Workshop on real-time for multimedia (RTMM), Catania, Italy, June 29, 2004

Contents

Part II - Presentations

Workshop on Real-time for Multimedia

Special Focus on Real-time Middleware for Consumer Electronics

Catania, Sicily, Italy, June 29, 2004

in conjunction with the
16th Euromicro Intl Conference on Real-Time Systems
Catania, Sicily, Italy, June 30 - July 2, 2004

Introduction to workshop proceedings

Peter van der Stok

The workshop has been realized thanks to the EUROMICRO technical committee who offers the infrastructure to organize a number of satellite workshops together with the ECRTS conference. In 2004 the Real-Time Multimedia (RTMM) workshop was organized among others. The special focus of this RTMM workshop was: "Real-time Middleware for Consumer Electronics". The RTMM workshop provided a forum to bring together researchers and industrial users in the area of middleware for consumer electronics. Specifically the real-time aspects associated with video steaming were emphasized.
The results obtained from FABRIC, an FP5 IST European project, incited the FABRIC partners to spend an extra effort to disseminate the newly gained insights to a larger audience. The satellite workshops offered by EUROMICRO technical committee seemed to be an ideal vehicle. To widen the audience beyond the partners, a general call for papers was issued in collaboration with the RTMM workshop. The accepted contributions led to a program that was FABRIC oriented in the beginning and widened its scope during the late morning and the afternoon sessions. The workshop was concluded with a panel that discussed with its chairman and the workshop audience a varied set of subjects centered around middleware for Consumer Electronic devices.
The workshop started with an introduction of the results of the FABRIC project. These same results were detailed in the three following lectures. The publish/subscribe model is taken as central communication mechanism. This has a profound influence on the way the application programs are written. Streams are published anonymously and a displaying device has to subscribe to the stream and does not directly know the physical origin of the video that is displayed. The software and hardware architectures that support this paradigm are explained. It is shown that the publish/subscribe paradigm and application development can be mapped onto the principles of the

Universal Plug and Play (UPnP) standard. The successful mapping increases the possibility that the same communication paradigm and programming model can be used on other standards, thus paving the way to interoperability between the many standards that exist in the home network arena. The performance aspects of this mapping are discussed in detail. With specific care, meaning a new approach to writing communication stacks, the copying of data within a processor can be reduced to a minimum, thus guaranteeing no or very little performance decrease.
A clear advantage of the publish/subscribe paradigm is the way the QoS management software can be constructed. A good separation between "local" and "global" concerns was realized. It is also shown that the QoS management scales well with network size and it adapts easily to evolving network reconfigurations.
After the FABRIC-specific presentations, two presentations about standardization followed. One addressed the area of standardization in the home network domain and reported on the differences and commonalities between the standards and their objectives. The other addressed the MPEG21 standard and how this standard can assist in making systems more robust by describing the quality attributes of software components.
The last two talks described two specific subjects relevant to the workshop: group management and OSGi. The first subject shows how the inclusion of group membership within the middleware can be done and the advantages of such a group membership scheme in a highly dynamic network environment. The last talk concluded with a demonstration of how a service outside the home network can be included in the home network and made accessible to applications in the home network with the aid of OSGi. The chosen home network standard was HAVi.
The panel concluded with a very lively debate between panelists and audience. After the opening issues like "what are relevant real-time concepts for multi-media?" and "how should that affect the UPnP and HAVi standards?", the predictability and ensuing future of wireless communication systems were discussed. The discussion quickly focussed on the question what useful properties the resources for video streaming need to have.
It is a pleasure to thank the ECRTS organization, the contributors and the panelists for a very successful and certainly repeatable workshop experience.

Workshop organizers

- Jean-Dominique Decotignie, CSEM, Switzerland
- Petru Elles, Linkoeping University, Sweden
- Gerhard Fohler, Malardalen University, Sweden
- Christian Hentschel, University of Cottbus, Germany
- Johan Lukkien, Technical University of Eindhoven, The Netherlands
- Hermann de Meer, University of Passau, Germany
- Raj Rajkumar, CMU, USA
- Liesbeth Steffens, Philips Research, The Netherlands
- Peter van der Stok, Philips Research, The Netherlands

Part I – Papers

The FABRIC project: Overview and Results

Peter van der Stok
Philips Research
Prof Holstlaan 4
5656 AA Eindhoven; The netherlands
Peter.van.der.Stok@philips.com

Abstract

The FABRIC project aims at the integration of middleware standards used in home networks to provide high quality streaming of video over a heterogeneous network. without introducing new standards.

1 Introduction

At this moment we are confronted by a large set of communication and interoperability standards in the home-networking domain. Example are HAVi [1] and UPnP [2][3]. Within the context of a single standard, devices can be introduced into the home by simply unpacking them and switching them on, possibly accompanied by connecting them to some wiring infra-structure. Currently, devices from different standards are completely isolated from each other. Each standard offers specific advantages to the equipment adhering to it and we can safely assume that these standards are here to stay. At the same time the advance of new technology (like 3G roaming devices) will result in newer standards. The users and purchasers of this equipment are not interested and probably not aware of the incompatibilities of standards. Confronted with the cited incompatibilities future customers will NOT acquire these new technologies. Integration of the different standards is absolutely essential to allow a transition from the current islands of technology to one integrated provision of services (roaming or wired) within the home and between the home and service providers.

The "Federated Applications Based on Real-time Interacting Components" (FABRIC) [13] project aims at developing *an architecture* in which several interoperability standards and technologies in the home networking context can be integrated. More than integration alone, a FABRIC application manages the complete network to satisfy *Quality of service (QoS)* requirements. The design is guided in this process by the requirements of a chosen application: multiple roaming multimedia streams. The leading architectural pattern is the publish/subscribe pattern as specified in the High level Architecture (HLA) IEEE 1516 standard [4].

The FABRIC-architecture especially caters for dynamic network configurations allowing frequent additions, removals and roaming of devices.

The FABRIC-architecture supports the provision of real-time multimedia streaming. The timing specification and realization by the FABRIC communication media provides the end-to-end Quality of Service (QoS) required by the multimedia streaming. Reactions to system changes need to be done in a timely manner to support high quality video streaming over wireless media with rapid bandwidth fluctuations (smaller than 10 ms). Configuration changes and video stream settings need to be communicated to the involved devices within time scales of a few 100 ms.

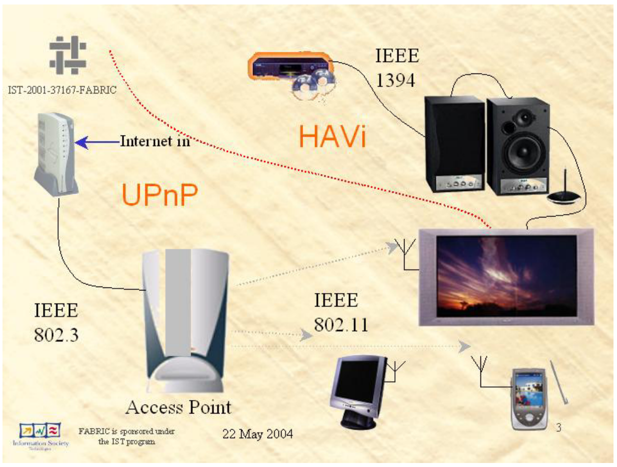


Figure 1, example home network

Figure 1 shows a possible home network configuration. A set of UPnP devices are connected wired with IEEE 802.3 (Ethernet) and wireless with IEEE 802.11 (WiFi). The HAVi devices are interconnected with IEEE 1394 (Firewire) [9]. The PDA based on UPnP and IEEE 802.11 should be able to establish a connection between the HAVi DVD player and the portable screen. This operability aspect is covered by one part of the FABRIC project. The second part enables the streaming of video over the heterogeneous network while maintaining a sufficiently high video quality for the end-user. The paper is structured in the following way. First the architecture is discussed with emphasis on the legacy and standardisation aspects. The distribution of the functionality and its robustness against network (re)configurations are discussed. Consecutively, some extra-functional properties are discussed and the way they are viewed by the FABRIC consortium. The paper concludes with the deceptions and pleasant surprises we experienced looking back at the end-results.

2 Standardisation and legacy

The architecture is discussed in [5]. Another overview is presented in [10]. Two aspects are shown (1) the interoperability between devices that allows an application on a device of one standard to control or communicate with a service on a device on another standard, and (2) the streaming control over a network with fluctuating bandwidth. The publish-subscribe paradigm has been used for the first aspect. More information on its consequences is described in [6]. The second aspect is fully described in [7].

The chosen architecture has to fulfil some requirements with respect to legacy and standardisation aspects. For the acceptance of the FABRIC results it is important not to introduce yet another new standard. In [8] it is described how the standards already very much fulfil the interoperability requirements within a given standard. There is clearly no need for yet another standard. Traditionally, interoperability between standards is attained by a gateway. The gateway receives the description for an function request, translates the request and send it on to the designated device. Today's major problem with this approach is the incompatibility of semantics between the proposed standards. For example a device in one standard can contain the function "integrated TV and DVD" which is not supported in another standard that only knows the separate DVD and TV functions. Requests to one service cannot be translated to another without unwanted side effects.

The description of the middleware services can be done in a fixed way, by providing standardised names for functions or numbers, or in a more dynamic way by providing self-describing services. The advantage of self-description of the service is to render the middleware standard independent of the offered services (they need not be standardised) but only a standardisation of the service description language is needed. In the case of a gateway, the addition of a new service means adapting the gateway every time a new service is introduced.

It is the great achievement of the FABRIC project that the semantics of the service are assumed to be known to the application that wants to control the given service. The FABRC results make it possible that an application accessing a service with HAVi semantics can run on any other device without knowing the middleware standard of the application-hosting device and the service-hosting device.

The FABRIC layer on top of a given middleware standard abstracts from the syntax of the middleware standard. However, the FABRIC consortium was obliged to introduce a service description language (based on XML).

An alternative approach is to not add a new service description language but to pass the service description to the application in the syntax of the middleware standard used by the service. Also in this case the gateway does not need to convert the contents of the messages. However, the application must be middleware aware thus reducing the abstraction level of the application. The FABRIC consortium has expressed a clear preference for the higher abstraction such that the application parses services description expressed in the FABRIC syntax.

The existence of legacy devices (i.e. devices belonging to a given middleware standard without a FABRIC software layer) is an important aspect. The proposed architecture allows applications on FABRIC-enabled devices to interoperate with services on other FABRIC enabled devices. To describe the legacy and inter-operability possibilities, the following notation is used. X-application is an application that executes on a device with a middleware of standard X. F-application is an application that executes on a FABRIC-enabled device. X-service is a service that conforms to standard X and runs on a device with middleware standard X. An F-service is a service that executes on a FABRIC-enabled device. The following possibilities exist:

1. An F-application accesses an F-service
2. An F-application accesses an X-service
3. An X-application accesses an X-service

It is not supported that an Y-application accesses an X-service when X is unequal to Y. This unsupported possibility is the purpose of most X-Y gateway designs under the restrictions mentioned above.

Possibility 1 is the main goal of FABRIC. Possibility 3 is the main goal of any middleware standard X. The existence of FABRIC-enabled devices does not hamper this access in any way.

Possibility 2 is subject to some boundary conditions. An F-application can only access an X-service when there is a FABRIC-enabled device that runs the middleware of standard X. The second condition is that not all X-services will be accessible but only a predefined set of services. As for the X-Y gateway case, known semantics exist that can be translated from middleware X syntax to FABRIC syntax by the FABRIC-enabled device that runs middleware X. The application has the possibility to perform some standardized commands on the legacy X-service.

Concluding we see that standardisation, abstraction and legacy facilities are closely related concepts. Higher and richer abstractions lead to more standardisation obligations. Legacy facilities lead to lower and/or poorer abstraction for the specific legacy purposes.

3 Distribution

Distribution is of-course of prime importance in a home network. Just providing the possibility to interconnect devices is not enough. The home network is assumed to be very dynamic where portable devices are added and removed, other devices are switched on and switched off. In this network the “plug and play” facilities are important, but even more important is the “unplug and play” facility. The latter means that when a device is switched off, no essential controlling element is removed that leads to a complete break-down of the network. For example, disconnecting a DVD player should not affect the showing of a football game that is broadcast over the cable network. Such unwanted behaviour is possible when the DVD player contains the only DHCP server and the TV cannot update its IP address.

The above example cites the DHCP server, but the FABRIC architecture also relies on some central decisions servers. The publish-subscribe encourages a rather simple mechanism to provide tolerance against server removals. All essential information (e.g. the service description, the QoS information) is published on the network. The central repository (or decisions authority) receives this information and publishes its eventual results. The hot standby paradigm allows several standby servers to receive all exchanged information. When at a particular moment the actual server is removed another server can take over the required functionality.

To make this work properly, a server election strategy is needed. The first device that starts-up in the network with the required functionality is the server. The connection and activation of another device containing the same server leads to the activation of a standby server. Each server and standby server has published its unique identification number. When the current server is disconnected the standby server with the highest identifier takes over the server functionality. This algorithm is used at several places in the FABRIC architecture.

Other advantages of the publish-subscribe paradigm for the correct and easy operation of the home network are discussed in [6].

The distribution is most explicit in two architectural parts: (1) the service discovery and (2) the QoS framework. The service discovery relies on one entity per FABRIC-enabled device that publishes the available services in the FABRIC syntax. The concept of subnet is introduced. A *subnet* consists of the set of all interconnected devices of a given middleware X where there is a path between any two devices that does not pass through a gateway. Within a subnet a FABRIC server publishes the service descriptions of the legacy services compliant with middleware X. In the QoS framework [7], one server per device publishes the state of the network. Requests for new video streams or changes to existing video streams are also published. One entity, the resource manager receives this information and publishes changes to the settings. At every steaming device, these changes are received and executed locally.

In both architectural parts, a hierarchical structure can be discerned. Experience up till now does not allow us to decide how this architecture will scale with the number of interconnected devices.

4 Extra-functional properties

Extra-functional properties are those properties not covered by the functional properties, like timeliness, performance, efficiency, robustness, and many others dependent on a particular application domain. Within the FABRIC architecture, the performance and timeliness of the system is of crucial importance.

Given the layered structure of the FABRIC architecture and the publication of the video streams instead of a more traditional connection-oriented approach, there was a risk that the total communication overhead for the streaming would be too high. A considerable effort was put in an efficient transport of the stream at the source, the destination and in the gateway. Use was made of the “postmaster approach” developed in [11]. Communication entities are separated into two parts: (1) messages exchanged between protocol entities to steer the communication, and (2) the video frames that are communicated between the stream source and destination. Analysis has shown that thanks to this approach no unnecessary copying of buffers was introduced and nevertheless a high level of abstraction could be maintained at the application level.

Another performance aspect is the efficiency of the video transmission over the communication links. Scheduling of the network packets is based on WF2Q+ [12]. The scheduler is based on the sharing principle. It determines which part of the bandwidth is allocated to each stream. No real-time clocks are needed for its implementation. That means that synchronization must be provided by other mechanisms. The scheduler nicely separates the end-to-end delay requirements from the bandwidth sharing requirements.

Thanks to the implementation of a mock-up based on the HLA implementation available at TNO. Numbers were obtained that indicated the size of the objects necessary to implement the streaming behavior. The shown numbers describe non-optimized, compressed library sizes. The largest sizes are provided by the UPnP stack and the FABRIC layer represented by the HLA stack of TNO. The Controller and the Resource manager federates represent the sizes of the FABRIC services.

The Window federate and the MMplayer federate represent the video streaming application. From the table we can conclude that the FABRIC overhead with respect to the stack size of the middleware (e.g UPnP) is comparable.

Software	Compressed library size
UPnP stack	100 Kb
TNO HLA stack	100 Kb
Controller federate	41 Kb
MMplayer federate	18 Kb
Window federate	19 Kb
Resource manager federate	18 Kb

No investigations have been done with respect to security. Care has been taken not to introduce any constructs that would hamper the later addition of security mechanisms into the architecture.

5 Conclusions

It is a pleasure to look back at quite a successful project. The original aim of the project was to make giant steps forward while neglecting as much as possible "minor" implementation details. Neglecting the minor details proved to be the hardest job we faced. The project has come up with some concrete results:

1. A mock-up has been made that showed the FABRIC functionality by streaming a video over the network publishing all application information.
2. A clear design has been made that allows interoperability between applications and services that are related to heterogeneous middleware standards.
3. A QoS streaming framework has been designed that allows the sharing of wired and wireless communication networks between video streams, while maintaining control over the stream quality.
4. An application design has been proposed and worked out for Consumer Electronic applications based on the publish-subscribe paradigm.

The HLA publish/subscribe mechanisms and their associated rules have been a very good starting point and provided major guidelines for the project trajectory through design space. Adhering rigidly to these rules and concepts has enabled major new insights that we would not have acquired otherwise. In particular, the relation between abstraction, standardisation efforts and legacy has been a very valuable one.

However, it proved in the end that some HLA rules were too strict for the development of an in-home network. In particular the concept of the federation Object Model (FOM) necessitated that all devices contained before operation a complete description of all services that could be brought into the network. The FOM concept prevents a dynamic evolution of the network.

Pleasant surprises were obtained in the following areas:

- Identification of abstraction levels
- The elegant QoS management
- The distribution and failure management that came for free

6 Acknowledgements

The FABRIC project is funded by the fifth framework IST program of the European Union [13]. The author wants to acknowledge the contributions of the partners: THOMSON (FR), TNO (NL), CSEM(CH) INRIA (FR), SSSA (IT), MDH (SE), TUE (NL) and Univ of Passau (DE).

References

[1] *The HAVi specification*, Version 1.1, available from http://www.havi.org/, 2002

[2] *Universal Plug and Play Device Architecture*, version 1.0, 2000

[3] *UPnP AV Architecture*, version 0.83, June 12, 2002. Available at http://www.upnp.org/.

[4] *Standard for Modeling and Simulation High Level Architecture*, IEEE standard 1516-2000

[5] R. Verhoeven and J. Lukkien, *A publish-subscribe Architecture for Interoperable in-home Video streaming,* this workshop, 2004.

[6] J-D Decotignie and P. Dallemagne *Expressing Audio/Video communication using the publish-subscribe model.* This workshop, 2004.

[7] L. Rizvanovic and G. Fohler, *The MATRIX, A QoS framework for Streaming in heterogeneous Systems,* this workshop, 2004.

[8] Helmut Bürklin, *Interoperability between CE middleware standards,* this workshop, 2004.

[9] *IEEE Standard for a High Performance Serial Bus*. IEEE Std. 1394-1995

[10] P.vd Stok et al., "The FABRIC project" in *Software Architecture, First European workshop* , EWSA 2004, St Andrews, UK, May 2004, proceedings eds F. Oquendo, B. Warboys, R Morrison, pages 272-278. LNCS 3047, Springer, 2004

[11] F. Vamparys, *Mastering the complexity of the communication platform using a systematic approach,* PhD dissertation 1540, Swiss Federal Institute of Technology, Lausanne, 1996.

[12] J.C.R. Bennett and H. Zhang, *hierarchical packet fair queuing algorithms,* proc of the ACM SIGCOMM'96, pp 143-156, 1996.

[13] European IST project 2001-37167 – *FABRIC.* at http://www.extra.research.philips.com/euprojects/fabric/

A publish-subscribe Architecture for Interoperable in-home Video Streaming

Richard Verhoeven
Technische Universiteit Eindhoven
P.O. Box 513
5600 MB Eindhoven, The Netherlands
P.H.F.M.Verhoeven@tue.nl

Johan Lukkien
Technische Universiteit Eindhoven
P.O. Box 513
5600 MB Eindhoven, The Netherlands
J.J.Lukkien@tue.nl

Abstract

This paper gives an overview of an architecture that to solves a number of interoperability problems that occur within home networks due to heterogeneity in network technologies, middleware stacks and video encodings. The architecture focuses on the middleware heterogeneity and defines a method for scheduling the resources within the network, independent of this heterogeneity. This research was carried out within the FABRIC project1.

1. Introduction

Network technologies are improving rapidly and more devices become network connected, not only in the computing environment, but also in the home environment. The number of households with fast Internet connections is growing and with the introduction of additional PCs in the home, also in-home networks are emerging. The PC network typically transports digital data over UTP cables, while the audio-video CE devices, like TVs, VCRs, DVD players and tuners, typically transport analogue data over SCART cables. As a result, it is difficult to connect these two technologies. With the introduction of fully digital audio-video devices, digital video broadcast, and additional network connectors in CE devices (for IEEE 1394[8] and Ethernet[3]), it becomes easier to connect the two networks. As long as the devices support the same digital encodings, it should be possible to exchange the video content between the devices, independent of their location on the network or their technological background (PC or CE).

However, the network technology and software control protocol on the devices might not be compatible. Furthermore, new network technologies like fiber, and new control protocols can appear during the life cycle of the device, thereby introducing legacy and interoperability issues.

The user or owner of a CE device often expects that if it is possible to connect two devices with a cable, then it is also possible to use that cable to exchange data.

In this paper, we describe the FABRIC architecture to address some of the issues that occur in these in-home networks. The main issue is the heterogeneity of the middleware and the resulting middleware dependency of the applications that are built on top of it. Another issue is the cooperation between the different devices, such that the available network and processing resources are properly divided for improved user experience. Next, there is the issue that the life cycle of CE devices is long and that devices are added or replaced incrementally, such that all kinds of legacy devices can exist in all kinds of configurations. Last, there is the issue that with the introduction of wireless and mobile technologies, network configurations and bandwidth availability become much more dynamic.

The architecture not only has to address these issues, it also has to work within certain boundary conditions. The main requirement for the architecture is that no new network protocols are introduced, as it would take a lot of time and effort to finalize them. Furthermore, a new protocol would create additional heterogeneity and cause interoperability issues with legacy devices. Another requirement is that the architecture has to fit within the restrictions of the device, that is, it is not an option to require that every device supports every middleware technology.

The paper is organized as follows. In section 2, the different forms of heterogeneity are identified, together with the approach taken by the Digital Home Working Group (DHWG[5]) and the approach taken by the FABRIC architecture. In section 3, different levels of real-time requirements are identified with respect to handling video content in the home environment. In section 4, the FABRIC architecture is introduced and its key elements are described. Section 5 contains concluding remarks.

[1] This work was funded by the European IST project IST-2001-37167, called FABRIC. It was carried out in collaboration with Philips, Thomson, TNO, CSEM, INRIA, SSSA, MDH, UCL and UP.

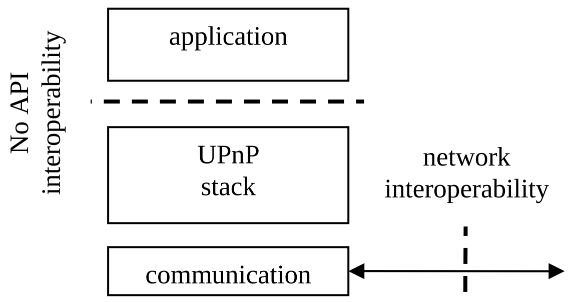


Figure 1. UPnP only provides network interoperability, not API interoperability.

2. Heterogeneity

In the in-home network, there are several levels of heterogeneity, which are also mentioned by the DHWG. First of all, different network technologies are present and new ones will appear in the future. Between PCs, a wired IEEE 802.3 network[3] is used, while a wireless IEEE 802.11 network[11] might exist as well. Between digital video devices, IEEE 1394, also known as Firewire[8], might be used, for example to connect to a video camera. For legacy video devices, analogue connections based on SCART are very common. In the future, fiber network technology can become available for in-home use. Other technologies will appear after that. At any stage, different technologies will coexist within the home.

These different network technologies also have different properties with respect to real-time behavior. For IEEE 802.3, the available bandwidth is more or less fixed, it is shared between a known number of devices and transmission is best effort. For IEEE 802.11, the bandwidth varies with the distance between the devices, it is shared between an unknown number of devices and transmission is best effort but may suffer from signal interference and fluctuations. For IEEE 1394, the bandwidth is also fixed; it provides isochronous channels with reservation, such that transmission is guaranteed. For future technologies, like fiber, the properties might differ again. When real-time video is transmitted over the network, the properties of the underlying network technologies become important when the resources become limited, that is, when over-dimensioning is not possible. Furthermore, when different technologies are connected, the properties of the end-to-end link become very complex and differ from those expected by the end device. In particular, traditional gateways between technologies focus on translating message formats without taking the functional level into account.

Secondly, on top of the network technologies, software stacks are available to provide transport, routing, discovery and control services. An example of such a stack is UPnP[2], which uses a collection of IP based protocols to perform tasks like device and service discovery, eventing and remote procedure calls. The control of audio/video streams is specified in the UPnP AV architecture[12], while the actual transport of the video stream is based on other standards and possibly unrelated to IP networks. Another example is the HAVi stack[1], which is specifically designed for IEEE 1394 and depends on the features provided by this technology. As a result, the HAVi stack specifies the complete stack from the network technology up to and including the applications.

Finally, other areas of heterogeneity are the encoding of the audio/video content and the rendering capabilities of the devices in terms of resolution and processing power. As a result, content might not be accessible for reasons not understood by the end user.

Within the DHWG[5], the heterogeneity problem is solved by selecting one technology for each heterogeneity level. For the network technology, the IP protocol stack is assumed to be present, which provides abstraction of the physical and data link (L2) layers. This abstraction is reasonable for non-real-time traffic like control messages, but not suitable for real-time traffic, where the capabilities of the hardware might be exploited, such as the support for video transport in IEEE 1394 and the differences between wired and wireless network technologies. Therefore, it might be necessary to apply L2 techniques like QoS and make them accessible to higher layers, possible up to the applications. On top of IP, the UPnP framework is selected as the middleware stack to perform the discovery and control operations, where different device control protocols can be added for different types of devices. UPnP is not designed with real-time behavior in mind and is not used for the transport of the streaming content with its real-time properties. Within DHWG, the audio/video content is standardized to JPEG and PNG for images, LPMC for audio, and MPEG2 for video. A device is allowed to provide content in a different encoding format, but has to provide the same content in the standardized formats as well, for example by transcoding it. For the media transport protocol, HTTP is the prime candidate.

Within the FABRIC project, the heterogeneity problem is addressed from a different angle. Instead of assuming that everything can be unified under UPnP over IP and that HTTP is the proper transport protocol for every network technology, we assume that some form of heterogeneity will always exist, and that somewhere between the application and the network technology some adjustments are necessary to achieve interoperability. Furthermore, UPnP only provides interoperability with respect to the network protocol, but it does not provided any interoperability with respect to application development, as depicted in Figure 1. That is, without an abstraction layer, a UPnP application is bound to one particular UPnP middleware stack. The FABRIC architecture is designed with this API interoperability in mind and independent of UPnP, such that it is possible to design applications for interoperability, based on the loose coupling provided by the model presented by the RTI.

3. Real-time levels

The tasks within the home network with respect to streaming video have different real-time properties, which have to be supported by the architecture as well.

- The video frames have hard deadlines, namely the time when the video frame has to be presented on the screen to have fluent audio and video playback. If a video frame cannot be presented on time, it could just as well not be sent over the network. Although the deadline is hard, it is usually not critical.
- The control operations from the user have soft deadlines, which are determined by the user perception of the complexity of the operation. For example, when the user presses the Play button, the video should start within 500 ms say. However, when the user presses the Freeze button, the video should hold within 100 ms. When a soft deadline is not met, the user can get annoyed and regard the device as not responding or broken. A soft deadline can also affect other parts of the system, like the amount of video buffering during stream startup. If it is not possible to meet a soft deadline due to a high level of heterogeneity, the user should be informed of the delay.
- Feedback information on the quality of a network connection and other soft state information has no strict deadline, although the information becomes less valuable over time, as it is used to adjust an active video stream to current network conditions.
- Content, device and service discovery is typically best effort. Typically, the discovery mechanism can be performed on request of the client or the provider regularly indicates that it is available on the network. The different mechanisms have an impact on bandwidth usage and the consistency of the network view.

When the usage scenarios for video streaming become more complex, additional real-time requirements occur. For example, when the audio and video channels are rendered by different devices, the playback has to be synchronized within 100 ms to assure lip synchronization. Or, when the user moves an active video from one screen to another, it should be performed within two seconds, where the synchronization between the videos is within two frames.

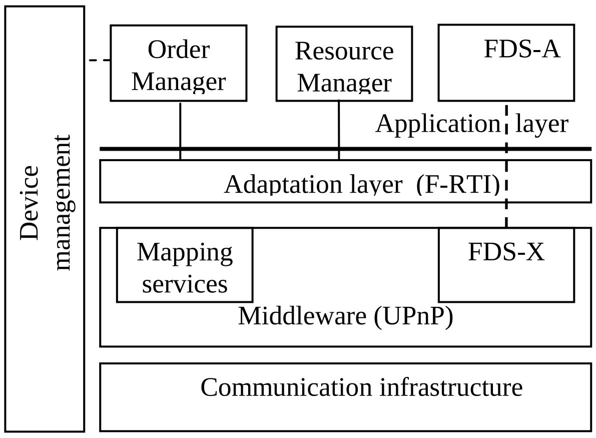


Figure 2. The FABRIC Architecture

4. Architecture

Within FABRIC, an architecture is chosen where an adaptation layer is placed between the applications and the middleware stack, such that the existing middleware stacks can be reused. Further, the adaptation layer is such that the application is independent of the actual middleware stack, thereby allowing the replacement of one middleware stack by another. The adaptation layer is such that dependencies of the application on the lower layers are minimized. An overview of the architecture is given in Figure 2 and the elements are further discussed in the following subsections.

A typical scenario for the architecture is the following. A device connects to the home network and uses a discovery mechanism to announce itself and to find other devices and content. Through some interface, a user selects specific video content and presents it on a specific device.

4.1. Adaptation layer

The adaptation layer is based on the run-time infrastructure (RTI), as defined by HLA[6] and IEEE 1516[7]. HLA stands for High-Level Architecture and is developed as a framework to create large scale real-time simulations. The RTI presents an anonymous publish&subscribe facility, where the application does not know the actual location of the provider of the information. By subscribing to information, an application can indicate that it wants to be informed of the arrival or modification of certain content. By publishing information, an application informs the applications that have subscribed to that type of information that it has changed. Within the RTI, it is possible to indicate whether the information should be transmitted reliably or unreliably. For the FABRIC architecture, an additional transport type is proposed to indicate a stream of information with a certain resource requirement in terms of bandwidth and delay[9]. This extended version of the RTI is called the F-RTI and provides the adaptation layer between the application and the middleware.

On top of the F-RTI, applications are built according to the publish&subscribe principle. The type of information that can be published is described in an object model, called a FOM in HLA terms. For in-home networks, this object model consists of a number of service types, content types and interactions (which are similar to asynchronous method invocations). The service and content types are based on existing middleware standards, like UPnP and HAVi, where common and useful features are available in the object model, while more exotic features are currently left out.

The scenario mentioned earlier uses the publish&subscribe mechanism as follows: after connecting, the device publishes its services and content, and subscribes to relevant interactions. The user application is subscribed to these services and content types and will discover the new objects. Through the application the user sends an interaction to instruct a rendering service to subscribe to specific content. After subscription, the rendering service will receive the content and render it.

Note that the application does not directly access the device that provides the content. Instead, the publish&subscribe mechanism is used to publish through an interaction that certain content has to be presented on a certain renderer. The provider of that content subscribes to these interactions as well, such that it can perform the necessary operations.

For the purpose of resource management, the object model is extended with two types: status and order cells. The devices use the status cells to publish the status of their resources and subscribe to the order cells to improve cooperation between devices. Although this management functionality is typically located within the OS, in this architecture it is lifted to the application level, such that the management function can extend past the boundaries introduced by heterogeneity.

The main advantage of the use of the F-RTI with an abstract interface on top is the fact that the application can remain the same independent of the underlying middleware or hardware. When new technologies are provided at the middleware or hardware layers, the F-RTI can incorporate them without affecting existing applications. That is, the RTI factors out common functionality of middleware stacks and focuses on the mappable parts.

One shortcoming of current middleware is hereby identified viz. that it does not standardize on the API – UPnP does not have an API to begin with. Therefore, applications that are built on top of one UPnP stack can not easily switch to a better UPnP stack, as indicated in Figure 1.

4.2. Mapping services

The F-RTI is built on top of existing middleware and reuses the capabilities of that middleware and the underlying hardware as good as it can. For example, if the underlying middle- and hardware supports bandwidth reservation, as is the case with IEEE 1394, the F-RTI can make use of that. However, if a required feature is not supported, the middleware can either be extended with mapping services that provide this functionality, like a bandwidth regulation scheme, or rely on the best effort method. In any case, the F-RTI can get most out of what is provided by the technology that is already in the device.

By reusing existing protocols, both at the application side and at the network side, there is no extensive standardization effort involved in constructing the architecture. And since there is a large overlap between the functionality provided by the middleware and that required by the F-RTI, the additional processing and resource overhead of the mapping services are expected to be small.

4.3. Directory services

To support legacy devices across the network, the architecture provides directory services, FDS-X and FDS-A. The FDS-X is a middleware specific service that acts as a proxy for the legacy devices within middleware X. The FDS-A is a directory service for the middleware specific functionality that is provided by the legacy devices, such that applications that are aware of this additional functionality are able to access it. As a result, such applications become middleware dependent, but the affected part of the application would only handle the advanced or exotic options.

4.4. Application level gateway

The introduction of the F-RTI does not prohibit that different middleware stacks or network technologies are present within one home network. In this case, we assume that a middleware stack is directly related to a network technology, such that there is a clear boundary, namely the device that connects both network technologies and acts as a gateway. Since this device is connected to both technologies, it also provides the related middleware stacks.

Each of these middleware stacks is equipped with its own F-RTI implementation, such that a gateway application can be built on top of that, as depicted in Figure 3. This gateway application acts as a proxy for everything that it discovers.

Since any application built on top of the F-RTI can only use the operations provided by the F-RTI, these are the only operations that the gateway has to support. The main restriction of such a gateway is the absence of cycles in the network topology. For wired networks, this is not a severe restriction, but for wireless networks, precautions should be taken with respect to ad-hoc

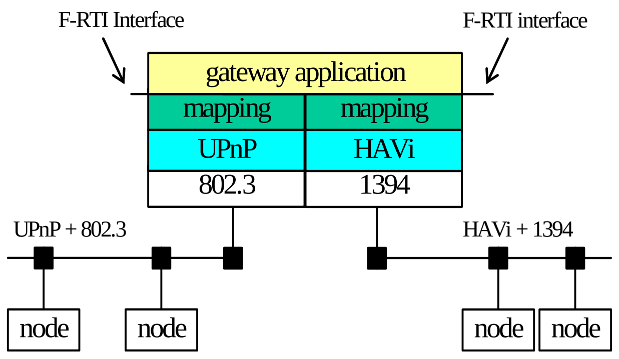


Figure 3. The application level gateway between UPnP and HAVi

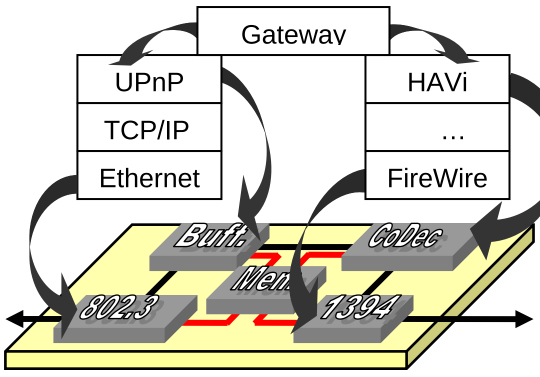


Figure 4. Software stack controlling the hardware plane.

networks, piconets and scatternets. For example, two mobile devices might be connected in a BlueTooth piconet together with small collection of other devices. When the collection of devices arrives home, the two mobile devices can both connect to the wireless access point and provide bridge functionality for the piconet. As a result, services and content would be available multiple times.

4.5. Streaming the video

Once the video content is streaming over the network, the role of the middleware stack is reduced and the main issues are related to moving the streaming content from storage to the network in the content provider, and from the network to the decoder hardware in the content renderer.

Within the FABRIC architecture, a content oriented approach is used, where a content renderer subscribes to certain content in order to receive it. Conceptually, the video content is represented by a value which is regularly updated by the content provider, i.e. 25 fps. Within the actual device, the optimization of this step can be such that the content is moved directly from the storage to the network, and from the network to the decoder. That is, the streaming content remains close to the communication infrastructure and does not go through the middleware stack, as depicted in Figure 4.

In the case of the application level gateway, the same principle holds. Conceptually the stream passes through the application and the middleware stacks, but the implementation can optimize these operations, such that the content is moved directly from one network interface to another, with minimal overhead.

The publish&subscribe mechanism provided by the F-RTI is anonymous and hides the actual location of the objects. For the streaming video content, this could be provided by using anonymous multicast addresses, although it depends on the mapping to the actual network technology whether it results in efficient resource usage.

4.6. Cooperation

When the home network contains a large collection of devices, it is likely that the resources within the network are insufficient to support all the requested video streams. Therefore, resource allocation algorithms are required such that the resources are properly divided common the different devices. For this to work, it is necessary that all devices cooperate: they are not “selfish”, they report their resource requests and resource status correctly and they obey the “orders” that are calculated by the resource manager. The exchange of information to come to a distributed decision is transmitted via the F-RTI as well in the form of publications to status and order cells. Through the F-RTI, all relevant devices are informed of the status and of the decisions, independent of the underlying middleware or hardware technology, since the gateway will transport the messages across technology boundaries. By using the F-RTI, the resource management is performed at the application level, where all relevant information is available. At lower levels, the same priority values might be used for different properties, thus hiding information that is relevant for the decision procedure.

As depicted in Figure 2, the architecture uses local order managers and a central resource manager. The order managers publish their status in a status cell, which is collect by the resource manager that subscribes. After scheduling the resources, the resource manager publishes the schedule in the order cells which are related to active video streams. The local order managers receive the relevant orders and act upon them by adjusting the local schedules and video streams.

Although one central resource manager is used, the publish&subscribe mechanism does not prohibit to have distributed resource managers, for example one for each subnet or a local resource manager in each device. Since the information that is published will arrive at every resource manager that subscribes, it is possible that each resource manager can create the same schedule based on the same status information.

The granularity in time of the status and order updates is based on the ability of the devices to adjust their behavior. For example, if a wireless device is not able to switch between access points within five seconds, it does not make sense to exchange status and order information every 100 ms. Furthermore, a stable low-quality video is perceived by users as better than a video switching between high and low quality too often. Such perception information can also be used for the time granularity of status and order updates.

The order manager has to report resource status information and adjust active video streams. These tasks require access to very system-specific local components, for which it is impossible to provide a general interface at the abstraction layer. Therefore, the order manager has direct access to the local components on the device through the device management, thus bypassing the abstraction layer. This is considered not to be a big issue, since the order manager is typically a small and system specific application.

5. Conclusions

The presented architecture provides an alternative solution to heterogeneity problem of the home environment, in comparison to the architecture provided by the DHWG, in a form of an abstraction layer on top of UPnP. As a result, applications can be written independent of the used middleware stack, whether it is a completely different middleware, or just a different implementation. During the development of a mock-up of the architecture, this proved to be a very useful feature.

To create an accurate schedule, the resource manager has to collect sufficient information about the home network and the devices within it. Due to legacy and transparent network devices, it will be difficult to create an accurate model of the existing network topology.

The actual algorithm for creating the network-wide schedule is still a research topic. The performance of the exchange of status and order information is part of that research, which does not depend on the availability of the complete architecture. In fact, the general idea would also work when the information is broadcast over the L2 layer. For production level CE devices, it would require standardization of that L2-based protocol.

Acknowledgments

The FABRIC architecture is developed within the European IST project IST-2001-37167. Therefore, we would like to acknowledge the partners of that project for their contributions: Philips(NL), Thomson(FR), TNO(NL), CSEM(CH), INRIA(FR), SSSA(IT), MDH(SE) and UCL(UK) / UP(DE). Special thanks to Gerhard Fohler from MDH, who contributed the resource manager framework, to Jean-Dominique Decotignie from CSEM, who contributed the gateway structure and the idea of the content-oriented approach, and to Peter van der Stok from Philips, who contributed the directory services and managed the project.

References

[1] The HAVi specification, Version 1.1, available from http://www.havi.org/, 2002

[2] UPnP Forum, http://www.upnp.org/

[3] IEEE Standard for Local Area Networks: Carrier Sense Multiple Access With Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications, No.: 802.3 – 1985

[4] Guideline of Transmission and Control for DVD-Video/Audio through IEEE1394 Bus, Version 1.0, September 2002, DVD Forum

[5] Digital Home White Paper, Final Version, June 2003, Digital Home Working Group

[6] "Creating Computer Simulation Systems - An introduction to the High Level Architecture" by Frederick Kuhl, Richard Weatherly, and Judith Dahmann, Prentice Hall.

[7] IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Federate Interface Specification, No.: 1516.1-2000

[8] IEEE Standard for a High Performance Serial Bus – Firewire, No.: 1394-1995

[9] Deliverable D5 – Standardization Proposal, M.H.H. Brassé and J.J. Boomgaardt, FABRIC project. Available at http://www.extra.research.philips.com/euprojects/fabric/

[10] European IST project 2001-37167 – FABRIC. Available at http://www.extra.research.philips.com/euprojects/fabric/

[11] IEEE Standard for Information Technology - LAN/MAN - Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, No.: 802.11, 1999

[12] UPnP AV Architecture, version 0.83, June 12, 2002. Available at http://www.upnp.org/.

The MATRIX: A QoS Framework for Streaming in Heterogeneous Systems

Larisa Rizvanovic
Department of Computer Engineering
Mälardalen University
P.O.Box 883
S-721 23 Västerås, SWEDEN
larisa.rizvanovic@mdh.se

Gerhard Fohler
Department of Computer Engineering
Mälardalen University
P.O.Box 883
S-721 23 Västerås, SWEDEN
gerhard.fohler@mdh.se

Abstract

The work presented here was carried out within the FABRIC project[1]*, which aimed at the integration of middleware standards used in home networks to provide high quality streaming over heterogeneous networks.*

In this paper we propose an adaptive QoS framework for efficient resource management, called the Matrix approach. The Matrix is a concept to abstract from having detailed technical data at the middleware interface. In stead of having technical data referring to QoS parameters like: bandwidth, latency and delay we only have discrete portions that refer to levels of quality. The underlying middleware must interpret these values and map them on technical relevant QoS parameters.

1 Introduction and Rationale

The aim of the FABRIC project was to develop *an architecture* in which several interoperability standards and technologies in the home networking context can be integrated. In addition, a FABRIC handles the complete network to satisfy *End-to-End Quality of service (QoS)* requirements.

Efficient transport of streams with acceptable playout quality in the FABRIC network requires management of both networks and CPUs. As resources will be limited in the home environment, *guarantee* mechanisms for continuous stream transport are demanded, e.g., it will not be acceptable to interrupt a football game for the start up of another stream.

One of the key issues for resource management is an efficient representation of the fluctuating system state and resource allocation decisions, to provide a small interface to decouple device scheduling and system resource allocation.

Event based vs. controlled state update

The overhead to transport the information needed for a 100% accurate view of the system with very fine grain granularity capturing highly fluctuating resources such as wireless networks will be prohibitively high on the network; scheduling activities for all events will overload CPUs. In addition, such information would be too fine-grained fluctuating, as resource management has to operate at larger granularity. We believe that the tradeoffs between accuracy of system state information and efforts to transport and process have to focus on efficiency providing the minimum relevant information for resource management only.

Step-by-step vs. system resource management

A distributed resource management approach will be severely impeded by overheads, as devices have to exchange information to determine a global system view for scheduling decisions. In addition, scenarios such as high fluctuations on a network link demand more scheduling activities, which in turn will create more network overhead, resulting in increased fluctuations.

A step-by-step approach, in which decisions about how much of a resource is dedicated to a given version of a stream are taken locally by the devices in sequence overcomes some of the problems of the fully distributed approach. It is impeded, however, by several shortcomings as well: the decisions taken on each device suffer from the limited view of the state of the device and the next one on the route. Suppose the resource on both devices can provide ample availability at the moment, resulting in the choice of a high quality/high bandwidth version of the stream to be transmitted. If any of the devices on the route to the play-out can handle only a lower quality version, the resources used here for the high quality will be wasted. Transcoding complicates the issue and is not considered. Propagating the state information back and forth along the route results in the overhead given earlier and delays the actual scheduling in each device further. Rather, a sender-based approach with global knowledge is appropriate.

[1] FABRIC is the European IST project IST-2001-37167.

Interfacing device – system resource management

Decoupling communication and synchronization between devices and resource management enables operation without explicit consideration of intricate details of device schedulers. By providing an abstraction level suitable for resource management, it facilitates a component-based approach as well. Instead of the resource manager probing local schedulers for state information, the devices can provide information about relevant state information and estimations about changes with appropriate, individual granularity themselves. Failures in devices or communication will not block or delay resource management.

Chosen design

We propose a global abstraction of device states as representation of the system state for resource management and as interface to decouple device scheduling and system resource allocation.

This global abstraction, called Matrix contains information about device states in a format appropriate for resource management. The accuracy of the information represented is suitable for resource management, abstracting over fluctuations or changes, which will overload scheduling: the very fine grain resolution of values is mapped into a very small number or discrete values. Devices are responsible for providing information about their states, only for changes relevant for resource management, i.e. in the pre-processed reduce value range. Thus the overhead to keep system wide state information fresh is dramatically reduced and no explicit communication or synchronization between resource management and local schedulers is needed. Devices update relevant information at the appropriate pace to the Matrix, on which resource management bases decisions.

Diffusion of these decisions to devices is carried out via the Matrix as well, i.e., orders for resource allocation on individual devices is put in high level abstractions of limited value ranges into the Matrix, from where the devices pick up orders to translate them into local scheduling policies or parameters. Thus global resource management is independent of detailed knowledge about local schedulers, which can be replaced easily, supporting a component based approach.

Without the need for explicit costly communication and negotiation between devices, decision about which version of streams to transport or the decomposition of end-to-end delays can be performed by global resource management, reducing overheads and resource waste due to limited local device knowledge.

The Matrix provides a logical abstraction of the view of the system state, not an actual centralized implementation requirement. Rather, the Matrix is represented in a distributed way. Not necessarily do all devices in the system have to use the Matrix, but a mix of direct explicit communication and Matrix abstraction is conceivable for resource management, although benefits of the data abstraction would obviously be reduced.

2 Operation

The Matrix enforces complex stream quality demands to be translated into a few resource interfaces.

QoS levels

The information in Matrix is prepared, ie. largely reduced in range, suitable for resource management, abstracting over fluctuations or changes, which will overload scheduling. Thus, the Matrix will contain only a very small, discrete range of few Quality of Service (QoS) performance levels, for example High, Medium and Low (see Figure 1).

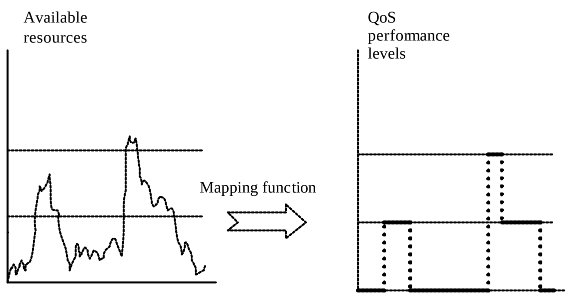


Figure 1: Mapping various kind of traffic specification to few QoS levels

It is crucial to map various kinds of traffic specifications to these few QoS levels in a manner that gives a fair abstraction of the system while reducing overheads. We assumed a linear mapping between bit rate and quality of a stream as the complex relationship between user perceived quality and resource demands was beyond the scope of the project. However, a comprehensive QoS mapping mechanism that translates representation of QoS at different system layers (i.e., application, network) and also considers user perceived quality [6, 7, 8] is a task of future work.

The fact that the Matrix approach has to present a valid picture of resources could involve a need for constant updates coming from the bandwidth fluctuations. But in our opinion, only two occurrences will cause the update of the Matrix elements:

- A new connection (a subscription by render device to a certain video content)
- A significant change of available resources in the system (e.g. when a change to a different quality level occurs for a significant amount of time).

In the Matrix, information about available local resources will be used, by the "resource manager". The resource manager will try to achieve an optimal (feasible) scheduling solution for all streams in the system, with respect to their priorities. Then, this global view of resources, end-to-end time constraints, will be transformed to a few scheduling factors for each link in the path of streams. These decisions, expressed in a very small number of discrete values, are spread to devices by the Matrix as well, from where the devices pick up orders to translate them into local scheduling policies or parameters.

3 Architectural design aspects

This section introduces a more detailed overview of the Matrix approach. The Matrix is composed of several entities that constitute an effective mechanism for monitoring and scheduling available resources in the system. Figure 2 shows the data flow (information flow) between the Matrix components. The functions of these components are further discussed in the following subsections.

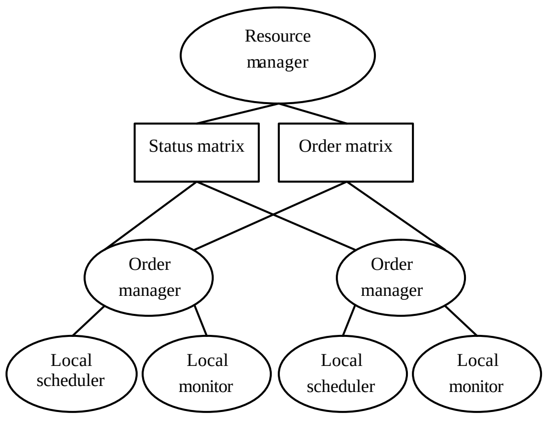


Figure 2: Information flow between Matrix's components

3.1 Resource manager

The resource manager will be used to schedule and reserve resources, within system. One part of stream scheduling is providing end-to-end timing constraints. So, the resource manager will provide sub deadlines (sub delays) for each device in the system by real-time methods such as outlined in [2].

Each time a new connection is about to be made, the resource manager has to determine if sufficient resources are available to satisfy the desired QoS of the new connection, without violating QoS of existing connections. Likewise, each time a resource variation occurs, that affects an active video stream, the resource manager has to make an adjustment of streams and resources.

In order to deal with resource reservation, the resource manager has to have knowledge about currently available resources in the system. This information is obtained from the Status Matrix. If there are enough resources to support the requested connection, the resource manager puts orders for resource reservation into the Order Matrix. Orders can be seen as an interface between the resource manager's global view of resources and set of entities (order manager, local scheduler and local monitor), which called "local enforcement mechanism".

If resources are not sufficient to carry on with a connection, the requested connection might be impossible to accommodate. The only solution will be to drop some of the existing connections. If the policy states it, this will be done as in the case of a sudden reduction in bandwidth by dropping of existing ones. In the case the user gives priority to a stream, it will be reduced in quality or dropped last. These policies and user interaction are outside the scope of this work.

3.2 Status Matrix and Order Matrix

The Status and the Order Matrix act as an interface between the resource manager and devices in the system. For each resource in the system, there are a two (sub) matrices (Status and Order).

The Status Matrix contains information about available resources in the system. Each resource is represented by its

- current value (out of the limited number range)
- current granularity, i.e., the time interval until which the current value is likely to not change, and
- likelihood that 2) holds.

These values are provided by the devices (order managers). A single link on, say, wired switched Ethernet, will have a high granularity interval and high likelihood, whereas a wireless link in a mobile environment might result in small values for each. While accurate and correct predictions will not be possible, these values support better estimates for the decisions of the resource manager than very pessimistic values only. Should the granularity interval be less than is useful for resource management, the associated value for the device state can be assumed 0.

The *Order Matrix* contains directions for resources reservation on the devices, made by the resource manager. Each device is presented by one element in the Order Matrix, from where the device picks its order in form of

- delay (sub-delay)
- value (out of the limited number range, QoS performance levels)

Source and destination devices/resources, as user priority request, are specially marked in both matrices.

3.3 Order manager

An order manager is responsible for allocating resources at a device. It maps global resource reservation constraints (orders), made by the resource manager, to the concrete scheduling specification for a local scheduler.

Another task of the order manager is to provide the Status Matrix with information about locally available resources, in form of well defined QoS levels; for both CPU and bandwidth.

The information about available resources is determined repeatedly, but not periodically by the order manager. The accuracy of the information depends on a chosen temporal granularity. Hence, one order manager is responsible for

- determining of the available resources at the devices
- transforming various kind of traffic specification into a few QoS levels and providing the Status Matrix with them
- allocating resources at devices and providing parameters to local schedulers

3.4 Local scheduler

A local scheduler is responsible for scheduling of device's local CPU resource or outgoing network packets. It is placed on a device and together with the order manager, it enforces local resource reservation.

As mentioned before, the order manager provides parameters to the local scheduler, which then performs the actual scheduling of the streams from this node.

3.5 Local monitor

A local monitor will perceive changes in the bandwidth availability class with a given granularity. Received and specified performances will be compared by the local monitor and the outcome of the comparison will be sent as feedback information to the order manager. If the significant resource change has been observed, it will be reported to the Status Matrix, otherwise it is ignored.

4 Evaluation

The FABRIC application interface is derived from the HLA API. HLA is an interoperability standard that is based on an anonymous publish/subscribe mechanism that allows for distributed applications (federates) to exchange data in a loosely way [5].

Federates express their interest in receiving data of specific object class (attributes) by using "subscribe to" mechanism. Likewise, federates that own object attributes can update their attributes values by using "publish".

We have implemented the Matrix approach using HLA within FABRIC.

The resource manager and the order managers are implemented as federates. The Status/Order matrix is a collection of objects.

The resource manager subscribes to quality requests, expressed in the Status matrix, that are published by streaming applications. The order manager subscribes to the Order matrix to receive orders from the resource manager.

By using HLA, messages are only sent when attributes of the matrix elements are actually updated.

The HLA offers the dynamic addition and removal of members of a federation. This HLA's characteristic is well suit for a redundancy of devices in the Matrix approach. Federates (order managers or a resource manager on a device) can fail without a major impact on the overall functionality. The lost federate (device) can be replaced during run time with some shadow federate (device).

5 Acknowledgments

The Matrix approach is developed within the FABRIC project. The authors would like to acknowledge our partners in the project: Philips(NL), Thomson(FR), TNO(NL), TUE (NL), CSEM(CH), INRIA(FR), SSSA(IT) and UP(DE) .

References

[1] Deliverable D3: Application Requirements, FABRIC project, http://www.extra.research.philips.com/euprojects/fabric/
[2] Deliverable D4: Streaming specification, FABRIC project, http://www.extra.research.philips.com/euprojects/fabric/
[3] Deliverable D8: Streaming design: analysis, control and scheduling, FABRIC project, http://www.extra.research.philips.com/euprojects/fabric/
[5] IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Federate Interface Specification, No.: 1516.1-2000
[6]Tatsuya Yamazaki and Jun Matsuda "On QoS Mapping in Adaptive QoS Management for Distributed Multimedia Applications"
[7] "A Survey of Quality of Service Architectures" Aurrecoechea, Campbell and Hauw, Center for Telecommunication Research, Columbia University, NY
[8] Fukuda K., Wakamiya N., Murata M., and Miyahara H., "QoS Mapping between User's Preference and Bandwidth Control for Video Transport", Proc. 5 th International Workshop on Quality of Service (IWQOS'97), Columbia University, New York, US

Expressing Audio/Video communications using the publish-subscribe model

Jean-Dominique Decotignie
CSEM
Jaquet-Droz 1
2007 Neuchâtel, Switzerland
jean-dominique.decotignie@csem.ch

Philippe Dallemagne
CSEM
Jaquet-Droz 1
2007 Neuchâtel, Switzerland
philippe.dallemagne@csem.ch

Abstract

The publish-subscribe model is an appealing communication model for transferring audio and video contents in the home environment. The model has been used with great success for years in different contexts among which some real-time ones. .

In this paper, we explain the benefits of the model and show how communication of multimedia contents in the extended home environment can be expressed under this model. The result departs from the classical client-server view and requires some rethinking of the applications. It leads however to an elegant object view of the system.

The last part of the paper is dedicated to expressing quality of service constraints under the publish-subscribe model. We show that, the model leads to a simpler approach that is also potentially more efficient than the conventional client-server model.

1. Introduction

The publish-subscribe model has been in use for more than three decades and, despites its advantages, has remained rather confidential. The belief of the FABRIC project (IST 2001-37167 project FABRIC Federated Applications Based on Real-Time interacting Components) is that it is a good candidate for the exchanges of audio and video contents in the extended home environment.

In most of the cases, the publish-subscribe model is used in environments with limited or no temporal constraints (some notable exceptions are reported in [1][2]) whereas the transfer of streams of images leads to temporal constraints. If we add that these contents must be transferred on wireless LANs that are subject to rapid changes in the quality, it is mandatory to take into account Quality of Service (QoS) in the model.

This paper aims at expressing the lessons learnt in the FABRIC project on the use of the publish-subscribe model for the distribution of audio and video content in the home environment and the expression of quality of service. In a first part, we will describe the publish-subscribe model and its advantages. The modeling of audio-video transfers in the home will then be described through the example of UPnP. The second part of the paper explains how the various concepts used in UPnP can be mapped to the model. In particular, we show that the quality of service constraints may be expressed in the model but require some change in the approach.

2. The publish-subscribe model

The publish-subscribe model covers a number of approaches and variations that are out of the scope of this paper. The idea is not new and has been in use in commercial industrial networks since the early 80s [1] [3]. Reference [4] is an interesting overview of the model while Thomesse [2] compares the model to others with regards to temporal behavior.

The basic idea of the model is to distinguish two roles. The publisher is the source of some information. The subscriber is the sink. The subscriber declares its interest without the need to know the publisher. In its declarations, it mentions the some characteristics (name, type, some attributes, etc.) of the information it interested to receive. Provided the name is unique, specifying the name in the subscription identifies unambiguously the required information. The source of this information is the publisher that has declared itself as the producer of this information.

Alternately, the subscriber may declare its interest to a type of information in which case it will receive it from all the publishers that publish information of this type. The same applies when only some aspects, such as the value of some attribute, of the information is given by the subscriber.

A third role is sometimes added, the distributor. The distributor is in charge of matching the interest of the subscribers with what the publishers have declared. It is also in charge of transferring the information from a given publisher to one or more subscribers. This role is usually played by the communication middleware.

The publish-subscribe model thus encompasses 1-to-N multicast distribution of information as well as N-to-1 (sometimes referred as anycast) and N-to-N distributions.

There are a few advantages of the model over the more conventional client-server model:

- Location independence – The information flows without the need to know the identification of the publisher (on the subscriber side) and the subscriber (on the publisher) side. From the application view-point, the flow is thus independent of any addressing scheme and thus of the location of the parties.
- Ease of adding fault tolerance – A faulty publisher may be replaced by another one without any change to the subscribers which might even not notice the change.
- Absence of synchronization – (flow control) in the client-server model, the server should be ready when the client issues requests. In the publish-subscribe model, the publisher produces the information when necessary (at the occurrence of some event or at a given frequency). The subscriber need not be ready at that time. It may even be disconnected. Whether the published information is buffered inside the communication middleware or not is a design choice. In many designs, buffering is supported with limits in space or in time. In the first case, when the buffer overflows, the old information is destroyed. Alternately, the new information may be thrown away. Buffering may also be based on a maximum residence delay. When the information has sojourned for a too long period of time, it is dropped either because of validity expiration or it is assumed that no subscriber will use it.
- Absence of blocking – In the client-server model, the client is blocked until the server responds to its request (some client server models such as in Corba allow some form of asynchrony). In the publish-subscribe, the publisher just produces the informa-tion on the network and continues its operations. The subscriber is later informed of the new information and may consume it at its own pace.
- A better prediction of performances – The temporal behavior of the producer is independent of the temporal behavior of the subscriber. It is thus easier to predict the performances of each entity.
- Possibility of transfer optimization – When more than one subscriber consumes the information, the middleware may easily use some form of multicast or broadcast mechanism. Instead of transferring the same information twice when there are two subscribers, the information is only sent once.
- Ease to realize gateways as a gateway is just an entity that subscribes to all items on one side and publishes these items on the other side [6].

The model has been used with success in many types of application such as distribution simulation [8], control of industrial processes [3], multiprocessors, etc. To the best of our knowledge is has not been applied to applications in which the bulk of transfers are continuous media information.

3. UpnP Audio/Video architecture

To exemplify the peculiarities of the audio and video stream transfers in the extended home context, let us look at how UPnP defines the related services. UPnP AV specifications define the interaction model between UPnP AV devices and control points. The control points include PDAs, Laptops, SetTop Boxes and any other Displays that have the ability to control the media streams. The specifications define three types of logical devices on the network: Media Servers, Media Renderers and Control Points (see Figure 1). There are on the other hand four types of “services” hosted by the devices:

- The *Content Directory Service* enumerates the available “content” (videos, music, pictures, and so forth).
- The *Connection Manager Service* determines how the content can be transferred from a Media Server to a Media Renderer.
- The *AV Transport Service* controls the flow of the content (play, stop, pause, seek, etc.).
- The *Rendering Control Service* controls how the content appears (volume/mute, brightness, etc.).

MediaServers are used to locate audio and video contents available on the network. As such, each Media Server will host three of the four defined services: Content Directory, AV Transport and Connection Manager. The MediaServer will also be the source of the content when a connection is established.

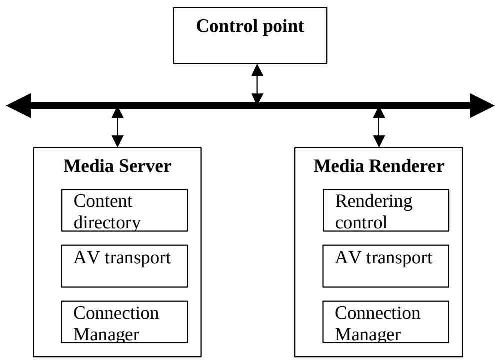


Figure 1 – UPnP AV architecture

MediaRenderers are used to render contents (display content, produce sound, etc.,) they receive. Each one must host a Rendering Control service as well as a Connection Manager and a AV Transport service.

The third logical entity that appears on Figure 1 is the Control Point that coordinates the operation of the servers and the renderers. They do not host any service. They are simply specialised control points as defined in the UPnP standard. AV Control points use UPnP servi-ces to offer the users a number of functionalities such as:

- Locate the existing Server/Renderer devices in the network, i.e. discovery,
- Enumerate the available content for the user to choose from, i.e. advertising,

- Query the Server and Renderer to find a common transfer protocol and data format for the selected content,
- Configure the Server and Renderer with the desired content and selected protocol/format, i.e. Server/Renderer Setup,
- Initiate the transfer of the content according to the user's desires with different functionalities such as: play, pause, seek, etc, i.e. flow control,
- Adjust how the content is rendered by the Renderer such as volume, brightness and so forth, i.e. rendering characteristics.

4. Publish-subscribe and continuous media content

Here we would like to discuss the way the various concepts and elements of audio/video communications in the extended hoe environment can be mapped to items that are published and to which entities subscribe. We will use UPnP AV architecture[5] as a reference for the concepts and elements. It is outside the scope of this paper to discuss the validity of these concepts and elements.

Furthermore, we will assume an object view as done in the FABRIC project. This means that the only items that are published are objects with their attributes. Let us look at the 3 entities defined by UPnP and how their operations may be modeled in terms of published and subscribed objects.

4.1. Media servers

The media servers publish the stream objects. Here we designate as streams any audio or video content flowing on the network from a server to a renderer. As such a stream object possesses a number of attributes among which the stream data (that will flow from the source to the destination), the characteristics of this flow (i.e bit rate, encoding, etc), and some identification (title, published date, …). This information is sufficient to support browsing of available contents and selecting one as well as to determine the resource that are necessary to transport the flow of data on the network and render the result. In other words, the *Content Directory Service* can be implemented easily using objects and the publish-subscribe architecture.

Establishing a flow of data from the publisher to the subscriber is automatic as soon as there is a publisher and at least a subscriber for a given stream data. There is hence no need for a start and a stop of flow as long as "non pausable" streams are concerned[1].

For "pausable" streams, the control point may wish to pause, resume or stop the flow. There are at least two ways to model this aspect in our context (publish-subscribe objects). The first solution is to define a command attribute on the stream object. This attribute is not published by the source of the stream but by another entity such as a control point that will produce a new value to change the attribute and thus operate on the flow of data. The second option is to allow explicit operations on objects. In such a case, the operation on the flow is represented by another object to which the stream producer subscribes. This object may possess attributes that are parameters of the operation. Note that only constructor operations may be implemented that way because there is no way to return values. Selectors are still implemented as attributes published by the original object publisher. The result of a constructor operation may be observed through a selector.

Both options implement the *AV transport service* defined by UPnP. Concurrent operations on the same stream by different network entities are not possible because, in both cases, only the publisher of the attribute (first option) or the operation object (second option) may invoke the operation. This may be a problem in some cases as more than one entity may wish to operate on the stream. It may be solved through the idea of ownership as in HLA [8]. Note than the problem is identical in the client-server approach.

4.2. Media renderer

Media renderers display or reproduce streams to the users. The rendering control service may be implemented by defining a renderer object with all the necessary attributes (brightness, volume, contrast, etc.). The renderer would subscribe to this object and a control point will publish it to operate on the renderer.

In our approach, the renderer has to subscribe to a given stream object as instructed by another entity. This is performed explicitly by changing an additional attribute of the renderer, the displayed object. It is changed using a constructor operation implemented using one of the two options described above (§4.1). Note that the publish-subscribe technique is here superior to the client-server approach in that it is easy to implement "picture in picture" by just adding a new attribute to the renderer object.

As in the server case, concurrent operations on the renderer require some mechanism to inform a publisher that it should stop publishing. It might be assumed that there is no need to inform the new publisher because it is likely to be the entity that invoked the stop. The attribute ownership services that HLA defines seem to be an elegant solution to the problem[8].

4.3.Control Points

Control points operate on the servers and renderers. All the user interactions typically go through the mediation of control points. A control point hence does not implement any service but uses services from other

[1] A pausable stream is a stream that can be paused at some point and resumed later at the same point. A movie in a DVD player is an example. A video broadcast is example of a non pausable stream.

entities. In the publish-subscribe approach, a control point is a subscriber of information published by the media servers and a publisher when it comes to operate on the media servers and the media renderers.

4.4.Managing connections

UPnP defines a service that has not yet been explicitly mentioned, the *Connection Manager Service*. This service is in optional and used to obtain information on the supported protocols and a handle on an instance of a AV transport service. This may be easily implemented using additional attributes on already defined objects.

4.5. QoS concerns

In to the client-server approach, a connection is established from the client to the server. This connection exhibits some quality of service and, at connection establishment, some negotiation may take place to decide on the best compromise between the client needs and the network capability.

In the publish-subscribe model, the data starts to flow as soon as there is match between a publisher and a subscriber. The flow of data is started without any explicit request from any party. In other words, the network infrastructure must be informed on the requested quality of service for this flow.

This may be done using a new object, a connection object , that captures the necessary QoS parameters. In FABRIC, we rejected this approach because it creates a new object that is not part of the application domain. Furthermore, the object has no clear producer or subscriber. Additionally, some networks may not support connections.

In FABRIC, there is no additional object but a few new attributes on existing objects. QoS attributes are used at two stages. First, there is a need to match the capabilities of a renderer with the parameters of a stream at a server. Second, the stream should be transported from the server to the renderer with a given quality.

To support the first aspect, we decided that, if the server can produce the same content with different quality of service, this should be reflected by as many stream objects as the number of qualities. The renderer on its side defines its capabilities through additional attributes on the renderer object. The match between the capabilities of a renderer and a given stream may be easily done. The renderer is then instructed to subscribe to the selected stream.

The second aspect is the transport of the information and in particular the network guarantees. When a subscription is made to a publish stream, the network middleware tries to establish a path that is able to support the given stream. If it succeeds, the flow strarts. If it fails, the subscriber is informed and it may subscribe to a lower quality. This is repeated until success.

If, the quality of the link degrades, the network may no longer be able to transport an existing stream. Rather than dropping information, the subscription is cancelled and a new one with lower requirements may be requested as described above.

There are a number of advantages in the policy described above. First, the negotiation mechanism is simpler. Second, using discrete levels of quality, different streams may be more easily shared among subscribers. If each connection leads to a different QoS, sharing is clearly impossible. Third, there is a clear distinction between the negotiation at the application level and the negotiation with the network. Fourth, the network middleware is free to implement any transport mechanism including a multilayer one.

Other aspects such as synchronization and redirection of streams will be addressed during the talk.

5. Conclusion

In this paper, we have shown that using the publish-subscribe model has a number of advantages over the client-server for expressing AV communication and the related QoS. Still it requires some re-design of the applications. All the capabilities present in the conventional models can be expressed in a way that leads to a better software architecture because it reduces coupling. The control mechanisms are also simpler. All these solutions have been mapped with success onto UPnP.

References

[1] R. Gueth et al., "Broadcast-Based Communication in Distributed Control", *Proc. EUROCON'84*, Brighton, Sept. 26-28, 1984, pp.321-5.

[2] Thomesse J.P., "Time and industrial local area networks", *COMPEURO'93*, Evry (France), May 23-25, pp.365-374, 1993.

[3] J.-P. Thomesse et al., "An industrial instrumentation local area network", *Proc. of IECON*, pp.73-78, 1986.

[4] P. Eugster et al., ”The many faces of publish/subscribe”, ACM Computing Surveys 35 (2), June 2002, pp. 114-131.

[5] J. Ritchie, T. Kuehnel, “UPnP AV Architecture”, version 0.83, June 12, 2002, available at www.upnp.org.

[6] J. Dingel, D. Garlan, C. Damon, “A feasibility study of the HLA bridge”, internal report CMU-CS-01-103, March 15, 2001.

[7] A. Snyder, “The essence of objects: concepts and terms”, IEEE Software 10 (1), Jan. 1993, pp.31-42.

[8] IEEE Standard for Modelling and Simulation (M&S) High Level Architecture (HLA) – Federate Interface Specification (IEEE Std 1516.1-2000)

Interoperability between CE middleware standards

Helmut Bürklin
THOMSON R&D France SNC
CS17616
35576 Cesson-Sévigné Cedex, France
helmut.burklin@thomson.net

Abstract

Current standards for audio-visual home networking differ in a large number of aspects. Some characteristics of two prominent standards, UPnP and DHWG are compared with regards to device models, underlying communication system, stream handling and user presentation of controls. A comparison is made with the characteristics of the interoperability middleware as developed in the FABRIC project.

1. Introduction

In the forthcoming years, home devices will be interconnected to share their functionalities and offer the user a living in a smarter home. Consumer electronic devices will share audio-visual content (such as a broadcast video stream or an internet connection) and services (for instance recording a stream). For such a sharing to be effective, all the devices have to "speak the same language" : this is interoperability.

Unfortunately there is no such single language agreed yet: since the arrival of the first digital camcorders, 10 years ago, several different protocols have arrived on the market, and have gained acceptance in different areas. One can assume that at least some of the existing standards are here to stay, and that given the continuous technical evolution, the differing business interests, even more standards will be developed in the future, based or not on the existing ones. As examples of standardisation activities underway can be given DENI (now in CEA R7.6), DHWG or DVB-HN.

In the following the main characteristics of two of the more complex specifications shall be discussed, HAVi (Home Audio Video interoperability) and UPnP (Universal Plug and Play). They will be explained below in order to outline their main functionalities. From these descriptions, we will differentiate the common and the specific functionalities.

The FABRIC[1] project defined an interoperability architecture for in home video streaming on the base of the 'High Level Architecture' HLA, defined in standard IEEE 1516. The solutions chosen in this project are also sketched – without details – to compare to the 'classical' approach of established standards.

2. Comparison of main features

Before going in the details of the comparison, some basic differences between the contemplated standards shall be evocated: HAVI [1] was created as a common software architecture for audiovisual devices over a specific connection fabric, IEEE 1394 (Firewire)[6], which due to its inherent bandwidth reservation and stream transport capabilities looks promising for building an audio-visual network. Together with the functionalities of any private native API, it also defined a standardised HAVi Java API, the graphics function of which are based on a subset of Java AWG 1.1.

UPnP [2][3] is a protocol specification, enhancing the standard IP stack, and is as such network independent (though usually deployed over IEEE 802.3 - Ethernet). For the transport of audiovisual content UPnP relies on well known Internet standards as HTTP or RTP. The Digital Home Working Group [5] prepares Implementation Guidelines widely based on the UPnP architecture, where appropriate we will indicate some specificities of this approach.

The FABRIC [7] project has come up with an architecture in which applications use the publish/subscribe facilities to set up and control the video streams in the network. FABRIC-enabled devices shall contain applications that run above the F-RTI, run

[1] This work was funded by the European IST project IST-2001-37167, called FABRIC. It was carried out in collaboration with Philips, TUe, TNO, CSEM, INRIA, SSSA, MDH, UCL and UP.

time infrastructure, a software layer extended from the RTI as defined in IEEE 1516 [4], and that itself uses the services of an embedded HAVi or UPnP stack, dependent on the underlying network.

2.1 Device Model

All home-networks dealing with the distribution of audio-visual contents are made up of physical devices connected to some communication fabric. The architectural model tends to refine this classification: physical devices in general contain controllable entities (called 'device' in UPnP) and controlling entities (these contain the UPnP 'control points', but are often not given a name, mostly called 'application').

The mapping of physical devices (boxes) to logical devices is not identical for the different specifications: HAVi physical devices are grouped in 3 classes, FAV, IAV, BAV, i.e. physical devices able to host other devices, self contained devices, and devices that need a host. This has no analogy in the other standards. The first two of these may contain a controlling application. UPnP allows a physical device (root device) to expose sub-devices and control points. DHWG will in its first release of the Guidelines restrict devices to be either 'servers' (sourcing some content) or 'players' (rendering content and controlling the servers), both being possibly contained in a physical device.

Despite these differences, the model of the (controlled) device is quite identical: all standards specify such an entity, that in most cases maps to (the controllable part of) the box that the end user purchases, and this controllable entity contains functional groupings, often corresponding to a physical entity inside a device: these groupings carry different names: sub-units, sub-devices, functions, … In UPnP (and in the following of this article) they are called 'services', in HAVi their abstraction is called a 'functional component module' (FCM).

In FABRIC's publish/subscribe model devices are represented by federates, all devices based on the same underlying middleware form a Federation. The services available inside a device are published as objects of a given class, which in consequence corresponds to a service type. In difference to HAVi and UPnP does HLA not define operations (methods) on objects, but object attributes may be directly changed by other federates.

2.2 Addressing

To access services or devices in a network, the applications in the network should be able to identify and uniquely address the connected services.

On a Firewire network addressing is based on the node_id that can change at every bus reset, e.g. when a device is attached or disjoint from a given bus. But HAVi messages (that are exchanged between different devices or software modules inside a single device) use as identifier the SEID, composed of a handle that identifies a software element inside a device and the GUID, a unique identifier for each physical device. The mapping of GUIDs to node_ids is handled inside the HAVi stack in a manner invisible to the application.

UPnP devices are identified by their IP address. In a home network, generally private IP addresses are used, i.e. addresses that are not world wide unique, but only unique inside one organization, which is of no consequence for a home network. Uniqueness of IP addresses inside the home network is guaranteed by autoIP, or if a server is present by DHCP. The uniqueness of the ASCII coded unique device name (UUID) in UPnP is to be guaranteed by the manufacturer, it is usually built from the Ethernet MAC address and an installation time.

HLA does not allow subscribing explicitly a given object, but only to its class. In Fabric this difficulty is circumvented through the subscription of 'regions', groups of objects (here a single one) sharing certain characteristics. Since the device is not an object but a federate, it will be represented by a unique federate designator, provided by the RTI. A possible algorithm for its construction is to use the IP address concatenated with a local unique ID, or to use the GUID for HAVi.

2.3 Basic Communication

HAVi systems provide a generic messaging system for communication between software modules. The location of the communication peer is transparent to the application, i.e it does not matter whether the communication peer resides on the same host or not. Messages can be send synchronous, i.e. the call returns only after the response is available, or asynchronous, in which case the caller has to match the correct response by himself. Furthermore, a raw message mode allows a client to use a proprietary protocol where the HAVi request/response protocol does not apply.

Nevertheless, many of the HAVi functionalities are directly dependent on the availability of 1394 features: the connection or disconnection of a device triggers a system wide bus reset event, which allows to each device in the system to immediately update its knowledge about all available devices/services. The 1394 bus is a unique shared but collision free communication medium, of a given capacity (mostly 200 or 400 Mb/s, though a physical layer with 800Mb/s and more is now available). HAVi relies on the resource reservation protocol defined for this medium that allows guaranteed and timely stream delivery.

UPnP is built on top of the IP protocol, so intrinsically independent of the underlying physical medium. It relies on SOAP, a XML-based protocol transmitted over HTTP, for remotely invoking the actions exposed. As SOAP relies on the standard HTTP protocol, remote method invocation is always reliable. But UPnP does not specify if the remote invocation is synchronous or asynchronous.

Fabric enabled applications run above the RTI, following the anonymous publish subscribe paradigm. The F-RTI maps this to the services of HAVi or UPnP respectively. Gateways include mapping services for several middleware standards, so two distinct F-RTIs. The gateway application can in this case fulfil the role of a proxy between the two underlying home network technologies.

2.4 Discovery

Devices and services (their software representations DCM and FCM) in HAVi are strongly typed: they rely on fixed standard templates that fully define the available methods. Proprietary extensions are possible, but cannot be known from other devices/services. Each (physical) device contains a registry service that makes the list of all software modules present in the system available to all other modules and applications. The (distributed) registry is guaranteed to be always up to date since connections or disconnections of devices are always made known through system wide events (bus reset). The Registry API provides rather high-level search functions where search attributes can be combined with logical operators to perform complex searches.

In UPnP devices and services can fully describe themselves (services, names, …) by XML pages. A set of templates of some basic classes and devices has been defined by the UPnP committee, which can be completed by vendor specific extensions, allowing the advertisement of private methods, including their parameters. But since the semantics of a method stays unknown if it is not described in the standard, some prior knowledge by the controller is needed before he invokes such a method. Control point become aware of newly available devices and services by listening to the advertisements broadcasted by these devices. They also send queries in form of SSDP messages to all devices, only devices with the required type, name, .. or containing the specified service type will respond. Details of the identified services can be read in the corresponding XML pages.

In Fabric, the discovery service is split in two parts; (1) the FABRIC Directory Service (FDS-X) is part of the RTI that links to the discovery middleware of the underlying standard (HAVi, UPnP…), and (2) the FDS-application (FDS-A) that can publish and subscribe to discovery objects, has access to RTI functionality.

2.5 Eventing

UPnP allows a control point to subscribe to the events generated by a service. It registers for all the events generated by the service, not for a specific type. A contrario, HAVi allows a software element to subscribe to its event manager for a specific type of event.

Both layers generate events reflecting changes in the states of the services / software elements. For UPnP, events are changes in the value of some state variables, for HAVi changes in the state of software elements. Only HAVi generates events for changes in the system, essentially the network.

FABRIC: after proper subscription, a federate will receive a notification each time the attribute value of any object of the subscribed type has changed. The UPnP equivalent is the eventing mechanisms in which subscribers are informed of the change of service state variables.

2.6 Stream Management

HAVi streaming is entirely based on the IEEE 1394 and IEC 61883 standards. As such, it specifies stream set up between three partners, the controlling entity, the source, and the destination. Since the HAVi system not just specifies protocols for inter-device communication, but also the (internal) software architecture, streams internal to a device are handled in the same manner as streams between devices, i.e. for the application there is no difference of setting up a connection between the tuner and the display inside a TV or the tuner of a satellite receiver and the display of the TV. The stream manager software element inside each device is responsible for setting up and tearing down connections, allocating necessary resources (bandwidth and 1394 channel) and also of verifying the transport type. Connections can be specified also over different media (e.g. analogue cable) and may be of the point to point or broadcast type.

In UPnP streaming is in the hand of two different services, the Connection manager (for setting up and tearing down) and the Transport service (for control, e.g. trick modes). The AV architecture UPnP doesn’t rely on any specific protocol for conveying streams, so it has to support multiple protocols of this kind, defined out of the scope of the UPnP protocol itself. UPnP does not specify any resource reservation and handles so far no multicast streams.

Fabric innovates with the ‘content oriented approach’ in which the application indicates to a designated display that it has to present specified content. The location of the display and the content is irrelevant to the application and the underlying technology (the RTI and the middleware) provides the means to transport the actual content from its source to the destination.

2.7 Presentation, User Controls

The UPnP protocol offers the possibility to the vendor to define an presentation HTML (3.0 or later) page specific to the device. This page can also optionally embed control and eventing facilities.

HAVi principally defines two different methods by which a device manufacturer is able to present its own user interface and controls to the end user through a third party display device. The level 1 UI is based on the Data Driven Interaction (DDI) in which user interface elements can be loaded from a device, and displayed by a DDI Controller. The DDI Controller generates HAVi messages in response to user input, it also responds to HAVi messages sent by the DCM as a result of changes in device state. The level 2 UI is constructed by Java bytecode applications uploaded from the controlled device to the display device, if this contains a Java Virtual Machine (FAV).

FABRIC does not cover the aspect of user controls.

2.8 Device/Service ownership

HAVi specifies a 'Resource Manager', but this is not used for the allocation of network or system resources, as bandwidth or processing power, but to control access to devices and services. The Resource Manager allows clients to reserve a group of device resources (e.g. services) in an all-or-nothing fashion and so to ensure that clients that have reserved resources can rely on not being disturbed by other applications that try to use the same resources. Reservations can also be done for some future time period ('scheduled actions'). The resource management system performs a number of validations for scheduled actions at scheduling time. Pre-emption of resource reservations is possible under control of a human user.

UPnP (and so DHWG) do not dispose of a comparable characteristic.

In HLA each object attribute has a single owner at a given time. Only the owner is able to modify the value of the attribute. Ownership can be released As a consequence, service ownership in the manner of HAVi is implementable in a straight forward manner. But to allow to share attributes between several editors it has been agreed that attribute ownership will generally be relinquished as soon as possible.

2.9 Content formats

In the analogue area audio-visual content in the home followed a single standard by region, at least for content present in electronic form (i.e. NTSC for the US, SECAM for France, PAL for most other European countries). Nowadays multiple standards and transport formats are used: MPEG1 is popular on computers, DV on camcorders, MPEG2 on DVD and TV – but while DVD uses program stream encapsulation, TV programmes are broadcast as transport streams. With the progress in compression technologies new formats are arriving (e.g. all the MPEG4 flavours) and will continue to appear for a long time.

HAVi limits the used content to those the transport of which is standardised in ISO/IEC 61883. For video, all available devices happen to support both, MPEG2-TS and DV, so interoperability problems did not appear, but this was not guaranteed by the specification.

UPnP allows the use of any content format, signalled in the content directory by its mime-type. This approach allows easy integration of future (even private) formats, but lacks any guarantee for the user that content available on one server be playable on another device.

DHWG tries to resolve the problem by defining as well mandatory as optional formats, the server presenting content always in a format suitable to the renderer, if ever this is possible. But transcoding may be technically difficult (and not perfect) and since the content owner - which is nowadays very often not the consumer ! – may prohibit transcoding, this approach is not bullet-proof either.

Fabric did not do special work on content formats.

3. Conclusion

Existing middleware standards have been shown to be different, as well in their scope, as in the implementation choices. This can partly be explained by their dependency on underlying low level protocols, or the relation to the middle and high level protocol stack around IP. Other differences can be found that are due to the scope: defining a protocol architecture for communication between devices, or defining a middleware architecture covering also communication aspects between software modules inside a single device. But also strong commonalities can be found, especially in the underlying device and service model. These commonalities in the models suggest that it is possible to develop an application programming interface of a network interoperability layer through which services of either standard are accessible. It could be shown how the discussed functionalities of the existing standards can still be implemented if this API is based on an anonymous publish/subscribe paradigm.

References

[1] *The HAVi specification*, Version 1.1, available from http://www.havi.org/, 2002

[2] *Universal Plug and Play Device Architecture*, version 1.0, 2000

[3] *UPnP AV Architecture*, version 0.83, June 12, 2002. Available at http://www.upnp.org/.

[4] *Standard for Modeling and Simulation High Level Architecture*, IEEE standard 1516-2000

[5] *Digital Home White Paper*, June 2003, Digital Home Working Group Available at http://www.dhwg.org/.

[6] *IEEE Standard for a High Performance Serial Bus*. IEEE Std. 1394-1995

[7] P.vdStok et al., "The FABRIC project" in *Software Architecture, First European workshop* , EWSA 2004, St Andrews, UK, May 2004, proceedings eds F. Oquendo, B. Warboys, R Morrison, pages 272-278. LNCS 3047, Springer, 2004

[8] European IST project 2001-37167 – *FABRIC*. at http://www.extra.research.philips.com/euprojects/fabric/

Middleware for MPEG applications

Jean H.A. Gelissen
Philips Research Laboratories Eindhoven
Prof Holstlaan4, 5656 AA Eindhoven, NL
jean.Gelissen@philips.com

Abstract

The goal of the MPEG Multimedia Middleware (M3W) is to improve application portability and interoperability through the specification of a set of APIs (syntax and expected execution behaviour) dedicated to multimedia. This is an end-user centric view of middleware[1]*. This approach towards middleware will allow:*

1. *Application developers to quickly develop and easily maintain multimedia applications;*
2. *Application and Service providers to limit cost of application provisioning;*
3. *Manufacturers to provide compelling and interoperable application development environment.*

1 Introduction

The objectives of the newly started M3W initiative in the context of the MPEG (ISO-IEC JTC 1-SC 29-WG 11) standardisation effort are the definition of a set of APIs supporting the following functionality:

- Allowing multimedia applications to execute functions provided by the middleware in a standard way without the need for an in-depth knowledge of the middleware;
- Allowing multimedia applications to trigger the update / upgrade or extension of the API, e.g. because they are not present or outdated.

To satisfy these objectives the following M3W technologies will be required:

- Functional APIs to allow the applications to access the multimedia functions;
- APIs to manage the execution behaviour (e.g. resource (memory, power, processor, network usage) of the multimedia functions;
- APIs to manage the isolation (e.g. memory protection) of the multimedia functions;
- APIs to manage the 'life-cycle' of the components of the M3W (e.g. identification, download, installation, activation, de-activation, un-installation).

[1] Note that the multimedia application domain is specific for the deployment of middleware in the MPEG context, the mechanisms described in this paper are generally applicable to middleware.

Where the 'traditional' MPEG community will specifically address the first of these technologies (functional APIs for multimedia applications) the ITEA projects ROBOCOP [1] and Space4U [2] will contribute, as explained in this paper, on the other technologies. In general MPEG [3], [4] has specified these goals as follows:

- M3W shall enable identification of middleware components needed by an application, localization of middleware components, retrieving of middleware components, consistency checking of the overall middleware components with respect to the platform, registering of middleware components, instantiation of middleware components and removing of middleware components. All these activities together constitute the lifecycle management of the middleware components.
- M3W shall enable resource management mechanisms: evaluation of the resources currently available on the device with respect to the resources requested by the application and the operation context of the device as well as the user. These mechanisms together enable resource management policy (e.g. priority, graceful degradation of application execution and the like).
- M3W shall enable fault management mechanisms to allow the handling of exceptions, prevent the propagation of software faults and the isolation of software components that exceed the execution space allocated to them.

Traditionally consumer electronics devices, one of the main deployment areas of MPEG standards, did not have any software at all and all functionality was realized in hardware. From ~1980 onwards software was introduced in parts of these systems starting in the control part. Gradually more and more software was introduced in several parts of the systems. At the same time parts that differentiate the products between the different vendors, were more and more obtained from external vendors. In recent years also the role of the different players in the field has changed, the semiconductors vendors focus on providing a general applicable platform that can serve a big market while the consumer electronics vendors focus on offering to their customers features and services that differentiate their product from the competitor's offer. All these developments have made the two traditional players (semiconductors and consumer electronics vendors) less dependent on each other and opened the opportunity for new players in this field in the so called middleware that bridges the gap between the platform that the semiconductors vendors provide and the applications that the consumer electronics vendors offer to their customers. All these developments together are illustrated in Figure 1.

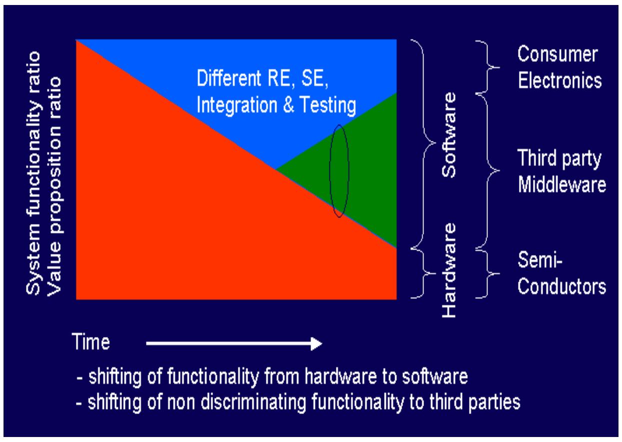

System functionality ratio
Value proposition ratio
Different RE, SE, Integration & Testing
Software
Hardware
Consumer Electronics
Third party Middleware
Semi-Conductors
Time
- shifting of functionality from hardware to software
- shifting of non discriminating functionality to third parties

Figure 1, Trends in consumer devices

Concluding, today's situation in the consumer electronics industry can be characterized, besides of course the development of the systems parts that make the actual difference with the competitors offers, as 'integration & testing'.

2 Separation of concerns

This situation, as illustrated in the previous chapter, is not different from the developments in the semiconductors industry some years ago that resulted in the establishment of a market for third[2] party modules that are integrated into the semi conductor's company's end products. In a market situation as sketched above the roles and responsibilities of the different providers need to be clear to avoid the overlap or missing of functionalities. Figure 2 illustrates the different roles of the (three) actors that together compose a system.

A number of de-facto industry driven and / or international standardization bodies are active on the interface layers between the three above-mentioned architectural areas for different reasons:

- The application layer interface (API between the middleware and the application) aims at the unification of interfaces to allow the services to operate on a large installed base of (consumer) devices. In the stationary domain bodies like ATSC with OCAP and DASE, DVB with MHP and for example OMA in the mobile domain are very active on defining these interfaces.

- The platform layer interface (API between the platform and the middleware) aims at the unification of the interfaces to allow the platforms to be used in a large variety of appliances. Also in this area there is a lot of activity both in the stationary domain, for example with the CE-Linux and STAPI initiatives and the recently announced UH-API from Samsung and Philips. In the mobile world the MIPI initiative is recently started with a similar aim.

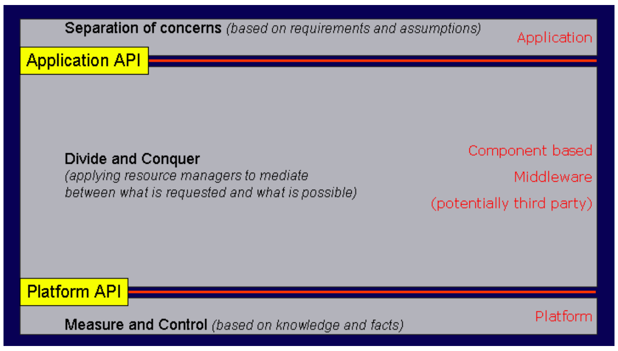

Separation of concerns (based on requirements and assumptions)
Application
Application API
Divide and Conquer
(applying resource managers to mediate between what is requested and what is possible)
Component based
Middleware
(potentially third party)
Platform API
Measure and Control (based on knowledge and facts)
Platform

Figure 2: Middleware SW components in their context

A (non exhaustive) list of examples of the entities that could be positioned in the three systems layers of Figure 2 is as follows[3]:

- Application layer
  - Electronic Program Guides
  - Advisors / Agents
  - DVD control program
  - TV control program
  - Storage management
  - Asset management
  - …

- Middleware layer
  - Content Management Support
  - WEB-DVD
  - Photo CD
  - Video Editing
  - PVR
  - Internet Browser

[2] Third party to the system integrator, mostly being the consumer electronics company, next to the traditional semiconductors company and the consumer electronics company.

[3] Note that international standardization bodies (like MPEG, HMEG, TVA, DVB, ATSC, CELF, MIPI) are active in these fields, however the standards are very often known from the resulting API definitions that can be fond at the interfaces between the (three) layers as depicted in this overview.

    - Media decoders and encoders
    - Media filters
    - MHEG
    - …

- Platform Layer
    - Operating System
    - (Software) Streaming
    - System Services
    - Device Drivers
    - Media decoders and encoders
    - …

Note that a number of the mentioned examples are already the result of international or industry de-facto standardization activities and are as such (potentially) available from multiple sources. Furthermore it should be noted that with respect to the media decoders and encoders, positioned in both the middleware and platform layers in the lists above, for the near future these functions probably still depend a lot on ASICs and as such are positioned in the platform layer. It is expected however that there will be a gradual shift of these functions towards software on programmable media processors and / or re-configurable computing devices and as such these functions become part of the middleware layer. The positioning of these functions in actual devices will however always be an economic trade-off and the move to the programmable media processors and re-configurable computing devices will only happen when the application needs it (due to a changing application mix requiring multiple decoders in different settings), when decoders will be (or need to be) down-loaded along with the content or when it becomes cheaper. The last one might seem strange but could happen in the case of re-configurable computing devices.

3 The need for trustworthiness

The integration and testing of the (third party) Middleware SW components into complete systems can follow two different approaches:

- Glass or White box (components are subject to / for change)
    - Need for in-depth expertise to understand / modify
    - Adaptation with every update / upgrade / extension
- Black box (components have to be taken as they come)
    - Hard to understand and control
    - Need for a 'glue layer' to fit into proprietary part

However both approaches will miss the desired effect in the future for technical and economical reasons. From the technical point of view the increasing complexity of the systems does not scale with the currently applied approaches. From the economical point of view the effort required for the integration & testing might easily become more important than starting an in-home development.

To solve both of these issues a better control of the integration as well as the execution process are needed to allow the introduction of open dynamic multi supplier systems in the targeted market places. This requires that mechanisms / measures are being developed to deal with:

- Robustness (active during execution and guaranteeing the expected functionality)
- Reliability (active during component lifetime management and guaranteeing the expected usefulness)
- Security (active during critical sections of diverse nature and guaranteeing the expected behaviour (making use of all the functionalities described before)

These three categories are often referred to as the essential providers of "Trustworthiness". In parallel it is also essential that that mechanisms / measures are being developed to allow the transition from *design* time (static) to *run* time (dynamic) for *configuration* and *consistency* management. Figure 3 shows the added extra functionalities next to the traditional functional section of the software component to enable the intended level of robustness and reliability.

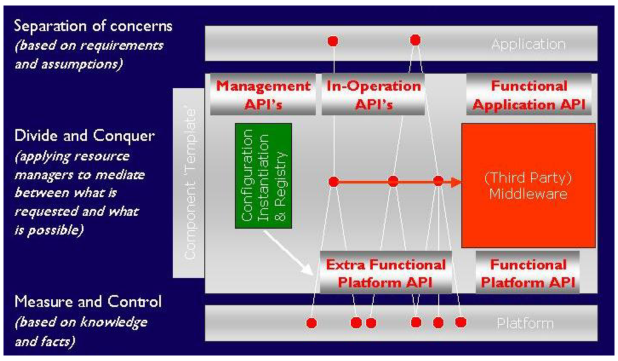

Separation of concerns
(based on requirements and assumptions)
Application
Management API's
In-Operation API's
Functional Application API
Divide and Conquer
(applying resource managers to mediate between what is requested and what is possible)
Component 'Template'
Configuration Instantiation & Registry
(Third Party) Middleware
Extra Functional Platform API
Functional Platform API
Measure and Control
(based on knowledge and facts)
Platform

Figure 3: Extra functionalities in SW components

For completeness Figure 4 is added in this introduction to map the three APIs that are discussed in the remainder of this document on the introduced diagram.

4 API requirements

Note the lists that are given in the paragraphs below are not intended to be complete.

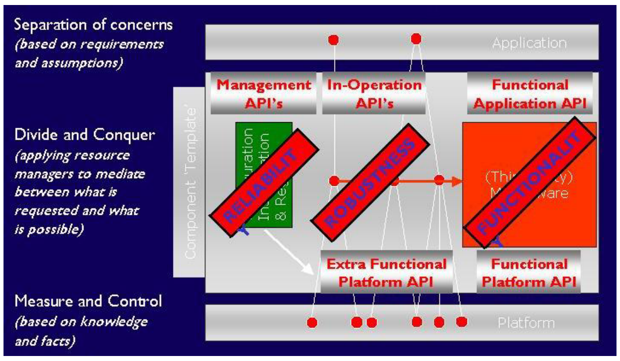

Separation of concerns
(based on requirements and assumptions)
Divide and Conquer
(applying resource managers to mediate between what is requested and what is possible)
Measure and Control
(based on knowledge and facts)
Application
Management API's
In-Operation API's
Functional Application API
Component 'Template'
RELIABILITY
ROBUSTNESS
FUNCTIONALITY
Extra Functional Platform API
Functional Platform API
Platform

Figure 4: Extra functionalities in SW components

4.1 Multimedia APIs

The Multimedia APIs are intended to give the requesting application access to multimedia functionalities provided by the middleware components. A non-exhaustive list of such APIs can be found below:

- Media Processing Services (including coding, decoding and trans-coding);
- Media Delivery Services (through files, streams, messages);
- DRM Services;
- Access to data (which data?);
- Access to Metadata; Metadata Edit; Metadata Search.

4.2 Management APIs

To allow the handling and management of the middleware components, procedures are required to deal with essential processes like service localization, decision / validation, downloading, instantiation and de-installation / removal.

The decision process for instance has to validate if the resulting system after the instantiation of the new, updated or extended functionality still provides a valid system with respect to base requirements. Service composition including an in-depth analysis of all possible side effects serves as the basis for this process. These procedures and decision processes, and the associated management APIs, need to be supplemented with business models and rules to enable a middleware component trading sevelopment (often referred to as ECO system) to happen. If this document has to be read by people other than specialists it is necessary to give some background, maybe in an appendix.

Such Management APIs for instance are related for instance to:

- Component service discovery;
- Component service fetching / downloading;
- Component service instantiation;

4.3 In-operation APIs

The In-Operation APIs take the required extra-functionalities into account that are essential for the sound execution of the application that request the services of the middleware components witch on their turn request services from the underlying platform.

In this context the middleware components mediate between the functionality requested by the application (based on requirements and assumptions) and what is possible on the platform (based on knowledge and facts). These extra-functionalities include for instance the access to:

- Resource management (processing cycles, memory, bandwidth / throughput, power, UI real estate and the like);
- Real-time operation & guarantees (for instance to allow access to schedulers).

In general these requirements require a two-layer approach towards the middleware, the upper layer being the interface to the application, the lower layer being the interface to the (abstracted) operating system and system services to allow for the monitoring and control of the above mentioned extra-functionalities. The middleware component itself can already handle a large portion of the control and monitoring and only pass and abstraction towards the application.

Such In-Operation APIs for instance are for instance related to:

- Resource management (power, processor, memory);
- Fault or exception management.

4.4 Other APIs

It is assumed that other middleware functionality will be needed but will be defined outside M3W. A non-exhaustive list of such APIs can be found below:

- Execution environment APIs;
- Composition APIs;
- Rendering APIs;
- User Interface APIs;
- Communication APIs and Protocols (e.g. SMS & MMS messaging, streaming protocols.

References

[1] The ITEA ROBCOP project, http://www.extra.research.philips.com/euprojects/robocop/
[2] The ITEA Space4U project, http://www.extra.research.philips.com/euprojects/space4u/
[3] MPEG Multimedia Middleware: Context and Objective, output document w6335 of the 68th MPEG meeting, Munich Germany, April 2004.
[4] MPEG Multimedia Middleware: Requirements on the MPEG Multimedia Middleware V2.0, to be published as output document of the 70th MPEG meeting, Mallorca Spain, October 2004.

Group Management for In-home Ad hoc Networks

Malika Boulkenafed, Valérie Issarny, Jinshan Liu
INRIA
UR Rocquencourt
Domaine de Voluceau, BP 105, 78153 Le Chesnay Cedex, France
{Malika.Boulkenafed, Valerie.Issarny, Jinshan.liu}@inria.fr

Abstract

In the context of Ambient Intelligence where services should be accessible anytime, anywhere, the realization of a given service depends on the location of the involved entities and on the specific features of the entities. In general, a service will be realized through cooperation among a set of distributed sub-services that are selected according to the location from which the service is requested and the target service. In addition, as users move and resource availability changes over time, the realization of a given service should be adaptive so as to react to such environmental changes in a way that maintains as much as possible quality of service perceived by users (e.g., response time). This paper addresses the above issue through the management of dynamic groups over Mobile ad hoc networks. Applications may then execute on top of groups, being relieved from the complexity of the highly dynamic network environment.

1. Introduction

Mobile ad hoc networks (MANET) pave the way for ambient intelligence and more specifically ubiquitous networking, due to inherent support for any-time, any-where network access for mobile users. Nonetheless, the highly dynamic nature of mobile ad hoc networks poses tremendous challenges for the development of applications since the application's context keeps changing over time. One approach to master this complexity lies in the management of groups over MANET, i.e., applications execute on top of groups that manage the dynamic execution context, including mobility-related failures. There has been extensive research on group management and related group communication services in the context of fixed networks, with special emphasis on providing availability properties [1,2]. However, proposed solutions cannot be applied directly to mobile wireless networks due to the network's highly dynamic topology [3]. This has led to adapt the management of group membership to the specifics of MANET.

Group membership is primarily defined according to the functional property to be achieved by the group, e.g., collaborative editing, sharing a computational load, increasing performance, providing fault tolerant service. In general, a member may leave a group because it failed, explicitly requested to leave, or is expelled by other members. Similarly, a member may join a group because it explicitly requests it or recovers from failure. A group membership protocol must manage such dynamic changes in a coherent way, i.e., all members of the group must have a consistent view of the group's membership despite failures [2]. The highly dynamic topology of MANET introduces additional complexity in the management of group membership because connections may be transient and partitioned networks may never be rejoined together. However, group membership for MANET may still be defined as for fixed networks, i.e., according to the functional property to be realized. In this case, it is considered that the MANET allows restoring lost connections using the underlying routing protocol. Solutions then lie in the adaptation of the group communication service to the dynamic topology of the network [4]. Such an approach does not adapt the system's functions to the specifics of the network but rather adapt the implementation of traditional distributed system functions. However, it is advantageous to revise the definition of group membership so as to integrate the connectivity dimension in addition to the functional one [5]. This allows dealing with quality of service requirements by bounding communication latency [6] and/or supporting location-aware applications [7]. Specifically, connectivity-constrained group membership enables managing a dynamic (sub-)network that is configured according to both connectivity constraints and the functional property to be implemented, while hiding mobility-induced link failures to applications [8]. Connectivity constraints may vary from 1-hop to multi-hop connectivity, where unbounded multi-hop connectivity corresponds to the aforementioned connectivity-unaware group membership addressed in [4]. Connectivity constraints may then be fixed according to the network's connectivity (i.e., number of hops) as in [8,9] or the respective geographical position of the group's members as in [5]. The definition of group membership may further be extended with integrity constraints (e.g., security constraints, size) [10].

In general, group management in MANET embeds a number of design dimensions relating to connectivity among group members and the non-functional properties that are offered to applications. In this paper, we provide a characterization of the key attributes of group membership in ad hoc networks (§2). We then discuss exploitation of group management over MANET, in the context of ambient intelligence scenarios relating to quality of service management in the home network (§3). Finally, we conclude with a summary of our contribution and our future work (§4).

2. Group Membership Attributes

Groups are first defined with respect to a given functional property. We denote such a property by f. Without loss of generality, we assume that the property is offered by any node, as opposed to being an aggregation of some functions provided by grouped nodes. We use the Boolean function $support(x, f)$ to denote that node x offers function f. Note that f may characterize various features supported by nodes, and resembles to resources considered in resource discovery protocols. The following introduces the network model that we consider and then the non-functional attributes relevant to group membership in MANET.

Network model: We consider a WI-FI-based ad hoc network consisting of a set $\mathcal{N}$ of N nodes, and assume that every node x of $\mathcal{N}$ has a unique identifier $Id(x)$. However, we do not fix the routing protocol that is used. We further introduce the following functions to reason about the connectivity of nodes, for $x \in \mathcal{N}$ and a time period $\mathcal{T}$ that is such that the network does not change over $\mathcal{T}$.

- $Proximity(\mathcal{T}, x, p)$ returns the geographical distance in meters between the location of x and geographical position p over $\mathcal{T}$.
- $Distance(\mathcal{T}, x, y)$ returns the geographical distance in meters between the respective locations of x and $y \in \mathcal{N}$ over $\mathcal{T}$.
- $Connectivity(\mathcal{T},x)$ returns the set of all nodes of $\mathcal{N}$ with which x can communicate using the underlying network protocols over $\mathcal{T}$; note that due to the asymmetric nature of wireless networks in general, $y \in Connectivity(\mathcal{T},x)$ does not imply $x \in Connectivity(\mathcal{T},y)$.
- $DualConnectivity(\mathcal{T}, x)$ returns the set of all nodes y of $\mathcal{N}$ such that $y \in Connectivity(\mathcal{T}, x)$ and $x \in Connectivity(\mathcal{T}, y)$.
- $Hops(\mathcal{T}, x, y)$ returns the number of hops for communication between x and y for any y belonging to $Connectivity(\mathcal{T}, x)$.

Finally, we denote by $\mathcal{G}^f$, a group realizing function f, i.e., $\mathcal{G}^f = \{x \mid x \in \mathcal{N} \text{ and } support(x, f)\}$

Location: We now define functions characterizing group membership with respect to constraints set on the relative location of member nodes.

Location-unaware groups as, e.g., addressed in [4], are defined solely with respect to the functional properties offered by the group members. Hence, LocationUnaware$(\mathcal{G}^f)$ always holds.

Proximity-based groups as, e.g., addressed in [5,10], set that group members should be in a given geographical area, whose location may be fixed a priori or set relative to the position of group members. Let pos denotes a referenced geographical position and $dist$ denotes the maximal geographical distance that is allowed, we get:

$$Geographical\_Proximity(\mathcal{G}^f, \mathcal{T}, pos, dist) \Leftrightarrow \forall x \in \mathcal{G}^f : Proximity(\mathcal{T}, x, pos) < dist$$

$$Relative\_Proximity(\mathcal{G}^f, \mathcal{T}, dist) \Leftrightarrow \forall \{x, y\} \in \mathcal{G}^f : Distance(\mathcal{T}, x, y) < dist$$

Bounded groups defined with respect to the number of hops separating node members as, e.g., addressed in [8,9], are defined in a similar way based on the maximal number of hops, noted $hops$, between nodes:

$$Bounded(\mathcal{G}^f, \mathcal{T}, hops) \Leftrightarrow \forall \{x, y\} \in \mathcal{G}^f : Hops(\mathcal{T}, x, y) \leq hops$$

Note that the above functions are not exclusive of each other and may be combined for the definition of a given group.

Openness: Group membership may be restricted to authorized nodes. We model such a constraint using the notion of *security domain*: a security domain $\mathcal{S}^f$ sets nodes of $\mathcal{N}$ that trust each other towards realizing function f. Practically, a security domain is managed by a trusted third party to which nodes may authenticate and register themselves; nodes then get a signed certificate that they may use to authenticate themselves with other nodes belonging to $\mathcal{S}^f$. Secure group communication may further be enforced through the implementation of group key agreement within the group. We get:

$$Closed(\mathcal{G}^f, \mathcal{S}^f) \Leftrightarrow \forall x \in \mathcal{G}^f : x \in \mathcal{S}^f$$

Connectivity: Group membership may require full, partial or even loose connectivity among nodes. In general, connectivity constraints may be combined with any of the aforementioned location-related constraints and may apply to both open and closed groups. Loose connectivity consists of relying on the connectivity enabled by the underlying network over time and thus does not impose any specific constraint. A fully connected group is further characterized by:

$$Connected(\mathcal{G}^f, \mathcal{T}) \Leftrightarrow \forall \{x, y\} \in \mathcal{G}^f : y \in DualConnectivity(\mathcal{T}, x)$$

Partial connectivity is defined according to the client and server roles of nodes with respect to function f. We use the function $client(x, f)$, resp. $server(x, f)$, to denote that x is client of f, resp. server of f. We get:

$$Partial(G^f, T) \Leftrightarrow \forall x \in G^f, \exists y \in G^f : server(x, f) : y \in DualConnectivity(T, x)$$

Note that $Connected(\mathcal{G}^f) \Rightarrow Partial(\mathcal{G}^f)$. Also, symmetric communication links may not be required between client and server nodes depending on the interaction patterns required by the application. However, we consider dual connectivity only, as this is the most common case for applications. The definition of partial connectivity with uni-directional reachability is further direct to infer. Finally, we enforce full connectivity among server nodes.

QoS awareness: Group membership may further be constrained for the sake of enhanced quality of service, i.e., members of the group must meet a number of Quality of Service (QoS) attributes. Various QoS attributes may be considered. In particular, the following attributes appear to be the most dominant in the context of MANET [11]: reliability, security, performance and transaction that relate to service-level attributes, and CPU load, memory, bandwidth and battery that relate to resource-level attributes. Then, a *QoS-aware group* may restrict group membership to nodes meeting the QoS attributes that are fixed for the group among the above.

In addition, group membership may be constrained so as to limit the probability of a node leaving a group. For instance, exploiting the movement of nodes has been suggested as an additional criterion for integration within a group [8]. In general, disconnection of a node from a group may be due to the node's mobility and/or the node's resource scarcity. The former may be anticipated based on information on the node's movement, and the latter may be anticipated based on information about resource-level QoS attributes of the node.

As part of the IST Ozone project[1], we have introduced a group management service that is generic with respect to the above membership attributes, and realizes three basic functions: (i) decentralized discovery of group members, (ii) initialization of the group that is carried out by one of the group members, called *leader*, and (iii) management of the group's dynamics that consists in periodically checking for changes in group membership and electing a new leader [13]. We are further implementing the group service on top of the Ozone middleware that is aimed at ambient intelligence applications and supports mobile Web services [14].

3. QoS Management in the Home Network

Group management may be exploited in the context of home networks where multimedia content available from servers within the home (e.g., Internet server, content storage server) is accessed from various sources (e.g., fixed displays in the home, PDAs). The issue that arises in this context relates to bandwidth management so as to ensure quality of service to users. In this context, we define a group, called $\mathcal{G}^{QoSHome}$, and dedicated to dynamic bandwidth management in the home network.

Membership constraints associated with the $\mathcal{G}^{QoSHome}$ group are:

- *Geographical_Proximity*($\mathcal{G}^{QoSHome}$, *pos_home, dist)* , i.e., members of the group should be located within the home.
- *Partial*($\mathcal{G}^{QoSHome}$, $\mathcal{T}$), i.e., client nodes should be connected with server nodes.

The *QoSHome* function relates to implementing dynamic bandwidth management within the home network, according to content access from client nodes. We use results of [12] to handle dynamic bandwidth management. Briefly stated, reference [12] introduces a solution to distributed weighted fair scheduling in a one-hop ad hoc network, hence allowing only one node to transmit on the channel at a time. Each flow in the network is assigned a weight that defines the flow's bandwidth requirement with respect to other flows. Assignment of weights in the ad hoc network then relies on:

- A centralized Bandwidth Manager (BM) that performs admission control and dynamic bandwidth management. The BM receives bandwidth requirement of a server node at the beginning of a connection, and gauges proportion of channel time, which may be equal to 0% if the flow cannot be accommodated due to the current network's load. The BM further performs dynamic bandwidth management following changes in available bandwidth, either due to the termination of flows or tuning of the perceived bandwidth used for estimating bandwidth requirement.
- A per-flow Rate Adaptor (RA) at the application level that controls the packet transmission rate of associated flow according to the time channel proportion allotted to the flow by the BM.

A per-node Total Bandwidth Estimator (TBE) that estimates the total network bandwidth for each flow sourced at the node it resides on, which is what a given flow perceives to be the total bandwidth of the network at a particular instant in time.

Hence, management of the $\mathcal{G}^{QoSHome}$ group allows nodes of the home network to share the network's bandwidth for concurrent access to the multimedia content available within the home. Precisely, group initialization is customized so that all peer nodes are known to each other, further enabling client nodes to establish connection with server nodes. The group then manages bandwidth allocation among nodes, as connections are initiated and terminated. The architecture of [12] maps quite directly to the one of the group service: each member runs a TBE, and a RA per flow it transmits. However, while the BM is centralized and hosted on a single node in [12], the BM task is fairly distributed among all nodes in our design; it is executed by the node acting as group leader, which changes every

[1] http://www.extra.research.philips.com/euprojects/ozone/

period. Note that we could actually assign the BM task to a single node, i.e., one of the fixed server nodes since they are resource-rich and are expected to remain in the group. However, our design adheres to the decentralized principle for group management in MANET, allowing for its exploitation beyond home networks and for handling server failures.

4. Conclusion

Group management appears as a key middleware functionality for assisting the development of applications over MANET. Group management takes care of managing a dynamic sub-network on top of which the application executes towards implementing given functional and non-functional properties. Group management over MANET has actually given rise to various studies over the last couple of years, each concentrating on specific applications. However, a distinctive set of key attributes may be identified for MANET-based groups, which may further be exploited to design a generic group service that is to be customized by applications.

This paper has introduced key attributes for group management over MANET, in particular based on applications published in the literature. Those attributes amount to setting membership constraints in relation with the location, connectivity, authentication and supported QoS of group members. We then have discussed one specific group instance over MANET, which relates to QoS management in the home network. We are currently implementing a group service that is generic with respect to the above membership attributes, whose design may be found in [13]. We will then implement the group instance that has been discussed so as to experiment with our solution and assess it in terms of performance.

Acknowledgment: This work has received the support of the European Commission through the IST programme as part of the IST Ozone project.

References

[1] D. Powell *et al.*. "*Group Communication (Special Issue)*". CACM 39(4). April 1996.

[2] G. Chockler, I. Keidar, and R. Vitenberg. "*Group Communication Specifications: A Comprehensive Study*". ACM Computing Surveys 33(4). December 2001.

[3] C. Basile, M-O. Killijian, and D. Powell. "*A Survey of Dependability Issues in Mobile Wireless Networks*". Technical Report, LAAS CNRS, France. February 2003.

[4] J. Luo, P. Eugster, and J-P. Hubaux. "*PILOT: Probabilistic Lightweight Group Communication System for Mobile Ad Hoc Networks*". Technical Report No IC/2003/35. EPFL. May 2003.

[5] R. Meier, M-O. Killijian, R. Cunningham, and V. Cahill. "*Towards Proximity Group Communication*". In Proceedings of the 1st Workshop on Middleware for Mobile Computing (in Conjunction with Middleware'2001). 2001.

[6] B. Hugues and V. Cahill. "*Towards Real-time Event-based Communication in Mobile Ad Hoc Wireless Networks*". In Proceedings of the 2nd International Workshop on Real-time LANs in the Internet Age. July 2003.

[7] J. Sanneblad and L. E. Holmquist. "*Using Ad Hoc Network Games to Support Face-to-Face Interaction in Public Spaces*". Demo at UIST. 2002.

[8] G-C. Roman, Q. Huang, and A. Hazemi. "*Consistent Group membership in Ad Hoc Networks*". In Proceedings of ICSE'2001. 2001.

[9] L. Briesemeister and G. Hommel. "*Localized Group Membership Service for Ad Hoc Networks*". In Proceedings of the International Workshop on Ad Hoc Networking (IWAHN). 2002.

[10] A. Meissner and S. B. Musunoori. "*Group Integrity Management Support for Mobile Ad Hoc Communities*", Proceedings of the 1st Workshop on Middleware for Pervasive and Ad Hoc Computing Computing (in Conjunction with Middleware'2003). 2003.

[11] J. Liu and V. Issarny. "*QoS-aware Service Location in Mobile Ad Hoc Networks*". In Proceedings of the 5th IEEE International Conference on Mobile Data Management. January 2004.

[12] S. Sha, K. Chen, and K. Nahrstedt. "*Dynamic Bandwidth Management for Single-hop Ad Hoc Wireless Networks*". In Proceedings of the 2003 IEEE International Conference on Pervasive Computing (PerCom). 2003.

[13] M. Boulkenafed, D. Sacchetti, V. Issarny. "*Using Group Management to Tame Mobile Ad Hoc Networks*". In Proceedings of the IFIP TC8 Working Conference on Mobile Information Systems. 2004. To appear.

[14] V. Issarny, D. Sacchetti, F. Tartanoglu, F. Sailhan, R. Chibout, N. Levy, and A. Talamona. "*Developing Ambient Intelligence Systems: A Solution based on Web Services*". Journal of Automated Software Engineering, 2004. To appear.

Combining HAVi and OSGi for Remote Control of Consumer Electronic Devices in Heterogeneous Home Networks

Isabell Jahnich
University of Paderborn / C-LAB
33098 Paderborn
Germany
Isabell.Jahnich@c-lab.de

Michael Ditze
University of Paderborn / C-LAB
33098 Paderborn
Germany
Michael.Ditze@c-lab.de

Reinhard Bernhardi
Siemens Business Services / C-LAB
33098 Paderborn
Germany
Reinhard.Bernhardi@c-lab.de

Abstract

In particular in recent years the idea of Smart Homes has been established. Due to the increasing number of integrated technologies, particular emphasis is put on the convergence of heterogeneous networks. Beside numerous specified middleware solutions, HAVi (Home Audio Video interoperabililty) exposes as an auspicious standard to connect devices within the home entertainment environment. To support interactions with different home areas and the Internet, a gateway is required. As the most established gateway technology the specified Service Platform of OSGi (Open Service Gateway initiative) acts as a middleware that allows to interoperate devices and services in heterogeneous networks. The popularity of HAVi and OSGi builds a promising foundation for the seamless symbiotic integration of these standards.
Consequently, this paper describes an efficient approach for the combination of HAVi and OSGi. It is based on a current representation of connected CE (Consumer Electronic) devices as Bundles within the OSGi environment and a communication that is realised via dedicated HAVi applications. Unlike other approaches interactions between heterogenous networks is supported and additionally remote controlling of connected HAVi devices is permitted.
Keywords: HAVi, OSGi, Service Gateways, middleware

1. Introduction

Interaction between heterogeneous networks within the home appears the environment to be "intelligent". In particular the ability of remote controlling Consumer Electronic devices e.g. from appliances that are based on industrial busses and a seamless connection to the Internet that allows accessibility of devices from arbitrary locations make a Smart Home attractive to end users. The connection of home entertainment devices requires adequate transfer rates for multimedia data. On the one hand, HAVi is an auspicious technology that supports the interoperable connection of Consumer Electronic devices. While this standard has been established in the home area, the biliteral interaction with non-HAVi networks, primarily the Internet, is not realised yet.
On the other hand, OSGi offers a Service Gateway, a network point that acts as an entrance to another environment, to connect heterogeneous networks. We propose to establish a biliteral communication between HAVi and additional networks by integrating the Service Gateway from OSGi. Unlike other approaches, integrated HAVi devices are represented within the OSGi environment, and their functionalities are mapped to registered Services. Moreover interaction between both standards is realised by dedicated applications that are on the one hand HAVi software elements and on the other hand accessible by OSGi components.
Since the OSGi standard provides the possibility of connecting heterogeneous networks supported by numerous hosted Services and because of its popularity we believe that this gateway realisation is a suitable technology. Even though the OSGi specification envisions the connection to HAVi networks, a detailed description of the interface is not realised yet.
Finally, there are numerous advantages with respect to the combination of OSGi and HAVi: Primarily, the seamless interaction between the entertainment environment and other local, non-HAVi networks, the remote accessibility of HAVi devices via the Internet from any web capable device, and the possibility add logging functionality, e.g. of errors and events within the HAVi environment that is not supported.
After a short introduction to both technologies ([1],[2]) in section 2 and the related work (see Section 3) section 4 presents the architecture that outlines the interface between HAVi and OSGi. Section 5 deals with a detailed description of related components. The approach has been developed within the European ITEA East-EAA-project [1]. The introduced architecture has been implemented and validated under conditions that are described in section 6.

2. Introduction to HAVi and OSGi

2.1. HAVi

The HAVi standard provides a home network specification for seamless interoperability between digital audio and video consumer devices. Appliances compliant with the HAVi specification implement a set of system services that allow them to interoperate over IEEE 1394 connections.
HAVi differentiates between *controller* and *target* devices. Controllers may control target devices within the network, and are classified as FAV (Full Audio Video) and IAV (Intermediate Audio Video). The former requires a JVM (Java Virtual Machine). Target devices may be BAVs (Base Audio Video) and LAVs (Legacy Audio Video). While BAVs

1 East-EAA is an ITEA Eureka international project funded by the German Bundesministerium für Forschung und Wissenschaft, that aims to standardize the future automotive middleware, by featuring a wide scale of car manufacturers and tool suppliers, http://www.east-eea.net

are controllable via uploadable bytocode or native code, the LAVs are only accessible via proprietary protocols.
The HAVi software architecture is based on the following software elements: *1394 Communication Media Manager* (CMM), *Messaging System*, *Registry*, *Event Manager*, *Stream Manager*, *Resource Manager*, *Device Control Module (DCM)*, *Functional Component Module (FCM)* and *DCM Manager*.
DCMs are specified software elements that represent devices within the home network. Their provided services are accessible via the HAVi Messaging System. DCMs are usually associated with several FCMs which present the functions of devices.
While provided services of the above mentioned software elements are mutual usable, registered HAVi applications are also able to address the discussed components.

2.2. OSGi

The *Open Service Gateway initiative* advances and promotes an service gateway for the management of multiple applications and services. The gateway is able to connect any type of local network to further service providers over the wide area network. Local networks that are attached to the service platform can be found in home-, vehicle-, mobile- and other environments.
The underlying *Framework*, integrated *Bundles* and *Services* are main components of the OSGi architecture. As the central point of the gateway the Framework supports the deployment of Service applications known as Bundles. A Bundle acts as a container for Services that may register with the Framework to be shared with other installed Bundles. These Services are on the one hand able to support locally connected networks. On the other hand, they assist the use of external Services offered via the Internet.
The specification defines some standard Services, like the mandatory Log Service, the optional HTTP Service and the Device Access. While the former supports the error logging within the Framework, the HTTP Service enables the gateway to act as a lightweight HTTP server. With the aid of the Device Access the coordination of automatic detection and attachment of connected devices is guaranteed.

3. Related Work

In this context related work primarily concerns the connection of networked devices, particularly HAVi devices, to the Internet and their remote controlling via the Web Browser.
The specified Web Proxy FCM of HAVi allows interested software elements to access the Internet [1]. It supports several Internet protocols and can be hosted by any HAVi device. Due to the fact that this FCM only implements the client side of the client/server protocols, remote accessibility is not supported. With respect to the HAVi specification the Web Proxy FCM is solely used by networked devices to access the Internet. Our approach that regards the combination of HAVi with the Service Gateway of OSGi allows a bilateral communication with the Internet and any non-HAVi local networks.
In [3] Harold et al. present a Web FCM for HAVi, which extends the existing Web Proxy FCM with Web server and TCP/UDP functionality for remote accessibility to the HAVi network. Harold et al. proposes the use of a a residential gateway on the home side that is connected to the HAVi environment. Therefore, this gateway is realised as a HAVi device with a special application that provides the HTTP server among other things. To access the HAVi environment appropriate applications are downloaded and HAVi API calls are sent via the Internet. While this solution requires a transformation of HAVi API calls to XML and a HAVi application download, the architecture that is presented in this paper supports a current representation of all connected devices with registered services for the FCMs. Therefore a download is not necessary.
Ahmed et al. [4] also deals with device controlling via Internet and Web Browser. With aid of a home management broker (HMB) which acts as an administrator, local networks are connected to the Internet. Furthermore this architecture is based on numerous broker objects that provide different services. Compared to our approach, a bilateral communication is not offered.
The specified Device Access of OSGi is also used in [5]. It presents a Device Discovery and Description Framework called 3DF to bind several home network protocols, making the devices of on network discoverable in the other. While this framework is responsible for the current representation of connected devices, the solution does not deal with the remote use of HAVi functionality.
In [6] an OSGi based infrastructure for context awareness with a residential gateway is introduced. Event-triggered define the context aware behaviour. Unlike our approach, the concrete interface between HAVi and OSGi is not discussed.
[7] describes a gateway architecture connecting the HAVi part of a home network with Internet terminals to allow the control of HAVi devices from Web Browsers. Key of this solution is the representation of HAVi devices as UPnP devices. This architecture is limited to UPnP based networks, while our combination with OSGi allows a connection to heterogeneous networks.

4. Architecture

Our approach is based on a actual representation of HAVi devices within the gateway environment. Furthermore, their functions are provided as services that may be accessed by the installed bundles. Figure 1 depicts important components for the fusion.

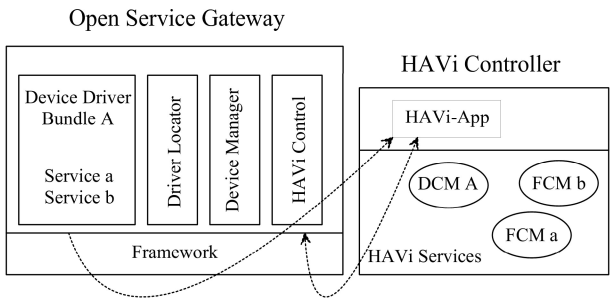


Figure 1. Architecture of combination

Each connected device within the HAVi network, more precisely its DCM, is represented as a Device Driver Bundle in the Service Gateway. Its functionalities are mapped to registered services and the communication within the HAVi environment is realised by a dedicated application. Core of the combination is the HAVi Control module. Its implemented HAVi Listener supports the notification of HAVi events within the

OSGi framework and in addition to that, it is responsible for the current representation of the Device Drivers. The specified Device Access allows the localisation of adequate Drivers. For these purpose the depicted Device Manager and Driver Locator are used.

5. Solution

In this section the outlined components above will be explained in detail. Moreover, basic interactions among discussed modules will be regarded.

5.1. Device Driver

The Device Driver Bundle represents devices of the HAVi network. As HAVi devices are accessible via their DCMs each registered DCM has an associated Device module within the OSGi environment. Core elements of this bundle are the Activator, a Driver and Device class for the realisation of the specified Device Access, and a service for each device function. Additionally, the HAVi Listener is required to notice interested events from the network area. Figure 2 presents a class diagram of a possible Device Driver Bundle. It depicts a radio device with amplifier and tuner functionality. After its installation methods of both services are provided for controlling the radio device.

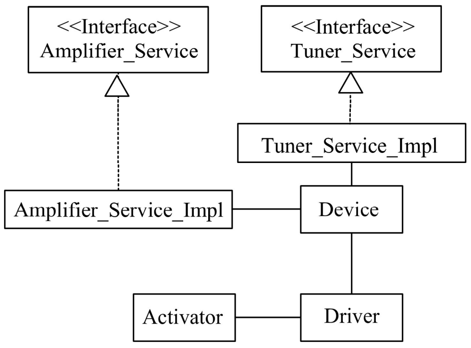


Figure 2. Device Driver class diagram

5.2. HAVi Control

With the aid of the HAVi Listener, the HAVi Control module (see Figure 3) supports the notification of HAVi events within the OSGi environment. In particular system events are of interest, e.g. newly plugged devices, whose occurrence initiates the specified mechanism of Device Access. In case of newly registered HAVi DCMs a Device Service is registered within the Framework that is only known as an abstract HAVi appliance at this time. Services may be accessible only once the adequate Device Driver has been found, the Device Driver Bundle is installed and its Services are registered. The registered Device Service initiates the Device Manager to control the Device Access mechanism.

5.3. Device Manager and Driver Locator

The specified Device Access of OSGi supports the localisation, installation and registration of Device Drivers for

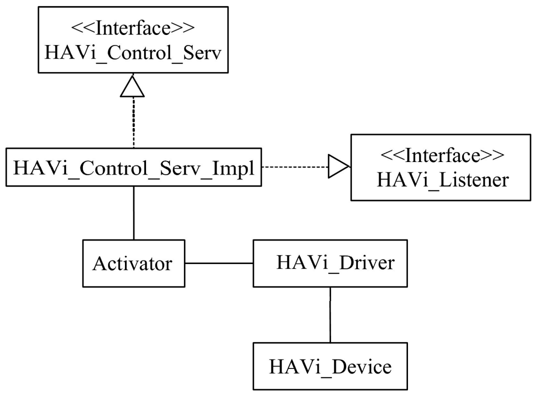


Figure 3. HAVi Control class diagram

plugged devices within a connected network. In this case it supports the localisation and installation of adequate Device Driver Bundles for HAVi devices so that for each FCM a service is registered. Core of the Device Access is the Device Manager that notices registrations of Device services. After that it locates a suitable Driver with the aid of the Driver Locator. The locator knows where to find the drivers and supports its download and installation. To select the right driver a device description is forwarded from the HAVi network, via the HAVi Control bundle to the Device Manager and Device Locator.

5.4. Plugging New Devices

The following scenario describes the activities within the OSGi environment after new devices are plugged to the HAVi network.
If new appliances are connected to the 1394 bus a *NewDevices* event is generated that is subscribed by a dedicated HAVi application. Consequently this module is noticed by the specified Event Manager when new devices are plugged to the home entertainment area. The HAVi application notices the HAVi Listener within the installed HAVi Control Bundle of OSGi. Thus, the mechanism of the specified Device Access will be initiated. The registration of a Device Service after the trigger of the above mentioned event type comes along with a notification of the Device Manager. Now this element is responsible for the attachment with a suitable Driver Service. With support of the registered Driver Locator Service and a device description the appropriate Driver will be located and the suitable Device will be instantiated. Finally, the suitable Device Driver Bundle is installed and the functionalities of the connected devices will be registered as an OSGi Service.

5.5. Remote Control

With the support of servlets [2] HAVi devices are accessible via the Internet. Due to the mapped OSGi Services to the HAVi device FCMs, functionality is accessible from all registered Bundles. Therefore, dedicated servlets allow the use of FCM Services. The following scenario, depicted in Figure 4, describes detailed actions of remote controlling connected HAVi devices.

2 small programs that run on the service gateway

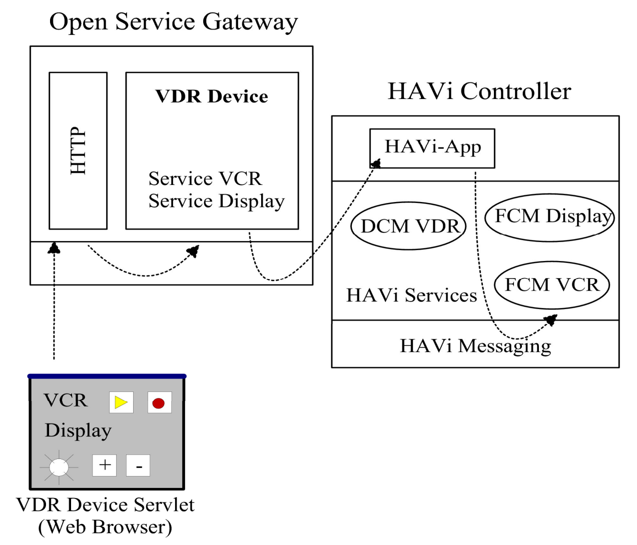


Figure 4. Remote control of CE devices

After a HAVi device, e.g. a digital VDR (Video Disk Recorder) is plugged into the HAVi network, it is represented within the OSGi environment. Moreover, device functionality is provided by registered Services. An appropriate servlet presents a Graphical User Interface for the HAVi VDR and with aid of the HTTP Service, functions of the FCMs are remotely usable within the Web Browser. In this context, the servlet contains a web page for controlling the specified VCR FCM and Display FCM that are registered software elements of the VDR device within the HAVi environment. Via the dedicated HAVi application the control commands are forwarded with the support HAVi messaging to the DCM and to the target device.

6. Implementation and Validation

While the HAVi environment is based on IAVs developed by dmn Software-Entwicklung GmbH, the Java Embedded Server [8] is used as Open Service Gateway, being compliant to the Version 2 of the OSGi specification. It requires a Java Run Time environment. In our demonstrator we use JRE 1.4 running under Linux with the kernel 2.4.20. As validation, the remote control of an integrated HAVi capable VDR is realised. With support of a dedicated Servlet, the VDR is accessible via the Internet from heterogeneous devices, e.g. PDA, PC or mobile phone. Due to the missing JVM within the VDR device, the Java Native Interface (JNI) technology is used.

7. Conclusion and Future Work

This paper introduces an approach to combine HAVi and OSGi. Within the described architecture HAVi devices are represented as Bundles within the Service Gateway and their functionality is mapped to registered Services. Due to this mapping these Services can be shared by installed Bundles and a remote controlling is possible. The communication between both standards is realised by a dedicated HAVi application. Furthermore, parts of the specified Device Access are used to guarantee a current representation of the HAVi network within the OSGi environment.

Unlike related approaches, the fusion with the Service Gateway does not only allow for the connection to the Internet, but also the interaction with non-HAVi local networks. Thus, remote control from the Internet, and also from other local networks is supported.
Primarily, the device representation Bundles should be available for all HAVi capable devices. Therefore, at least we have to implement the specified FCMs as Services, so that e.g. manufacturer of devices are able to provide the gateway representations of their devices. Furthermore, the current representation of HAVi devices within the OSGi environment could be such optimised that the appropriate Services are only registered on request. Therefore, only in case of received control commands within the Service Gateway, the Device Driver Bundle is installed. Finally, we have to integrate the specified Driver Selector to select the most suitable Bundle for a device.

References

[1] HAVi Consortium: Specification of the Home Audio/Video Interoperability (HAVi) Architecture (Version 1.1), avaiable at: http://www.havi.org

[2] Open Service Gateway Initiative: *OSGi Service Platform Specification, Release 3.* March, 2003

[3] Harold, P.: *Adding full Internet protocol functionality to HAVi.* 20th International Conference on Consumer Electronics (IEEC), Chicago, 2001

[4] Ahmed, F. and Madisetti, V. and Jiao, Y. and Dasigi, V.: *Web-Enabled Information Appliances for Broadband Residential Networks.* Yamacraw Industrial Avisory Board Worksphop, October, 2000

[5] Wils, A. and Matthijs, F. and Berbes, Y. and Holvoet, T. and De Vlamnick, K.: Device Discovery via Residantial Gateways. ICCE 2002 Digest of technical papers (Rowe, W.A., ed.), pp.96-97, 2002

[6] Zhang, D. and Wang, K. and Leman, K. and Hunag, W.: *OSGi Based Service Infrasturcture for Context Aware Connected Homes.* 1st International Conference on Smart Homes and Health Telematics (ICOST2003), September 24, 2003, Paris, France

[7] Baier, R. and Gran, C. and Scheller, A. and Stolp, R.: *Control of CE Devices through a HAVi/IP Gateway.* International Symposium on Consumer Electronics, Erfurt, August, 2002

[8] Sun Microsystems, Inc.: *The Connected Home Powered by Java Embedded Server Software*, California, USA, 2001

Part II – Presentations

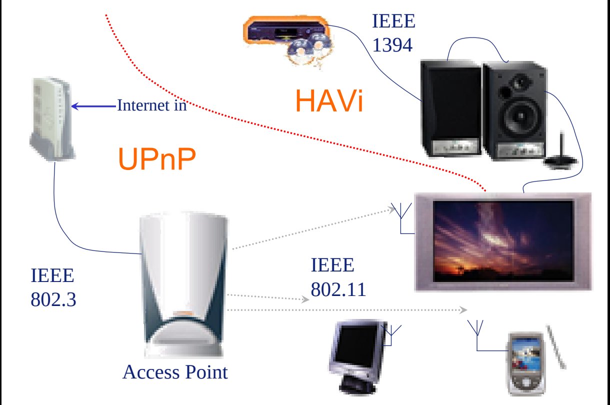
PHILIPS

The FABRIC project
Overview and Results

Peter van der Stok
29 June 2004
RTMM-ECRTS workshop

Participants consortium

PHILIPS INRIA TU/e
THOMSON csem MÄLARDALENS HÖGSKOLA
TNO UNIVERSITÄT PASSAU



FABRIC Goal

An architecture for:

1. Video streaming over heterogeneous network
2. with satisfaction of QoS requirements of streams

Base design on IEEE1516 HLA standard

Tangible results

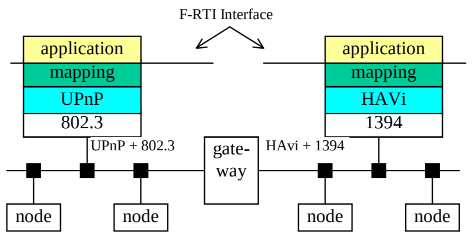
1. Application design
2. Interoperability over middleware stds
3. Streaming QoS framework
4. Mock-up based on FABRIC-RTI

Extra-functional properties

1. Distribution
2. Performance (overhead, delays)
3. Footprint (demonstrator numbers)
4. Security (no extra barriers identified)
5. Standardization/ Abstraction
6. Legacy awareness

PHILIPS

Publish/Subscribe

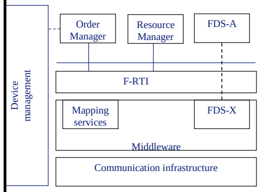


FABRIC RTI based on High Level Architecture, (IEEE 1516)

Research 7

PHILIPS

Architecture elements



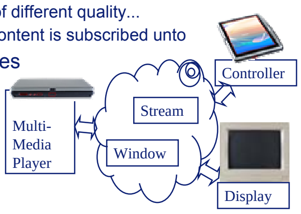
- FDS-A and FDS-X
  - implement the directory (discovery) services
- Resource & order managers
  - global & local part of the FABRIC resource management using the Matrix
    - in a federation, only a single resource manager is active
- Device management
  - (OS-related) access to local hardware
- F-RTI
  - the FABRIC RTI
- Mapping services
  - middleware services that map the F-RTI onto the middleware
- FOM (not drawn)
  - data exchange model

Research 8

PHILIPS

Application adaptation

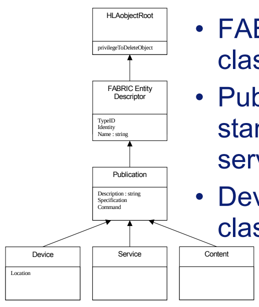
FABRIC: content oriented
- public “content space” (data space): everything is just content
  - even control information...
  - even stream instances of different quality...
  - content is published & content is subscribed unto
- diminishes dependencies



Research 9

PHILIPS

FOM



- FABRIC Entity Descriptor class
- Publication class for standardized devices, services and content
- Device, Service and Content classes for filtering

Research 10

PHILIPS

MATRIX element

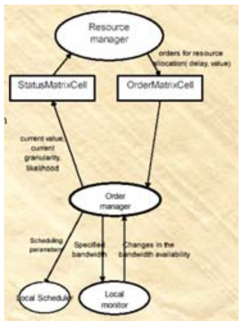
order manager on a device (locally)
- estimates resource availability
- maps these values to the appropriate QoS level

resource manager (globally)
- collects information about available resources from the Status matrix elements
- puts resource reservation orders in Order Matrix

order managers (locally)
- get these global orders
- make local resource reservation

local scheduler
- gets scheduling specification (bandwidth, delay)

local monitor (on next node in the stream path)
- gets information about specified performance
- compares it with actual received performance
- deviation performance is then sent to the order manager


Resource manager
orders for resource allocation(delay, value)
StatusMatrixCell
OrderMatrixCell
current value, current granularity, likelihood
Order manager
Scheduling parameters
Specified bandwidth
Changes in the bandwidth availability
Local Scheduler
Local monitor

Research 11

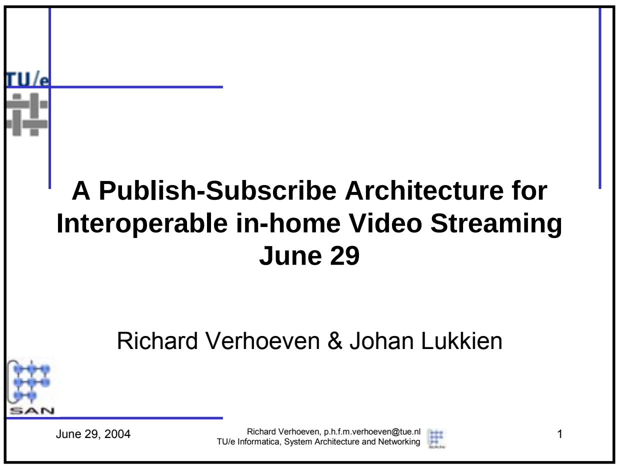
PHILIPS

Personal view

Points of attention
- additional standardization needs
- translation complexity between service descriptions

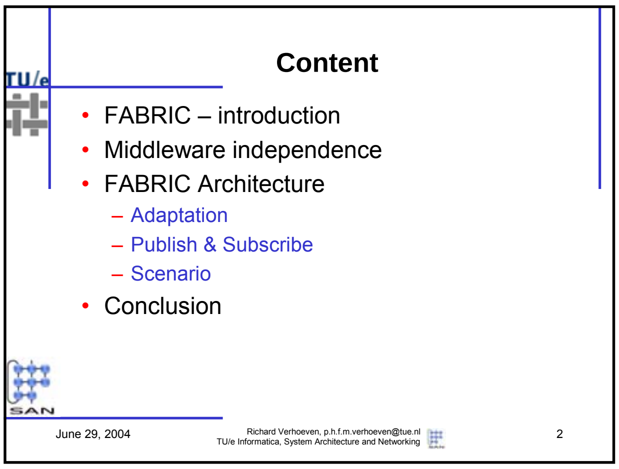
Pleasant surprises
- abstraction levels
- elegant QoS management
- distribution for free
- handling differences in service semantics

Research 12

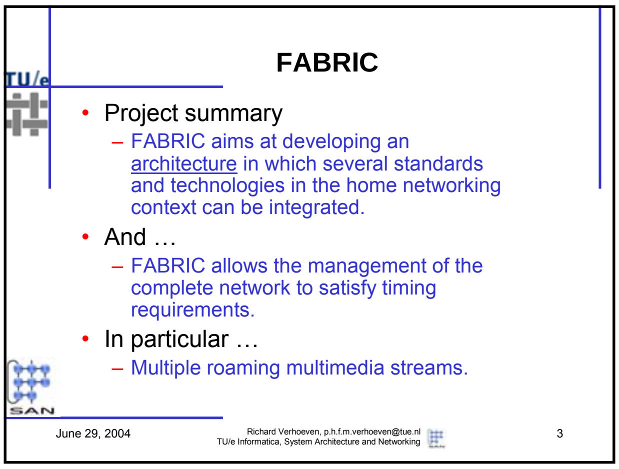


A Publish-Subscribe Architecture for Interoperable in-home Video Streaming June 29

Richard Verhoeven & Johan Lukkien

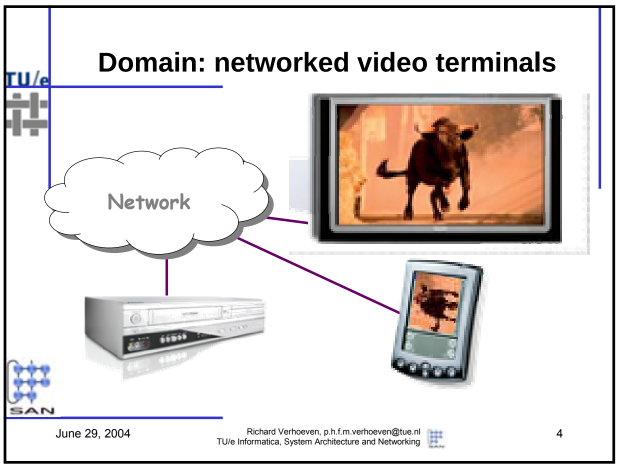


Content

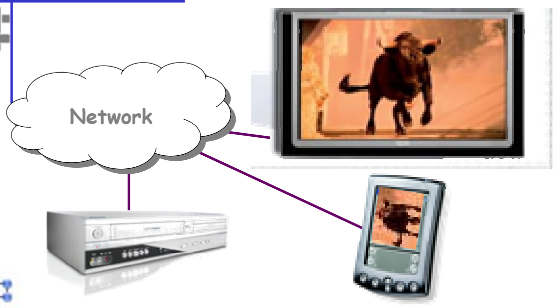


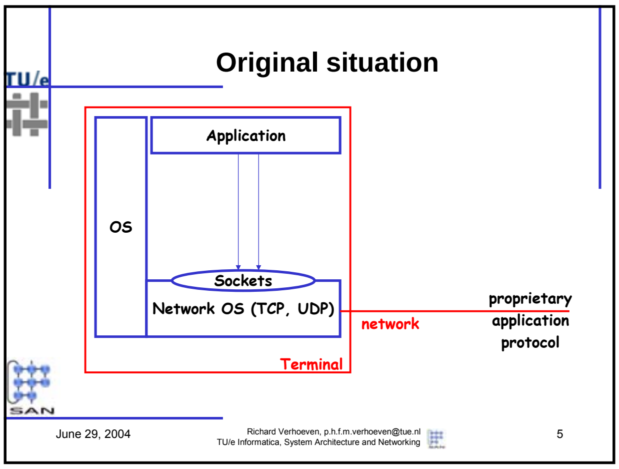
FABRIC

- Project summary
  - FABRIC aims at developing an architecture in which several standards and technologies in the home networking context can be integrated.
- And …
  - FABRIC allows the management of the complete network to satisfy timing requirements.
- In particular …
  - Multiple roaming multimedia streams.

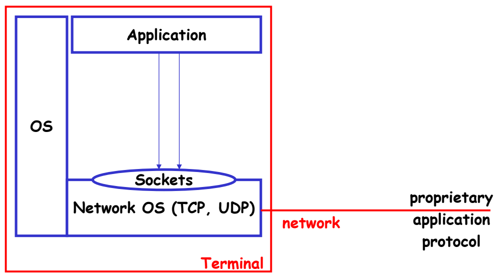


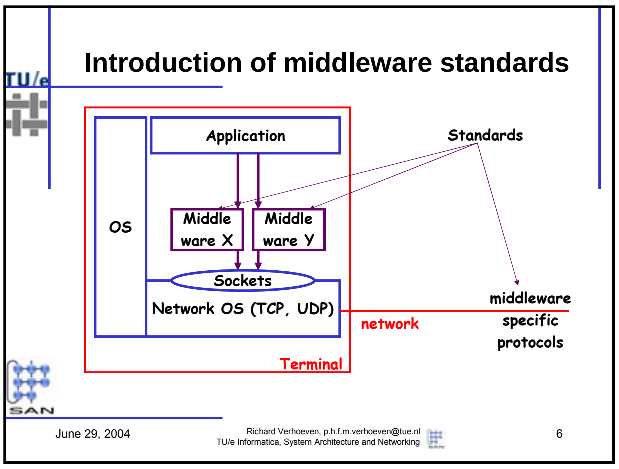
Domain: networked video terminals



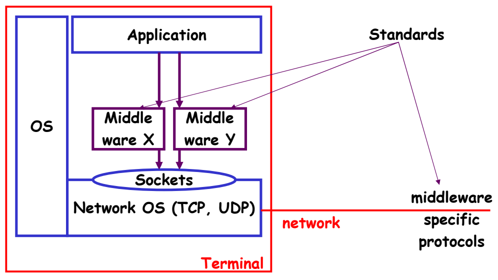


Original situation





Introduction of middleware standards



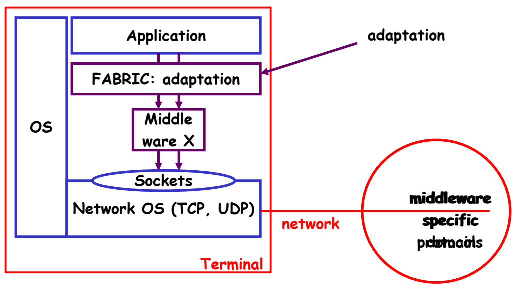
Areas of heterogeneity

- Streams
  - MPEG-2
  - MPEG-4
- Middleware
  - UPnP
  - HAVi
- Network
  - IEEE 802.3 / Ethernet
  - IEEE 802.11 / Wireless LAN
  - IEEE 1394 / Firewire

Heterogeneity and applications

- User is only interested in movies
  - Independent of network technology
  - Independent of encoding
- Applications should be portable
  - Independent of middleware
- Applications should be interoperable
  - Over different technologies
  - Including extra-functional properties
- Application development should not be aware of it!

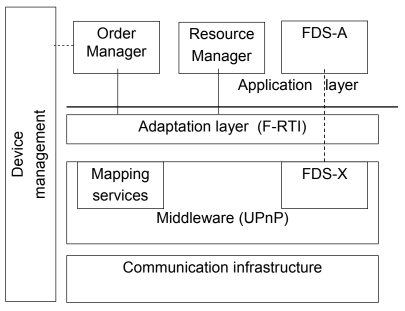
FABRIC: middleware independence



Adaptation Requirements

- Loose coupling
  - Devices may come and go
    - No sender / receiver pairing
  - Different capabilities
- Extendable
  - New functionality, new device types
- Generalization of different middleware
  - Standardized API
  - No new heterogeneity
- Low overhead
  - Resources, delay, complexity
    - Only control information through the stack

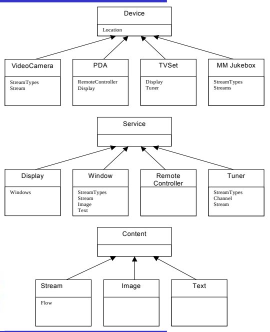
FABRIC Architecture



Anonymous Publish & Subscribe

- Information published on network
  - Anonymous data objects at API
  - Applications subscribe to information (content)
- Data object model based on video domain
  - Include common features (e.g. title, resolution)
  - Ignore exotic features (e.g. actors, goofs)
- Anonymous interactions for notification
  - Play, stop, pause, …
- Data distribution management to reduce traffic
- FABRIC approach based on HLA standard
  - Adaptation layer is called RTI – runtime infrastructure
  - IEEE 1516.1

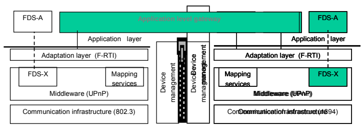
Example object model



- Objects for devices, services and content
- Attributes of Window service indicate content to be shown
- Subscribe to attribute of Stream content object starts the flow of that stream
- Operations on streams are based on interactions

Application level functionality

- Directory services
- Gateway functionality
- Resource management



Scenario

- Mobile device connects to the in-home network
- It discovers (display) services and content
- User combines a display service and content
- Display service is instructed to use content
- Resource manager can deny or approve

TV Set

- Provide display services
- Report status
- Receive video content

MM Player

- Provide recording and playback services
- Report status
- Send and receive video content

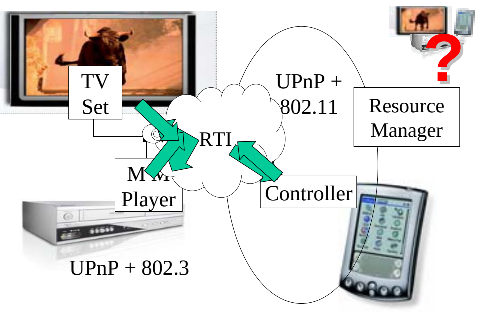
Controller

- Provides user interface
  - Present services and content
- Present status
- Create and control video streams
  - React to advice from resource manag

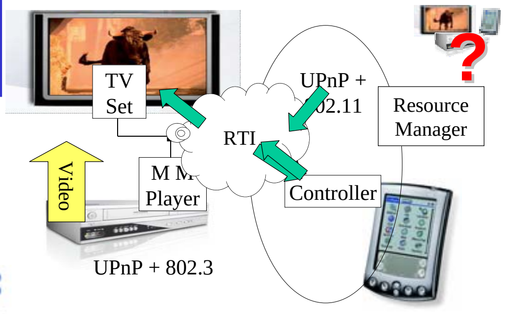
Resource manager

- Monitors resource usage
- Instructs applications to adjust video streams
  - Close one stream, open another
- Location unknown
  - One active, available everywhere

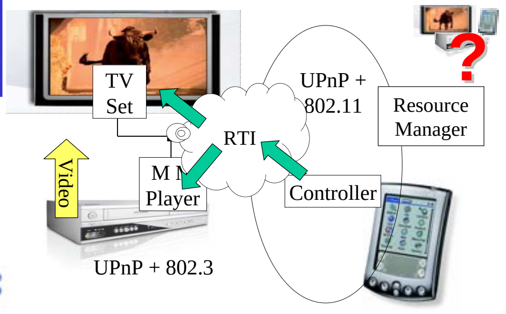
Connect and discover



Display high quality content & abort

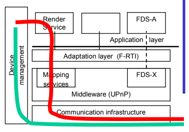


Display low quality content & start

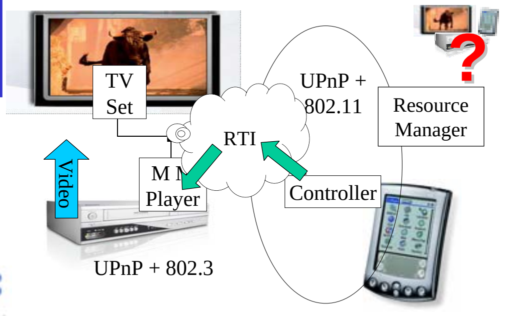


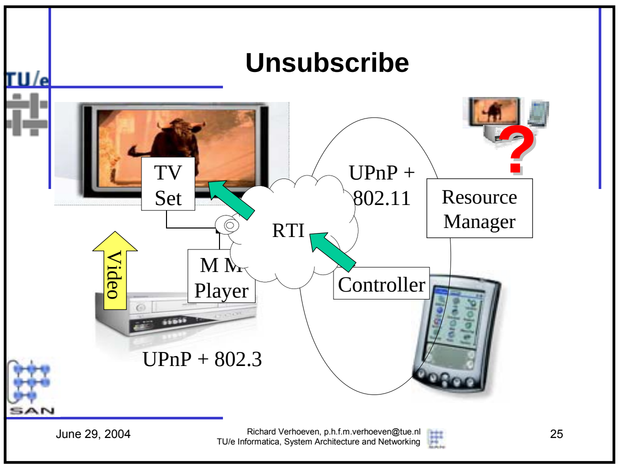
Streaming the video

- Conceptually
  - Over the F-RTI
  - Resource reservation
- Actually
  - Directly from storage to network or from network to decoder
- Mapping
  - Reuse network technology as good as possible

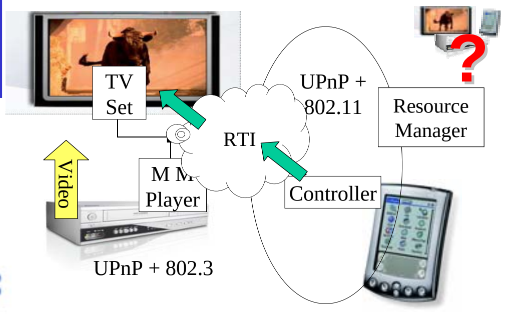


Stop

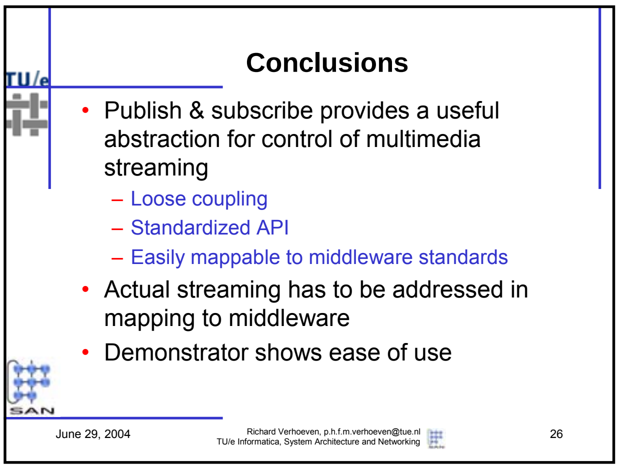




Unsubscribe



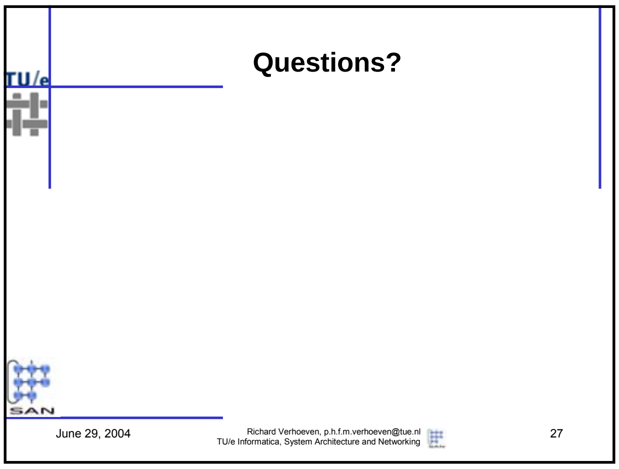
June 29, 2004 Richard Verhoeven, p.h.f.m.verhoeven@tue.nl TU/e Informatica, System Architecture and Networking 25



Conclusions

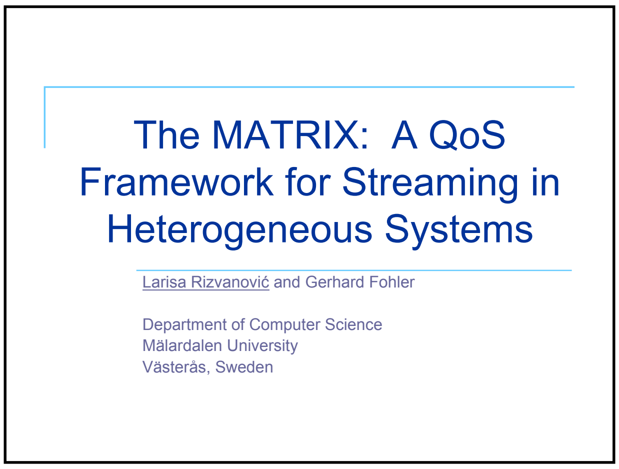
- Publish & subscribe provides a useful abstraction for control of multimedia streaming
  - Loose coupling
  - Standardized API
  - Easily mappable to middleware standards
- Actual streaming has to be addressed in mapping to middleware
- Demonstrator shows ease of use

June 29, 2004 Richard Verhoeven, p.h.f.m.verhoeven@tue.nl TU/e Informatica, System Architecture and Networking 26



Questions?

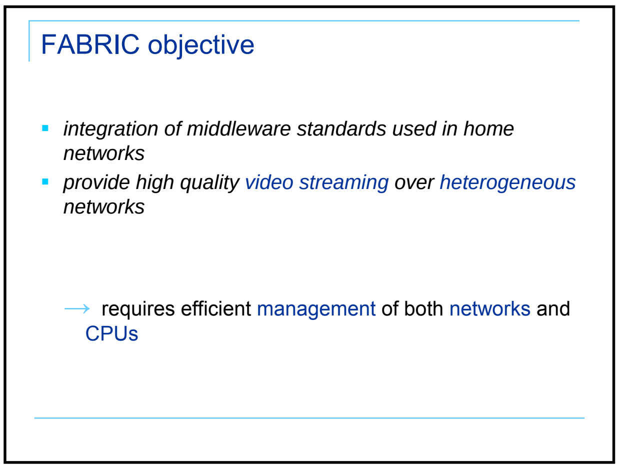
June 29, 2004 Richard Verhoeven, p.h.f.m.verhoeven@tue.nl TU/e Informatica, System Architecture and Networking 27



The MATRIX: A QoS Framework for Streaming in Heterogeneous Systems

Larisa Rizvanović and Gerhard Fohler

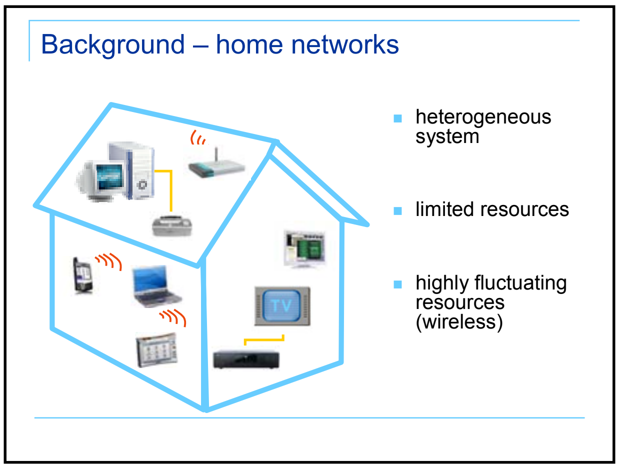
Department of Computer Science
Mälardalen University
Västerås, Sweden



FABRIC objective

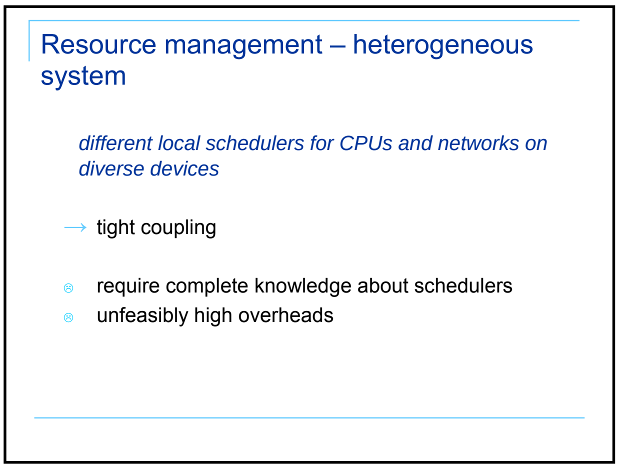
- *integration of middleware standards used in home networks*
- *provide high quality video streaming over heterogeneous networks*

→ requires efficient management of both networks and CPUs



Background – home networks

- heterogeneous system
- limited resources
- highly fluctuating resources (wireless)

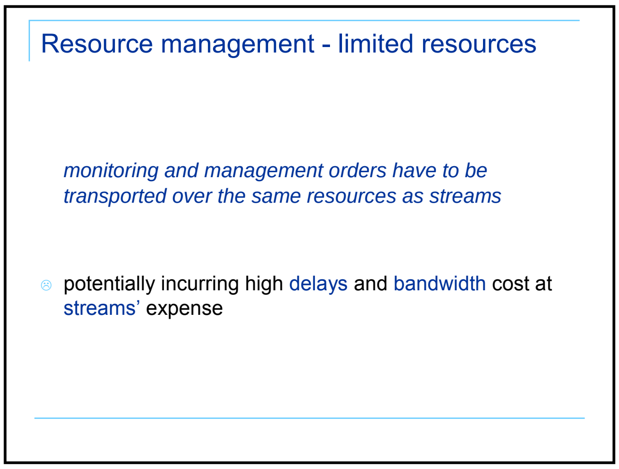


Resource management – heterogeneous system

different local schedulers for CPUs and networks on diverse devices

→ tight coupling

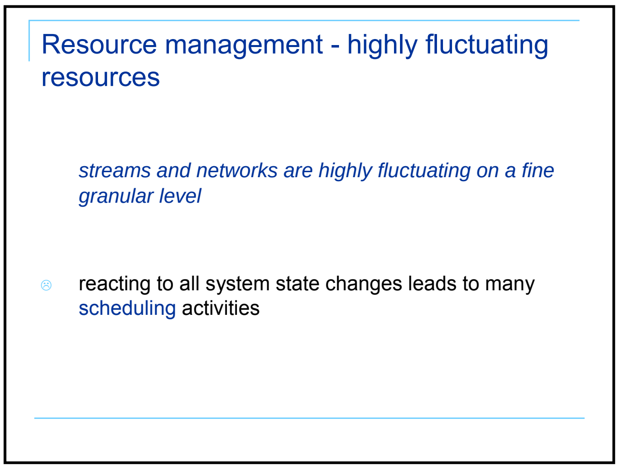
☹ require complete knowledge about schedulers
☹ unfeasibly high overheads



Resource management - limited resources

monitoring and management orders have to be transported over the same resources as streams

☹ potentially incurring high delays and bandwidth cost at streams' expense



Resource management - highly fluctuating resources

streams and networks are highly fluctuating on a fine granular level

☹ reacting to all system state changes leads to many scheduling activities

Key issues for resource management

- efficient representation of the fluctuating system state
- resource allocation decisions
- dissemination of orders

System state representation

desirable:

- fine granularity
- accurate
- fresh

+

☹ high NW and CPU overhead

System state representation (cont…)

tradeoffs between

- accuracy of system state information
- efforts to transport and process

we want to use

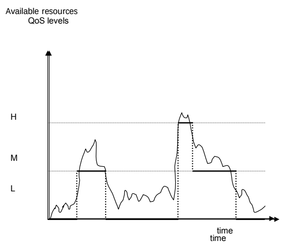
⟶ minimum relevant information
as needed for resource management

Our approach – MATRIX

- system wide abstraction of device states
  - representation of the system state for resource management
  - interface to decouple device scheduling and system resource allocation

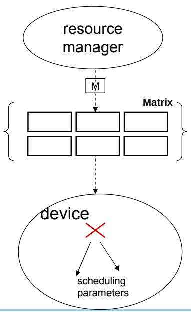
MATRIX - granularity

- spatial granularity – reduce precision of values
  - small, discrete range
  - few QoS
- temporal granularity – update intervals
  - device provide information
  - filter (too) high fluctuations
  - not more often than resource management can handle



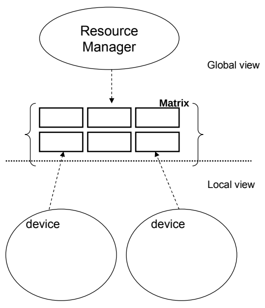
MATRIX - dissemination of resource management's decisions

- order - high level abstractions of limited value ranges
- devices translate orders into local scheduling parameters



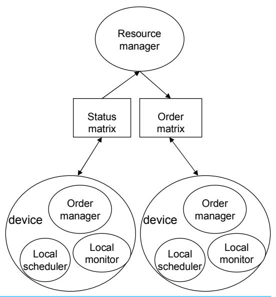
MATRIX – loose coupling

- data processing at appropriate levels
  - resource management (global, high level)
  - device handling (local, low level)
- loose coupling between
  - system resource management
  - device internal details, specific scheduling etc



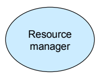
Architectural design aspects

- Resource manager
- Status matrix
- Order matrix
- Order manager
- Local scheduler
- Local monitor



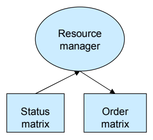
Architectural design aspects (cont …)

- Resource manager
  - responsible for scheduling and reserving of resources in the system
    - determine available resources
    - make adjustments of streams and resources



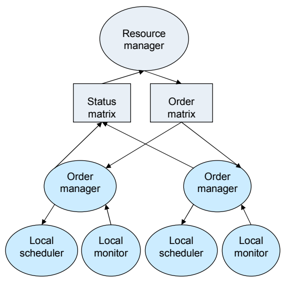
Architectural design aspects (cont…)

- Status matrix
  - offers information about available quality of resources
  - predicts how likely the quality will be available in the near future
- Order matrix
  - offers means to reserve amounts of resources (delay, value)



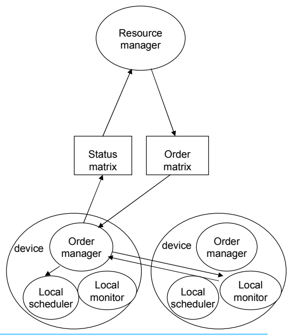
Architectural design aspects (cont…)

- Order manager
  - determines the available resources at the devices
  - transform various kind of traffic specification into a few QoS levels
  - maps global resource reservation constraints to the concrete scheduling specification
- Local scheduler
  - scheduling of device’s local CPU resource or outgoing network packets
- Local monitor
  - monitors available network bandwidth on the receiver node



One operational scenario

- order manager estimates resource availability locally (QoS levels)
- resource manager collects system state information from the Status matrix
- resource manager puts directions for resource reservation in the Order matrix
- order manager maps global constraints to local resource reservation
- local scheduler gets scheduling specification (bandwidth, delay)
- local monitor gets information about specified performance
- local monitor compares specified with actual received performance



HLA (High Level Architecture)

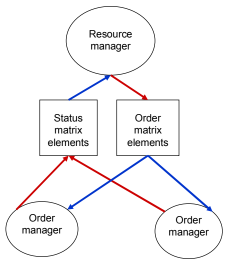
- interoperability standard based on an anonymous publish/subscribe mechanism
- subscribe – federates express their interests in receiving data of specific object class
- publish - federate intends to create instances of the class and update attributes of those instances

MATRIX and HLA

- Status matrix
  - information produced by various Order managers
  - collection of objects
- Order manager
  - information produced by Resource manager
  - collection of objects
- Resource and Order manager
  - HLA federates

MATRIX and HLA (cont…)

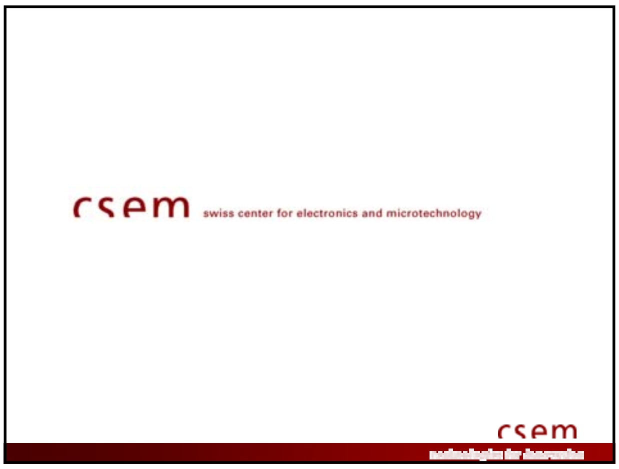
- Resource manager
  - subscribes to Status matrix elements
  - publishes Order matrix elements
- Order manger
  - publishes Status matrix elements
  - subscribes to Order matrix elements

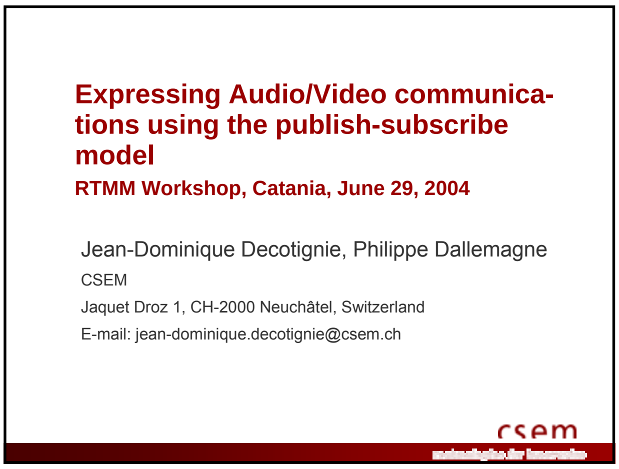


Conclusions

MATRIX – global abstraction of device states

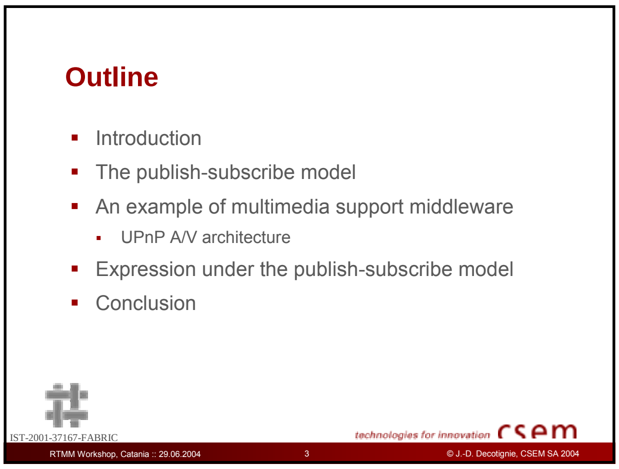
- information prepared
  - ☺ reduced system state
- device report only changes relevant for resource management
  - ☺ reduced NW and CPU overhead
- data handling at appropriate level
  - ☺ loose coupling


csem swiss center for electronics and microtechnology

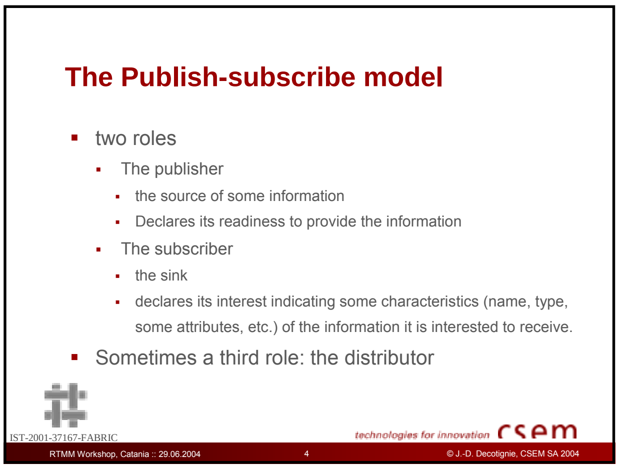

Expressing Audio/Video communica-tions using the publish-subscribe model

RTMM Workshop, Catania, June 29, 2004

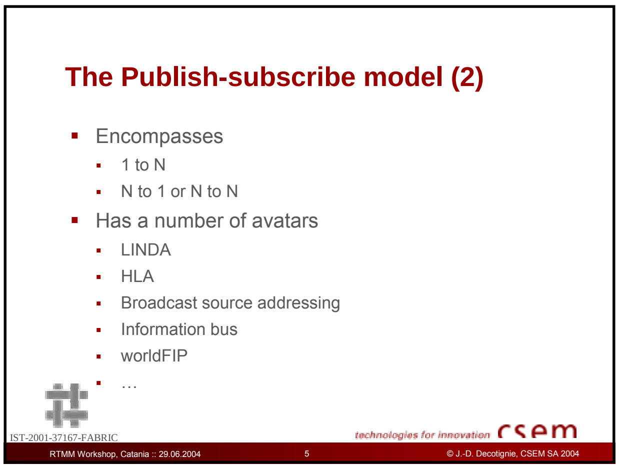
Jean-Dominique Decotignie, Philippe Dallemagne
CSEM
Jaquet Droz 1, CH-2000 Neuchâtel, Switzerland
E-mail: jean-dominique.decotignie@csem.ch


Outline

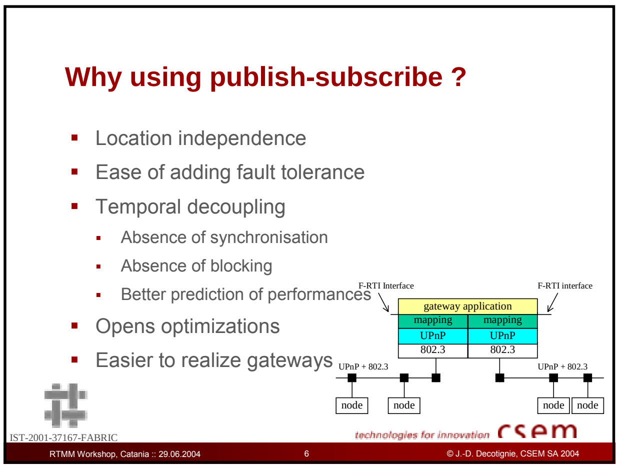
- Introduction
- The publish-subscribe model
- An example of multimedia support middleware
  - UPnP A/V architecture
- Expression under the publish-subscribe model
- Conclusion


The Publish-subscribe model

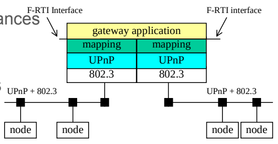
- two roles
  - The publisher
    - the source of some information
    - Declares its readiness to provide the information
  - The subscriber
    - the sink
    - declares its interest indicating some characteristics (name, type, some attributes, etc.) of the information it is interested to receive.
- Sometimes a third role: the distributor

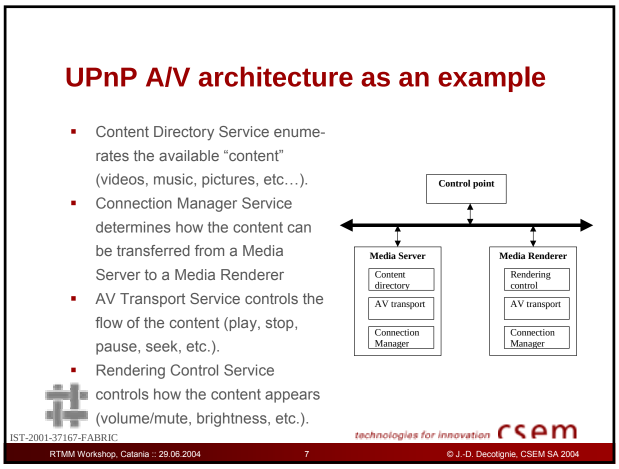

The Publish-subscribe model (2)

- Encompasses
  - 1 to N
  - N to 1 or N to N
- Has a number of avatars
  - LINDA
  - HLA
  - Broadcast source addressing
  - Information bus
  - worldFIP
  - …


Why using publish-subscribe ?

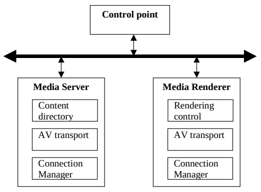
- Location independence
- Ease of adding fault tolerance
- Temporal decoupling
  - Absence of synchronisation
  - Absence of blocking
  - Better prediction of performances
- Opens optimizations
- Easier to realize gateways

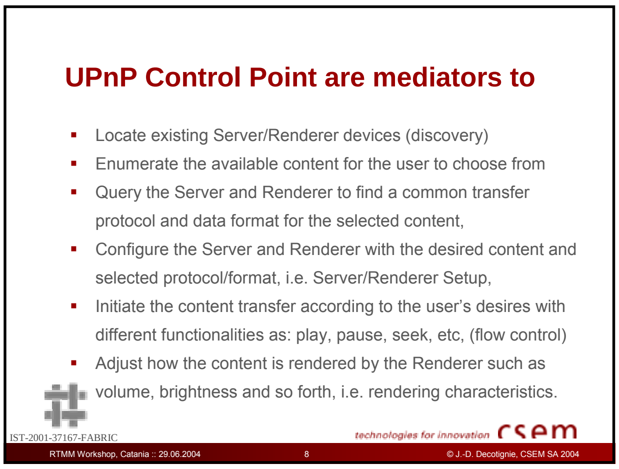




UPnP A/V architecture as an example

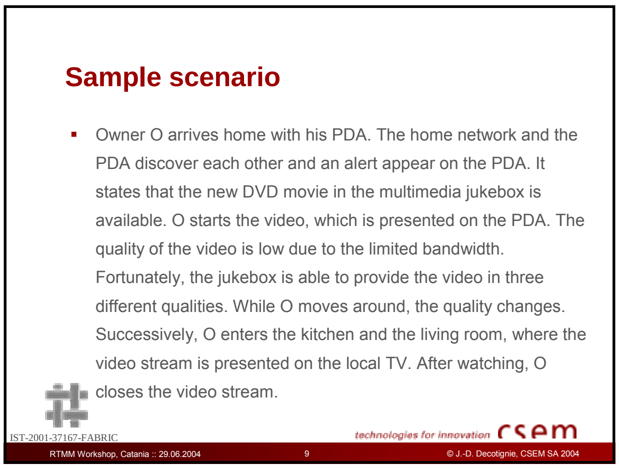
- Content Directory Service enume-rates the available “content” (videos, music, pictures, etc…).
- Connection Manager Service determines how the content can be transferred from a Media Server to a Media Renderer
- AV Transport Service controls the flow of the content (play, stop, pause, seek, etc.).
- Rendering Control Service controls how the content appears (volume/mute, brightness, etc.).





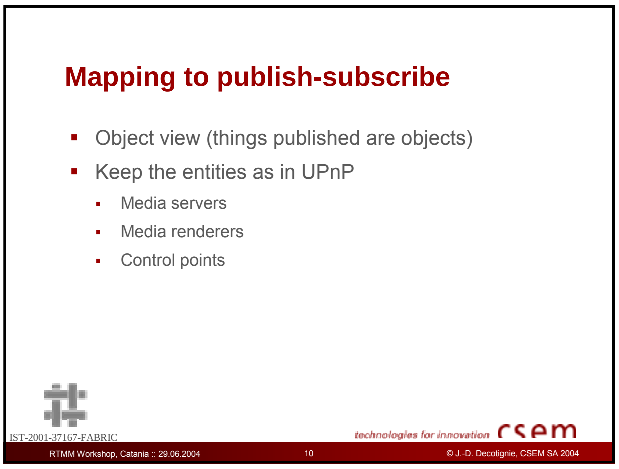
UPnP Control Point are mediators to

- Locate existing Server/Renderer devices (discovery)
- Enumerate the available content for the user to choose from
- Query the Server and Renderer to find a common transfer protocol and data format for the selected content,
- Configure the Server and Renderer with the desired content and selected protocol/format, i.e. Server/Renderer Setup,
- Initiate the content transfer according to the user’s desires with different functionalities as: play, pause, seek, etc, (flow control)
- Adjust how the content is rendered by the Renderer such as volume, brightness and so forth, i.e. rendering characteristics.



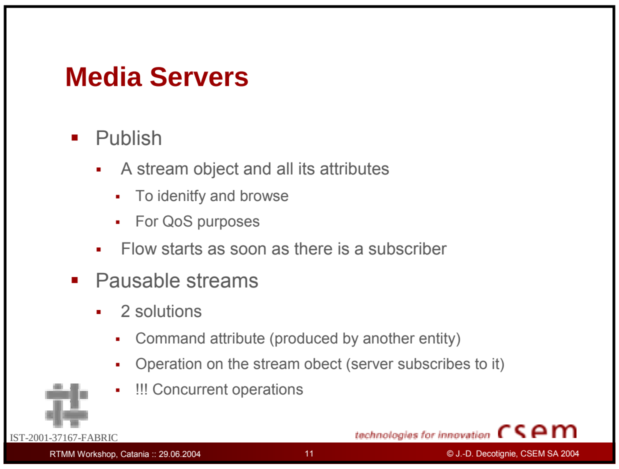
Sample scenario

- Owner O arrives home with his PDA. The home network and the PDA discover each other and an alert appear on the PDA. It states that the new DVD movie in the multimedia jukebox is available. O starts the video, which is presented on the PDA. The quality of the video is low due to the limited bandwidth. Fortunately, the jukebox is able to provide the video in three different qualities. While O moves around, the quality changes. Successively, O enters the kitchen and the living room, where the video stream is presented on the local TV. After watching, O closes the video stream.



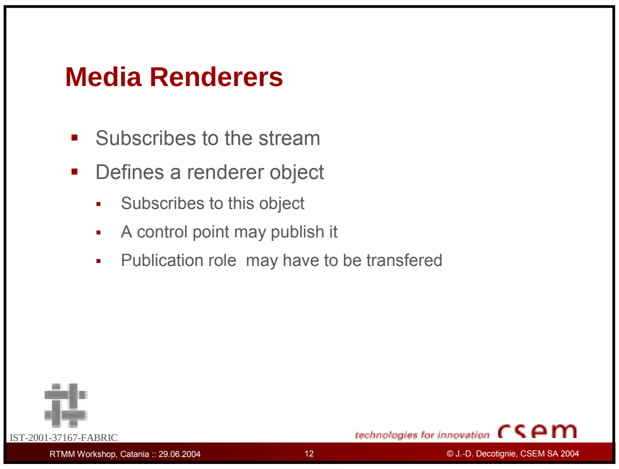
Mapping to publish-subscribe

- Object view (things published are objects)
- Keep the entities as in UPnP
  - Media servers
  - Media renderers
  - Control points



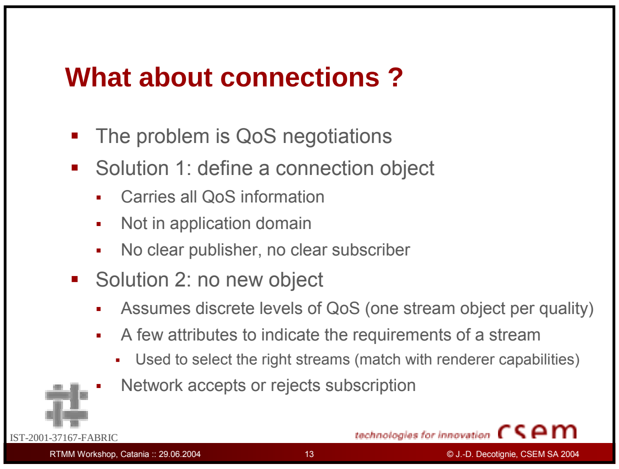
Media Servers

- Publish
  - A stream object and all its attributes
    - To idenitfy and browse
    - For QoS purposes
  - Flow starts as soon as there is a subscriber
- Pausable streams
  - 2 solutions
    - Command attribute (produced by another entity)
    - Operation on the stream obect (server subscribes to it)
    - !!! Concurrent operations



Media Renderers

- Subscribes to the stream
- Defines a renderer object
  - Subscribes to this object
  - A control point may publish it
  - Publication role may have to be transfered

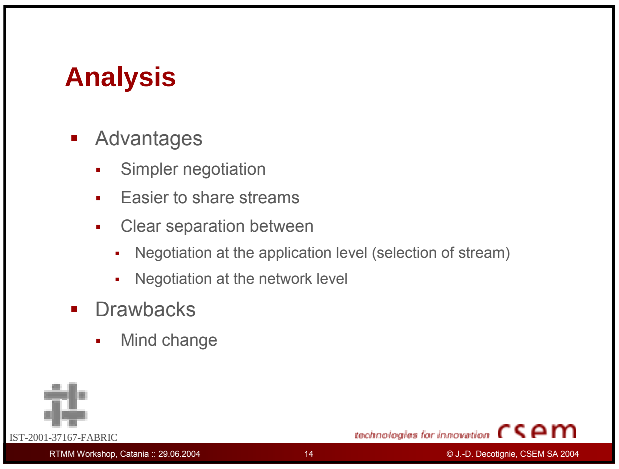


What about connections ?

- The problem is QoS negotiations
- Solution 1: define a connection object
  - Carries all QoS information
  - Not in application domain
  - No clear publisher, no clear subscriber
- Solution 2: no new object
  - Assumes discrete levels of QoS (one stream object per quality)
  - A few attributes to indicate the requirements of a stream
    - Used to select the right streams (match with renderer capabilities)
  - Network accepts or rejects subscription

IST-2001-37167-FABRIC technologies for innovation csem

RTMM Workshop, Catania :: 29.06.2004 13 © J.-D. Decotignie, CSEM SA 2004

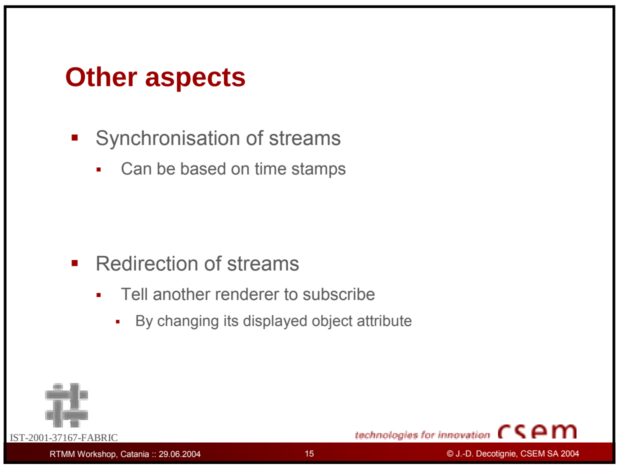


Analysis

- Advantages
  - Simpler negotiation
  - Easier to share streams
  - Clear separation between
    - Negotiation at the application level (selection of stream)
    - Negotiation at the network level
- Drawbacks
  - Mind change

IST-2001-37167-FABRIC technologies for innovation csem

RTMM Workshop, Catania :: 29.06.2004 14 © J.-D. Decotignie, CSEM SA 2004

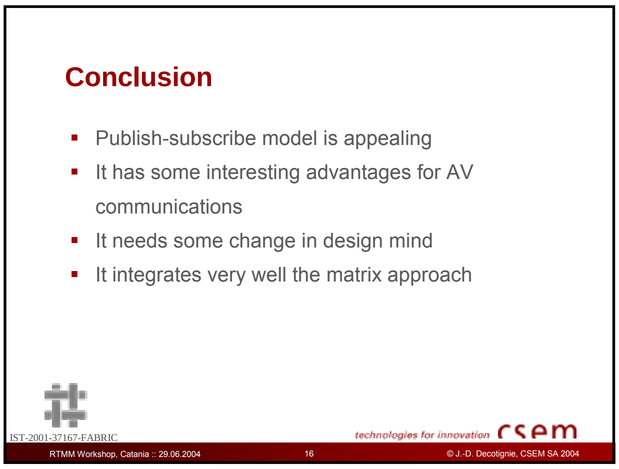


Other aspects

- Synchronisation of streams
  - Can be based on time stamps
- Redirection of streams
  - Tell another renderer to subscribe
    - By changing its displayed object attribute

IST-2001-37167-FABRIC technologies for innovation csem

RTMM Workshop, Catania :: 29.06.2004 15 © J.-D. Decotignie, CSEM SA 2004



Conclusion

- Publish-subscribe model is appealing
- It has some interesting advantages for AV communications
- It needs some change in design mind
- It integrates very well the matrix approach

IST-2001-37167-FABRIC technologies for innovation csem

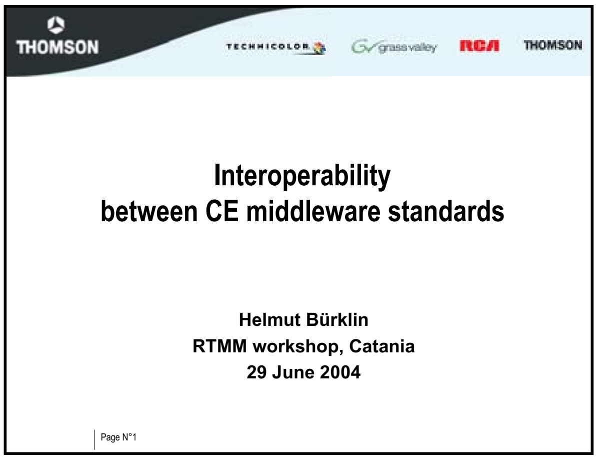
RTMM Workshop, Catania :: 29.06.2004 16 © J.-D. Decotignie, CSEM SA 2004

csem

Thank you for your attention.

csem

technologies for innovation

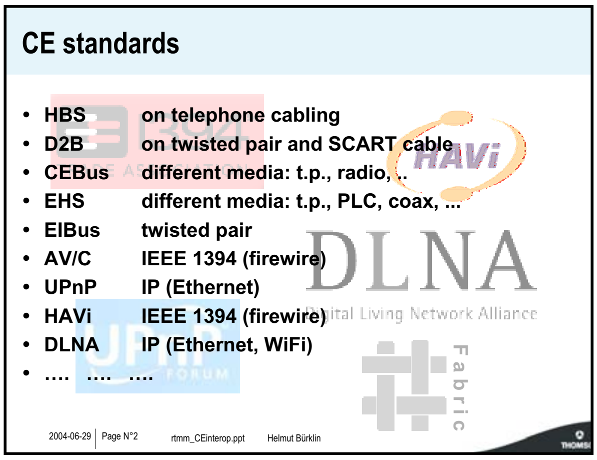


THOMSON — TECHNICOLOR — grass valley — RCA — THOMSON

Interoperability between CE middleware standards

Helmut Bürklin
RTMM workshop, Catania
29 June 2004

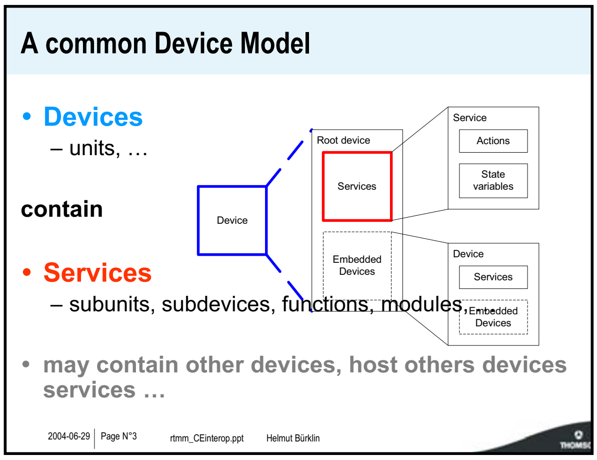
Page N°1



CE standards

• HBS	on telephone cabling
• D2B	on twisted pair and SCART cable
• CEBus	different media: t.p., radio, ..
• EHS	different media: t.p., PLC, coax, ...
• EIBus	twisted pair
• AV/C	IEEE 1394 (firewire)
• UPnP	IP (Ethernet)
• HAVi	IEEE 1394 (firewire)
• DLNA	IP (Ethernet, WiFi)
• ….	…. ….

2004-06-29 | Page N°2 rtmm_CEinterop.ppt Helmut Bürklin

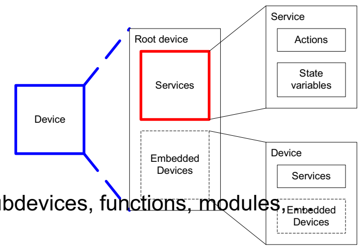


A common Device Model

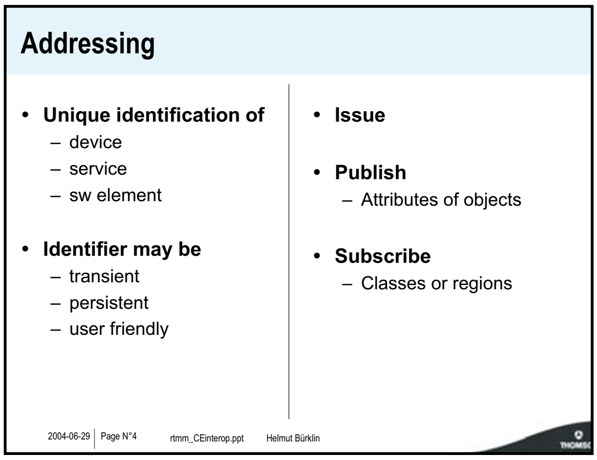
- **Devices**
  - units, …

contain

- **Services**
  - subunits, subdevices, functions, modules, …
- **may contain other devices, host others devices services …**



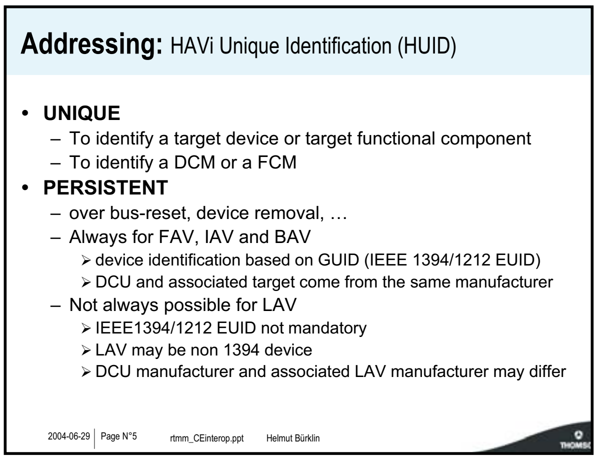
2004-06-29 | Page N°3 rtmm_CEinterop.ppt Helmut Bürklin



Addressing

- **Unique identification of**
  - device
  - service
  - sw element
- **Identifier may be**
  - transient
  - persistent
  - user friendly
- **Issue**
- **Publish**
  - Attributes of objects
- **Subscribe**
  - Classes or regions

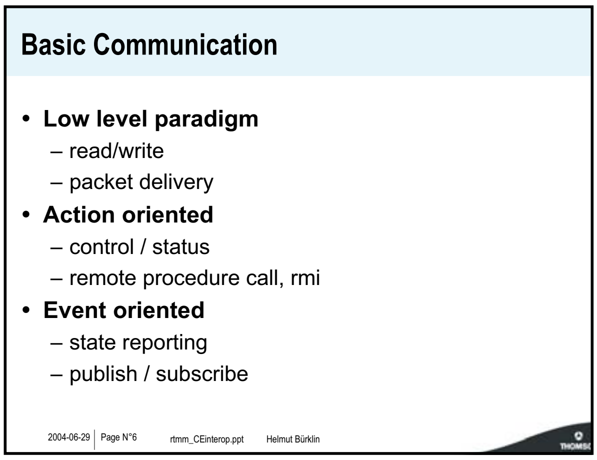
2004-06-29 | Page N°4 rtmm_CEinterop.ppt Helmut Bürklin



Addressing: HAVi Unique Identification (HUID)

- **UNIQUE**
  - To identify a target device or target functional component
  - To identify a DCM or a FCM
- **PERSISTENT**
  - over bus-reset, device removal, …
  - Always for FAV, IAV and BAV
    - ➢ device identification based on GUID (IEEE 1394/1212 EUID)
    - ➢ DCU and associated target come from the same manufacturer
  - Not always possible for LAV
    - ➢ IEEE1394/1212 EUID not mandatory
    - ➢ LAV may be non 1394 device
    - ➢ DCU manufacturer and associated LAV manufacturer may differ

2004-06-29 | Page N°5 rtmm_CEinterop.ppt Helmut Bürklin

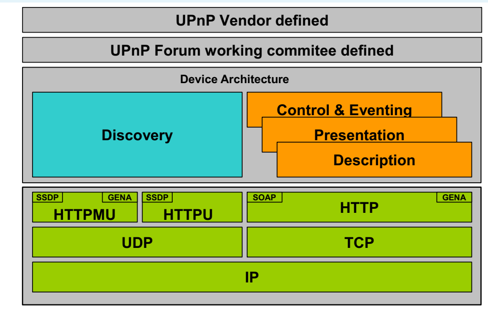


Basic Communication

- **Low level paradigm**
  - read/write
  - packet delivery
- **Action oriented**
  - control / status
  - remote procedure call, rmi
- **Event oriented**
  - state reporting
  - publish / subscribe

2004-06-29 | Page N°6 rtmm_CEinterop.ppt Helmut Bürklin

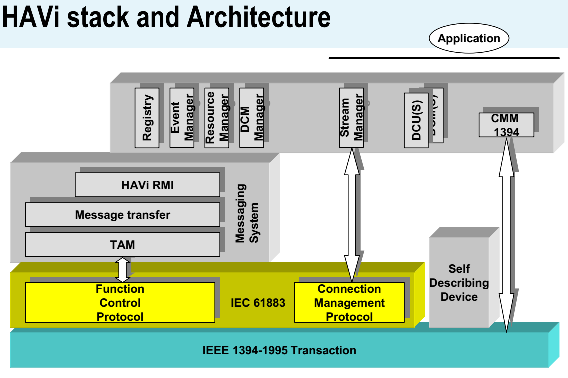
UPnP Protocol Stack



SSDP: IETF Draft Simple Service Description Protocol
GENA: IETF Draft General Event Notification Architecture
SOAP: IETF Draft Simple Object Access Protocol

2004-06-29 | Page N°7 rtmm_CEinterop.ppt Helmut Bürklin

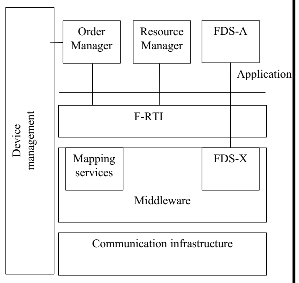
HAVi stack and Architecture



2004-06-29 | Page N°8 rtmm_CEinterop.ppt Helmut Bürklin

Fabric Architecture

- **Publish / Subscribe**
- **Standardised RTI Interface**
  - Middleware independent
- **Run Time Infrastructure**
  - Dependent on choosen middleware



2004-06-29 | Page N°9 rtmm_CEinterop.ppt Helmut Bürklin

Discovery

- **Need to discover**

– devices	(user sees)
– device capabilities	(user uses services)

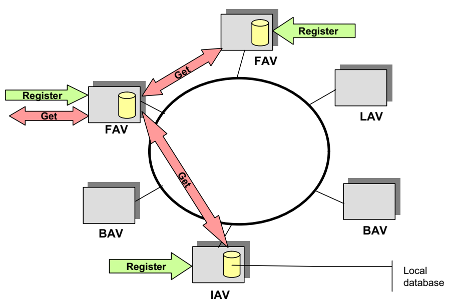
- **Registry**

– local	(possibly huge)
– central	(single point of failure)
– mixed	(synchronisation)

- **Service description**
  - fixed templates
  - self describing (XML)

2004-06-29 | Page N°10 rtmm_CEinterop.ppt Helmut Bürklin

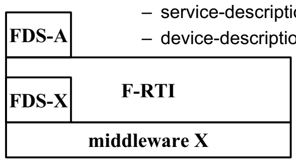
Discovery : HAVi REGISTRY distributed data base



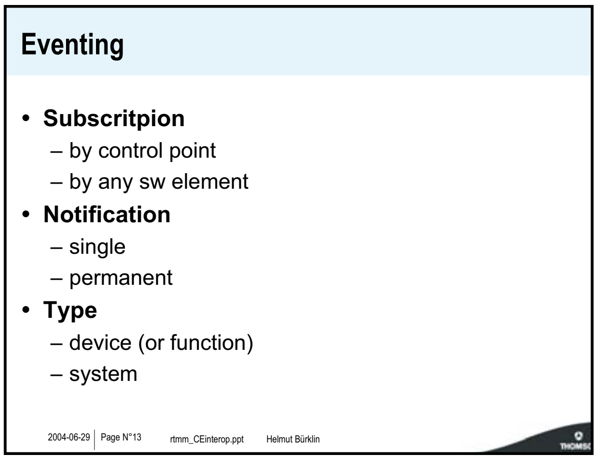
2004-06-29 | Page N°11 rtmm_CEinterop.ppt Helmut Bürklin

Discovery : Fabric Directory Service

- **The Fabric Directory Service has two parts**
  - middleware dependent – FDS-X
  - middleware independent – FDS-A
- **FDS-A of hosting device publishes**
  - service-descriptions
  - device-description



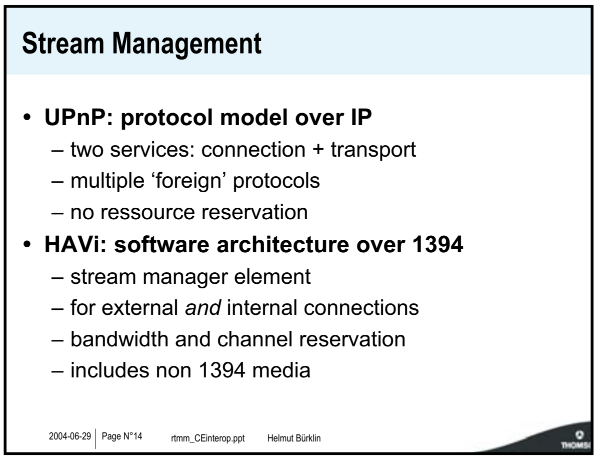
2004-06-29 | Page N°12 rtmm_CEinterop.ppt Helmut Bürklin



Eventing

- **Subscritpion**
  - by control point
  - by any sw element
- **Notification**
  - single
  - permanent
- **Type**
  - device (or function)
  - system

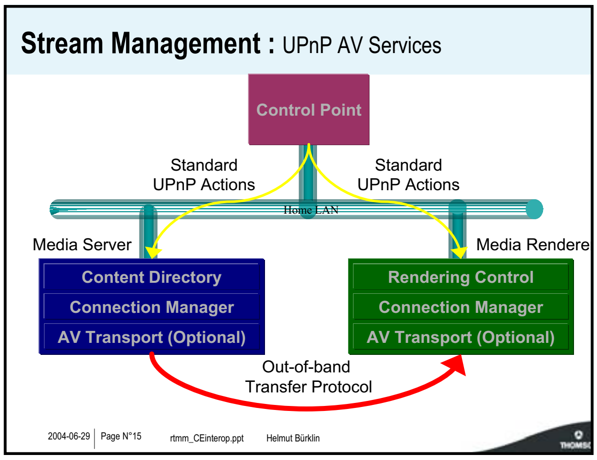
2004-06-29 | Page N°13 rtmm_CEinterop.ppt Helmut Bürklin



Stream Management

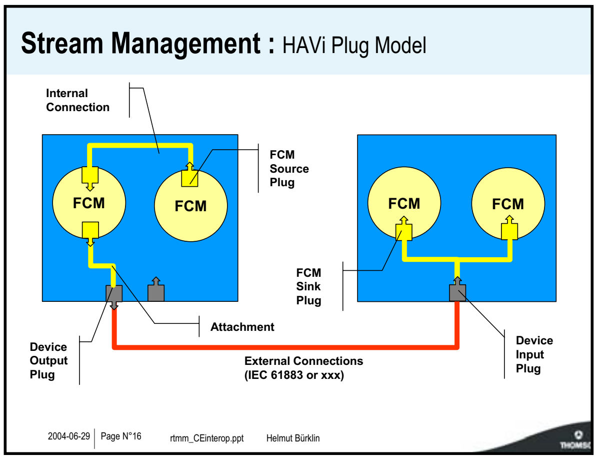
- **UPnP: protocol model over IP**
  - two services: connection + transport
  - multiple 'foreign' protocols
  - no ressource reservation
- **HAVi: software architecture over 1394**
  - stream manager element
  - for external *and* internal connections
  - bandwidth and channel reservation
  - includes non 1394 media

2004-06-29 | Page N°14 rtmm_CEinterop.ppt Helmut Bürklin



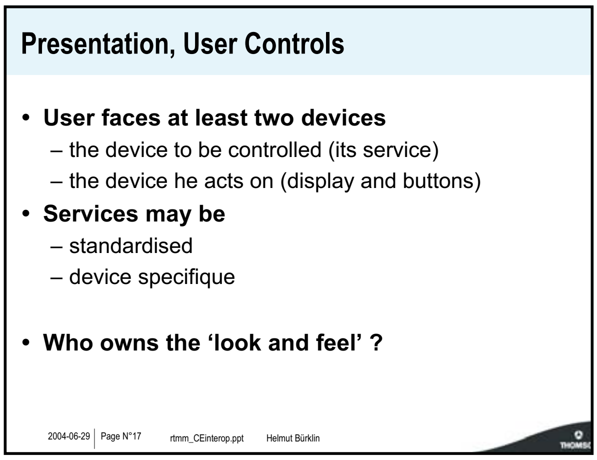
Stream Management : UPnP AV Services

2004-06-29 | Page N°15 rtmm_CEinterop.ppt Helmut Bürklin



Stream Management : HAVi Plug Model

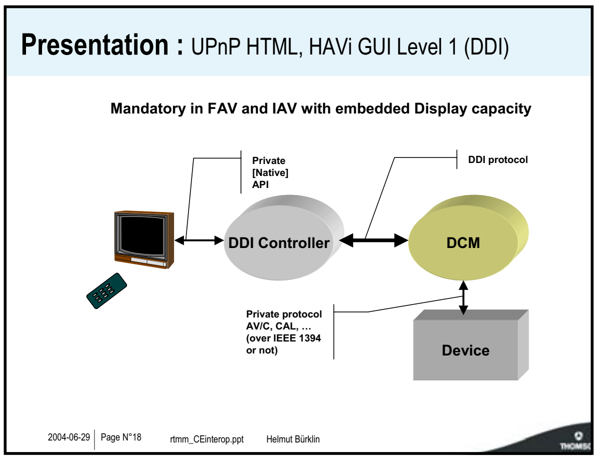
2004-06-29 | Page N°16 rtmm_CEinterop.ppt Helmut Bürklin



Presentation, User Controls

- **User faces at least two devices**
  - the device to be controlled (its service)
  - the device he acts on (display and buttons)
- **Services may be**
  - standardised
  - device specifique
- **Who owns the 'look and feel' ?**

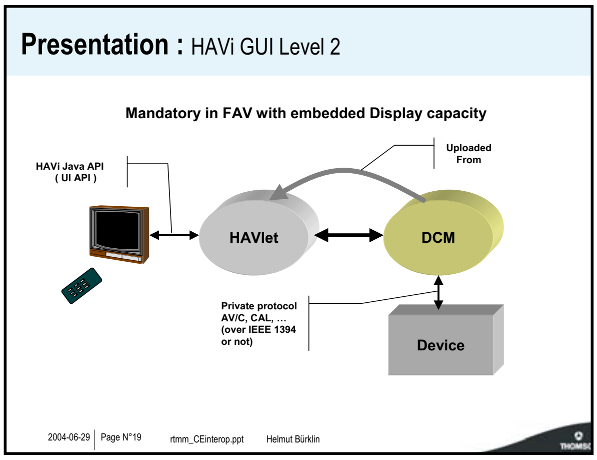
2004-06-29 | Page N°17 rtmm_CEinterop.ppt Helmut Bürklin



Presentation : UPnP HTML, HAVi GUI Level 1 (DDI)

Mandatory in FAV and IAV with embedded Display capacity

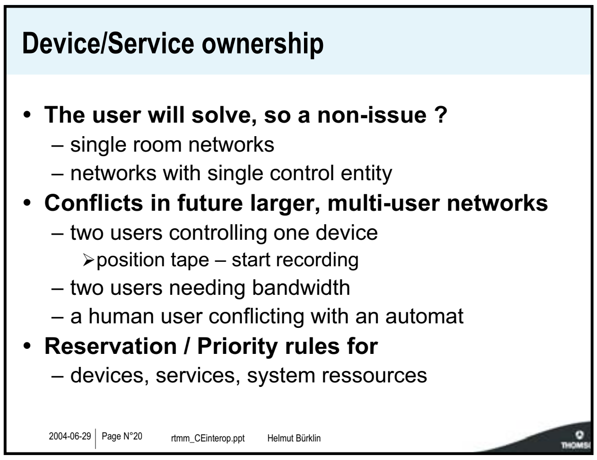
2004-06-29 | Page N°18 rtmm_CEinterop.ppt Helmut Bürklin



Presentation : HAVi GUI Level 2

Mandatory in FAV with embedded Display capacity

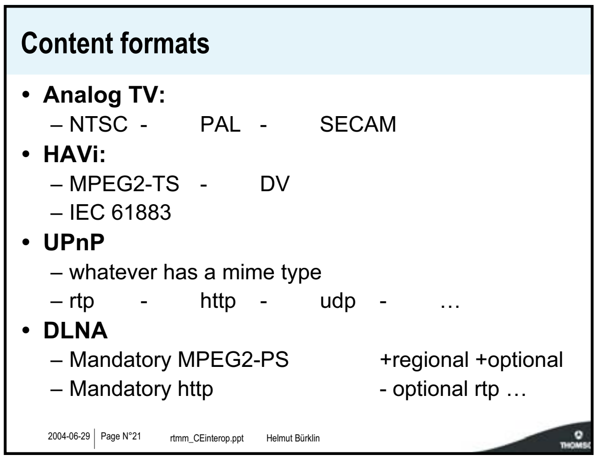
2004-06-29 | Page N°19 rtmm_CEinterop.ppt Helmut Bürklin



Device/Service ownership

- **The user will solve, so a non-issue ?**
  - single room networks
  - networks with single control entity
- **Conflicts in future larger, multi-user networks**
  - two users controlling one device
    - ➢position tape – start recording
  - two users needing bandwidth
  - a human user conflicting with an automat
- **Reservation / Priority rules for**
  - devices, services, system ressources

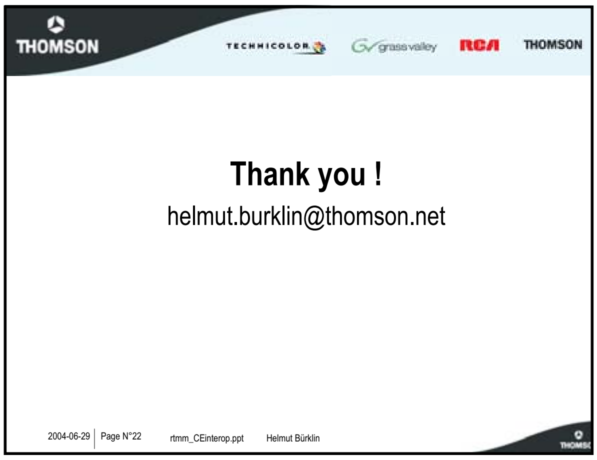
2004-06-29 | Page N°20 rtmm_CEinterop.ppt Helmut Bürklin



Content formats

- **Analog TV:**
  - NTSC - PAL - SECAM
- **HAVi:**
  - MPEG2-TS - DV
  - IEC 61883
- **UPnP**
  - whatever has a mime type
  - rtp - http - udp - …
- **DLNA**
  - Mandatory MPEG2-PS +regional +optional
  - Mandatory http - optional rtp …

2004-06-29 | Page N°21 rtmm_CEinterop.ppt Helmut Bürklin

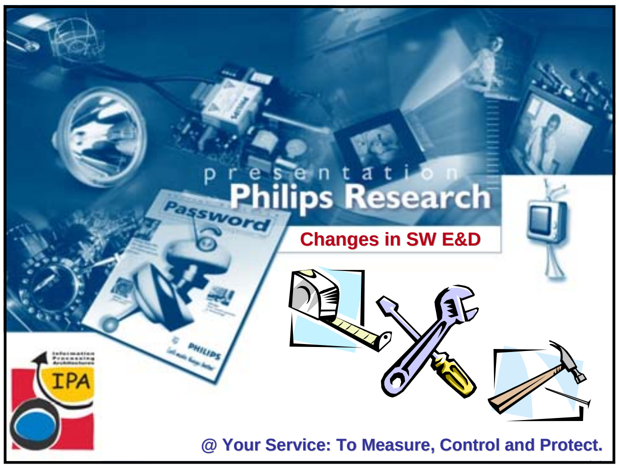


THOMSON TECHNICOLOR grass valley RCA THOMSON

Thank you !

helmut.burklin@thomson.net

2004-06-29 | Page N°22 rtmm_CEinterop.ppt Helmut Bürklin

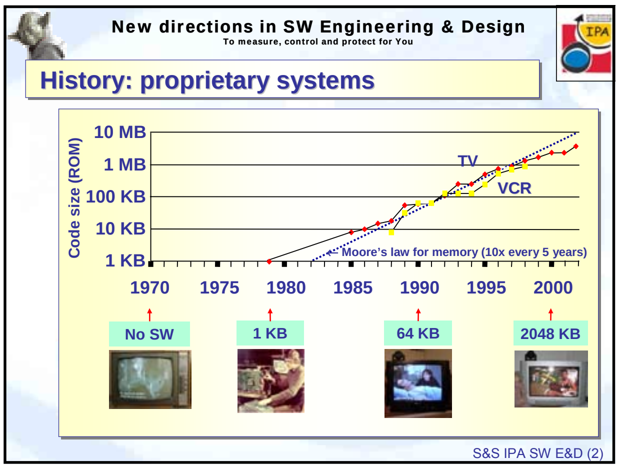


presentation

Philips Research

Changes in SW E&D

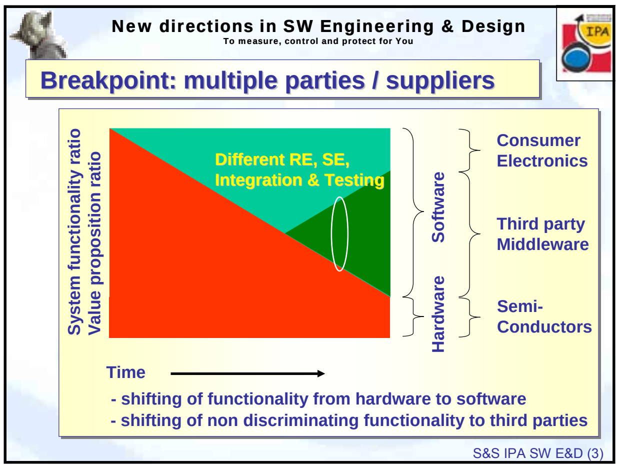
@ Your Service: To Measure, Control and Protect.



New directions in SW Engineering & Design
To measure, control and protect for You

History: proprietary systems

S&S IPA SW E&D (2)

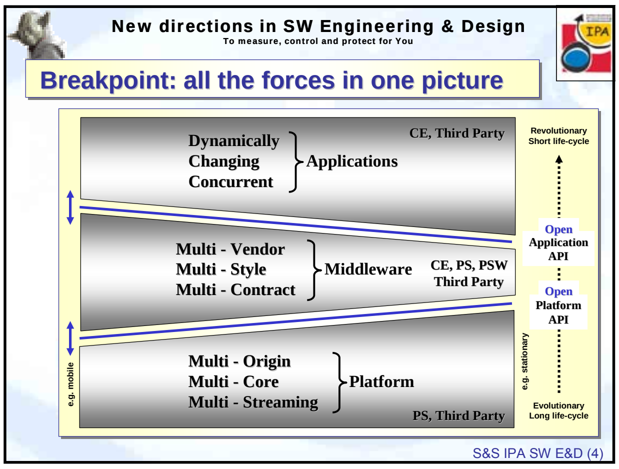


New directions in SW Engineering & Design
To measure, control and protect for You

Breakpoint: multiple parties / suppliers

- shifting of functionality from hardware to software
- shifting of non discriminating functionality to third parties

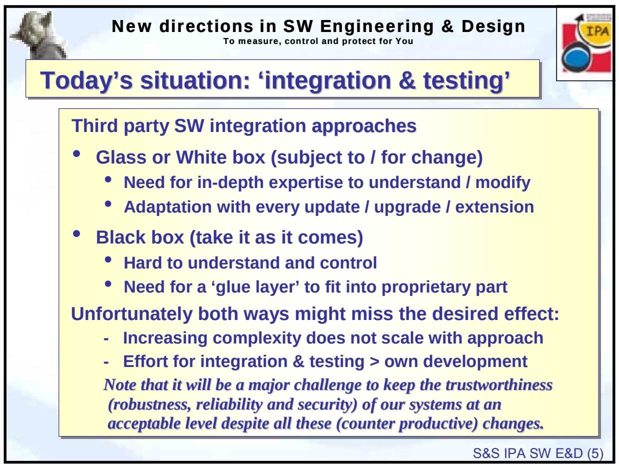
S&S IPA SW E&D (3)



New directions in SW Engineering & Design
To measure, control and protect for You

Breakpoint: all the forces in one picture

S&S IPA SW E&D (4)



New directions in SW Engineering & Design
To measure, control and protect for You

Today's situation: 'integration & testing'

Third party SW integration approaches

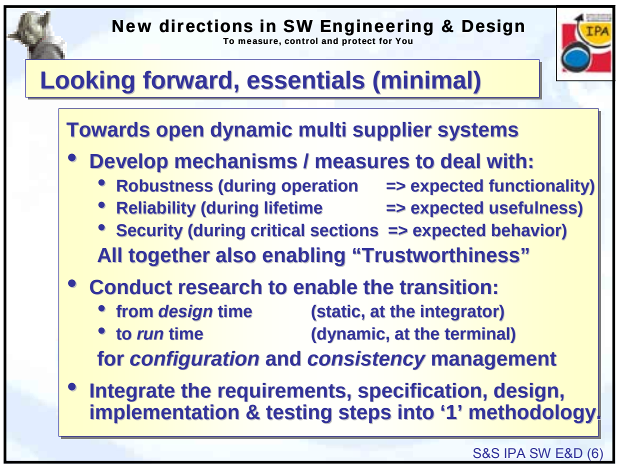
- Glass or White box (subject to / for change)
  - Need for in-depth expertise to understand / modify
  - Adaptation with every update / upgrade / extension
- Black box (take it as it comes)
  - Hard to understand and control
  - Need for a 'glue layer' to fit into proprietary part

Unfortunately both ways might miss the desired effect:

- Increasing complexity does not scale with approach
- Effort for integration & testing > own development

Note that it will be a major challenge to keep the trustworthiness (robustness, reliability and security) of our systems at an acceptable level despite all these (counter productive) changes.

S&S IPA SW E&D (5)



New directions in SW Engineering & Design
To measure, control and protect for You

Looking forward, essentials (minimal)

Towards open dynamic multi supplier systems

- Develop mechanisms / measures to deal with:
  - Robustness (during operation => expected functionality)
  - Reliability (during lifetime => expected usefulness)
  - Security (during critical sections => expected behavior)

  All together also enabling "Trustworthiness"
- Conduct research to enable the transition:
  - from *design* time (static, at the integrator)
  - to *run* time (dynamic, at the terminal)

  for *configuration* and *consistency* management
- Integrate the requirements, specification, design, implementation & testing steps into '1' methodology.

S&S IPA SW E&D (6)

New directions in SW Engineering & Design
To measure, control and protect for You

Motivation for “Trustworthiness”

Application domain research and development

Technology research, development & deployment
- A basic architecture including the of enabling SW IP
- Models for resource (processor, power, …) management
- Models for fault & terminal management and security

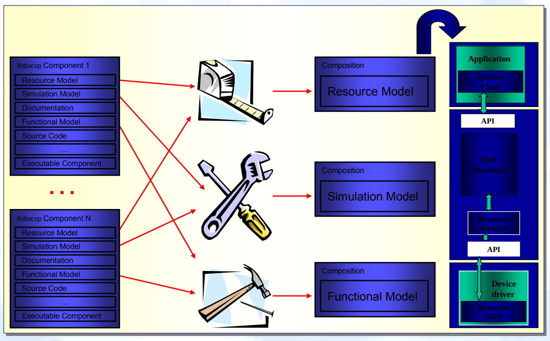
Deployment in stationary and mobile domains
- Robustness: CPU, Memory, Power, Fault Management
- Reliability: Internal Roadmap Alignment, Product family

Standardization efforts e.g. MPEG M3W

S&S IPA SW E&D (7)

New directions in SW Engineering & Design
To measure, control and protect for You

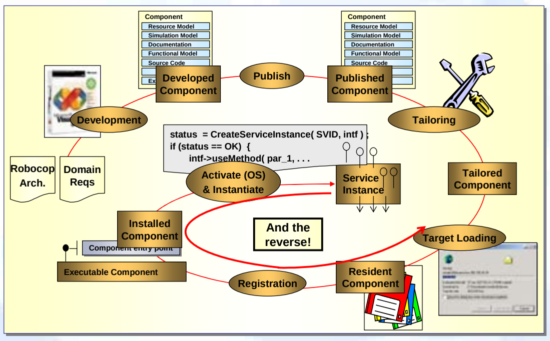
Trustworthiness: Robustness



S&S IPA SW E&D (8)

New directions in SW Engineering & Design
To measure, control and protect for You

Trustworthiness: Reliability



```
status  = CreateServiceInstance( SVID, intf ) ;
if (status == OK)  {
     intf->useMethod( par_1, . . .
```

S&S IPA SW E&D (9)

New directions in SW Engineering & Design
To measure, control and protect for You

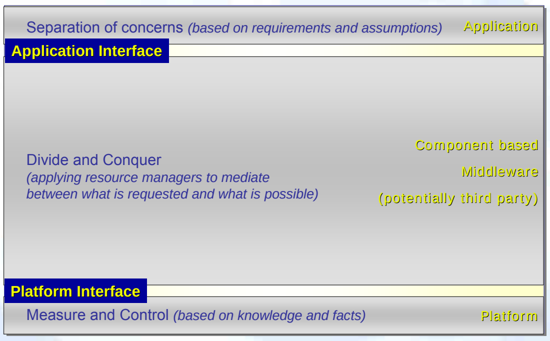
Motivation for “from Static to Dynamic”

- Life cycle management of software components includes the validation and certification of composed systems also after updates / upgrades / extensions
- However:
  - There will not be middleware from one supplier
  - There will not be one service provider to the terminal
  - The user might not want to be depending on a single terminal / middleware / service provider
- This requires a shift from
  - ‘factory / service provider’ validation / certification towards
  - ‘end-user terminal’ validation / certification

S&S IPA SW E&D (10)

New directions in SW Engineering & Design
To measure, control and protect for You

SB* : Context aware SW components ?

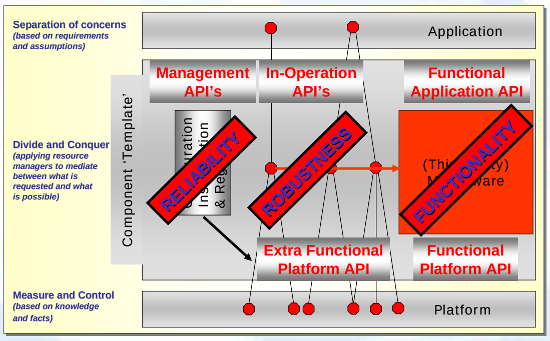

Separation of concerns (based on requirements and assumptions)
Application
Application Interface
Divide and Conquer
(applying resource managers to mediate between what is requested and what is possible)
Component based
Middleware
(potentially third party)
Platform Interface
Measure and Control (based on knowledge and facts)
Platform

** SB: Silver bullet, the ultimate solution ??*

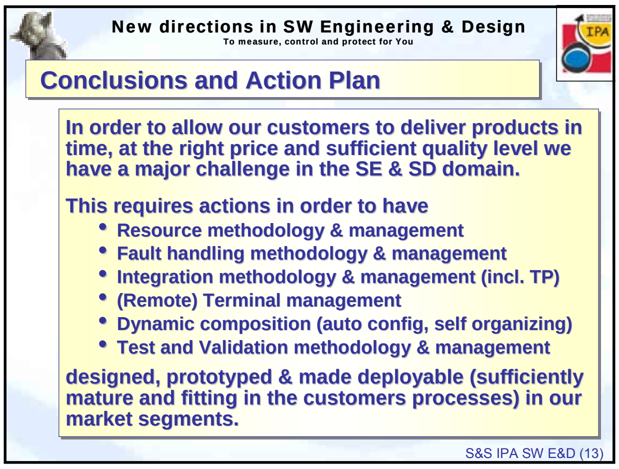
S&S IPA SW E&D (11)

New directions in SW Engineering & Design
To measure, control and protect for You

Extra functionalities in SW components


Separation of concerns
(based on requirements and assumptions)
Application
Management API’s
In-Operation API’s
Functional Application API
Divide and Conquer
(applying resource managers to mediate between what is requested and what is possible)
Component ‘Template’
RELIABILITY
ROBUSTNESS
FUNCTIONALITY
Extra Functional Platform API
Functional Platform API
Measure and Control
(based on knowledge and facts)
Platform

S&S IPA SW E&D (12)


New directions in SW Engineering & Design

To measure, control and protect for You

Conclusions and Action Plan

In order to allow our customers to deliver products in time, at the right price and sufficient quality level we have a major challenge in the SE & SD domain.

This requires actions in order to have
- Resource methodology & management
- Fault handling methodology & management
- Integration methodology & management (incl. TP)
- (Remote) Terminal management
- Dynamic composition (auto config, self organizing)
- Test and Validation methodology & management

designed, prototyped & made deployable (sufficiently mature and fitting in the customers processes) in our market segments.

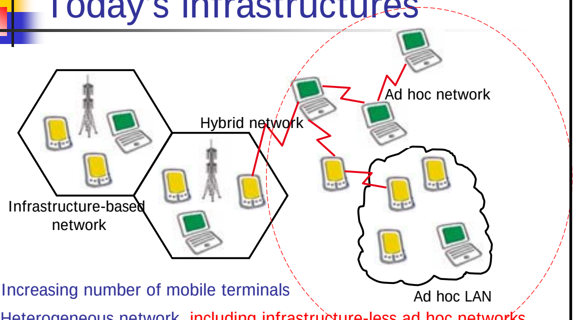
S&S IPA SW E&D (13)

Group Management for In-home Ad hoc Networks

Valérie Issarny
Projet INRIA ARLES
INRIA-Rocquencourt
Joint work with Malika Boulkenafed & Jinshan Liu

INRIA

Today's infrastructures



→Increasing number of mobile terminals
→Heterogeneous network, including infrastructure-less ad hoc networks

In-home Ad hoc Networks

- Advantages
  - Infrastructure-less networking
  - Allows for spontaneous networking
  - Complements infrastructure-based networks
  - ➢ Enabler of the extended home environment
- But
  - High dynamics
  - Changing execution context
- ➢ Group management for dealing with the dynamic execution context

Background

- Adapt solutions aimed at wired networks
  - Adaptation to the network's dynamic topology by restoring lost connections through routing
- Constrain group membership *wrt* ad hoc network features
  - Connectivity constraints (geographical and/or routing-based)
  - Integrity constraints
- ➢ Group membership dependent upon the application
  - ➢ Generic group management for ad hoc networks

Outline

- Attributes of group membership for MANET
- Group service design
- Application to QoS management in the home network
- Conclusion

Group Membership Attributes

- Location
  - Location-unaware *vs* proximity-based and/or bounded
- Openness
  - Open *vs* closed *wrt* security domain
- Connectivity
  - Full *vs* partial *vs* loose connectivity
- QoS awareness
  - Service-level and resource-level QoS attributes

Group Service Design

- Requirements
  - Minimize resource consumption
  - Decentralized solution
  - Mask the network's dynamic topology
- Functions
  - Discovery of group members
  - Group initialization
  - Management of the group's dynamic

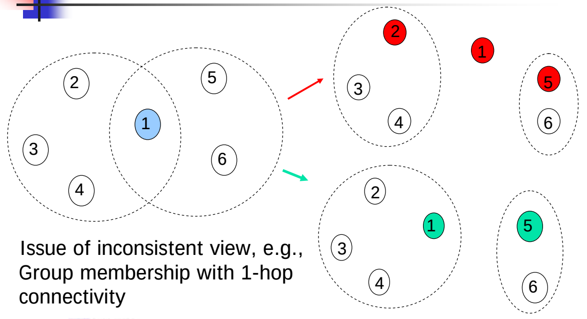
Periodic discovery process

- Broadcast *Disc(groupId, senderId)* over 1 hop
  - QoS constraints: check conformance of local QoS attributes prior sending
  - Closed constraint: include certificate + encrypt data
- Handle *Disc* msgs
  - Location constraints: inclusion and forwarding of messages accordingly

Group initialization

- **Functions**
  - Enforces connectivity constraints
  - Enforces global QoS constraints
  - GKA protocol for closed groups
- **Base algorithm**
  - Management by *leader* node
  - Local list of peers sent to *leader* using *Join*
  - Computation of group membership according to the lists of peers

Group membership at *leader l*



Issue of inconsistent view, e.g., Group membership with 1-hop connectivity

Group membership at *leader l (Cont'd)*

- **Notations**
  - *P(n):* Peers of node *n*
  - *R:* Receipt of *Join* by member nodes
  - *G(l):* Leader's peers that belong to *R*
  - $I(l) = \cap_{(j \in G(l))} P(j)$: Peers of *leader* that discovered each others and meet membership constraints
  - $U(l) = \cup_{(j \in G(l))} P(j)$: Set of all peers

Group membership at *leader l (Cont'd)*

- **Group members**
  - ***G(l) = I(l) = U(l)*****:** All peers of *G(l)* have identical view on group membership; *G(l)* as group membership.
  - ***(G(l) = I(l) and I(l) ≠ U(l)) or (G(l) ≠ I(l) and I(l) = U(l))*****:** Nodes belonging to *U(l)* but not *G(l)* do not meet location constraints *wrt leader*; *G(l)* as group membership and exclusion message to discarded nodes.
  - ***(G(l) = U(l) and I(l) ≠ U(l)) or (G(l) ≠ U(l) and I(l) ≠ U(l) and G(l) ≠ I(l))*****:** Nodes of *I(l)* meet location constraints *wrt* nodes of *G(l)* but some nodes of *G(l)* do not meet such constraints; *leader* establishes group membership (e.g., *I(l)*)

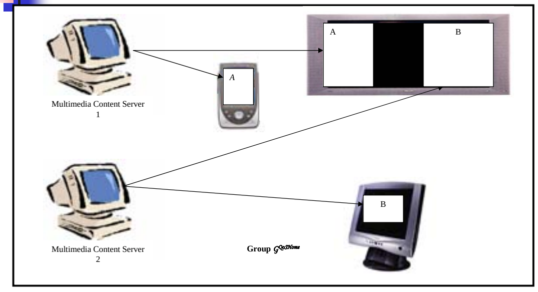
Management of the dynamics

- Dynamically update the period according to the past behavior of the group
- Change *leader* every period

Assessment

- Theoretical complexity in O(N)
- Inconsistent group membership due to the periodic discovery process
  - Required connectivity can still be established although violating location-based constraints (e.g., from 2 to 3 hops)
    - Does not impact upon the functional behavior
  - Required connectivity broken
    - Exception to application

In-home QoS Management



Membership attributes

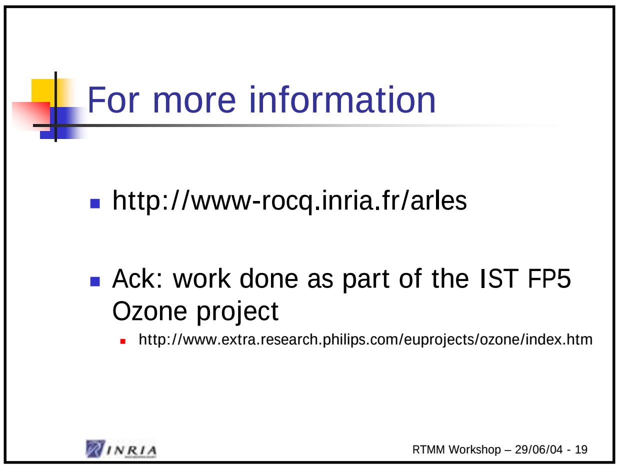
- Location
  - Geographical proximity
- Partial connectivity
  - Client nodes connected to server
- QoS awareness
  - Resource-level attributes (CPU, memory, battery)

Dedicated group management

- Dynamic bandwidth management [Sha *et al.*, PerCom'2003]
  - Centralized bandwidth management according to the server's requirements and available bandwidth
- Group managing bandwidth allocation
  - Centralized at the leader node
    - Fair load distribution

Conclusion

- **Contribution**
  - Generic group management for ad hoc networks
- **On-going work**
  - Prototype implementation of the Group service within the WSAMI middleware for mobile Web services
  - Experiment
- **Future work**
  - Group-based development of mobile applications


For more information

- http://www-rocq.inria.fr/arles
- Ack: work done as part of the IST FP5 Ozone project
  - http://www.extra.research.philips.com/euprojects/ozone/index.htm

RTMM Workshop – 29/06/04 - 19

Cooperative Computing & Communication Laboratory

Combining HAVi and OSGi for Remote Control of CE Devices in Heterogeneous Home Networks

Isabell Jahnich
isabell.jahnich@c-lab.de

29 June 2004

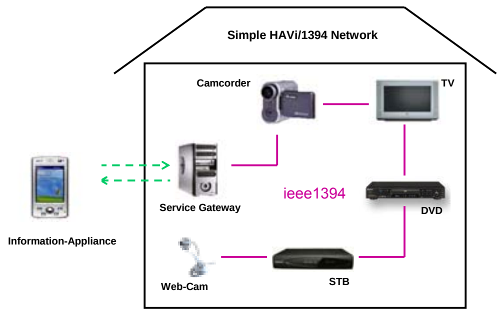
Outline

Motivation

- Increasing number of integrated network technologies within Smart Homes
- Convergence of these heterogeneous networks

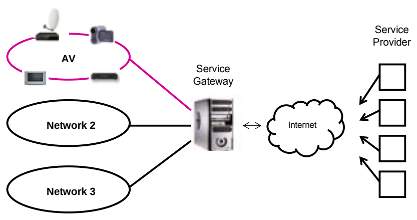
- HAVi exposes as an auspicious standard within the home entertainment environment
- Gateway is required to support interaction with different home areas
- OSGi = most established gateway technology

Motivation



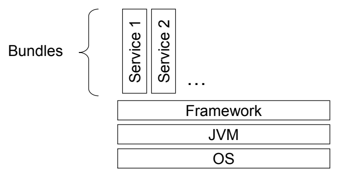
Open Service Gateway Initiative

- Specification for the network delivery of services



OSGi Architecture

- Framework: supports the deployment of service applications (Bundles)
- Bundle: comprises Java classes and resources that implement Services

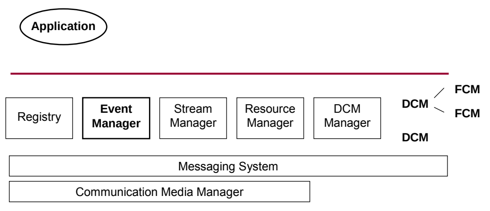


Home Audio Video Interoperability

- Home network standard for seamless interoperability between CE devices
- Based on IEEE 1394
- Specifies services to locate, query, control and extend devices
- Classification of devices
  - Controllers: HAVi capable devices that are able to control non-HAVi capable ones
  - Targets: non-HAVi capable devices

HAVi Architecture

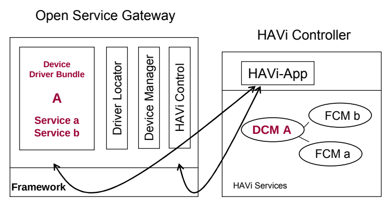
- Based on specified software elements that provide services within the home network



Requirements of a Combination

- Independency of both systems
- The service gateway should support the flexibility of the HAVi network
- Adequate handling of events within the other system
- Reliability, modifiability, reusability of components
- Functional extension
  - Connection of the HAVi network to heterogeneous networks
  - Remote control of CE devices

Architecture of the Combination



Device Representation

- Connected HAVi devices (DCMs) are represented by single bundles (Device Driver)
- FCMs are mapped to OSGi services
- Communication with its DCM via dedicated HAVi Applications
- GUI for the represented HAVi device (e.g. servlet)
- Components of the Device Access mechanism

⇨ Plugging and unplugging of devices come along with installing and uninstalling of the bundle representations

HAVi Control

- HAVi Event Listener for interested HAVi events, e.g.
  - New device
  - Gone device
  - …
- Integrated HAVi Applications subscribes to interested events and notices the listener
- Initiation of the specified Device Access in case of newly connected HAVi devices

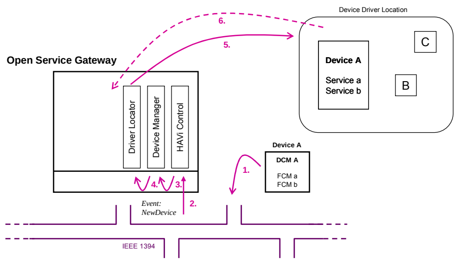
Device Access of OSGi

- Supports the coordination of automatic detection and attachment of existing devices
- Newly plugged devices are represented as device services
- Device driver is responsible for attaching the suitable device service under control of the *Device Manager*
- *Driver Locator* supports the manager to find the adequate driver

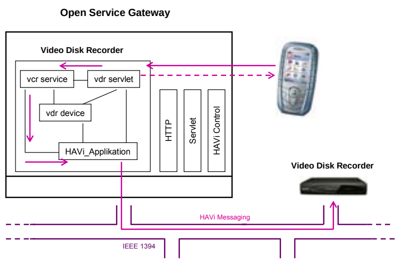
Device Access of OSGi

- HAVi Control bundle is able to register a device service of any HAVi device (no details)
- Device Driver bundle contains a driver service with a detailed device description
- Device Manager controls the localisation and installation of the adequate Device Driver with aid of the
  - device description from the HAVi system
  - Device Driver service with its device description within the Device Driver bundle
  - Driver Locator

Device Access



Remote Control



Thanks for your attention!

csem swiss center for electronics and microtechnology

Round Table
RTMM Workshop, Catania, June 29, 2004

Valérie Issarny, INRIA, F
Liesbeth Steffens, Philips Research, NL
Helmut Bürklin, Thomson, F

Chair: Jean-Dominique Decotignie, CSEM, CH

Our distinguished panelists

Valérie Issarny
INRIA (F)

Liesbeth Steffens
Philips Research (NL)

Helmut Bürklin
Thomson (F)

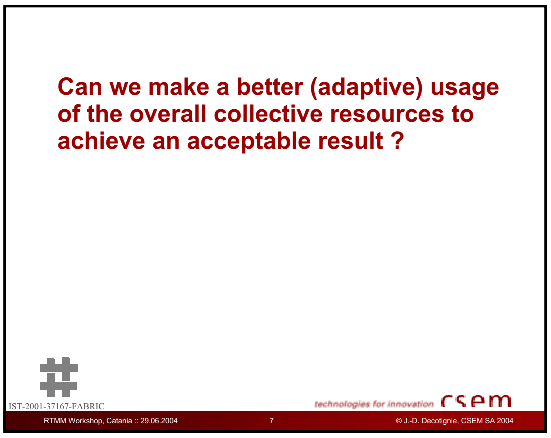
Is Real-Time a real concern ?

HAVi supports real-time constraints whereas UPnP leaves it free to the implementation. How do you see the issue solved in UPnP ? Is it necessary to solve it ? Is there still something to do ? Same for the platforms: QoS aware OSs and middlewares

- Special purpose technology vs general purpose
- « Let solve the problems when they appear »

What about predictability ?

- wireless communication have fluctuating properties and more and more applications compete on the use of the same spectrum. Where do you see the future for wireless multimedia communications ?

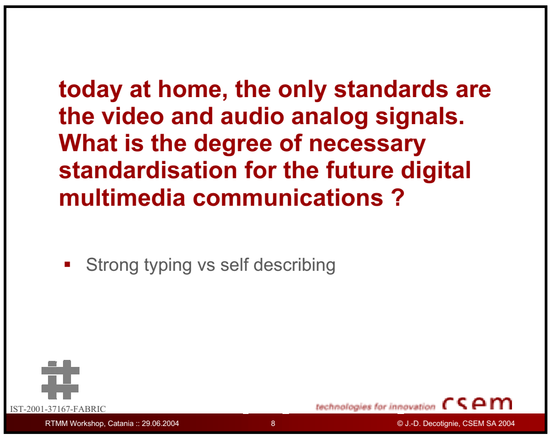


Can we make a better (adaptive) usage of the overall collective resources to achieve an acceptable result ?

IST-2001-37167-FABRIC

technologies for innovation csem

RTMM Workshop, Catania :: 29.06.2004 7 © J.-D. Decotignie, CSEM SA 2004



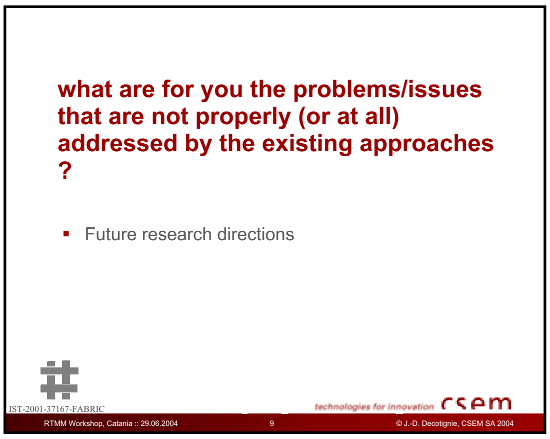
today at home, the only standards are the video and audio analog signals. What is the degree of necessary standardisation for the future digital multimedia communications ?

- Strong typing vs self describing

IST-2001-37167-FABRIC

technologies for innovation csem

RTMM Workshop, Catania :: 29.06.2004 8 © J.-D. Decotignie, CSEM SA 2004



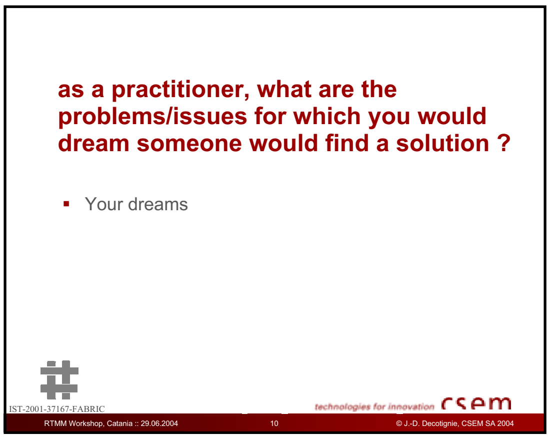
what are for you the problems/issues that are not properly (or at all) addressed by the existing approaches ?

- Future research directions

IST-2001-37167-FABRIC

technologies for innovation csem

RTMM Workshop, Catania :: 29.06.2004 9 © J.-D. Decotignie, CSEM SA 2004



as a practitioner, what are the problems/issues for which you would dream someone would find a solution ?

- Your dreams

IST-2001-37167-FABRIC

technologies for innovation csem

RTMM Workshop, Catania :: 29.06.2004 10 © J.-D. Decotignie, CSEM SA 2004

csem

Thank you for your attention.

csem

technologies for innovation