, to illustrate the composability of the media tools built from our Mbus-based architecture.

The Mbus-enabled version of RAT also forms the basis of the audio gateway tool, described below.

- **Audio Gateway v0.1**

This is the initial release of AudioGate, an ISDN Mbone Gateway, jointly developed by TELES and UB TZI. This initial version is primarily intended as a proof of concept for the following key components: usage of rat as audio engine for conversion from line-oriented to packet-oriented audio and vice versa as well as realizing audio communications with ISDN boards in Linux PCs and integrating this facility into the multimedia communication environment developed by the MECCANO project.

AudioGate provides a dial-in interface that allows users to call a phone number and automatically be transferred into a preselected Mbone session. AudioGate uses an ISDN BRI to connect to the phone network and currently allows one voice call (to the same conference). If a Calling Line Identification Presentation (CLIP) service is supported by the caller and the ISDN network, the caller's phone number is provided in the SDES item to identify the person on the phone to the other parties in the Mbone conference.

Version 0.1 of AudioGate is a minimal workable prototype that does not yet exploit the Mbus as core coordination tool between system components and hence intentionally provides only limited functionality. The focus for this version of AudioGate was the "low level" integration of ISDN call signaling and device handling. This tool will gradually be revised to include all the concepts for Mbus-based gateways developed in the MECCANO system architecture (and thus use the same platform as an H.323-SIP, H.323-to-ISDN, or SIP-to-ISDN gateway). With this evolution, AudioGate will also gain functionality and provide a variety of additional services for the caller from the telephone network - such as dynamic conference selection, muting/push-to-talk, etc. - which are

expected to be mostly controlled through touch tone signaling.

These tools are also included as part of deliverable D4.2 and are more fully described there.

References

1. MERCI project Home Page
<http://www-mice.cs.ucl.ac.uk/multimedia/projects/merci>
2. The Internet Multimedia Conferencing Architecture
draft-ietf-mmusic-confarch-00, July 1997
3. RTP: A Transport Protocol for Real-Time Applications
IETF rfc1889
4. SDP: Session Description Protocol
IETF rfc 2327
5. SIP: Session Invitation Protocol
IETF rfc 2543
6. SAP: Session Announcement Protocol
draft-ietf-mmusic-sap-00, November 1996
7. SAP Security Using Public Key Algorithms
draft-ietf-mmusic-sap-sec-04, September 1998
8. Session Announcement Protocol: Version 2
draft-ietf-mmusic-sap-v2.00, May 1998
9. Microsoft Research IPv6 stack
<http://www.research.microsoft.com/msripv6/>