A Mobility-aware Broadcasting Infrastructure for a Wireless Internet with Hotspots

Cristian Hesselman, Henk Eertink
Telematica Instituut
The Netherlands
{cristian.hesselman, henk.eertink}@telin.nl

Ing Widya, Erik Huizer
University of Twente
The Netherlands
{widya, huizer}@cs.utwente.nl

ABSTRACT

In this paper, we consider the problem of adaptively delivering live multimedia broadcasts (e.g., for applications such as TV, radio, or e-cinema) to a potentially large number of mobile hosts that roam about in a wireless internet with hotspots. We take a user-oriented approach based on an application-level delivery infrastructure consisting of and managed by (value-added) service providers. The service providers are mobility-aware and offer broadcasts in configurations that are optimized for wireless links and mobile hosts. In hotspots, mobile hosts may be able to simultaneously reach several localized service providers through different interfaces. Within this context, we present the design of a lightweight application-level protocol that enables mobile hosts to select a service provider from which they want to receive a broadcast. Mobile hosts use the protocol to begin receiving a broadcast and to remain connected to the same logical broadcast as they move across subnets. The protocol is independent of the actual stream control protocol (e.g., RTSP) that service providers might use. We show how our protocol can be realized with the existing protocols SIP and SDP. The realization uses SIP in combination with SDP's offer-answer model in a new way.

Categories and Subject Descriptors

C.2.1 [**Computer Communication Networks**]: Network Architecture and Design – *network communications, wireless communication;* C.2.2 [**Computer Communication Networks**]: Network Protocols – *applications*.

General Terms

Algorithms, Design

Keywords

Multimedia broadcasting, negotiation, hotspots/overlays, mobility

WMASH'03, September 19, 2003, San Diego, California, USA.
Copyright 2003 ACM 1-58113-768-0/03/0009…$5.00.

1. INTRODUCTION

In this paper, we consider the problem of adaptively delivering live multimedia broadcasts (e.g., TV, radio, or e-cinema broadcasts) to a potentially large number of mobile hosts that roam about in a wireless internet with hotspots [1-4]. Our approach is user-oriented and is based on a managed multi-domain application-level delivery infrastructure that is mobility-aware and offers broadcasts in configurations that are optimized for wireless links and mobile hosts ('mobile friendly' configurations).

We first discuss how we model such delivery infrastructures in Section 1.1. We then consider the contributions of our work in Section 1.2 and provide a comparison with related work in Section 1.3. We conclude with an overview of the rest of the paper in Section 1.4.

1.1 Approach

A key requirement for our model is that it must be flexible enough to cover the most important multimedia streaming architectures: (1) end-to-end Internet architectures in which access to services (in our case, broadcasting services that support 'mobile friendly' configurations) and Internet access are completely decoupled; (2) architectures in which users have trust relationships with value-added service providers that aggregate broadcasts and transparently offer them to their users in a 'mobile friendly' manner; (3) architectures in which users have trust relationships with content providers that offer broadcasts to their users through transparent value-added service providers; and (4) walled garden architectures often used in telco environments in which access to (value-added) services is strongly coupled with network access.

The main difference between architectures 2 and 3 lies in the selection of 'content channels' (e.g., "BBC World News"). Architecture 2 assumes that the value-added service provider has a directory of available channels out of which a user can choose. This is similar to a traditional cable TV network, but generalized to arbitrary (mobile) networks. Architecture 3 is a 'mobile friendly' version of Internet content distribution networks: the user directly accesses the source of a channel (e.g., at cnn.com), and is then re-routed [5] over a content distribution network to an available 'mobile friendly' media server that is 'closest' to the user (e.g., http://nearest-server.akamai.com/CO231234).

1.1.1 Roles

To be able to model architectures 1 through 4, we distinguish three roles [6]: origin streamers, access streamers, and ISPs.

An origin streamer acts as the original source of one or more broadcast channels. An origin streamer may simulcast each channel using various encodings and bandwidth levels.

An access streamer receives broadcast channels from origin streamers and can deliver them to mobile users in a 'mobile friendly' manner in the form of a (typically relatively small) set of broadcast configurations. We define a configuration (cf. [7, 8]) as a bundle of streams of well-formatted RTP packets [9] of a certain bandwidth that contain encoded multimedia information (e.g., a 64 kbps MP3 audio configuration, a 32 kbps G722.1 audio configuration, a 24 kbps G722.1 audio configuration, and so on). Each access streamer supports its own set of configurations. In this paper, we consider these to be a subset of a universal ordered list of configurations. In addition to being quality-aware, an access streamer is also mobility-aware in that it can deal with the changing IP addresses of mobile hosts (also see ISP below). An access streamer may furthermore be mobility-aware in that it is aware of the notions of home and foreign access streamers (see Section 1.1.2). An access streamer uses a pool of media servers to stream broadcast channels to mobile users. It may use these servers to provide value-added services for origin streamers by increasing the available set of configurations for its customers, for instance by transcoding high-bandwidth HDTV streams to a low bandwidth format (e.g., [2, 8, 10, 11]).

An Internet Service Provider (ISP) provides IP connectivity to mobile hosts at various bandwidth levels. To limit the complexity of our model, we assume that ISPs provide a best-effort service. We furthermore do not require an ISP to support a network-level mobility handling solution such as Mobile IP [12].

In this way, we clearly distinguish roles that concentrate on serving mobile hosts (access streamers and ISPs). In addition, we clearly distinguish application-level roles (streamers) from network-level roles (ISPs). Another reason for introducing access streamers is that they can be used to perform application-level multicasting in an environment where the end-to-end availability of IP multicast is limited [13].

In this paper, we concentrate on type 2 architectures (users have agreements with access streamers that aggregate streams coming from origin streamers) and exclude type 3 architectures. To be able to cover type 4 architectures, we assume that ISPs have set up agreements with access streamers. Other architectures can be covered using a simplification of the work presented in this paper (e.g., type 1 architectures can be covered by combining the roles of an origin and an access streamer in one domain and by removing the access streamer-ISP agreements).

Figure 1 shows an instance of our model. In this example, user Bob has a subscription with access streamer AS1 and uses a multi-mode mobile host with an 802.11b and a UMTS interface. For simplicity, we have omitted origin streamers from the picture.

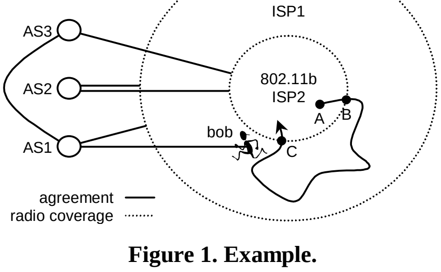


Figure 1. Example.

1.1.2 Agreements

A configuration completely defines the quality level that the user will perceive. In our model, each access streamer 'markets' its configurations by associating them with a user-oriented quality label (e.g., "CD" quality audio). We therefore define a user's agreement with an access streamer in terms of the quality levels (i.e., labels) the user has access to (typically in combination with a pricing model) [6].

The availability of an access streamer's services may be restricted to users that connect to ISPs with which the access streamer has set up an agreement (e.g., in Figure 1, AS1 offers its services to hosts that connect to ISP1). This means that the services of such access streamers are local to the involved ISPs, which may require users to switch from one access streamer to the other when they roam across ISPs (e.g., from AS1 to AS3 when Bob roams from ISP1 into ISP2). Since a user has an agreement with only one access streamer (his home access streamer), this means that access streamers need to set up application-level roaming agreements [6] amongst each other (e.g., between AS1 and AS3). In our model, application-level roaming agreements declare which configurations users of the home access streamer can use from the foreign access streamer.

A switch from one access streamer to another involves a handoff to a media server of the target access streamer [32] and may involve a handoff between subnets (e.g., between the UMTS and 802.11b subnets of Figure 1). The latter may be subject to a handoff policy [3, 4]. Mechanisms that move a mobile host to another media server and handoff policies are however outside the scope of this paper.

Our model also covers agreements between origin streamers and access streamers (e.g., to describe if and how access streamers are allowed to manipulate streams), between users and their home ISPs (subscriptions to get network access), and roaming agreements between ISPs (to get access to networks of foreign ISPs) [6]. They are however out of the scope of this paper.

1.2 Contributions

In this paper, we consider the design of a lightweight application-level protocol that enables mobile hosts to select the access streamer that provides the "best" configuration out of N alternative access streamers (e.g., at point A, Bob's mobile host must select among configurations of AS1 and AS3). Mobile hosts use the protocol to begin receiving a broadcast (e.g., at point A in Figure 1) and to remain connected to the same logical broadcast as they move across subnets (e.g., at points B and C in Figure 1). The protocol is independent of the actual stream control protocol (e.g., RTSP [14], WindowsMedia, and so on) that access streamers might use on their media servers.

The contributions of this paper consist of the design of the application-level protocol and a realization of it using the Session Initiation Protocol (SIP) [15] and the Session Description Protocol (SDP) [16]. The realization uses SIP with SDP's offer-answer capabilities [17] in a multiparty fashion to select the access streamer that provides the "best" configuration. This differs from [17] where SDP's offer-answer capabilities are used between two parties to select a codec. The streams of the "best" configuration will be set up in a later phase (e.g., using RTSP).

1.3 Related Work

Many application-level infrastructures for delivering multimedia streams to (mobile) Internet hosts have been proposed in the literature (e.g., [11, 13, 18, 19]). The work that comes closest to ours is that of Dutta et al. [20]. Their MarconiNet system also uses a managed environment with agreements between administrative authorities and also considers mobility handling in combination with quality issues. However, their model is different in that it does not use autonomous intermediary (our access streamers) that each support their own set of configurations (quality levels). As a result, their model does not include roaming agreements between intermediaries and their system does not support a protocol that enables mobile hosts to select a "best" intermediary (access streamer). We furthermore take a user-oriented approach and do not assume a particular mobility handling technique such as MarconiNet hosts hopping between localized IP multicast groups (though our model allows it). Finally, we provide a more complete and detailed view on agreements and explicitly consider multi-mode mobile hosts. We do not cover security issues, which Dutta et al. do.

1.4 Overview

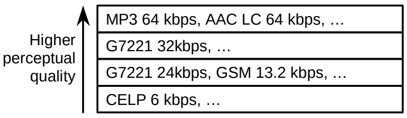
The rest of this paper is organized as follows. In Section 2, we first explain how we describe configurations and agreements using SDP. Next, we discuss our application-level protocol and how it uses/relates to the descriptions of Section 2. In Section 3, we consider the features that the protocol must posses; in Section 4 the part of the protocol that allows a mobile host to begin receiving a broadcast; in Section 5 the part that allows a mobile host to remain connected to the same logical broadcast as it moves across subnets; and in Section 6 a realization of our protocol using SIP and SDP. We end the paper with conclusions and an outlook on our future work in Section 7.

2. DESCRIPTIONS

To keep as close to the existing state of the art as possible, we use the Session Description Protocol (SDP) [16] to describe broadcast configurations.

2.1.1 *Universal List of Configurations*

An entry in the universal list of configurations consists of a set of configurations that provide approximately the same perceptual quality level. The entries in the list are ordered according to the perceptual quality the configurations in each entry provide. The top most entry contains configurations that provide the highest quality level; the configurations in the bottom entry provide the lowest quality. Figure 2 shows a (hypothetical) example for audio configurations in which the 'quality spectrum' is divided in four categories.



MP3 64 kbps, AAC LC 64 kbps, …
G7221 32kbps, …
G7221 24kbps, GSM 13.2 kbps, …
CELP 6 kbps, …

Figure 2. Example of universal list of configurations.

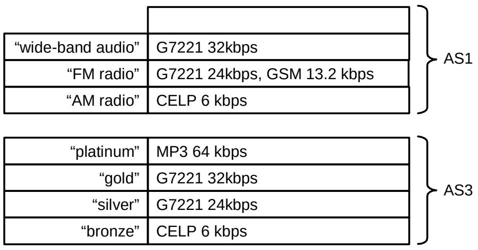
2.1.2 *Configurations of Access streamers*

Each access streamer supports a subset of the configurations in the universal list and associates some or all of them with the broadcast channels it offers. Access streamer AS1 (see Figure 1) could for instance support the configurations of the last three entries of the universal list, while AS3 could support one configuration of each entry (say the first ones). In SDP, AS1 can describe its alternative configurations like this:

```
m=audio 0 RTP/AVP 96 97 98
a=rtpmap:96 G7221/16000
a=fmtp:96 bitrate=32000
a=fmtp:96 bitrate=24000
a=rtpmap:97 GSM/8000
a=fmtp:97 bitrate=13200
a=rtpmap:98 MP4A/LATM/8000
a=fmtp:98 bitrate=6000
```

Each `rtpmap` and `fmtp` line together define a configuration. We note that the bitrate parameters in the `fmtp` lines in the above example are for illustrative purposes only. In reality, these parameters are codec-specific. Also note that the payload types in the media line (`m=`) merely form alternatives and do not express an ordering as is normally the case in SDP. For simplicity, we omitted all other SDP lines other than `m=`.

As a running example, we assume that access streamer AS1 associates the quality labels "wide-band audio", "FM radio", and "AM radio" with its configurations (see Figure 3, top), while AS3 uses "platinum", "gold", "silver", and "bronze" for its configurations (see Figure 3, bottom). Figure 3 only shows the configurations of the universal list that AS1 and AS3 support.



		AS1
"wide-band audio"	G7221 32kbps	
"FM radio"	G7221 24kbps, GSM 13.2 kbps	
"AM radio"	CELP 6 kbps	
"platinum"	MP3 64 kbps	AS3
"gold"	G7221 32kbps	
"silver"	G7221 24kbps	
"bronze"	CELP 6 kbps	

Figure 3. Quality levels (labels) of AS1 and AS3.

2.1.3 *Subscriptions*

We express the agreement between a user and his home access streamer (a subscription) in terms of a subset of the quality levels (labels) of the home access streamer. As an example, we assume that Bob's subscription with AS1 contains all of AS1's labels, indicating that Bob is allowed to receive broadcasts from AS1 at any of the configurations that AS1 supports. We expect that the quality labels and the associated configurations that a user is allowed to use will be part of a user profile. In this paper, such a profile is stored on the mobile host the user is logged onto.

When Bob has selected a broadcast (e.g., the audio-only broadcast "BBC World News") from an electronic program guide, his mobile host submits SDP that includes the configurations Bob is allowed to use according to his subscription with his home access streamer (see sections 4, 5, and 6 for more details):

```
s=BBC World News
…
m=audio 0 RTP/AVP 96 97 98
a=rtpmap:96 G7221/16000
a=fmtp:96 bitrate=32000
a=fmtp:96 bitrate=24000
a=rtpmap:97 GSM/8000
a=fmtp:97 bitrate=13200
a=rtpmap:98 MP4A/LATM/8000
a=fmtp:98 bitrate=6000
```

2.1.4 *Roaming Agreements*

In our model, different networks may be served by different localized access streamers. Access streamers therefore set up application-level roaming agreements amongst each other so that their users can receive broadcasts in networks served by foreign access streamers. An application-level roaming agreement includes information on the configurations that users from the home access streamer can receive from the foreign access streamer. For example, the roaming agreement between AS1 and AS3 could specify that AS3 only offers its configurations of the lowest two categories to users of AS1. As a result, Bob's mobile host will only be able to use the G722.1 (24 kbps) and CELP (6 kbps) configurations of AS3 when he receives an audio broadcast from AS3. Using the universal list of configurations (see Section 2.1.1) and AS1's labels in Bob's user profile (see Section 2.1.3), Bob's mobile host will nonetheless display the quality labels of Bob's home access streamer ("FM radio" or "AM radio"). This is what makes our platform user-oriented.

3. PROTOCOL FEATURES

In Section 2, we considered the application-level agreements of our model in more detail and showed how we describe broadcast configurations using SDP. In this section, we discuss the requirements of an application-level protocol that makes use of this information to enable mobile hosts to seamlessly access multimedia broadcasts from arbitrary places within a wireless internet. The requirements for the protocol are that it must be fast, that it does not require access streamers to maintain a large amount of state, and that it has reasonable bandwidth consumption.

3.1.1 Rapid Operation

In our model, a mobile host may need to switch to another access streamer, configuration, subnet, or ISP in 'mid-call' (e.g., in Figure 1, when Bob moves from ISP1 to ISP2). To accomplish this in a reasonably smooth manner, the protocol must be fast. This means that the protocol must use a minimal number of round trips and that it should exploit parallelism where possible (e.g., send requests to multiple access streamers simultaneously rather than sequentially). The protocol should use UDP as a transport service for two reasons: (1) because we are designing a signaling protocol that does not require long-lived connections; and (2) because UDP messages can be sent right away without having to wait for a TCP connection to be established.

3.1.2 Minimize Access Streamer State

Internet design principles [21, 22] state that the amount of state that the network needs to maintain should be minimized to aid scalability. In our model, this means that our protocol should not require access streamers to maintain a large amount of state. For example, the protocol should not require multiple access streamers to temporarily allocate media processing resources (e.g., transcoders) for multiple configurations and hold these resources until the mobile host has decided which of these configuration it is going to use. If maintaining state is unavoidable, access streamers should use soft state rather than hard state. Minimizing the amount of state at access streamers moves most of the protocol state and functions to the hosts.

3.1.3 Reasonable Bandwidth Consumption

The bandwidth levels at the wireless edges of the Internet will always remain several orders of magnitude less than those in the backbone. The protocol must therefore be reasonably bandwidth efficient, but taking into account the relatively high bandwidth levels that streaming applications usually require. The signaling protocol should preferably use textual messages because they are easier to extend, process, and debug than binary encoded messages. At the same time, the use of UDP limits the amount of information we can transfer (e.g., an access streamer's set of available configurations) to the maximum size of a UDP packet.

4. INITIATION PROTOCOL

In this section, we discuss the part of our protocol that allows mobile hosts to begin receiving a broadcast (e.g., at point A in Figure 1). The protocol uses SDP to describe configurations and operates in the context of established agreements, the universal list of configurations, and the configurations supported by individual access streamers (see Section 2).

We discuss the requirements of the initiation protocol in Section 4.1, and cover its messages and behavior in Section 4.2.

4.1 Requirements

To begin receiving a broadcast the user has selected from an electronic program guide (e.g., at point A), a mobile host has to (1) discover the access streamers available on each of its network interfaces, (2) request access to the available access streamers, (3) select the access streamer that provides the "best" available configuration (a best of N selection amongst the access streamers available on the mobile host's network interfaces), and (4) connect to a media server of the access streamer that provides the "best" configuration to receive the broadcast.

4.1.1 Discovering Access Streamers

An ISP can inform an inbound mobile host of the access streamers it can use when the mobile host first connects to the ISP. For example, when Bob turns on his mobile host at point A, ISP1 can point the mobile host to AS1 and AS2 while ISP2 can point the mobile host to AS2 and AS3. An ISP would typically convey this information together with other information such as an IP address and the address of a DNS server, for instance using DHCP [23].

Mobile hosts also need to be able to discover if access streamers are going off-line or are coming on-line (e.g., because an ISP has only contracted an access streamer to provide its services during rush hour). This requires mobile hosts to regularly poll access streamers and set a time out on their responses, or it requires access streamers to push such events through an announcement protocol (cf. the announcement protocols (that are part of) SLP [24], SIP [25], and UPnP [26]). We will however not consider such protocols in this paper.

Without agreements between access streamers and ISPs, mobile hosts need to revert to other discovery techniques to figure out which access streamers they can use (e.g., using a well-known multicast group [27]).

4.1.2 Requesting Access

In our platform, access streamers control which users they admit. A foreign access streamer (e.g., AS3) admits users based on the existence of an application-level roaming agreement with the user's home access streamers (e.g., Bob's home access streamer AS1). If such an agreement exists, the foreign access streamer contacts the home access streamer to authenticate the user (e.g., AS3 contacts AS1 to authenticate Bob). Home access streamers admit their users based on local information and do not have to contact other access streamers.

Access streamers will typically communicate with each other through a AAA protocol like Diameter [28]. Such protocols are however outside the scope of our work.

4.1.3 Best of N Selection

A mobile host keeps track of the configurations available from the access streamers it has gained access to. The mobile host combines this information with the ordered universal list of configurations (see Section 2.1.1) to select the access streamer that provides the "best" available configuration. In our platform, "best" is determined by the policies of the user (which determine the "best" quality level) and the policies of the mobile host (which determine the "best" configuration per quality level). In this paper, we assume that the "best" quality level is the one that provides the highest perceptual quality. Other user-level aspects that can define "best" are power consumption and cost [29].

The set of configurations available from a particular access streamer depends on the user's subscription with his home access streamer, the roaming agreement between the access streamer and the home access streamer, the available resources on the access streamer's media servers, and the capabilities of the mobile host.

In our platform, an access streamer controls which users can use which of its configurations. A foreign access streamer first contacts a user's home access streamer to find out which of the home access streamer's configurations the user is allowed to use (typically as part of the authentication process). It then applies the application-level roaming agreement it has set up with the user's home access streamer to determine which of its own configurations the user is allowed to use. This set may differ from the set of configurations the user is allowed to use at his home access streamer. For example, to determine which of AS3's configurations Bob is allowed to use, AS3 contacts AS1 and applies the roaming agreement between AS3 an AS1 to the set of configurations Bob is allowed to use at AS1 (see Figure 3, top). While Bob can use the two G722.1 configurations, the GSM configuration, and the CELP configuration of AS1 (quality levels "wide-band audio", "FM radio", and "AM radio"), the roaming agreement between AS1 and AS3 declares that he can only use the CELP and 24 kbps G722.1 configurations of AS3 (see the example of Section 2.1.4). Home access streamers determine this information locally and do not have to contact other access streamers.

In our platform, an access streamer is responsible for keeping track of its available application-level resources (e.g., availability of transcoders that need to be activated to provide certain configurations, if any). These resources determine which of an access streamer's configurations are available.

The capabilities of the mobile host may rule out the use of certain configurations of an access streamer because the mobile host does not support the corresponding codec. For example, Bob might log onto a host that is not equipped with a G.7221 codec. This further restricts the set of configurations Bob can receive from AS1 to just the CELP configuration (see Figure 3). We assume that a mobile host stores its own capabilities.

4.1.4 Connect to Media Server

In our platform, mobile hosts are responsible for selecting a media server of the access streamer that can provide the "best" configuration. The media server will host the media processing resources (if any) to provide the "best" configuration.

4.2 Messages and Behavior

After the user has selected a broadcast channel, the initiation protocol uses two messages to begin receiving a broadcast: ADMISSION and CONNECT.

4.2.1 ADMISSION Requests

A mobile host first sends an ADMISSION request to each access streamer it has discovered on its interfaces. For efficiency, an ADMISSION request serves two purposes: to request access to an access streamer, and to find out which of an access streamer's configurations are currently available on which media servers. An ADMISSION request must therefore contain (1) the user's identity (e.g., Bob@AS1), (2) the user's credentials (e.g., username and password), (3) a timer value that the mobile host uses to propose an interval for refreshing the authentication state that the access streamer maintains (see Section 5), and (4) optional SDP that contains a subset of the configurations the user is allowed to use at his home access streamer.

The purpose of the SDP is to limit the amount of information conveyed between the mobile host and the access streamer if the user or the mobile host is only interested in a subset of the configurations the user is allowed to use at his home access streamer (e.g., if the user logs onto a mobile host without stereo capabilities, then it of no use to include stereo configurations in the ADMISSION request).

The ADMISSION request that Bob's mobile host sends to access streamer AS3 could then look like this:

```
User: Bob@AS1
Credentials: …
Refresh-Interval: 2 seconds
s=BBC World News
…
m=audio 0 RTP/AVP 96 97 98
a=rtpmap:96 G7221/16000
a=fmtp:96 bitrate=32000
a=fmtp:96 bitrate=24000
a=rtpmap:97 GSM/8000
a=fmtp:97 bitrate=13200
a=rtpmap:98 MP4A/LATM/8000
a=fmtp:98 bitrate=6000
a=inactive
```

The `a=inactive` line informs the access streamer that the SDP describes capabilities and that it should not begin to stream [17]. Unlike [17], we use this mechanism in a multiparty fashion because a mobile host generally sends ADMISSION requests to multiple access streamers. As far as we know, this multiparty use of `a=inactive` lines is new.

To speed up the initiation process, the mobile host should submit ADMISSION requests simultaneously.

4.2.2 ADMISSION Responses

An access streamer responds to an ADMISSION request with an ADMISSION response. An ADMISSION response contains (1) an indication of whether or not the request succeeded plus a reason if the request failed, (2) an admission token that the mobile host must use in further communications with the access streamer, (3) a timer value that indicates which interval the access streamer wants the mobile host to use for refreshing authentication state (see Section 5), and (4) SDP that specifies the configurations of this access streamer that are available to the user. The SDP also contains URIs (e.g., SIP or RTSP URIs) that point to media servers that can host the configurations.

The ADMISSION response that Bob's mobile host receives from AS3 could look like this:

```
Result: OK
Admission-Token: …
Refresh-Interval: 10 seconds
s=BBC World News
…
m=audio 0 RTP/AVP 96 98
a=rtpmap:96 G7221/16000
a=fmtp:96 bitrate=24000
a=sip:bbc.co.uk
a=rtsp://bbc.co.uk
a=rtpmap:98 MP4A/LATM/8000
a=fmtp:98 bitrate=6000
a=sip:bbc.co.uk
a=inactive
```

In this example, we have stored the URIs (SIP and RTSP URIs in this case) in SDP `a=` lines. They immediately follow the `rtpmap` and `fmtp` lines of a configuration.

The ADMISSION request-response procedure is similar to that of SDP's offer-answer model [17]. However, unlike [17] the configurations in an ADMISSION response may differ from those in the corresponding request. This is because a foreign access streamer might support a different set of configurations than a user's home access streamer.

If the response indicates that the request has failed, the admission token and the SDP must be empty. An ADMISSION request can fail because the access streamer has no roaming agreement with the user's home access streamer, because it does not support the requested broadcast channel, or for reasons internal to the access streamer serving the request.

4.2.3 CONNECT Requests

When the mobile host has determined which configuration it considers "best", it checks the ADMISSION response of the access streamer that provides the configuration, selects a URI, and sends a CONNECT request to the media server to which the URI points.

A CONNECT request is control protocol-specific. It could for instance map to a SIP INVITE [15], or to a sequence of RTSP messages consisting of a DESCRIBE followed by a SETUP followed by a PLAY [14]. In any case, a CONNECT request has to contain (1) the admission token, (2) the IP address of the interface through which the mobile host wants to receive the broadcast, and (3) SDP that describes the selected "best" configuration.

Assuming that Bob finds the configuration that provides the highest perceptual quality the best one, Bob's mobile host selects AS3's 24 kbps G722.1 configuration. The CONNECT request it transmits to AS3 then looks like this:

```
Admission-Token: …
s=BBC World News
…
c=IN IPv4 12.34.56.78
m=audio 10000 RTP/AVP 96 98
a=rtpmap:96 G7221/16000
a=fmtp:96 bitrate=24000
```

The absence of an `a=inactive` line indicates that the mobile host requests the access streamer to begin streaming. The port number in the `m=` line must be non-zero [17].

4.2.4 CONNECT Responses

An access streamer responds to a CONNECT request with a CONNECT response. A CONNECT response only needs to contain an indication of whether or not the request succeeded plus a reason if the request failed. A CONNECT response may contain additional control protocol-specific information (e.g., SIP information).

If the response indicates the request failed, the SDP must be empty. A CONNECT request can fail because of an invalid admission token or for reasons internal to the access streamer serving the request.

4.2.5 Example

Figure 4 shows a typical message exchange at point A of the example of Figure 1.

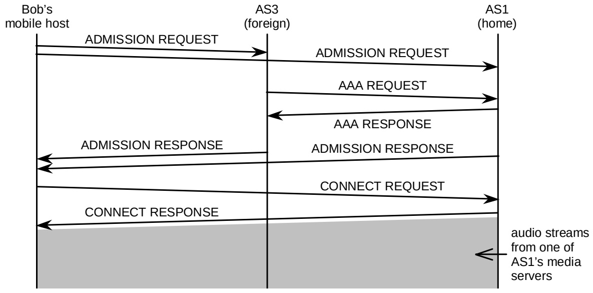


Figure 4. Example initiation behavior at point A.

5. ROAMING PROTOCOL

In this section, we discuss the part of our protocol that allows mobile hosts to continue to receive a broadcast while they move across subnets (e.g., at points C and D in Figure 1). The protocol uses SDP to describe configurations and operates in the context of established agreements, the universal list of configurations, and the configurations supported by individual access streamers (see Section 2).

We discuss the requirements of the roaming protocol in Section 5.1, and cover its messages and behavior in Section 5.2.

5.1 Requirements

While receiving a broadcast, a mobile host has to (1) refresh authentication state at available access streamers, and (2) handle the assignment of a new IP address to one of its interfaces. The result may be that the mobile host has to switch to a new "best" configuration, possibly involving another access streamer or another interface.

5.1.1 Refresh Authentication State

Similar to [28], access streamers should maintain authentication softstate on users they have admitted. This reduces the delay between requests and responses because an access streamer does not have to interact with the user's home access streamer (e.g., to reauthenticate the user) on a per-request basis. As a result, mobile hosts will be able to switch from one access streamer to another more quickly (see Section 5.1.2). Maintaining authentication state also reduces the load on the user's home access streamer, which benefits the scalability of the system.

In our platform, authentication state consists of a user ID (e.g., Bob@AS1) and the set of configurations the user is allowed to receive from his home access streamer.

The mobile host and the access streamer must be able to negotiate the refresh interval when the mobile host first requests access to the access streamer (see Section 4.2).

Refreshing the authentication state should allow a mobile host to detect changes in the set of available configurations at an access streamer in the absence of a protocol that announces such events (cf. the announcement protocols (that are part) of SLP [24], SIP [25], and UPnP [26]).

5.1.2 Mobility Handling

When a mobile host roams into a new subnet, new access streamers may become available. Some of these new access streamers might provide a configuration that is "better" than the one the mobile host currently uses. As a result, the mobile host may want to switch to a configuration of one of the new access streamers, possibly also involving a handoff to another subnet. A mobile host may also want to switch to a new "best" available configuration when the user roams out of a subnet.

When a mobile host assigns a new IP address to one of its interfaces, it has to discover the access streamers that are available on the subnet (see Section 4.1.1). In addition, it needs to request access to newly discovered access streamers (see Section 4.1.2) and go through best of N selection (see Section 4.1.3) to determine which access streamer currently provides the "best" configuration. When roaming, best of N selection involves detecting which configurations are available at newly discovered access streamers, as well as detecting which are available at access streamers that the mobile host discovered previously.

If the new "best" configuration belongs to another access streamer, the mobile host must disconnect from the access streamer it is currently using and connect to the new one. The SDP in the request to the old access streamer describes the configuration the mobile host was using; the request to the new access streamer specifies the new "best" configuration. The new "best" configuration may provide another perceptual quality than the old one (adaptation).

If the new "best" configuration belongs to the access streamer the mobile host was receiving the broadcast from, the mobile host has to reconnect to the same access streamer.

We assume that an access streamer can detect a disconnected mobile host (e.g., using RTCP [9]) and can then automatically clean up any resources the host was using (e.g., transcoders) in case the disconnect request does not reach the access streamer.

There are other events besides a change of IP address that can result in the mobile host switching to a new "best" configuration (e.g., an access streamer's configuration becoming unavailable), but we will not discuss them in this paper.

5.2 Messages and Behavior

The roaming protocol reuses the ADMISSION messages of the initiation protocol for newly discovered access streamers and the CONNECT messages to switch to another access streamer. For simplicity, we do not consider disconnect messages in detail and only note that they are control protocol-specific as well.

The roaming protocol requires two additional messages: AVAILABILITY and REFRESH.

5.2.1 AVAILABILITY Requests

A mobile host uses an AVAILABILITY request to determine the available configurations of an access streamer that it has discovered previously. The request has to contain (1) the admission token, and (2) optional 'inactive' SDP that contains a subset of the configurations the user is allowed to use at his home access streamer (cf. the SDP in an ADMISSION request).

To speed up best of N selection, a mobile host should submit ADMISSION and AVAILABILITY requests simultaneously. An ADMISSION request fails if the target access streamer does not offer the broadcast channel the mobile host is receiving.

5.2.2 AVAILABILITY Responses

An access streamer reacts to an AVAILABILITY request with an AVAILABILITY response. The response has to contain (1) an indication of whether or not the request succeeded plus a reason if the request failed, and (2) 'inactive' SDP that specifies which of this access streamer's configurations are available to the user. The SDP also contains URIs that point to media servers that can host the configurations (cf. the SDP in an ADMISSION response).

An AVAILABILITY request can fail because of an invalid admission token or for reasons internal to the access streamer serving the request.

A mobile host uses the ADMISSION and AVAILABILITY responses it receives to determine the new "best" configuration.

5.2.3 REFRESH Requests

A mobile host uses a REFRESH request to regularly refresh the soft authentication state at an access streamer. A REFRESH request must contain the same information as an AVAILABILITY request.

A mobile host must send REFRESH requests to the all the access streamers it has discovered, including to the one it is receiving the broadcast from. A mobile host must transmit a REFRESH request before the end of the refresh interval.

In this paper, we assume that mobile hosts and access streamers do not adjust the refresh interval they negotiated through the ADMISSION request-response pair. As a result, it is not strictly necessary that REFRESH requests and responses include information on the refresh interval.

5.2.4 REFRESH Response

An access streamer reacts to a REFRESH request with a REFESH response. The response has to contain (1) an indication of whether or not the request succeeded plus a reason if the request failed, and (2) optional 'inactive' SDP that specifies which of this access streamer's configurations are available to the user. The SDP also contains the URIs that point to media servers (cf. the SDP in an ADMISSION response).

A REFRESH request can fail because of an invalid admission token or for reasons internal to the access streamer serving the request.

5.2.5 Example

Figure 5 shows a typical message exchange at point C of the example of Figure 1. Figure 5 assumes that Bob's mobile host was receiving the broadcast from AS1 between points B and C.

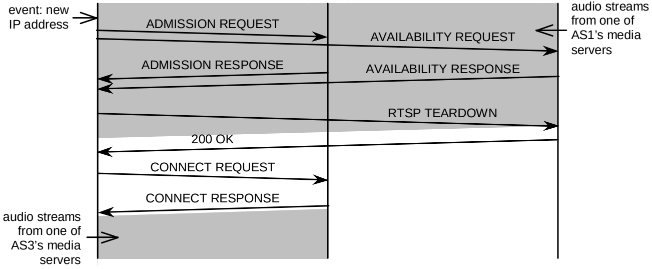


Figure 5. Example roaming behavior at point C.

6. REALIZATION

In this section, we consider the realization of our protocol using SIP and SDP.

In Section 6.1, we explain why we chose this combination to realize our protocol instead of other solutions. Next, we consider the mapping of our protocol's messages to SIP headers and SDP payloads in Section 6.2. In Section 6.3, we discuss the mapping of our protocol's behavior onto that of SIP.

6.1 Possible Solutions

Access streamers need to run an authentication service, a directory service (containing information on available configurations), and an initiation service. While we could have used dedicated protocols for each of these services (e.g., SLP [27] or UPnP [26] for the directory service and RTSP [14] for the initiation service), we chose the Session Initiation Protocol (SIP) [15] for all three of them. The primary reasons for this choice were the following.

First of all, SIP can also be used in a directory-like manner through OPTIONS messages or through (re-)INVITE messages that contain an SDP payload with `a=inactive` lines [17]. In addition, SIP supports shared secret user authentication by default while there is ongoing work within the IETF to extend SIP with user authentication based on public keys [30]. SIP is thus able to provide the three services we require.

The second reason is that there is also ongoing work on an extension that allows SIP to refresh state using SIP re-INVITEs [31]. We can use the extension for our protocol to refresh authentication softstate at access streamers.

Third, SIP has an extension that can push events to interested hosts [25]. We will thus also be able to use SIP for the announcement protocol of our platform (see Section 7).

Fourth, if we realize CONNECT requests and responses in SIP, we only need one protocol. This simplifies the realization and the implementation of the protocol of Sections 4 and 5.

The fifth reason for using SIP is that we consider SIP to be a generic application-level signaling protocol that will eventually become ubiquitously available.

6.2 Mapping to SIP Messages

We combine SIP headers, SIP header extensions and SDP attributes with SIP INVITEs and SIP responses to realize our protocol.

6.2.1 SIP INVITEs

Table 1 shows how the requests of our protocol map to SIP INVITEs, their headers, and SDP payloads. Headers and payloads that are optional are marked with an (o).

Table 1. Mapping to SIP INVITEs.

Request	**SIP (re-)INVITE**	
	Headers	**Payload**
ADMISSION	From, Authorization, Session-Expires (o), Min-SE (o)	SDP (o) with `a=inactive`
AVAILABILITY	Admission-Authorization-Token	SDP (o) with `a=inactive`
REFRESH	Admission-Authorization-Token, Session-Expires	SDP (o) with `a=inactive`

The AVAILABILITY, and REFRESH requests take the form of re-INVITEs because they are carried across an existing SIP association (a dialog), which is established by the initial INVITE that realizes the ADMISSION request.

The From field of an INVITE contains the ID of the user (e.g., Bob@AS1). The Authentication header [15] contains the user's credentials.

In an INVITE (ADMISSION request), the Session-Expires header specifies the refresh interval proposed by the mobile host. In a re-INVITE that is used as a REFRESH request, the Session-Expires header specifies the refresh interval the mobile host and the access streamer have agreed upon (see ADMISSION response in Table 2). The Min-SE header indicates the minimum interval that the host finds acceptable. We note that we are somewhat misusing the Session-Expires and Min-SE headers: in [31] they are used to refresh the state of a (multimedia broadcast) session, while we are using them to refresh authentication state (which may not yet involve a session).

The Admission-Authorization-Token is similar to a P-Media-Authorization-Token [33] but contains the admission token. At a later stage, we will replace the Admission-Authorization-Token header with the result of the identity work that's currently going on in the IETF [30].

The SDP describes configurations. For (re-)INVITEs used as an ADMISSION, AVAILABILITY, or REFRESH request the SDP contains an `a=inactive` line.

A CONNECT request maps to a control protocol-specific message, for instance to a SIP INVITE. It does not contain an `a=inactive` line.

6.2.2 SIP Responses

Table 2 shows how the responses of our protocol map to SIP responses, their headers, and SDP payloads.

Table 2. Mapping to SIP responses.

Response	**SIP Response**		
	Code	**Headers**	**Payload**
ADMISSION	200	Session-Expires, Admission-Authorization-Token	SDP with `a=inactive`
	401	WWW-Authenticate	-
	403	-	-
	422	Min-SE	-
AVAILABILITY	200	-	SDP with `a=inactive`
	403	-	-
REFRESH	200	Session-Expires (o)	SDP (o) with `a=inactive`
	403	-	-

A 200 (OK) indicates that an access streamer has successfully serviced a request. A 401 (Unauthorized) indicates that the mobile host sending the request must resubmit the request, this time including the user's credentials [15]. In our protocol, a 403 (Forbidden) signals that a user does not have access to an access streamer, either because the access streamer has no roaming agreement with the user's home access streamer (e.g., AS2 has no agreement with AS1), because the access streamer does not offer the broadcast channel, or because the user's authentication state has expired. A 422 (Session Interval Too Small) indicates that the access streamer finds the refresh interval in the Session-Expires header of an ADMISSION request too small and wants the mobile host to resubmit the request using the value of Min-SE as a lower bound [31].

The Admission-Authorization-Token contains the admission token that the access streamer hands to the mobile host. The WWW-Authenticate header forces a mobile host to provide its credentials if it previously submitted an INVITE (ADMISSION request) without the user's credentials.

The Session-Expires header in a 200 OK specifies the final refresh interval that the mobile host and the access streamer have agreed upon. The Min-SE header of a 422 indicates the minimum refresh interval the access streamer finds acceptable.

6.3 Mapping to SIP Behavior

We consider the behavior of our protocol at the SIP level during initiation and during roaming. We assume that CONNECT requests and responses also map to SIP messages.

6.3.1 Initiation

Figure 6 shows an example of the behavior of our protocol at the SIP-level during initiation at point A in Figure 1. After Bob has selected a channel, his mobile host simultaneously submits an INVITE (ADMISSION request) to access streamers AS1 and AS3. Both INVITEs contain `a=inactive` SDP lines (see Table 1).

In this example, the INVITE to AS3 does not contain Bob's credentials. As a result, AS3 responds with a 401 Unauthorized (ADMISSION response) containing a WWW-Authenticate header to force the mobile host to resubmit the INVITE with Bob's credentials [15].

After having successfully authenticated Bob, AS3 returns a 422 Session Interval Too Small. Using the Min-SE field in the 422, the mobile host picks a refresh interval that AS3 finds acceptable and resubmits the INVITE [31]. AS3 then responds with a 200 OK (ADMISSION response). If the mobile host would have provided the user's credentials and an acceptable session interval in the first INVITE, AS3 would have responded with a 200 OK right away.

In Figure 6, the mobile host decides that AS1 provides the "best" available configuration based on the 200 OKs of AS1 and AS3. As a result, the mobile host sends an INVITE (CONNECT request) to a SIP media server of AS1 with SDP specifying the "best" configuration.

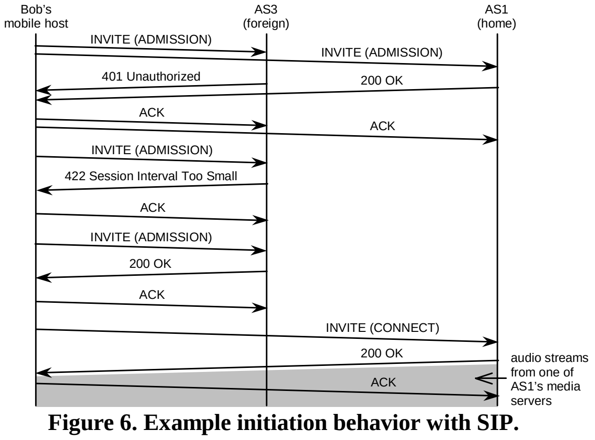


Figure 6. Example initiation behavior with SIP.

6.3.2 Roaming

Figure 7 shows an example of the behavior of our protocol at the SIP-level when the mobile host roams back into ISP2 (point C in Figure 1). The example assumes that the mobile host was receiving the broadcast from AS1 between points B and C. For simplicity, we have omitted the REFRESH messages that the mobile host regularly sends to AS1.

The mobile host sends a re-INVITE to AS1 (AVAILABILITY request) because it already knows of AS1, and an INVITE to AS3 (ADMISSION request) because it just 'rediscovered' AS3. In this example, the mobile host picks acceptable refresh intervals and includes its user's credentials in the (re-)INVITEs right away. Bob can furthermore be authenticated at AS1 and AS3, so both access streamers respond with a 200 OK.

Based on the configurations carried in the SDP of the 200 OKs, the mobile host decides that this time AS3 provides the "best" configuration. It therefore sends an RSTP TEARDOWN (a control protocol-specific disconnect request) to the media server of AS1 it is receiving the broadcast from, and an INVITE (CONNECT request) to AS3. The INVITE to AS3 contains the new "best" configuration. In this example, the switch from AS1 to AS3 also involves a handoff from the UMTS subnet to the 802.11 subnet.

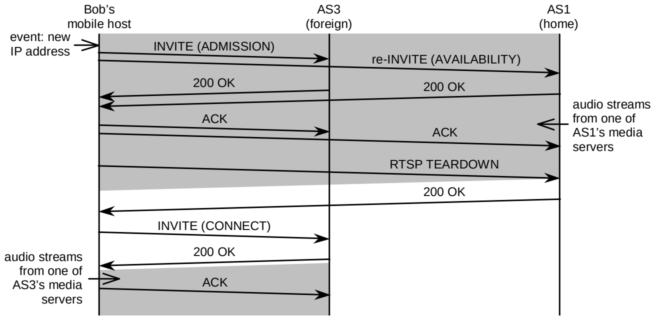


Figure 7. Example roaming behavior with SIP.

7. CONCLUSIONS AND FUTURE WORK

We presented an innovative protocol that enables mobile hosts to select among different configurations of the same broadcast that are offered by different service providers. Mobile hosts use the protocol to begin receiving a broadcast and to remain connected to the same logical broadcast as they move across subnets. The protocol is independent of the actual stream control protocol (e.g., RTSP) that service providers might use and can therefore easily be deployed in addition to existing streaming services. The main added value of our system is that it enables mobile hosts to seamlessly roam in heterogeneous environments, while effectively exploiting local resources. As a result, mobile users will be able to continue to receive a multimedia broadcast wherever they go. We have shown that the system can be realized with existing Internet standards, notably SIP and SDP.

We are currently working on a generalization of the protocol presented in this paper to support additional business models. We expect that it will only have an impact on the types of agreements that need to be settled between the various actors as well as on the initial selection of broadcast channels. We do not expect that it will influence the mobility handling part of the protocol.

We have implemented the protocol using open SIP [34] and are currently integrating it with a streaming solution. We expect that we will be able to realize seamless handoffs, as we will reuse the implementation of [4] that hands a mobile host off from one stream to another. The prototype software will be validated on the wireless testbed at the University of Twente [35], which provides a multi-domain 802.11, UMTS, and GPRS environment.

Future work includes the design and realization of an announcement protocol. The protocol should for instance be able to announce events such as a configuration becoming unavailable.

8. REFERENCES

[1] M. Haardt, W. Mohr, "The Complete Solution for Third-Generation Wireless Communications: Two Modes on Air, One Winning Strategy", IEEE Personal Communications, Dec. 2000

[2] E. Brewer, R. Katz, Y. Chawathe, S. Gribble, T. Hodes, G. Nguyen, M. Stemm, T. Henderson, E. Amir, H. Balakrishnan, A. Fox, V. Padmanabhan, S. Seshan, "A Network Architecture for Heterogeneous Mobile Computing", IEEE Pers. Communications, Oct. 1998

[3] M. Stemm, R. Katz, "Vertical Handoffs in Wireless Overlay Networks", ACM Mobile Networking, Special Issue on Mobile Networking and Internet, Spring 1998

[4] C. Hesselman, H. Eertink, A. Peddemors, "Multimedia QoS Adaptation for Inter-tech Roaming", 6th IEEE Symposium on Computers and Communications (ISCC'01), Hammamet, Tunisia, July 2001

[5] B. Cain, A. Barbir, R. Nair, O. Spatscheck, "Known CN Request-Routing Mechanisms", Internet Draft, draft-ietf-cdi-known-request-routing-03.txt, April 2003

[6] C. Hesselman, I. Widya, H. Eertink, E. Huizer, "A Comprehensive Framework for Broadcasting Multimedia Content in the Future Mobile Internet", 2nd IEEE Workshop on Applications and Services in Wireless Networks (ASWN'02), Paris, July 2002

[7] Kutscher, Ott, Bormann, "Session Description and Capability Negotiation", Internet Draft, draft-ietf-mmusic-sdpng-06.txt, March 2003

[8] D. Xu, K. Nahrstedt, "Supporting Multimedia Service Polymorphism in Dynamic and Heterogeneous Environments", Technical Report UIUCDCS-R-2000-2159, University of Illinois at Urbana-Champaign, USA, Oct. 2000

[9] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 1889, Jan. 1996

[10] A. Balachandran, A. Campbell, M. Kounavis, "Active Filters: Delivering Scaled Media to Mobile Devices", 7th Int. Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV'97), St. Louis, USA, May 1997

[11] E. Amir, S. McCanne, R. Katz, "An Active Service Framework and its Application to Real-time Multimedia Transcoding", ACM SIGCOMM'98, Vancouver, Canada, Sept. 1998

[12] J. Solomon, "Mobile IP — The Internet Unplugged", Prentice Hall, 1998

[13] J. Chennikara, W. Chen, A. Dutta, O. Altintas, "Application-Layer Multicast for Mobile Users in Diverse Networks", IEEE Globecom 2002, Taipei, Taiwan, Nov. 2002

[14] H. Schulzrinne, A. Rao, R. Lanphier, "Real Time Streaming Protocol (RTSP)", RFC 2326, April 1998

[15] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002

[16] M. Handley, V. Jacobson, "SDP: Session Description Protocol", RFC 2327, April 1998

[17] J. Rosenberg, H. Schulzrinne, "An Offer/Answer Model with the Session Description Protocol (SDP)", RFC 3264, June 2002

[18] S. McCanne, V. Jacobson, M. Vetterli, "Receiver-driven Layered Multicast", ACM SIGCOMM, Stanford, USA, Aug. 1996

[19] Y. Chawathe, "Scattercast: An Adaptable Broadcast Distribution Framework", Special Issue of ACM Multimedia Distribution, 2002

[20] A. Dutta, H. Schulzrinne, S. Das, A. McAuley, W. Chen, O. Altintas, "MarconiNet supporting Streaming Media over Localized Wireless Multicast", M-Commerce 2002 Workshop, Atlanta, USA, Sept. 2002

[21] J. Saltzer, D. Reed, D. Clark, "End-to-end Arguments in System Design", ACM Transactions on Computer Systems, Vol. 2, Issue 4, Nov. 1984, pp. 277-288

[22] D. Clark, "The Design Philosophy of the DARPA Internet Protocols", ACM SIGCOMM, Sept. 1988

[23] H. Schulzrinne, "Dynamic Host Configuration Protocol (DHCP-for-IPv4) Option for Session Initiation Protocol (SIP) Servers", RFC 3361, August 2002

[24] J. Kempf, J. Goldschmidt, "Mobile Code for Network Service Access", INET 2000, Yokohama, Japan, July 2000, http:// www.isoc.org/inet2000/cdproceedings/3c/3c_2.htm

[25] A. Roach, "Session Initiation Protocol (SIP)-Specific Event Notification", RFC 3265, June 2002

[26] Microsoft Corporation, "Universal Plug and Play Device Architectute", Version 1.0, June 2000, http:// www.upnp.org/download/UPnPDA10_20000613.htm

[27] E. Guttman, C. Perkins, J. Veizades, M. Day, "Service Location Protocol, Version 2", RFC 2608, June 1999

[28] P. Calhoun, J. Loughney, E. Guttman, G. Zorn, J. Arkko, "Diameter Base Protocol", Internet Draft, April 2003, draft-ietf-aaa-diameter-17.txt

[29] H. Wang, R. Katz, J. Giese, "Policy-Enabled Handoffs Across Heterogeneous Wireless Networks", 2nd IEEE Workshop on Mobile Computing and Applications (WMCSA 1999), New Orleans, USA, Feb. 1999

[30] J. Peterson, "Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)", Internet Draft, Feb. 2003, draft-ietf-sip-identity-01.txt

[31] S. Donovan, J. Rosenberg, "Session Timers in the Session Initiation Protocol (SIP)", Internet Draft, Nov. 2002, draft-ietf-sip-session-timer-10.txt

[32] S. Roy, B. Shen, V. Sundaram, R. Kumar, "Application Level Hand-off Support for Mobile Media Transcoding Sessions", NOSSDAV'02, Miami Beach, Florida, May 2002

[33] W. Marshall, "Private Session Initiation Protocol (SIP) Extensions for Media Authorization", RFC 3313, Jan. 2003

[34] oSIP webpage, http://www.gnu.org/software/osip/

[35] Wireless Campus, http://www.wirelesscampus.nl