IPv6 applications to demonstrate IP-layer mobility mechanisms

ENST Bretagne, option Informatique et Télécommunications
3rd Year Internship Report

François LEIBER

20th September 2004

Realized in **Keio University**, under the direction of:

Thierry ERNST - *WIDE, Nautilus6 project*
Jean-Marie BONNIN - *ENST Bretagne, RSM department*

Preface

This internship was realized between April and September 2004, for my 3rd year as a student of the *Ecole Nationale Supérieure des Télécommunications de Bretagne* (ENST Bretagne), option *Computer Science and Telecommunications*.

ENST Bretagne
Technopôle Brest-Iroise
BP 832, 29285 Brest cedex, FRANCE

It was done in the Jun Murai Lab. of Keio University, Shonan Fujisawa Campus (SFC), as part of the Nautilus6 project, of the WIDE organization (refer to Chapter 1.1 for more information on the latest).

Keio University, Murai Lab.
Shin-Kawasaki Town Campus
Kawasaki, Kanagawa, JAPAN

Acknowlegments

I would like to thank warmly:

- Thierry ERNST
- Jean-Marie BONNIN
- Romain KUNTZ
- Nautilus6 members
- Bruno DENIAUD
- Annie GRAVEY

Abstract

The recent advances in wireless technologies have brought us one step closer to the ubiquitous Internet which could change our way of living. But in order to achieve this, the underlying network technology must be enhanced with more advanced features like mobility support. The Nautilus6 project in which I am now working aims at completing all the steps for the deployment of IPv6 networks, and complementary mobility protocols such as Mobile IPv6 and NEMO Basic Support, from development to demonstration, which will make this ubiquitous Internet possible.
More precisely, I have been involved in three different activities: helping to develop a better IPv6 environment, demonstrating the technologies developed in Nautilus6, and improving the working environment of the project.

Keywords

IPv6, host and network mobility, streaming, adaptive applications, Zaurus

Résumé

Les récentes avancées dans les technologies sans-fil nous ont amenées un pas plus près de l'Internet omniprésent, qui pourrait changer notre manière de vivre. Afin d'atteindre ce but, la technologie réseau sous-jacente doit être complétée par des fonctionnalités plus avancées, comme le support de la mobilité. Le projet Nautilus6, dans lequel je travaille actuellement au Japon, a pour but de compléter les différentes étapes du déploiement des réseaux IPv6, accompagnés de protocoles complémentaires pour la mobilité tels Mobile IPv6 ou NEMO Basic Support, du développement jusqu'à la démonstration, et qui rendront cet Internet omniprésent possible.
Plus précisément, j'ai été impliqué dans trois différentes activités : aider a développer un meilleur environnement IPv6, démontrer les technologies développées dans Nautilus6, et améliorer l'environnement du travail du projet.

Mots-clés

IPv6, mobilité d'un noeud ou d'un réseau, streaming, applications adaptatives, Zaurus

Contents

1 Introduction **8**
- 1.1 Overview of Nautilus6 . . . 9
  - 1.1.1 WIDE organization . . . 9
  - 1.1.2 Nautilus6 project . . . 9
  - 1.1.3 Motivation . . . 10
  - 1.1.4 Goals . . . 10
  - 1.1.5 Tasks . . . 11
  - 1.1.6 Testbeds . . . 12
- 1.2 Internship objectives . . . 12

2 Technical generalities **14**
- 2.1 IPv6 environment set up . . . 14
  - 2.1.1 Installation and testing . . . 14
  - 2.1.2 Zaurus platform . . . 14
- 2.2 Streaming . . . 15
- 2.3 Videoconferencing . . . 16
- 2.4 Multicast . . . 17
- 2.5 Statistics gathering . . . 18
- 2.6 Adaptive Applications . . . 19
  - 2.6.1 Definition . . . 19
  - 2.6.2 Techniques . . . 19
- 2.7 Programming on Zaurus . . . 20

3 Demonstration Testbeds **22**
- 3.1 SOI . . . 22
  - 3.1.1 SOI project . . . 22
  - 3.1.2 Collaboration objectives . . . 23
  - 3.1.3 Scenarios . . . 23
  - 3.1.4 Plans . . . 24
- 3.2 InternetCAR . . . 25
  - 3.2.1 Presentation . . . 25
  - 3.2.2 iCAR camp . . . 26

3.3 E-Wheelchair 26
3.3.1 Goals 26
3.3.2 Constraints 26
3.3.3 Equipment 28
3.3.4 E-Bike 29
3.3.5 Other demonstration testbeds 30
3.3.6 Plans 30

4 Improving Nautilus6 working environment 31
4.1 Videoconferencing 31
4.2 Website 31
4.3 Ashina management 32
4.4 Communication 32

5 Conclusion 33

Bibliography 34

List of Figures 36

A Glossary 38

B IPv6 and mobility 42
B.1 IPv6 Features 43
B.2 Mobile IPv6 44
B.3 NEMO Basic Support 44

C Student Project 46
C.1 Cyberté Project 46
C.2 Bibliographic report 48
C.3 Implementation 48

D Additional figures 51

Chapter 1

Introduction

The past few years have seen a quick expansion of the needs for mobile devices, and more generally for wireless connections. Unfortunately, underlying network technology was not yet sufficient to provide us an ubiquitous Internet, available at any time and any place. The deployment of IPv6 networks, and the recent developments of complementary mobility protocols such as Mobile IPv6 (MIP6) and NEMO Basic Support (refer to appendix B for all those protocols) will finally give us the possibility to reach this objective, which can change our way of living in many different ways.

The Nautilus6 project in which I now work in Japan aims at developing all the technologies associated with mobility: test and validate them if they already exist, develop, standardize and implement them otherwise, design the operational environment to deploy those standards, and finally demonstrate them, so industrials will commercialize them and bring their features to reality for the general public.

In order to achieve those goals, now that protocols like MIP6 [1] and NEMO Basic Support [2] are stable and mature enough, a set of testbeds to demonstrate the underlying network technology is needed.
Those demonstrations are composed of several aspects:

- The technical aspect: what terminals to use, how to give them power, what physical medium for the network, etc.
- The network aspect: install a set of network protocols which will enable the features we want to demonstrate, for example MIP6 for host mobility support.

- The application aspect: develop, install and configure a set of applications which can be integrated nicely with the two other aspects. This means we must demonstrate what the protocols offer that we could not have before, or at least now have it in a more simple way, and what is the benefit of using them. allow us to have features we could not have before, or at least have them in a more simple way, and why those features are interesting with that kind of terminals.

For the last five months I have been working more particularly in the third direction, selecting, testing and validating applications to demonstrate IP-layer mobility. Moreover, I had already worked in the Cyberté project to develop an application which could adapt itself to the network conditions, a feature which can be very useful when associated with mobility (see Annex C for more complete information).

After I have presented in more details the global context and the Nautilus6 project, I will talk about the different demonstration environments we are working on, as well as the other activities in which I have been involved. Note that even if some of these activities are now completed, others are done on a longer perspective since I will be working here six more months after defending my presentation.

1.1 Overview of Nautilus6

1.1.1 WIDE organization

Started in 1988 for engineering universities in Japan, the goal of this computer engineering organization was the creation of a Widely Integrated Distributed Environment, for a large variety of systems, networks, technologies, and for the benefit of anyone, individuals, groups, national or international societies.

The experimental environment created by this project, called the WIDE Internet, has been the first step towards Internet in Japan. Finally, WIDE was interested very early in the IPv6 protocol, for scalability reasons, since many machines need to be inter-connected; this ubiquity of the network is today becoming a reality, making the technical innovations conducted by WIDE a necessity.

1.1.2 Nautilus6 project

Created in spring 2003 as part of the WIDE organization and based in Jun Murai Lab, Keio University, Japan, this project aims to de-

velop a complete and operational IPv6 mobility environment, in many different scenarios (see Appendix A for more information on IPv6, host and network mobility support).
Members mostly come from Keio University, but also from other institutions of WIDE. Links have been established with ULP Strasbourg, ENST Bretagne, INT Paris, France Telecom and Seoul National University, who all participate to the overall objectives of Nautilus6.

1.1.3 Motivation

Within the framework of the WIDE Internet, mobility features are necessary in order to achieve the ubiquity of the Internet we are aiming at, which means any terminal can connect at any time and any place, and that users can have constant connectivity while on the move. This kind of scenario is likely to become very frequent, for example with embedded networks inside of cars [3], bus, trains, airplanes, etc, which will be continuously connected when the vehicle moves; but also with new trends such as PANs (Personal Area Network), always connected thanks to one or more of its elements (mobile phone, PDA, etc) which will be used as mobile routers.

Moreover, networks nowadays tend to become all-IP, from personal computers to small mobile devices, but some features must be efficiently supported in order for this mobility to become operational; those should be finally available by the deployment of IPv6 networks, and the integration of IPv6 features such as host and network mobility support, ad-hoc networking and other technologies necessary to achieve the deployment of IPv6, multihoming, autoconfiguration, multicast, security, access control, etc.

1.1.4 Goals

All these features now need to be integrated, and there is a need to show how such mechanisms can actually be deployed in real environments, in an operational, integrated, efficient and secured way.

The goal of Nautilus6 is to address all these issues. More precisely, the mission of the project is:

- Firstly to test and use the IETF standards when they exist, or develop, standardize and implement them if needed;
- Secondly to design the operational environment (the list of necessary protocols and other technologies needed) to deploy those

standards, so that industrials such as ISPs will create applications and services using them;

- Then, to create a set of demonstrations of that environment to show its economic reality and convince financial deciders to start using it;
- Finally, to launch the new directions of research which seem the most promising.

1.1.5 Tasks

Many standardization activities have already been done within the IETF, which are the starting points for our project. More precisely, Nautilus6 has already defined these tasks:

- Test and use Mobile IPv6 implementations for BSD (done by KAME project) and for Linux (USAGI project); this was already done, but new implementations in development are still to be studied, such as the one which will be integrated in the Linux 2.6 kernel.
- Study, standardize and implement NEMO Basic Support; the protocol is now mostly operational, but some outdoor testing are still to be performed.
- Study multihoming issues associated with mobility [4], for example when more than one terminal are connected to the Internet in a PAN, or when one of the terminals is connected via more than one interfaces.
- Study seamless handover in IPv6 mobility (FMIP). The starting point is the Working Group Seamoby (for Seamless Mobility), which focuses on the performance aspect of mobility protocols, for example fast handover mechanisms for mobility hosts and routers, in order to have the smallest possible disruptions when using real-time services above heterogeneous networks.
- Security, authentication; some testing was already done with existing implementations, but the conclusion was that several things were missing before we could realize our own testbed now.
- Develop applications and services requiring or benefiting from the mobility features, which is my main working direction.

1.1.6 Testbeds

To realize the different necessary steps in the creation of a new protocol, the Nautilus6 project has installed three types of demonstration testbeds, usually used in parallel since the developed protocols are in different states of development:

- The *Indoor Testbed* is designed to implement and test the new protocols. Presently, NEMO Basic Support, multihoming and FMIP6 (protocol to realize seamless handover) are tested there.
- The *Demonstration Testbeds* are designed to validate and demonstrate the concepts (see E-Wheelchair, section 3.3).
- Finally, the *operational Testbeds* are designed to demonstrate the integration of several features in operational use (see Zaurus testbed, section 2.1.2).

More precisely, we need demonstration testbeds because our goal is to inform and convince:

- People about the usefulness of layer 3 mobility technologies, and therefore create a need in the general public for services requiring them.
- Financial deciders about the economic reality of such technologies, so industrials start producing it. This is very important for us, because one of the main barriers to the deployment of mobile technology today is not technical, but it is a mentality issue: people are unconfident about using new technologies, especially at a time when we regularly hear news about viruses, spam, etc. Even if it is on a completely different technical level, it makes people afraid of new computer and network technologies in general.
- Programmers about its maturity and possibilities, in order to have them develop mobility-aware applications.

1.2 Internship objectives

I have been engaged in three different main activities during this internship.

Demonstration Testbeds

As suggested by its title, *IPv6 applications to demonstrate IP-layer mobility mechanisms*, this internship is centered on the application

side of the demonstration testbeds of Nautilus6, more particularly on E-Bike, E-Wheelchair, and the collaboration with the SOI project. As we will see, each application and each aspect of those testbeds are aiming at demonstrating one or more particular aspect of the studied technologies, which were detailed in paragraph 1.1.5, including host and network mobility support, multihoming, seamless handover, etc.

One of the interesting directions for our demonstrations is adaptive applications, which can adapt themselves automatically to the conditions of the network; those applications are perfect to demonstrate many of the mobility aspects which we are studying, such as vertical or horizontal handovers (when changing the network interface, or when changing from one network to the other while keeping the same interface), multihoming, quality variations of the wireless network, etc. Moreover, streaming applications become very visual (the user can *see* the changes in quality) when they are modified to adapt to the network conditions.

IPv6 Environment

Before realizing complete demonstration testbeds, the IPv6 and Mobile IPv6 environment must already be completely installed, operational and tested, on desktops as well as on other platforms used for demonstration, like PDAs. Therefore we needed to make sure of this point before realizing other activities.

Miscellaneous

Finally, many other smaller tasks are necessary before the Nautilus6 project can run smoothly. One of them is to complete the website, another is to improve the working environment for the project members, for example by promoting simple and operational Videoconferencing, etc.

Chapter 2

Technical generalities

2.1 IPv6 environment set up

2.1.1 Installation and testing

Before joining Nautilus6, I had never worked in an IPv6 environment. Therefore, I started my internship by testing the IPv6 environment on my computer.

IPv6 applications

I found out which applications were already IPv6-capable, and tested them, more particularly applications involving streaming, such as live or file streaming and videoconferencing. I created a summary page on the Nautilus6 website [5], with conclusions on which applications types are today available in IPv6 or not.

2.1.2 Zaurus platform

The Zaurus is an advanced PDA running under Linux, produced by Sharp. It was used in one of the large Nautilus6 operational testbeds [6], where 40 MIP6-enabled Zaurus where given to WIDE members, in order to have some feedback on a large scale usage. The experiment was not very successful, and a final poll revealed several reasons:

- People use it as a simple PDA, without IPv6 connection.
- VoIP application was not very attractive: bad quality, no buddy list as in usual chat systems, battery life problem, etc.
- Streaming application was better than expected, but there was complaints because interesting content could not be found.

- Other types of applications were needed: videoconferencing, mailer, web browser, instant messaging, TV, Radio.

To enable MIP6, we used the original Sharp ROM based on Linux 2.4, modified by the USAGI team to support host mobility.
In order to improve the operational testbed and to use the Zaurus in other demonstration environments, I tested different applications which could benefit from IPv6 or MIP6. I also installed (with the help of Bruno Deniaud, working in ENST Rennes) a L2TP system, with a server in Nautilus6 office and a client on the Zaurus, which creates a IPv4 tunnel to give the possibility to have IPv6 connectivity even in IPv4-only networks.

2.2 Streaming

To have a high-quality, open and standard-based streaming (the word "streaming" designs a data transmission mechanism, where compressed multimedia streams are sent over a network like Internet and played by the client as they arrive) solution, the best solution is presently to rely on technologies such as (refer to the Glossary for further information):

- MPEG-4 for video encoding: presently the best quality / compression ratio, large range of available bitrates, open encoders and decoders available; there may be some licensing issues in the future.
- MP3 or AAC for the audio: AAC can provide far better compression than the very common MP3, but it is much more complex, less applications can read it, and you need to rely on proprietary encoders to have really good quality.
- MP4 file format: it is very fitted for streaming, since each stream in the file is independent and can be hinted; this word means that pre-calculated RTP packets are saved in the file, so the streaming server has almost nothing to do when a client asks for that file to be streamed.
- RTP and RTSP for the streaming.

After a study to determine which applications supporting those technologies we were going to use, we decided that:

- For the file server, we would use Darwin Streaming Server (DSS), the open-source version of the Apple Quicktime Streaming Server, with many evolved features, complete web administration, in fact one of the best presently available. The official

version does not support IPv6, but I was able to find a version modified by the 6NET project [7] to support it.

- For the live server (the application which encodes a stream coming from a webcam and sends it on a network), we will use mp4live, which is part of MPEG4IP, an end-to-end system to explore streaming multimedia. The package includes many existing open source packages and the "glue" to integrate them together (under the Mozilla Public License 1.1). The development is focused on the Linux platform but also ported to other ones. It is a tool for streaming video and audio that is standards-oriented and free from proprietary protocols and extensions. mp4live itself is not IPv6-enabled, but it will send the stream in IPv4 unicast to the Darwin Streaming Server, which will then reflect it to all the clients in IPv6.
- For the client, possible solutions are VideoLAN Client (very complex to change and cross-compile), MPlayer (but IPv6 RTSP does not work, moreover it is very complex), and mp4player from MPEG4IP. Since the latter is the only one being at the same time quite simple and really compliant with the technologies we described above, we decided to use it.

For the streaming server, I installed IPv6 DSS on my desktop computer for testing, then a second one on an official Nautilus6 server we set up for this. I also tested the DSS that is installed in ENST Rennes, but the conclusion was that except for a little more delay in the connection which does not really influence streaming, it was about the same performances.

Concerning MPEG4IP, I helped the developer of this application solve two annoying bugs: the first concerning the decompression of XviD (a very good, open and popular MPEG-4 video encoder) videos, the second concerning RTP streaming, where the socket was not correctly opened when using IPv6.
Now this package makes a very good solution for streaming, I could not find any other problems during all my testing.

2.3 Videoconferencing

Videoconferencing applications are an interesting type of streaming applications used for demonstration, since they also include strong real-time constraints: it is difficult to discuss with someone if the delay is more than one second. Moreover, they can also be used

to demonstrate multicast protocols when doing conferences with several locations at the same time. If we add the fact that such applications can be very useful for the communication inside an international project like Nautilus6, we understand that I spent a lot of time working on such applications:

- After testing, I created a summary of H.323 applications with all the compatibilities and possible problems, including Polycom, NetMeeting, GnomeMeeting, XMeeting and KMeeting.
- I now need to investigate other solutions, some of them in IPv6 multicast, to be used as an internal Nautilus6 tool as well as a demonstration application. I investigated in particular MCU solutions, SIP applications, but also VIC and RAT.

2.4 Multicast

Multicast is one of the native features of IPv6, and is very interesting for multimedia applications, especially in mobile networks, where one of the main issues is to reduce the bandwidth used. For example, if several people want to do multi-videoconferencing between two vehicles, it is much more interesting to transmit only one stream between the two vehicles and then divide it inside each vehicle, instead of doing standard unicast multi-videoconferencing, where the video and audio streams would flow more than once between the two vehicles.

I first realized a list of all the IPv6 multicast capable applications, which can be used in our two directions of testing:

- In the Indoor testbed, it is very convenient for testing different kinds of multicast applications inside mobile networks, and see the impact of other technologies on them: for example many multicast applications have real-time constraints, like videoconferencing applications, and in that case it can be interesting to study the impact of fast handover mechanisms (like FMIP6 presently tested in the Indoor Testbed) on the multicast stream quality when switching from one network to another.
- Koshiro Mitsuya created a IPv6 multicast tunnel between the Nautilus6 office and ULP, which means our office is now connected to the M6Bone, but also that we now need to test it.

We will also investigate other kinds of multicast protocols like XCast, where the list of all the destinations is included in the header, which simplifies greatly the work of the network, but limits its usage to a small number of destinations (early testing showed it was interesting for less than around ten destinations, which makes it perfect for multiple videoconferencing with a few different locations).

2.5 Statistics gathering

To validate the efficiency of the different developed protocols, we need to gather statistics while they are used. This is possible at four different levels if we do it for RTP/RTCP streaming:

- From the IP layer: only bandwidth and packet loss can be calculated.
- Directly from the RTP packets: there is no way to calculate the delay, since RTP is used on UDP. Jitter can be calculated manually as the median of the "relative transit time" of two different packets. Loss, doubles and late packets can be detected by the sequence numbers. All informations on streaming quality can be found at that layer, but it is rather difficult. Some programs exist which do this analyze automatically, but they present several problems: first most of them do not support IPv6; secondly they are usually used in a multicast environment, which means they can not listen to a unicast stream requested by an application which is already listening on the port, as it is the case in my streaming scenarios.
- From the RTCP packets sent by the client: contains loss, jitter and average bandwidth statistics, and round-trip delay can be calculated on the server side (time when RR received - time when last SR sent - time on the client side between SR reception and RR sent). RTCP statistics can possibly be noted "by hand" since there are only a few receiver reports by minute, for example by using Etheral [8] and by exporting the RTCP packets in a format like XML.
- Directly from the application that creates the RTCP reports, and which calculates internal information about RTP streaming, but this solution lacks genericity.

We did not do any statistics gathering for the moment, but as soon as we start setting up our demonstration testbeds, it will be-

come critically important to convince people about the efficiency of the demonstrated protocols.

2.6 Adaptive Applications

2.6.1 Definition

First, what do we mean with the word "adaptive"? In brief, it refers to the fact that the application can not do the same things depending on the network technology it can use. For example it is not possible to stream video if we use a wireless technology with insufficient bandwidth; and even when using high-bandwidth wireless technology like WiFi, the distance to the access point or network overload can greatly affect the available bandwidth: if the user wants to have the best quality video with the current network conditions, the application will have to detect those conditions and adapt the stream consequently.

2.6.2 Techniques

In a mobile environment where network conditions change frequently, this automatic adaptation problem is particularly important to guaranty the quality of applications which have strong real-time and bandwidth constraints like streaming applications, and therefore important to improve the user's experience. But it is not a simple issue: how to detect the bandwidth available for one application at a given moment ? Leads can be followed in several directions:

- Detect the medium used, and encode a fixed corresponding bandwidth in the application; this solution is not very good since the bandwidth of a medium can change a lot, like explained above, but it has the advantage of adapting very quickly, and can be enough for some cases (for example use video if we have WiFi or Ethernet access, and only sound otherwise). Another problem is how to detect the current medium, it is difficult to get information under the IP layer without modifying the kernel.
- Suppose the bandwidth is linked to another network property that we can easily detect, like the round-trip delay; unfortunately, this is usually too simplistic.
- Calculate the packet loss rate, use a lower bitrate if this rate is too high, and try using a higher bitrate if there has been no

loss during a given time; this solution is interesting, but it has to be carefully configured, and the application will probably be slow to adapt itself.

As we can see, there is no perfect solution, especially if we consider the wide range of possible scenarios we want to cover: combining different solutions would be the ideal.
Two solutions are planned to be used in Nautilus6 for the moment: detect the medium used and adapt approximately the application (for example use only sound when connected with cell phone technology) or use the last given solution for sharper but slower adaptation.

2.7 Programming on Zaurus

I will describe here the main application that I have tried to port to the Zaurus: the streaming video player *mp4player* from the MPEG4IP suite, so I could improve the Zaurus operational Experiment and use it in several demonstration testbeds.

Problem statement

To compile a Linux application for the Zaurus, there are two solutions: install a complete compilation environment on the Zaurus, or cross-compile (which means compiling on a platform for a different one) on a desktop. The first solution has the advantage of solving many problems, but the limited memory and processor power of a Zaurus makes it very uncomfortable; I tested some compilation environments on the Zaurus, but they were never complete enough for what I needed, and compilations would take hours. On the contrary, when cross-compiling we can benefit from the power and ease of use of desktop computers; unfortunately there are many other problems, which make cross-compiling a usually difficult task: in fact, many programs were not designed to be cross-compiled, so they make many assumptions during configuration and compilation time that prove to be wrong it that case (for example they compile and launch their own program to compile parts of the application, but when cross-compiling binaries can only be launched on the destination platform).

Cross-compilation environment

The first step was to install a complete cross-compiling environment, including all the necessary headers, libraries (such as Qt and Qtpia

[9], SDL [10], etc), utilities, and an emulator to test the Zaurus programs on the computer. For that I had to test many different solutions and spend a lot of time on Zaurus users forums to discuss problems people had already met, and finally I made a package [11] with all the files to install, several scripts to do the work automatically, complete documentation and a few examples. This package is now being used by other WIDE projects, for example by the iCAR project.

Cross-compiling

The second step was to use this environment to cross-compile applications, and more particularly MPEG4IP. There was many problems, and I only succeeded at the end of August in having binaries running on the Zaurus. Unfortunately, this did not solve everything, presently the sound (as well as other less important parts) is not working... Those problems are presently under investigation, but I do not have any real lead. In fact, the Sharp ROM is based on a Linux 2.4.18 kernel, heavily modified to be able to run on the somewhat particular Zaurus hardware, so it is not surprising that some PC applications do not work on it without modification. Some better ROMs (improved by open-source communities) can be found on the Internet, with more elaborate cross-compiling environments, but the ROM we use is presently the only way to have MIP6 on the Zaurus, even though it is not supported by USAGI anymore since they are now only working on the Linux 2.6 kernel.

Chapter 3

Demonstration Testbeds

Presently, one of our demonstrations which could be of great interest to advertise mobility technologies is in fact a collaboration with one of the most famous WIDE projects, the School Of Internet project.

We will then talk about our own demonstration testbeds, E-Wheelchair, E-Bike, as well as another collaboration we are doing with the iCAR project.

3.1 SOI

3.1.1 SOI project

The SOI Working Group was started in 1997 inside the WIDE organization and Keio University. It aims at using new information technologies, like the Internet, to do e-learnig, i.e. it aims at distributing high-level education, without geographical limitation and to anybody willing to learn. The problem is, how to create a complete and efficient university environment on the Internet? This is particularly important in South-East Asia, where the low economical development makes such education rare, where the presence of many small islands makes access to education in general difficult, but where the use of satellite technologies can provide Internet access everywhere, and with a correct downlink bandwidth.

SOI project has already distributed many courses which were followed by thousands of persons in streaming. To do this, the user only needs a good downlink bandwidth: even if there is a very slow uplink and a very long round-trip delay between the client and the server – as it is the case with satellite transmissions – the video will begin in the worst case a few seconds later, which is not a problem

in this situation since the application is not interactive, the stream only going from the server to the client.
SOI distributes two kind of courses: recorded courses available on-demand, and live courses which are listened at the same time by all the connected clients.

3.1.2 Collaboration objectives

The collaboration between the SOI and Nautilus6 projects aims at using the time that many students (but also many employees who are still willing to learn) spend everyday commuting between their accommodation and their campus or office. Why couldn't we use that time to allow them to follow high-level education? For example when leaving home, a student can begin to listen one course; when he is walking he only listens to the sound, but when he can stop, for example in a train or a bus, he has the possibility to also watch the video or a slide presentation synchronized with the sound.
As a first step, the SOI project wants to realize this for students from Keio University's SFC (Shonan Fujisawa Campus), but also for several usual of its professional clients.

3.1.3 Scenarios

Today, it is necessary to have a desktop or a laptop computer with a good Internet connection to follow those SOI courses. Nautilus6 is therefore interested in making possible the listening of those courses from a mobile platform, like a cell phone or a PDA (Portable Device Assistant).

- The advantage of the cell phone is that everybody already has one and that it is particularly light. On the other hand, it is presently difficult to deploy our own applications on it, they do not always have a headphone plug, the screen is not necessarily large enough to display slides correctly, the processor is not powerful enough to display video with correct quality, etc.
- On the contrary, PDAs are almost ideal tools. Let us consider for example the Zaurus (see section 2.1.2): it is almost a real computer running under Linux, which means that we have a lot of liberty to deploy our own applications on it; its large high-quality color display allows the user to comfortably watch videos and read slides or text; its processor unit is powerful enough to read streaming video, even with good quality; it is quite light and holds in one hand, which is an essential point for

an application which will probably be used while walking, or standing in a bus; finally, the autonomy while being connected to a wireless network and playing video and sound is about one hour and a half, which is enough for usual commuting time. Therefore, we naturally decided to use PDAs for this project.

Remains the problem of how to connect to the Internet: which wireless technology to use to listen to the course?

- On the way between the accommodation and the campus or office, we need to use a packet cellular phone technology like Air-H or b-mobile, for which some cards for PDAs do exist, but which only have a low available bandwidth, usually no more than 64 Kbps. In that case the user will be limited to listening the course, or he may also read text and slides accompanying the sound on the PDA screen.
- When the person arrives near his campus or office (or more simply every time he can connect to a wireless hotspot), he can continue listening to the course by using the private high-bandwidth wireless network like WiFi which provide at least several hundreds of Kbps, and he will then also be able to watch video if he wants too.

This is where a protocol like Mobile IPv6 and adaptive applications become interesting and greatly improve the user-friendliness of the application: MIP6 enables the connection to be kept when moving from one network to the other, whereas the adaptive part of the application makes the best usage of the bandwidth, changing the quality of the streams, and transmitting video or not.

3.1.4 Plans

As we have seen in section 2.7, I managed to cross-compile MPEG4IP for the Zaurus, but some sound issues still remain. Therefore, we plan to organize a meeting with Sharp: they will surely be interested if a SOI member joins the meeting, pointing out that if this experiment is successful it could be a big advertisement for the Zaurus.

But, if ever we do not succeed in deploying a streaming player on the Zaurus and since the collaboration with SOI is valuable for Nautilus6, we will try deploying the same applications on one of the tiny-PC which are beginning to be sold, like the Hagaki-PC from Mitsubishi [12] (*"hagaki"* means postcard in japanese, which represents the dimensions of the PC). The processor is x86-compatible

Figure 3.1: Hagaki-PC and Zaurus

and a standard Linux kernel is able to run on it, so we should not have the same problems as with the Zaurus...

In the next few months, I will continue to lead and develop the collaboration with SOI, and communicate about it (for example several symposiums are planned in the next few months) to promote our technologies.

3.2 InternetCAR

3.2.1 Presentation

The research focus of the InternetCAR working group [3], part of WIDE organization, is on technologies dedicated to connecting cars to the Internet. This main objectives are mobile Internet and the handling of global positioning and security information. The group is also studying the relevant digital architecture within the wider context of the evolution of the Internet ITS (Intelligent Transportation Systems). To this end, it is carrying out research on in-car systems, Internet access, middleware and applications, besides developing shared platforms and formulating specifications, for systems using IPv6.

This environment is a very good option to demonstrate technologies developed in Nautilus6 such as NEMO, which is why we try to set up a collaboration between our two teams.

3.2.2 iCAR camp

I participated to the NAIST iCAR camp in August to gather information and statistics on IPv6 streaming, with WiFi, Air-H and Bi-band technologies, and different kinds of handovers.
For that, I used MPEG4IP as the streaming client, and the two available IPv6 streaming servers, in Nautilus6 and in ENST Rennes (see 2.2). I also had to compress low-bitrate sound files (8-32 kbps, with AAC and Lame MP3) to be used with low-bandwidth connections like Air-H. I recorded all my experiment network traces with Ethereal so we could analyze it later on.

This camp was also a good opportunity to discover what kind of work was done in other WIDE projects, how Nautilus6 could use that work (like GPS sensor queries from a Zaurus PDA), and how we could bring up new kinds of demonstrations.

3.3 E-Wheelchair

3.3.1 Goals

By installing a permanently connected network with many different equipments on a wheelchair, the Nautilus6 projects [13] aims at demonstrating IPv6 mobility mechanisms, particularly NEMO, and how these technologies can be used as a base for an operational communication system. More particularly, it will be used to demonstrate the benefit of network mobility for all types of configurations (i.e. nested mobile networks and multihomed mobile network). Clients will naturally be all the people who usually use a wheelchair, i.e. those with a bad physical condition who require constant monitoring, or those who can just not move by themselves and who are looking for ways to improve their independence.

3.3.2 Constraints

Such a "sensible" system presupposes a list of constraints, that the features of IPv6 and NEMO Basic Support finally enable in a simple and efficient way:

- Mobility, to remain connected even while changing the attachment point to the Internet topology; two different mechanisms can be used. The first one where every equipment is mobility-aware with Mobile IPv6, like what we used in SOI since there was only one mobile terminal; or use the NEMO Basic Support

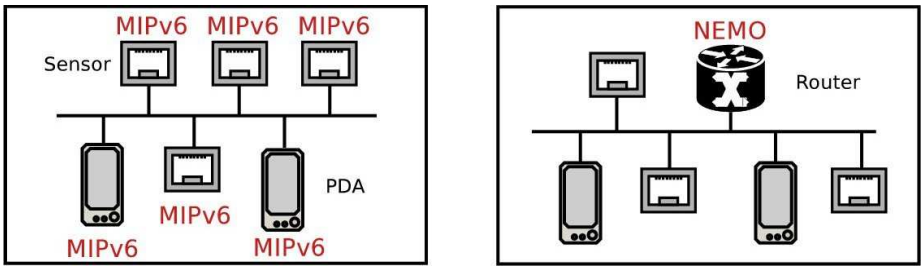

Figure 3.2: Difference between MIP6 and NEMO solution

protocol, where only the mobile routers need to handle mobility, the nodes inside the mobile network staying transparently connected in IPv6.

- Robustness, to avoid undesirable network failures. NEMO allows multihoming, which means several terminals can be connected at the same time to the Internet, or different network interfaces can be connected at the same time on one terminal; therefore, connection can be kept even if one network technology fails.
- Security and confidentiality, to be sure that the transfered informations can not be intercepted or modified: IPSec and AAA.
- Scalability, so that this system will stay operational even if many users begin using it, this property is a characteristic of IPv6.
- The quality of applications with strong real-time constraints (like videoconferencing) can be guarantied with the protocol which will be integrated inside the mobile routers, and which is presently under investigation: it will allow the applications to discuss with the mobile routers to get information on the network conditions, or to indicate their needs such as bandwidth requirement. Thanks to this protocol, applications will be able to adapt the network connection to their needs, or adapt themselves to the network conditions, like explained in section 2.6. Of course all the issues explained in that section are not solved, but now the detection of the network condition is only done by the mobile routers.
- User-friendliness: even if the proposed services can appear to be very useful, we need to be careful to keep the technology easy to use, all the more since this category of users usually has difficulties with it.

3.3.3 Equipment

For the monitoring, the wheelchair will combine information from different types of sources, all the more since the person can not necessarily use its five senses to communicate a possible problem: visual (digital camera), audio (microphone), sensory (movement or pression detector), medical (heart beat, blood pressure, etc) or about the environment (location, temperature, humidity, etc). Some of those sensors already exist, like those which were developed in other WIDE working groups, such as E-Care or iCAR (some of these sensors – a digital camera, temperature and humidity sensors – are already accessible for demonstration inside a mobile network, inside the Nautilus6 showroom [14]). The retrieved information can then be (at least partially) automatically processed, and alert the appropriate staff if necessary.

Then, we needed to decide which equipment we would use for demonstration purpose, and for the Mobile Router using NEMO Basic Support, depending on the autonomy the equipment can provide, its space and weight, and which system can be installed on it. For those reasons we concentrated our studies on SOEKRIS motherboard, Zaurus SL-C760 and SL6000, HP iPAQ and Sony VAIO U50/70; for the operating system, we had the choice between *BSD, or GNU/Linux with kernel 2.4 or 2.6.

- First, HP iPAQ and SOEKRIS motherboard where rapidly excluded of the possible solutions due to power or operating system issues.
- Secondly, in addition to all the limitations described in section 2.7 we have seen we can only use the Sharp PDA with Linux 2.4 kernel, which limits the possibilities for the Mobile IPv6 stack; moreover, we could not configure an ingress interface, preventing us to use the Zaurus as a Mobile Router.
- Finally, the Sony VAIO U50/70 appeared to be a perfect solution, a light-weight and independent x86-compatible computer, which means we can use any operating system, and therefore any Mobile IPv6 and NEMO Basic Support stack.

As a conclusion, we decided to use Sony VAIO with *BSD for the mobile router, with existing NEMO implementations. Inside the mobile network, all the equipments will be connected via an Ethernet hub, supporting Power over Ethernet to power the sensors. The

Figure 3.3: Figure of E-Bike with all the elements composing the NEMO

mobile router will be connected to the Internet via different technologies, WiFi and cell phone packet technologies. Those different interfaces will need the features of adaptive applications to help creating the best user experience. See [15] for further information on equipment.

3.3.4 E-Bike

Nautilus6 will first develop a second demonstration environment, based on the same communication system as E-Wheelchair, but with a more "fun" and convenient direction: E-Bike. It will be an IPv6 PAN installed on a bicycle, and made of several IPv6 devices with several access technologies: small computer (e.g. Sony VAIO), PDA (e.g. Zaurus), various IPv6 sensors (temperature/humidity sensor, 2-axis acceleration sensor, direction sensor), microphone, video-camera, GPS, etc.

Even if we have more issues concerning the size and the power of the different equipments of the mobile network, this testbed offers a much more convenient, easy-to-move and fun demonstration, to promote the underlying technology of E-Wheelchair.

3.3.5 Other demonstration testbeds

In fact, the goal is to create a basic connected system, adaptable according to anyone's wish. Every member of Nautilus6 could then be given a complete demonstration equipment, under the condition to use it regularly and in an original way. We could therefore integrate the demonstration technology into our lives to make it more mature, to help new ideas appear and improve old ones, and finally to become "live demonstrations" and convince people about the maturity and the interest of the underlying technology.

3.3.6 Plans

My objective in those demonstration testbed is more particularly to install a set of applications, which benefit from the mobility functions and help demonstrate them, while respecting all the constraints explained in section 3.3.2. The best examples are streaming and videoconferencing applications, but also monitoring tools that display information from the whole mobile network, such as data from the different sensors and the current connectivity.

E-Bike will be started in October with the first demonstrations planned during November, for example during the Ubiquitous Network Symposium or the Open Research Forum [16], and should be finished by the beginning of 2005. We should begin setting up the E-Wheelchair testbed by the end of the year 2004, benefiting from our experience with E-Bike.

Finally, I will investigate any other technology which could make the testbeds more efficient, like SCTP.

Chapter 4

Improving Nautilus6 working environment

Being a international working group, with people from different countries working at different places and only a few of them working full-time, there is always work to do to improve the working environment and the communication between members, which has proved to be an important point in the project.

4.1 Videoconferencing

After having tested different videoconferencing solutions (see section 2.3), we are now trying to motivate Nautilus6 members to communicate more using those great tools. We have a Polycom in the Nautilus6 room, so I tested it with other members in ENST Rennes and ULP, and I organized a meeting where members from six different locations joined and could discuss successfully about Nautilus6 projects.

4.2 Website

I helped clean, complete and update information on the Nautilus6 website, mostly concerning the Zaurus [6]: all the work I have done can be found on it, and I regularly update the informations on the Wiki page, a very convenient tool that we are now using to synchronize all our work.
I also forced a friend (Tristan Millner) to draw a logo for the project, and had it integrated to the website.

Finally I added a few pages to the website, for example a PHP page to test IPv6 connectivity [17].

4.3 Ashina management

Yasua Ashina is a Japanese student from Todai (Tokyo University), which is now in ENST Rennes for one month and a half, developing an IPv6 SIP application on Zaurus. In July and August, he came to K2 Campus every week; I managed him so he could become confident with the technologies that we use, and prepare him for his stay in France.

4.4 Communication

First, I submitted a paper with the title *Adaptive applications and usages of mobility in IPv6* for the *Journées Scientifiques Francophones*, in Tokyo.

I also participated to on WIDE meeting and the September WIDE camp to communicate about the work done in Nautilus6 to other WIDE members.

The communication will become more and more important as our demonstration testbeds begin to be operational, and we plan to participate in many conferences and meetings in the next few months.

Chapter 5

Conclusion

A lot has been said these last years have about ubiquitous Internet, or about the fact that one day, in the *future*, all our everyday life objects would be connected together in intelligent networks, allowing a whole new set of applications and services to be deployed for the general public, just like what happened – and is still happening – with the arrival of Internet; science-fiction authors have already imagined futuristic usages for those networks that could change and simplify the life of users, but most of these features had yet to be realized... With the development and deployment of IPv6 networks, associated with the mobility-related technologies studied in the Nautilus6 project, we are coming one big step closer to this goal. Of course, such protocols are for the moment recent or even still under development, which means that applications and services which take benefit from them are usually very simple; we are still now in the early stages, developing, implementing, testing and validating, but those basis will enable in the next few years the use of those still "futuristic" network applications, which we can begin to imagine with the demonstration testbeds we are developing, like E-Wheelchair. Therefore, I believe that the work we have done during the last five months, concerning IPv6, Mobile IPv6 and NEMO deployment, testing and demonstrating, and more generally everything we are now doing in Nautilus6 at the intersection of many mobility concerns, is a necessary basis for many other useful directions...

Concerning my personal improvement, this five months have made me completely at ease and independent under a Linux environment, since I have been doing many different works on this platform, and they made me discover many aspects and issues concerning networking, and more particularly the mobility aspect. It was a good oppor-

tunity to learn how to work as an member capable of independent actions, but integrated in a team and trying to reach certain goals common to everybody in the project. Finally, my stay in Japan was very interesting since I could discover a very different way of working and communicating, as well as a totally different culture...

Bibliography

[1] David B. Johnson, C. Perkins, and Jari Arkko. Mobility Support in IPv6. Request For Comments 3775, IETF, June 2004.

[2] Vijay Devarapalli, Ryuji Wakikawa, Alexandru Petrescu, and Pascal Thubert. NEMO Basic Support Protocol. Internet Draft draft-ietf-nemo-basic-support-03.txt, IETF, June 2004. Work in progress.

[3] Thierry Ernst and Keisuke Uehara. Connecting Automobiles to the Internet. In *ITST: 3rd International Workshop on ITS Telecommunications*, Seoul, South Korea, November 2002.

[4] C.W Ng, Eun Kyoung Paik, and Thierry Ernst. Analysis of Multihoming in Network Mobility Support. Internet Draft draft-ietf-nemo-multihoming-issues-00, IETF, July 2004. Work in progress.

[5] Test your IPv6 connectivity http://www.nautilus6.org/testipv6.php.

[6] Operational Testbed Experiment for Zaurus users http://www.nautilus6.org/operation/zaurus.html.

[7] 6NET Project http://www.6net.org/.

[8] Ethereal Network Protocol Analyser http://ethereal.sourceforge.net/.

[9] Trolltech, creators of Qt and Qtopia http://www.trolltech.com/.

[10] Simple DirectMedia Layer http://www.libsdl.com/.

[11] Complete cross-compilation package for Zaurus http://www.nautilus6.org/operation/download/cross-install-2.95.tar.bz2.

[12] Hagaki-PC from Mitsubishi
http://www.hagakipc.jp/.

[13] Thierry Ernst. E-Wheelchair: A Communication System Based on IPv6 and NEMO. ICOST, Singapore, September 2004.

[14] Nautilus6 IPv6 showroom
http://www.nautilus6.org/show.html.

[15] Romain Kuntz. Implementations, equipments, operating systems for NEMO demonstrations. Technical report, Nautilus6, July 2004. www.nautilus6.org/confidential/doc/NEMO-OS-EQUIP-IMPL-Romain_KUNTZ.pdf.

[16] Open Research Forum
http://www.kri.sfc.keio.ac.jp/english/event/orf.html.

[17] Test your IPv6 connectivity
http://www.nautilus6.org/testipv6.php.

List of Figures

3.1 Hagaki-PC and Zaurus 25
3.2 Difference between MIP6 and NEMO solution 27
3.3 Figure of E-Bike with all the elements composing the NEMO 29

B.1 Mobile Network moving with NEMO 45

C.1 Cyberté project 47
C.2 Original interface 49
C.3 Modified interface with bitrate display and control . . 50

D.1 New internal architecture of a Mobile Node 52

Appendix A

Glossary

AAC: The audio codec of the MPEG-4 standard. See MPEG-4.

H.323: An ITU-T recommendation for packet-based multimedia communications systems. This recommendation defines the different multimedia entities - Endpoint, Gateway, Multipoint Conferencing Unit (MCU) and Gatekeeper - and their interaction to compose a multimedia system. This recommendation is used for many voice-over-IP applications, and is heavily dependant on other recommendations, mainly H.225 and H.245. RADVision

Home Agent: In Mobile IPv6, the server which provides a Home Address to the Mobile Node, and creates a tunnel between them in order to redirect all the packets sent to the Mobile Node to its Care of Address. Refer to [1] for all the other technical terms concerning Mobile IPv6.

L2TP: Layer 2 Tunelling Protocol, PPP extension IETF Working Group. It uses a reliable signalling channel, a UDP unreliable data channel, tunnel level authentication, etc. It can be used to create Virtual Private Networks, avoid firewalls, have global addressing behind a NAT or have a remote IP address.

MCU: Multipoint control unit. A bridging or switching device that supports multipoint videoconferencing.

MPEG-4: It is a standard issued of the Motion Picture Experts Group (MPEG), which is a international standards group working under the direction and rules of the International Standards Organization (ISO) and of the American National Standards Institute (ANSI). Its development began in the mid-1990s and is comprised of more than 1,000 pages of documents outlining

a wide range of multimedia elements, file formatting, and presentation alternatives. Much of what is of interest these last years seems to be the video codec, the audio codec (Advance Audio Coding) and file format that are specified in sub-sections of the draft-standard. The Internet Streaming Media Alliance (ISMA), founded by the biggest international media companies (Sony, Philips, Apple, IBM, Sun, Time Warner, etc) tries nowadays to impose an open standard platform, where media compression will be achieved by the MPEG-4 standard.

Compared to MPEG-1 (one of which part is the famous MP3), which was designed for coding progressive video at a transmission rate of about 1.5 Mbps (designed for Video-CD and CD-i media), and to MPEG-2 which was designed for coding interlaced images at transmission rates above 4 Mbps (used for digital TV broadcast and DVD), MPEG-4 is a much more ambitious standard: it addresses speech and video synthesis, fractal geometry, computer visualization, and an Artificial Intelligence approach to reconstructing images. MPEG-4 addresses a standard way for authors to create and define the media objects in a multimedia presentation, how these can be synchronized and related to each other in transmission, and how users are to be able to interact with the media objects. Nowadays, even if the file format has some difficulties to impose itself (due to the competition with proprietary formats and the existence of high licensing fees), the video codec is becoming very popular through different compliant codecs, such as Quicktime 6, DivX or the open-source XviD (in fact, the other famous proprietary codecs – such as Windows Media Video 9 and Real Video 9 – are also MPEG-4 adaptations), and it is presently the best solution for streaming video.

Finally, MPEG-21 will provide in a few years a larger, architectural framework for the creation and delivery of multimedia. It will define seven key elements: *Digital item declaration*, *Digital item identification and declaration*, *Content handling and usage*, *Intellectual property management and protection*, *Terminals and networks*, *Content representation* and *Event reporting*.

PAN: Personal Area Network; a network in which all interconnected computers (usually small electronic devices, PDA, cell phone, watch, etc) are distributed locally around an individual person (e.g., connected via Bluetooth). Contrast with local area network (LAN) and metropolitan area network (MAN) and wide area network (WAN).

PDA: Personal Device Assistant. Any small hand-held device that provides computing and data storage abilities. Examples of PDAs include the Palm Pilot and the HP Ipaq.

RTP: Real Time Transport Protocol; it is both an IETF Proposed Standard (RFC 1889) and an International Telecommunications Union (ITU) Standard (H.225.0). It is a packet format for multimedia data streams, usually working over the User Datagram Protocol (UDP). RTP is used by many standard protocols, such as RTSP for streaming applications, H.323 and SIP for IP telephony applications, and by SAP/SDP for pure multicast applications. It provides the data delivery format for all of these protocols.
Information in the RTP header tells the receiver how to reconstruct the data, and describes how the codec bit streams are packetized. More precisely, RTP components include: a *sequence number*, which is used to detect lost packets; *payload identification*, which describes the specific media encoding so that it can be changed if it has to adapt to a variation in bandwidth; *frame indication*, which marks the beginning and end of each frame; *source identification*, which identifies the originator of the frame; and *intramedia synchronization*, which uses timestamps to detect different delay jitter within a single stream and compensate for it. RTP can be associated with the Real Time Transport Control Protocol (RTCP), which allow applications to monitor the data delivery, controlling the number of lost packets, round-trip time, jitter and other Quality of Service parameters.

RTSP: Real Time Streaming Protocol; it is a client-server multimedia presentation control protocol, designed to address the needs for efficient delivery of streamed multimedia over IP networks. It leverages existing web infrastructure (for example, inheriting authentication and firewall features from HTTP) and works well both for large audiences as well as single-viewer media-on-demand.
Contrary to HTTP, RTSP is designed to work with time-based media, such as streaming audio and video, as well as any application where application-controlled, time-based delivery is essential. It has mechanisms for time-based seeks into media clips (with commands such as SETUP, PLAY, PAUSE, etc), with compatibility with many timestamp formats, such as SMPTE timecodes. In addition, RTSP is designed to control multicast delivery of streams, and is ideally suited to full multicast so-

lutions, as well as providing a framework for multicast-unicast hybrid solutions for heterogeneous networks like the Internet.

SDP: The Session Overview Protocol describes multimedia sessions for the purpose of session announcement, session invitation and other forms of multimedia session initiation. Session directories assist the advertisement of conference sessions and communicate the relevant conference setup information to prospective participants; SDP is designed to convey such information to recipients. SDP is purely a format for session Overview - it does not incorporate a transport protocol, and is intended to use different transport protocols as appropriate including the Session Announcement Protocol (SAP), Session Initiation Protocol (SIP), Real-Time Streaming Protocol (RTSP), electronic mail using the MIME extensions, and the Hypertext Transport Protocol (HTTP). SDP is intended to be general purpose so that it can be used for a wider range of network environments and applications than just multicast session directories. However, it is not intended to support negotiation of session content or media encodings.

Streaming: It is a data transmission mechanism, where compressed multimedia streams (usually sound or video, or both) are sent over a network like Internet, and played by the client as they arrive (at least as soon as the reception buffer is full), which means that a user does not have to wait to download a large file before seeing the video or hearing the sound. This technique can be used in many different cases, such as unicasting (one server - one client, used for on-demand delivery) or multicasting (one server - many clients, used for video-conferencing, Internet radio, live-broadcasting, etc).
Even though the streaming is a recent technology (late-1990s), it is nowadays becoming widely used in a large variety of applications, due to Internet's democratization all over the world and the development of high-bandwidth accesses.

Appendix B

IPv6 and mobility

Most of today's Internet uses IPv4, or Internet Protocol version 4, which has proven to be robust, easily implemented and interoperable, and has stood the test of scaling to the size of today's Internet by using different mechanisms such as NAT. However, the initial design of IPv4 did not take into consideration several issues that are of importance today and suffers from problems in various areas: address space limitations, inefficient routing, lack of support for security, lack of autoconfiguration, lack of QoS support, and poor mobility support.

The IETF had two options. The first option was to fix IPv4 and risk the continued degradation of the Internet model, which would lead to more complex and volatile network services, lower performance, and less robust, less secure, and less manageable networks.

The second option was to replace IPv4 with a newer version and enable simple and stable network services, higher performance, and more robust, secure, and manageable networks. They wanted to avoid doing changes now to IPv4 and then repeat the same exercise a few years later on. Therefore, they made the decision to design a new protocol. As a result, the IETF defined IPv6 to fix the problems in IPv4 and to add many improvements for future networks.

The design philosophy of IPv6 is a scalable protocol that provides a large address space with a simple structure, an original end-to-end environment, a NAT-free network, fast processing, and many features needed by current and future applications. Migrating from IPv4 to IPv6, and IPv6 deployment should not be expensive. IPv6 should inter-operate with IPv4 and provide tools and mechanisms needed by hosts running different IP versions to communicate with

each other, and to enable applications to work with both IP versions.

B.1 IPv6 Features

IPv6 was designed with enhanced features compared to IPv4. Its major features are:

- Large Address Space: IPv6 provides a 128-bit address field. This extended address space is very essential, as IP addresses will be assigned to mobile phones, home appliances, motor vehicles, and other equipment. In addition, with such a huge address space, we can create multi-level hierarchies of addresses, simplifying routing problems – requiring simpler routing algorithms and less space needed for routing tables.
- New Types of Addresses: IPv6 introduces the concept of scoped addresses and defines three types of addresses: unicast (global, link local, site local), multicast, and anycast.
- Autoconfiguration: IPv6 provides hosts with the ability to configure themselves automatically without the use of a stateful configuration protocol. A host can also use router discovery to determine the addresses of routers, additional addresses, and other configuration parameters. This feature allows hosts to discover automatically all the information they need to connect to the Internet, without any human intervention.
- New Streamlined Header Format: in addition, IPv6 is much more flexible in its support of options through extension headers. Extension headers encode optional Internet-layer information. They are placed between the IPv6 header and the upper layer header in a packet and are chained together using the next header field in the IPv6 header. There are six different extension headers: Hop-by-hop Options, Destination Options, Routing, Fragment, Authentication, and Encapsulated Security. The next header field indicates to the router which extension header to expect next. If there are no more extension headers, the next header field indicates the upper-layer header (TCP header, UDP header, ICMPv6 header, an encapsulated IP packet, or other items).
- Better Network Management: IPv6 provides enhancements that allow better network management such as network renumbering, which make it simpler to move a whole network to a new

ISP by reconfiguring the router with the new routing prefix from the new ISP.

- Support for IPsec: the IETF has mandated support for Internet Protocol Security (Ipsec) with IPv6 so it will not be an optional extension, as was the case with IPv4.
- QoS: the IETF has specified two approaches, integrated services and differentiated services, to provide guaranteed and selectable Quality of Service (QoS) over the Internet. In addition, IPv6 provides flow labels that can be used to provide QoS by identifying the packets as belonging to a flow. These labels can be used in conjunction with a hop-by-hop routing extension header (allowing predefined routes) and the priority field (allowing for QoS). The flow label also serves as a key in the router cache to reduce the amount of processing. When a router first receives a datagram, it can cache the flow label and next hop so as to save time when the next datagram arrives with the same flow label. This technique reduces router processing time considerably.

As a result, IPv6 is feature-rich, fixing many of the problems of IPv4 and adding much new functionality.

B.2 Mobile IPv6

Mobile IPv6 is a protocol developed by the IETF MobileIP Working Group, which enables host mobility.

B.3 NEMO Basic Support

NEMO (for NEtwork MObility, a Working Group created at the IETF in november 2003) Basic Support, an backward compatible extension to Mobile IPv6 , is a protocol which adds mobility features to IPv6 routers, to manage the mobility of one entire network which changes its point of attachement to the Internet, and thus its reachibility in the Internet topology. Thanks to NEMO, only the computer that connects the mobile network to the Internet (a Mobile Router) needs to be added mobility features.

Its working is very close to Mobile IPv6, except that the Mobile Router informs the Home Agent about the IPv6 prefix it advertises; therefore, the HA knows how to route the packets whose destination

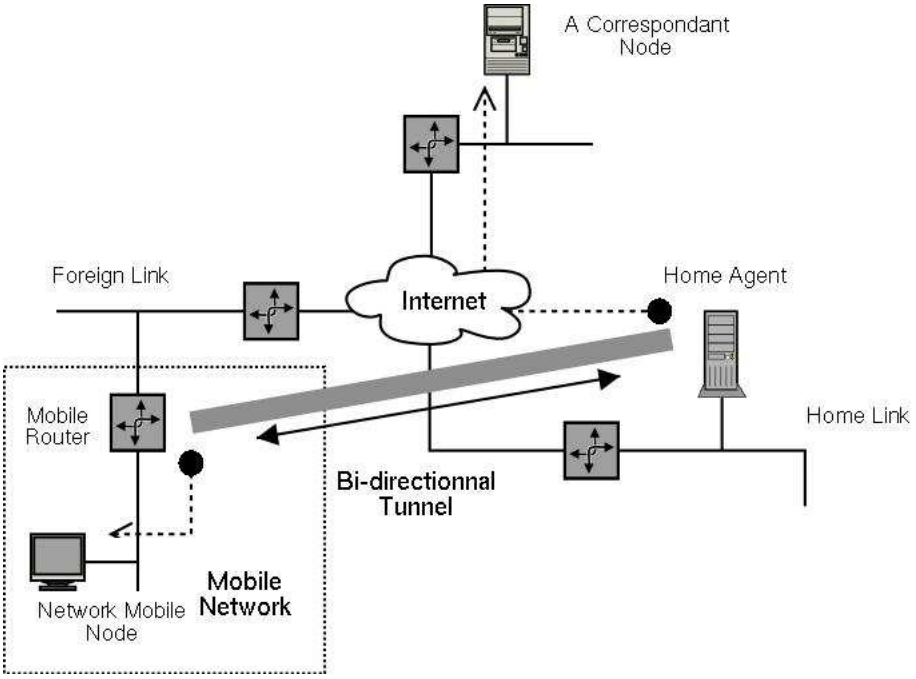

Figure B.1: Mobile Network moving with NEMO

node is inside the mobile network.

This approach is usually much better than running Mobile IPv6 on each node of the mobile network: this can be expensive especially for small devices, but can also generate a lot of signalization traffic each time the mobile router changes its point of attachement in the Internet topology.

Appendix C

Student Project

As an introduction to my internship in the Nautilus6 project, I realized an adaptive streaming application during my 3rd year in ENST Bretagne, in collaboration with the Cyberte Project [Cyberte].

C.1 Cyberté Project

Nowadays, there are frequently more than one network interface for each device, using different mediums: Ethernet, Wi-Fi or HiperLan, BlueTooth, Air-H, GPRS (or UMTS in the years to come), etc.
Each of those mediums have very different characteristics, such as: different bandwidths, different security levels, different packet loss rates, different cover zones; moreover, each medium has a number of other parameters, which can be seen as "costs": financial cost, energy cost, and so on.
In brief, one would want to continue using network connectivity without having to change manually from one interface to the other, each time he moves or each time he wants to use different applications; moreover he would also like to have the cost of his connections automatically minimized.

The Cyberté project (*www.cyberte.org*) was focused on the problems appearing when a mobile device has different interfaces (called multihoming, and usually each interface uses a different medium), and that it wants to do simultaneous or successive uses of these interfaces. The project regroups a mobile telecommunications operator (France Telecom - Orange), a wireless devices manufacturer (Cisco) and three research laboratories (ENST Bretagne, IRISA, LSIIT).
The main goal (presented on Figure C.1) was to create a *Multiple Interfaces Management Layer*, between the layers 2 and 3 of the net-

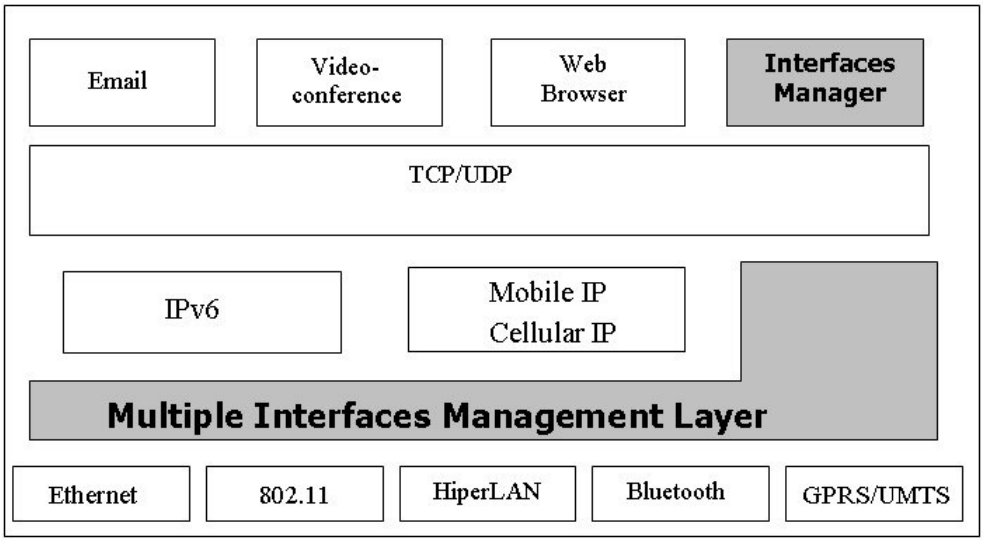

Figure C.1: Cyberté project

work stack, which would abstract the features of the different layers 2 (mostly wireless mediums), to offer a simple and unified view to the IPv6 layer. High-level applications would then be able to use IP features (usually through higher-level protocols such as TCP or UDP) without having to know which layer 2 protocol is used underneath, the Multiple Interfaces Management Layer automatically determining which is the best one to use. Finally, the *Multiple Interfaces Management Protocol* (MIMP) aimed to set up an interaction between the applications, the user preferences and the network interfaces to adapt in real time the use of the multiple interfaces (cf a complete Overview in Figure D.1).

Adapting IP protocols in mobile devices usually requires much work, due to the inherent characteristics of radio transmissions: low or very irregular bandwidth, high error rates. Moreover, due to their small size, mobile devices must imperatively minimize the energy consumption to maximise the batteries' life span, which brings new constraints on the protocol mechanisms (minimal broadcasting mechanisms, no network connection when not required, etc).
The differences between desktop or mobile environment have today to be managed at an application level. Applications cannot be directly adapted to be run on a mobile device, and require deep modifications, which are usually done for a few known mobile targets, and the solutions lack genericity. The Cyberté project has therefore

proposed generic mechanisms, allowing applications to adapt themselves automatically to the different execution environments.

Cyberté members had already developed some applications which took profit of the kernel-level generic interfaces, to be able to run while dynamically changing wireless network connections. Through these interfaces, the applications could retrieve information like the different parameters which were described in the previous paragraph, bandwidth, cost, security, QoS, etc. But the adaptation work is still important, the final goal of the project was to automate the interface management to make the adaptation very simple; my project was to be used as a demonstration of this adaptation.

Therefore, we needed an application which would intensively use network connections, and which could adapt itself to different bandwidths (a mail application can easily adapt itself to any bandwidth, but the benefit is far from critical...).
For these reasons, it was decided to use a streaming application, to watch multimedia streams (video, sound, etc) on computers over a network.
Starting with an existing open-source streaming application, the main goal was for the application to detect the changes in the current network conditions, and automatically adapt the bandwidth of the stream in consequence.

C.2 Bibliographic report

First, I wrote a bibliographic report, which was divided in two parts:

- A presentation of the different technologies which I was going to use during the project; a brief summary of those technical terms can be found in the Glossary.
- A comparison between different open-source streaming applications; the search criteria were the protocols, video codecs and file formats supported, whether it was developed under Linux or not, and IPv6-aware or not. The summary can be found in the Appendix, Table D.1.

C.3 Implementation

After studying these various streaming applications, I decided to use MPEG4IP on the client side (see Fig. C.2), Darwin Streaming

Server on the other side, and to modify only the client application.

In the end, the interface should not have been any different with the original release of the MPEG-4 player. That means that the user would have just requested his streaming server for a video as usual, which would be delivered to him in the best possible manner, without further manipulations.
More precisely, when a request for a multimedia file is done:

- The client fetches from the server a SDP file, which contains the description of the different available bitrates for the media.
- Thanks to Cyberte's Multiple Interfaces Management Protocol, he collects information about the currently available interfaces, and begins playing the optimal bitrate file.
- Each time a change occurs in the network interfaces, the MIMP executes a script to update the bandwidth information; the player once again chooses the optimal bitrate, and changes the playing file if necessary.

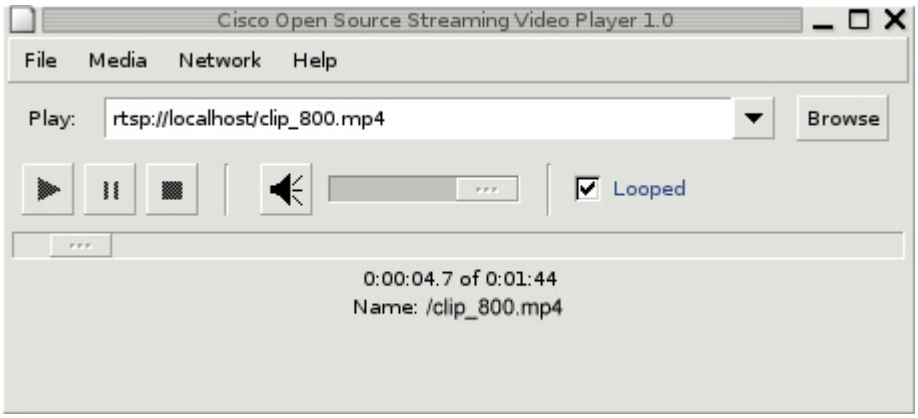

Figure C.2: Original interface

Finally, the result was a little different:

- I did not modify the SDP protocol to add the possibility to announce multiple bitrates for the same file. The different bitrates were directly encoded in the client application, but it is not a problem for a demonstration application.
- The script that could be called by MIMP to update the network informations inside the client was operational, but since I did

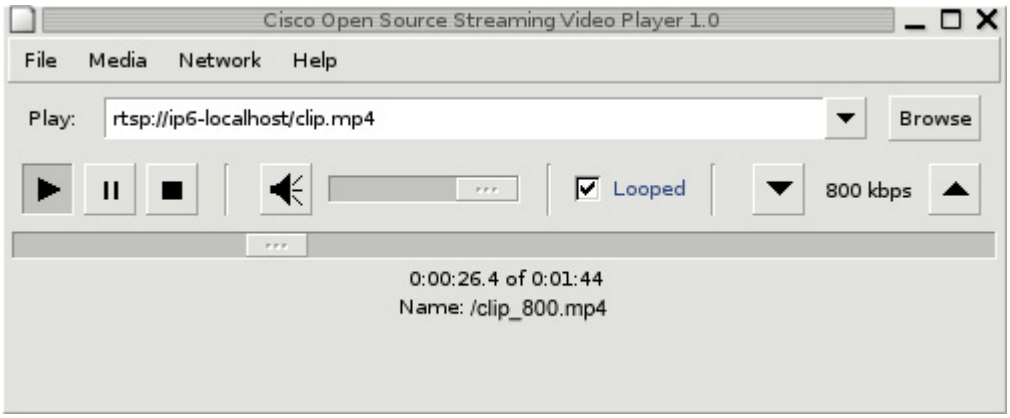

Figure C.3: Modified interface with bitrate display and control

not have a computer with this feature installed, I added for testing a text box displaying the bitrate of the currently read video, and two buttons for the user to change the bitrate while playing a video (see Fig. C.3).

Each time the user enters a new video address or changes the bitrate, the application adds to the file name an extension indicating the bitrate (for example, `clip.avi` would become `clip_800.avi` if the current bitrate is 800 kbps, and start playing the new file from the server, at the same moment the previous video was stopped.

Appendix D

Additional figures

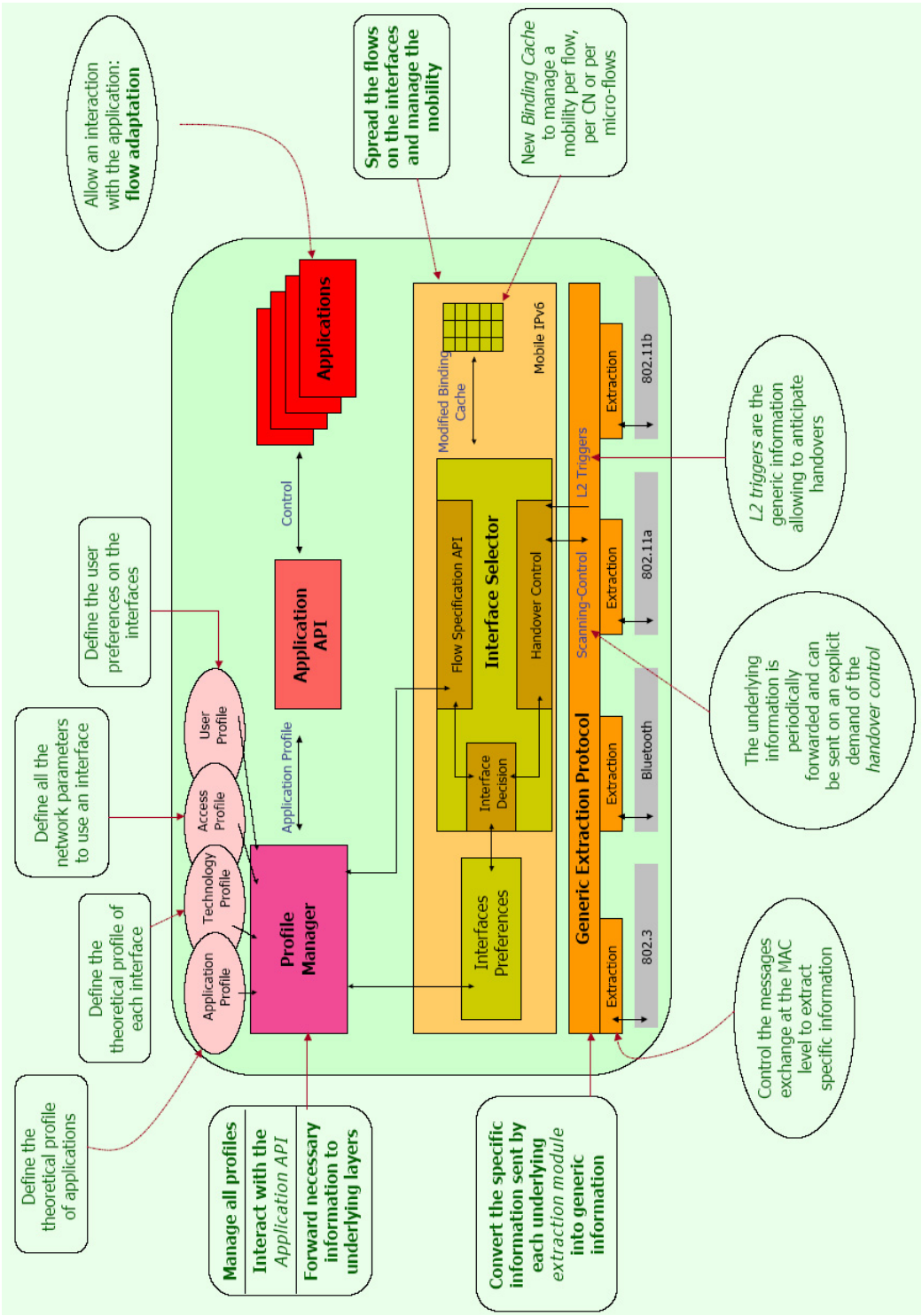

Figure D.1: New internal architecture of a Mobile Node

Application	Server	Client	Streaming protocols	RTSP	MPEG-4	Other video codecs	IPv6	License
Danubio Streaming Architecture	Application framework		RTP		X	MPEG, MJPEG	?	GPL
Live.com Streaming Media	X	X	RTP/RTCP	X	X	MPEG, H.263, MJPEG	Proposed	LGPL
	+ streaming libraries							
VideoLAN	X	X	RTP	Partial	X	MPEG1& 2, WMV, H.263, ...	X	GPL
Helix	X	X	RTP, RDT	X	Started	RealVideo	Proposed	RPSL
Darwin Streaming Server	X		RTP	X	X		No, but modified versions exist	APSL
MPEG4IP		X	RTP	X	X		X	Different open-source licenses

Table D.1: Streaming applications comparison in December 2003