Multimedia QoS Adaptation for Inter-tech Roaming Using Spatially Situated Multicast Groups

Cristian Hesselman, Henk Eertink and Arjan Peddemors
Telematica Instituut, the Netherlands
{hesselman, eertink, peddemors}@telin.nl

1. Introduction

We have recently proposed an application-level QoS adaptation service for roaming between wireless networks that are based on different technologies [1] ('inter-tech' roaming [2]). The service is a central part of the content distribution platform that we have introduced in [3]. In this extended abstract, we briefly discuss the platform (Section 2), the adaptation service (Section 3) and its supporting handoff protocol (Section 4). We then outline the requirements that we put on the network infrastructure (Section 5) and describe the purpose and structure of our demo (Section 6). We conclude with a short summary of our work (Section 7).

2. The Platform

Our platform supports the distribution of multimedia streams (e.g., a streamed TV channel) to a diverse set of mobile clients. The philosophy behind the platform is to find a balance between high scalability and the delivery of an optimal QoS to individual mobile clients. To accomplish this, the platform divides the coverage area of a wireless infrastructure into domains and restricts the amount of available 'QoS spectrum' in each domain to a few application-level service classes [4, 5]. A *service class* defines a *domain-specific* QoS level that the presentation resources (e.g., a display) of a mobile client receive. Each domain of a wireless infrastructure must support a *few* service classes and realize each class as a set of *site-local multicast groups* [6]. The size of this set may vary depending on whether the audio and video parts of a stream are encoded separately or jointly. It also depends on whether the encoding technique is layered [7]. We use *proxies* [4, 5, 8, 9] to bridge the differences between service classes. Our proxies connect to sets of site-local multicast groups for communications with mobile clients, and to a global multicast group to communicate with fixed clients. Proxies perform functions such as rate adaptation, transcoding, audio and video filtering, and so on. Proxies typically run on *gateway* hosts in an *access domain* [5, 9].

As an example, consider the distribution of a TV channel to mobile clients. Figure 1 shows how our system decomposes the TV broadcast.

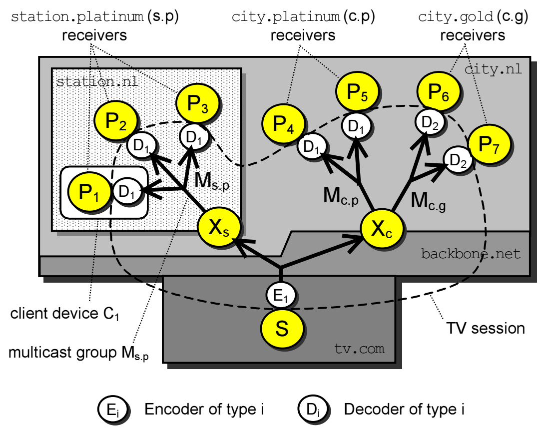

Figure 1. Example: a TV broadcast.

The server-side application S produces an uncompressed audio-video stream and injects it into a TV *session* (dashed line). The session delivers the stream to client-side application components P_1 through P_7. Each P_i represents the presentation resources of client device C_i and forms the sink of the TV stream. Clients C_1 through C_7 mostly consist of mobile devices that are distributed over domains `station.nl` and `city.nl`. The city domain operates a medium-range medium-speed Metropolitan Area Network (MAN). It *overlays* [10, 11] the short-range high-speed LAN that `station.nl` operates. Each client device is equipped with at least two network interfaces so that they can connect to `station.nl`'s LAN as well as to `city.nl`'s MAN. The clients are furthermore IP multicast enabled and are equipped with an RTP depacketizer, a decoder (e.g., an MPEG-4 decoder) and multimedia presentation resources (display and speakers). All clients thus have the necessary resources for their end-users to watch the TV channel transmitted by S, regardless of whether they are in `station.nl` or `city.nl`.

To simplify the discussion, we assume that all the client devices demand that the audio and video portions of the

TV stream are encoded together in a non-layered fashion. As a result, the station and city domains can realize their service classes as a single site-local multicast group. For ease of notation, we denote a service class as `domainName.className`. We denote the multicast group associated with `domainName.className` as $M_{d.c}$ with d the first letter of `domainName` and c the first letter of `className`. We depict each $M_{d.c}$ as a multi-point arrow.

As an example, consider `station.platinum` and its associated multicast group $M_{s.p}$. Players P_1 through P_3 receive the TV stream at class `station.platinum` because their respective client devices (C_1 through C_3) have joined $M_{s.p}$ on their LAN interfaces. The proxy of `station.nl` (X_s) has joined the global multicast group as well as $M_{s.p}$. It realizes the QoS level associated with `station.platinum` by manipulating the media stream coming from S and transmitting it onto $M_{s.p}$. X_s may for instance have to adapt the rate of the TV stream to that of class `station.platinum`.

3. Inter-tech Roaming

The clients involved in the TV session of Figure 1 will typically experience fluctuations in the availability of communications resources [12]. As a result, the QoS of the stream that the players receive needs to be *adapted* [13] so that the communications resources that are available can support it.

Our platform adapts the QoS of the TV stream that a player receives by transferring the player to another service class. We call this a *service class handoff*. As an example, consider player P_3 of Figure 1 and assume that the client device that hosts it (C_3) roams from `station.nl` to `city.nl`. Figure 2 shows the portion of Figure 1 that is relevant for this particular situation.

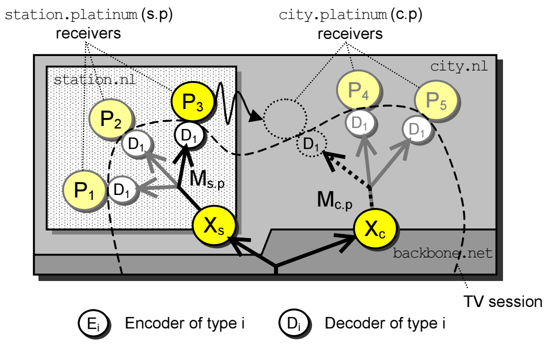

Figure 2. A service class handoff.

Since service classes are domain-specific, our platform must eventually handoff player P_3 from class `station.platinum` to class `city.platinum`. To achieve this, the platform unsubscribes C_3 from the multicast group associated with `station.platinum` ($M_{s.p}$) and joins it to the multicast group of `city.platinum` ($M_{c.p}$). The platform furthermore configures C_3's decoder and IP service so that their QoS characteristics correspond to the QoS level that `city.platinum` defines. Observe that in this particular example `city.platinum` is a suitable target class because it is also based on encoding type 1.

The result of the handoff is that P_3 receives the TV stream from X_c rather than from X_s. In addition, it receives the stream at the QoS level associated with class `city.platinum`. This QoS level will typically be lower than that of `station.platinum` because `city.nl` uses a relatively low-bandwidth MAN rather than the high-bandwidth LAN of `station.nl`. The end-user will therefore perceive a degradation in the QoS of the audio-video stream that P_3 presents when it moves from `station.nl` to `city.nl`. The QoS will usually improve when C_3 roams back into the station domain.

The networks that the station and city domains use will most likely be based on different technologies. Using the terminology of Pahlavan *et al.* [2], we therefore call the handoff of Figure 2 an *inter-tech* service class handoff. In this example, it is also an *inter-domain* [9] service class handoff because it transfers P_3 from the station to the city domain. Service class handoffs can also occur when a client roams in a network based on a single technology (intra-tech) or within a single domain (intra-domain). Combinations of these types of handoffs are also possible.

4. Inter-tech Handoff Protocol

The inter-tech service class handoffs that our platform supports are *mobile-controlled* [11]. In the example of Figure 2, this means that there is a *handoff component* on C_3 that decides whether a handoff to $M_{c.p}$ is required. The handoff component is part of our platform. It also executes a handoff should one be required.

Our handoff component uses the *packet loss* characteristics of the paths between C_3 and the two proxies to decide when to handoff to another service class. We make use of *beaconing* [10] to determine these loss characteristics. In the example of Figure 2, this means that proxies X_s and X_c multicast beacon messages into their domains at regular intervals. X_s and X_c include the domain that they belong to and the beacon interval that they use in the beacon messages.

Both proxies transmit their beacon messages onto the same *well-known* multicast group M_b. M_b is a site-local multicast group in both `station.nl` and `city.nl`. The beacons that X_s and X_c transmit do therefore not leave their respective domains.

The handoff component uses the beacon messages from X_s and X_c to determine if C_3 should use its LAN interface or its MAN interface to receive the TV stream. The handoff component joins C_3 to $M_{s.p}$ (class `station.platinum`) if it decides to use the LAN interface; it joins C_3 to $M_{c.p}$ (class `city.platinum`) if it decides to use C_3's MAN interface.

5. Infrastructure Requirements

Our approach requires a server in the network infrastructure that hands out IP addresses, for instance a DHCP [14] server. A mobile client must receive an IP address from this server whenever it roams into a new network. For example, C_3 needs to receive an IP address from the server when it roams from the MAN of `city.nl` into the LAN of `station.nl`. C_3 can join M_b and $M_{s.p}$ only after it has associated the new address with its LAN interface.

In our approach, the IP address of a mobile client changes regularly as a result of roaming. This is similar to the SIP solution for mobility [15]. Our model does not require a client to have a permanent IP address like in Mobile IP [16]. As a result, we also do not require any additional routing complexity from the IP layer, i.e. we do not need Mobile IP.

We further require base stations to be connected to a multicast router through a separate link or through a multicast-aware switch. To see why this is necessary, consider clients C_4 through C_7 of Figure 1. Suppose that C_4 and C_5 reside in one cell (A) and C_6 and C_7 in another (B). If the base stations of both cells are connected to the local multicast router through a shared link that does not involve a multicast-aware switch, the traffic of $M_{c.g}$ would be transmitted into cell A while there are no gold clients in that cell. Similarly, the traffic of $M_{c.p}$ would be sent into cell B while there are no platinum clients in that cell. This is clearly an undesirable situation.

6. Demo

Our demo implements the scenario of Figure 2 in which client C_3 roams between domains `station.nl` and `city.nl`. Figure 3 shows its organization. This figure also illustrates how the proxy and player components of Figure 2 are distributed over the various machines that the demo is made up of.

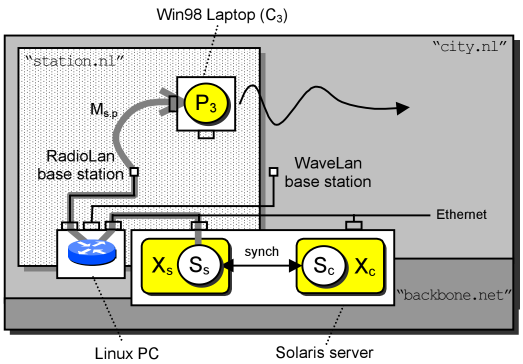

Figure 3. Demo.

6.1 Proxies

The Solaris machine in Figure 3 hosts the proxies X_s and X_c of Figure 2. For reasons of simplicity, we have implemented proxies X_s and X_c to act as broadcasting servers. That is, they generate the stream containing the TV channel locally rather than from a stream coming from the broadcasting server S (cf. Figure 1). X_s and X_c each consist of a QuickTime Darwin streaming server [17]: X_s consists of server S_s; X_c consists of server S_c.

S_s and S_c run synchronously as indicated by the arrow between them in Figure 3 and loop continuously. S_s locally reads a high quality movie from a hinted (i.e., encoded and RTP-packetized) QuickTime file and transmits it onto the multicast group that represents class `station.platinum`, $M_{s.p}$. Similarly, S_c locally reads a low quality version of the same movie from a different hinted file and transmits it onto the multicast group that represents class `city.platinum`, $M_{c.p}$.

X_s and X_c each also contain a process (not shown in Figure 3) that transmits beacon messages at regular intervals.

6.2 Networks

The RadioLan [18] base station in Figure 3 represents `station.nl`'s LAN (high capacity, short range). It offers a gross bandwidth of 10 Mbps. The RadioLan base station provides an indoor range of approximately 15 meters and operates in the 5.8 GHz band. The WaveLan [19] base station mimics the MAN of `city.nl` (medium capacity, medium range). It provides a gross over-the-air bandwidth of 1 Mbps. The WaveLan base station has an indoor range of approximately 30 meters and operates at a frequency of 2.4 GHz.

We have placed the RadioLan and WaveLan base stations next to each other. In this way, the WaveLan cell *overlays* the RadioLan cell.

6.3 Client

We use a Windows98 laptop to represent client C_3. The laptop is equipped with a RadioLan and a WaveLan network interface that have hard-coded IP addresses. The RadioLan interface represents C_3's LAN interface; the WaveLan interface represents its MAN interface.

The laptop runs the QuickTime client software package [20] to receive the streams that servers S_s and S_c transmit. It hosts a Java implementation of the handoff component. The handoff component uses the beacons that it receives from X_s and X_c to determine the network that the laptop should use (RadioLan or WaveLan). It also executes a handoff by calling the QuickTime API. For example, if the laptop roams out the RadioLan coverage area (see Figure 3), the handoff component observes that it begins to loose beacons from X_s. At some point, it will decide that it has lost too many beacons and call the QuickTime API to join the laptop to $M_{c.p}$ on its WaveLan interface. When the handoff is completed, the end-user sees the low quality version of the stream that S_c produces rather than the stream from S_s (see Figure 3). Figure 4 shows this post-handoff situation.

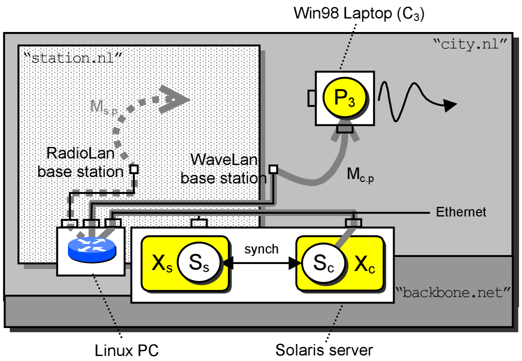

Figure 4. Post-handoff situation.

7. Summary

We introduced a platform that revolves around the notion of domain-specific application-level service classes. We focussed on the QoS adaptation service of our platform, in particular for inter-tech roaming scenarios. The service uses beacon messages that service providers transmit to realize inter-tech roaming. Mobile clients listen for these beacons and use them to determine when they should initiate and complete a handoff to a new service class. We used IP multicast to increase the scalability of our platform.

We feel that one of the major advantages of our approach is that it handles mobility at the application level. This allows our platform to deal with handoffs in an application-aware manner. In addition, we believe that our approach makes the selection and adaptation of QoS levels relatively simple.

Similar to the SIP solution for mobility, we think that our approach would typically be used in conjunction with Mobile IP. Our approach would support applications that use UDP while Mobile IP would support TCP-based applications.

A disadvantage of our approach is that a mobile client does not have a unique identifier. As a result, we will have to deal with authentication at the application-level rather than at the IP level (cf. Mobile IP).

8. References

[1] C. Hesselman, H. Eertink, A. Peddemors, "Multimedia QoS Adaptation for Inter-tech Roaming", paper submitted to the IEEE Symposium on Computers and Communications (ISCC'01), Tunisia, July 2001

[2] K. Pahlavan, P. Krishnamurthy, A. Hatami, M. Ylianttila, J. Makela, R. Pichna, J. Vallström, "Handoff in Hybrid Mobile Data Networks", IEEE Personal Communications, April 2000

[3] C. Hesselman, H. Eertink, "A Scalable QoS Adaptation Service for Mobile Multimedia Applications", 6th EUNICE Open European Summer School, Sept. 2000, Enschede, The Netherlands, http://wwwtgs.ctit.utwente.nl/eunice/summerschool/papers/paper4-4.pdf

[4] G. Fankhauser, M. Dasen, N. Weiler, B. Plattner, B. Stiller, "The WaveVideo System and Network Architecture: Design and Implementation", Technical Report, Zürich, Switzerland, June 1998

[5] N. Yeadon, F. Garcia, D. Hutshison, D. Shepherd, "Filters: QoS Support Mechanisms for Multipeer Communications", IEEE Journal on Selected Areas in Communications, Sept. 1996

[6] D. Lee, D. Lough, S. Midkiff, N. Davis, P. Benchoff, "The Next Generation of the Internet: Aspects of the Internet Protocol Version 6", IEEE Network, Jan/Feb 1998

[7] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", Internet draft, draft-ietf-avt-rtp-new-08.txt, July 2000

[8] I. Kouvelas, V. Hardman, J. Crowcroft, "Network Adaptive Continuous-Media Applications Through Self Organised Transcoding", Proc. of Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'98), July 1998, Cambridge, UK

[9] 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

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

[11] N. Tripathi, J. Reed, H. VanLandingham, "Handoff in Cellular Systems", IEEE Personal Communications, Dec. 1998

[12] A. Campbell, "A Research Agenda for QOS-aware Mobile Multimedia Middleware", NSF Wireless and Mobile Communications Workshop, Virginia, USA, March 1997

[13] D. Chalmers, M. Sloman, "A Survey of Quality of Service in Mobile Computing Environments", IEEE Communications Surveys, http://www.comsoc.org/pubs/surveys, 2nd Quarter 1999

[14] R. Droms, "Automated Configuration of TCP/IP with DHCP", IEEE Internet Computing, July-August 1999

[15] E. Wedlund, H. Schulzrinne, "Mobility Support Using SIP", 2nd ACM/IEEE International Conference on Wireless and Mobile Multimedia (WoWMoM'99), Seattle, USA, Aug. 1999

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

[17] Darwin Streaming Server, http://www.publicsource.apple.com/projects/streaming/

[18] RadioLan homepage, http://www.radiolan.com

[19] WaveLan homepage, http://www.wavelan.com

[20] Apple QuickTime Client 4.1, http://developer.apple.com/quicktime/