Distributed storage in heterogenous home networks

Ing. I.W.F. Paulussen

Coach: Dr.ir. G.E. Mekenkamp
Supervisor: Prof.ir. M.P.J. Stevens
Period: 1 March 2003 – 1 December 2003
Distributed storage in heterogeneous home networks

Confidential

Master Thesis
I.W.F. Paulussen
Distributed storage in heterogeneous home networks

Confidential

Master Thesis

Author: I.W.F. Paulussen
Student ID: 485461
Chair: Computers, Networks & Design (C.N.D.)
Supervisor TUE: Prof. Ir. M.P.J. Stevens
Company: Philips Research Laboratories Eindhoven
Company supervisor: Dr. Ir. G.E. Mekenkamp
Graduation period: 1 March 2003 till 1 December 2003
Dedicated to Caroline.

For all here patience and lonesome evenings.
Preface

After four years of learning and hard work the finish line is almost insight. The last hurdles to take before I really reach the finish line is writing and defending my master thesis. With the master thesis now in front of you, the hurdle of writing the master thesis is taken. This master thesis is based on work carried out in the group Storage Systems and Applications within the Philips Research Laboratories Eindhoven.

Acknowledgements

Without the support of the following people I probably would have not finished my part time study Informatie Technologische Wetenschap at the Eindhoven University of Technology. That's why I would like to thank:

Caroline, my father, mother, brother and the rest of the family.

All my friends who wished me luck for exams and where interested in how the study was progressing.

Dr. Ir. Gerhard Mekenkamp and Prof. Ir. M.P.J. Stevens for their supervision. With them I had fruitful discussions about my graduation assignment and they gave useful comments and remarks on my master thesis. I was pleased that Prof. Ir. M.P.J. Stevens still wanted to be my graduation professor after retiring!

The SPATION team members, Mauro Barbieri and Frank Crienen for useful discussions and ideas, reading the report and taking over a bit of my more work when I was at university for my study.

The part time student colleagues with whom I had a lot of fun during the lectures. We had a lot of nice discussions before and after the examinations. The ones who still need to graduate, good luck!

Philips Research who made it possible for me to do this part-time study.

Finally, all the ones that think, why don't I get an acknowledgement, thank you.

After these four years I can advice everybody, who's willing to do a part time study, to go for it, but you'll need a lot of perseverance.

Igor Paulussen

Nuenen, 26 November 2003
Abstract

Consumer electronic devices with large storage capacity become more and more interconnected in home networks. This gives users the opportunity to distribute huge amounts of multimedia content over multiple devices. In this situation retrieval of distributed content becomes a challenge.

The graduation assignment is fulfilled within the European Research project SPATION, which focuses on storing and retrieving multimedia content in a home network of connected CE devices and finds solutions to help the user to move, retrieve, and organize this content. The project investigates how retrieval can be integrated in a distributed home system built using existing middleware standards such as Universal Plug and Play (UPnP).

An investigated user requirement in the graduation assignment is that metadata in a distributed home storage environment must be handled in a way that it is transparent to the user. To reach this requirement the Universal Plug and Play (UPnP) standardized Content Directory Service (CDS) is investigated. This service provides a mechanism to share metadata between devices in the network. Taking this Content Directory Service as a starting-point, possible metadata aggregator approaches are investigated. The goal of the metadata aggregator is to offer a centralized place where metadata, which belongs to content distributed on different devices in the network, is aggregated.

This master thesis provides abstract solutions that can help the user with transparently accessing their distributed metadata. From these solutions important requirements for the metadata aggregator were found that are used as guidelines for the verification of the possible implementations of a metadata aggregator. Also, the “Super” CDS is introduced which makes it is really possible to build such an aggregator of metadata. This “Super” CDS, which is based on the standard UPnP Content Directory Service, offers a central place to access metadata from different UPnP Content Directory Services available in the home network. At the end, several possible implementation approaches are highlighted, all with their own advantages and disadvantages with respect to the metadata aggregator requirements.
Samenvatting

Consumenten Elektronica apparaten met een grote opslag capaciteit worden steeds vaker aan elkaar gekoppeld in huis netwerken. Dit geeft de gebruiker de mogelijkheid om grote hoeveelheden multimedia data te distribueren over meerdere apparaten. Hierdoor wordt het een uitdaging om de gedistribueerde multimedia data terug te vinden.

De afstudeeropdracht is uitgevoerd in het kader van het Europese onderzoeksproject SPATION. Dit project richt zich op het opslaan en het terugvinden van multimedia data in een huis netwerk dat bestaat uit aan elkaar gekoppeld CE apparaten. Het project levert oplossingen om de gebruiker te helpen met het verplaatsen, terug vinden en organiseren van deze data. In het project wordt onderzocht hoe, door gebruik te maken van bestaande middleware standaarden zoals Universal Plug and Play (UPnP), het terugvinden van data in een gedistribueerd huis netwerk kan worden vereenvoudigd.

Een gebruikers eis die onderzocht is in de afstudeeropdracht is dat de afhandeling van metadata voor de gebruiker transparant is. Om aan deze eis te kunnen voldoen is de in Universal Plug and Play (UPnP) gestandaardiseerde Content Directory Service (CDS) onderzocht. Deze service biedt een mechanisme aan om metadata te delen tussen apparaten in een netwerk. Met deze Content Directory Service als uitgangspunt zijn mogelijke benaderingen voor metadata aggregators uitgezocht. Het doel van de metadata aggregator is een centrale plaats aan te bieden waar metadata, die behoort bij multimedia data die gedistribueerd is over verschillende apparaten in het netwerk, wordt samengevoegd.

Dit afstudeerverslag draagt abstracte oplossingen aan die de gebruikers kunnen helpen om op een transparante manier toegang te krijgen tot hun gedistribueerde metadata. Met behulp van deze oplossingen zijn belangrijke eisen voor de metadata aggregator gevonden die worden gebruikt als richtlijn voor de evaluatie van de aggregator implementaties. Daarnaast wordt er een "Super" CDS geïntroduceerd die het mogelijk maakt om daadwerkelijk een metadata aggregator te bouwen. Deze "Super" CDS, die gebaseerd is op de standaard UPnP Content Directory Service, biedt een centrale plaats waar toegang verkregen kan worden tot alle metadata van verschillende UPnP Content Directory Services in het huis netwerk. Als laatste worden een aantal mogelijke implementaties met bijbehorende voor en nadelen belicht.
# Table of Contents

1. **Introduction** .......................................................................................................................... 1  
   1.1. Group: Storage Systems and Applications (SSA) ............................................................... 1  
   1.2. Graduation Assignment ...................................................................................................... 1  
      1.2.1. Problem situation ................................................................. 2  
      1.2.2. Assignment ........................................................................ 2  
      1.2.3. Goal .................................................................................. 2  
      1.2.4. Report structure ................................................................ 2  

2. **Distributed Storage** .............................................................................................................. 3  

3. **Analyzing user benefits and requirements** ......................................................................... 9  
   3.1. The SPATION project ....................................................................................................... 9  
   3.2. Application scenarios ....................................................................................................... 11  
   3.3. SPATION system setup ..................................................................................................... 11  
   3.4. Content navigation .......................................................................................................... 13  
      3.4.1. Video browser ................................................................. 14  
      3.4.2. Key-frame browsing ....................................................... 15  
      3.4.3. Video preview ................................................................. 15  
      3.4.4. Pictorial overview of a video .......................................... 16  
   3.5. User benefits and requirements ....................................................................................... 16  
      3.5.1. Access content in one central place .................................. 16  
      3.5.2. Obtain information about stored content ......................... 17  
      3.5.3. Quickly access content of interest .................................... 17  

4. **UPnP Basics** ......................................................................................................................... 19  
   4.1. UPnP Architecture .......................................................................................................... 20  
      4.1.1. Addressing ........................................................................ 21  
      4.1.2. Discovery .......................................................................... 22  
      4.1.3. Description ....................................................................... 23  
      4.1.4. Control ............................................................................. 23  
      4.1.5. Eventing .......................................................................... 23  
      4.1.6. Presentation .................................................................... 24
4.2. UPnP devices and services ..................................................24
  4.2.1. Media renderer device ...............................................25
  4.2.2. Media Server device ...............................................27
4.3. The Content Directory Service ..........................................27
  4.3.1. Browsing and Searching ...........................................28
  4.3.2. Object creation and file transfer .................................30
4.4. UPnP system set-up examples ...........................................31

5. UPnP based distributed storage: The “Super” CDS ....................33
  5.1. Metadata aggregator ..................................................33
    5.1.1. Overall concept ................................................33
    5.1.2. Metadata aggregator requirements ............................38
  5.2. “Practical” technical solution based on UPnP Content Directory Service ....38
    5.2.1. UPnP Content Directory Service ...............................39
    5.2.2. Announcing that a Content Directory Service is a “Super” CDS ....40
    5.2.3. Keeping “Super” CDS up to date using UPnP Eventing ...........41
    5.2.4. Merging metadata ...............................................43
    5.2.5. Storing self defined metadata ..................................43
  5.3. Analysis of UPnP based “Super” CDS .................................44

6. “Super” CDS implementation approaches ................................45
  6.1. One Content Directory Service for all media server devices in the network ..45
  6.2. Simple “Super” CDS implementation ................................46
  6.3. Simple “Super” CDS implementation with local caching ...............47
  6.4. “Super” CDS implementation on a media server ........................48
  6.5. “Super” CDS implementation on a media server using a cache ..........49
  6.6. “Super” CDS implementation using two Content Directory Services in one media server ..................................................51
  6.7. Summarizing the possible implementation approaches ..................54
  6.8. Useable extra possibilities for the building of a “Super” CDS ...........55
    6.8.1. Cache strategies ..............................................55
    6.8.2. Keeping cache up to date using UPnP eventing ..................55
    6.8.3. All media server devices provide a “Super” CDS ................56
  6.9. Verification of “Super” CDS implementations approaches ...............56

7. Conclusions ......................................................................57

8. Future work ......................................................................59
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>References</td>
<td>61</td>
</tr>
<tr>
<td>List of figures and tables</td>
<td>63</td>
</tr>
<tr>
<td>Abbreviations and Acronyms</td>
<td>65</td>
</tr>
<tr>
<td>Appendix A</td>
<td>67</td>
</tr>
</tbody>
</table>
1. Introduction

Consumer electronics (CE) devices will soon have a huge storage capacity, and enormous processing power, which will create the possibility to offer a wide range of functions. Additionally high speed wired or wireless networks provide the opportunity to interconnect these devices. This results in a home network with tremendous possibilities: connected devices can share both control (distribution of functionality) and content (distribution of storage). From the end users’ point of view, sharing control means that control functions of devices are available on other devices. Content sharing means that the stored content is transparently accessible on every device even if it is spread over the whole home network and it is not at a single physical location anymore. In this master thesis the focus will be only on sharing of content. The huge amount of data a user can store in the house becomes very difficult to manage due to the fact that retrieving data is more difficult in a distributed situation. Without an integrated storage solution, the users would have to search each device separately.

1.1. Group: Storage Systems and Applications (SSA)

The graduation assignment is carried out within the group Storage Systems and Applications (SSA) which is part of the sector Storage of the Philips Research Laboratories. The group Storage Systems and Applications aims at innovations at the system level of storage devices to enable increasingly advanced applications of data storage, primarily in the home consumer environment. One of the main topics of the group is video recording that traditionally is the most storage intensive application in the home. The group focuses on the explosive growth of available content combined with enormous increases in storage capacities. More content means less value if the user cannot find its way through it. Data searching and retrieval mechanism are studied in the group and integrated into the storage systems. Future storage systems all interact with networks, enabling among others distributed storage. This is also investigated in the group, both for the home environment and for mobile applications. To achieve this all, the group has capabilities in the field of storage application concepts, systems architecture, integration of technologies, video compression and analysis, and fast software-based prototyping.

1.2. Graduation Assignment

This graduation assignment has been fulfilled within the European SPATION project. The objective of SPATION is to find solutions for the problem how to move, organize and retrieve information (music, video, pictures, documents, etc.) in a heterogeneous home network. The focus of the project is on storing and retrieving information in a home network of connected CE devices, such as (personal) video recorders, TV’s, handheld devices etc. Interconnection of these devices and wireless communication with various portable devices will create a complex home system with the capacity to store many types of data and offer new ways of interacting with it. For example, the personal video recorder will get more and more storage capacity, and a powerful
processor to support many new functions. One of these functions will be to allow access to personal content and meta-data via handheld devices.

To offer the user the flexibility and ease of use that is custom for consumer electronics new solutions are required. This includes easy set-up and maintenance of the home system. Also the transfer of data between devices should be easy and transfer and adaptation of accompanying meta-data must be transparent to the user.

1.2.1. Problem situation

Many of the connected devices in the home will offer the possibility to store hundreds of hours of video, thousands of songs, and tens of thousands of photographs. There needs to be a solution to help the user to move, retrieve, and organize this information. From a technical perspective this means the need to:

- Find solutions that create a distributed storage space by interconnecting Consumer Electronics devices.
- Investigate how metadata can be handled in a distributed home storage environment in a way that is transparent to the user.

1.2.2. Assignment

The main problem that needs to be solved in the graduation assignment is how to deal with distributed storage in a heterogeneous home network

1.2.3. Goal

The goal of the graduation assignment is to find answers to the question, “how to deal with distributed storage in a heterogeneous home network”. To answer that question, a middleware standard, such as UPnP, must be investigated to see what this middleware standard offers to solve the distributed storage problem. Also, UPnPs Content Directory Service (CDS) will be investigated: what can be used from this service to handle multimedia data such as meta-data and (multiple) content files. When the Content Directory Service doesn’t offer the complete solutions to handle distributed storage, extensions to the protocol need to be made. At the end of the project the results need to be implemented and integrated in the SPATION demonstrator. This should make possible to show with the SPATION demonstrator the working of a distributed storage network.

1.2.4. Report structure

This Master Thesis gives an overview of the graduation assignment fulfilled within the SPATION project. The focus is on how the Universal Plug and Play (UPnP) standard can be used to provide transparent metadata management in home networks. To reach this goal the following structure in the master thesis is used: Chapter 2 focuses on the operation boundaries of the graduation assignment. The user benefits and requirements are analyzed in chapter 3. In chapter 4 the basics of the UPnP standard are discussed because the investigated metadata aggregator is built using this standard. The introduced metadata aggregator called “Super” CDS is substantiated in Chapter 5. Possible “Super” CDS implementation approaches are the topic of chapter 6. Conclusions and future work are presented in chapter 7 and 8.
2. Distributed Storage

In this chapter the operation boundaries of the graduation assignment are defined. The assignment needs to be fulfilled within the operation boundaries of the SPATION project, which focuses on the following:

- Distributed storage.
- Heterogeneous home networks.
- Using applications and resources transparently in a “small” home network.

One of the operation boundaries of this project is the projects decision to use UPnP as middleware stack. The question is: How do the operation boundaries of the SPATION project influence the graduation assignment that needs to solve the main problem of how to deal with distributed storage in a heterogeneous home network? To answer this question many other questions need to be answered first, like:

- What is distributed storage?
- What is a heterogeneous home network?
- What do transparency and scalability mean for home networks?

To answer the first question, the dictionary is consulted to get the meaning of the word distribute. According to the dictionary it means:

\textit{Distribute} separate something into parts and give a share to each person or thing

When in this explanation “something” is replaced by storage and “thing” is replaced by device the next sentence appears: separate storage into parts and give a share to each person or device. This already makes the link between the word distribute and the first question. But still distributed storage has a very broad scope and in the context of the graduation assignment it is still too vaguely described. To narrow this broad scope, books about distributed systems were consulted for the definition of a distributed system.

Tanenbaum [1] defines a distributed system as:

\textit{A distributed system is a collection of independent computers that appears to its users as a single coherent system.}

Coulouris [3] defines a distributed system as:

\textit{A distributed system is one in which hardware or software components located at networked computers communicate and coordinate their actions only by passing messages.}
These two simple definitions cover the entire range of systems in which networked computers can usefully be deployed. Both definitions have two aspects. The first one deals with hardware: the machines are autonomous. The second one deals with software: the users think they are dealing with a single system.

Coulouris adds to this that:

_The sharing of resources is a main motivation for constructing distributed systems._

And,

_The presence of computers everywhere only becomes useful when they can communicate with one another._

When this last statement is adapted to the vision of SPATION the statement should become even broader: The presence of _computational power_ everywhere only becomes useful when they can communicate with one another.

To deal with these above mentioned distributed (storage) system Coulouris defines some key challenges for distributed systems, like:

- Heterogeneity of its components
- Openness
- Security
- Scalability
- Failure handling
- Concurrency of components
- Transparency

One of these key challenges is heterogeneity of its components, but what is exactly heterogeneous. Heterogeneous literally means: made up of different kinds; varied in composition. This explanation helps with answering the question: what is a heterogeneous home network? With using the explanation a heterogeneous home network is: A network in the home made up of different kinds of devices. In practice for the SPATION project this means that the heterogeneous home system consists of a number of interconnected stationary devices (like Consumer Electronics (CE) devices and PCs) and various handheld devices. All these devices communicate with each other. For this interconnection multiple solutions are available, and the network in the home will be a mixture of solutions, such as wired network connections, wireless network connections (802.11), bluetooth etc. All these devices also have large processing power and have their own storage capacity that allows storing large amounts of content. When these devices are combined a network is created that allows users to control devices in the house and search for content stored on any of these devices with a handheld device.
Answering the questions, what is distributed storage and what is a heterogeneous home network, already gives a better view on the assignment, but many new questions arise. For example, Coulouris defined the seven key challenges, but what do they mean for the graduation assignment?

Not all these defined key challenges are equally important for the assignment. The problem is that distributed systems and distributed storage itself is not a new research area. For example the Internet can be seen as a large distributed storage network. A lot of papers and books are available on this distributed storage topic. The problem of these papers is that they are too specific and the technologies described in the papers have different requirements. For example scalability is very important in very large distributed networks and broadcasting messages over the network create an unacceptable network load. In contrast, the “small” home network doesn’t need to scale up and that’s why broadcasts are possible e.g. to announce the presence of a device to all other devices. Using broadcast messages distributed algorithms can often be implemented in a much simpler way.

Another important goal of a distributed system is to hide the fact that its processes and resources are physically distributed across multiple devices. A distributed system that is able to present itself to users and applications as if it were only a single (computer) system is said to be transparent. There are different forms of transparency in a distributed system, like:

<table>
<thead>
<tr>
<th>Transparency</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>Hide differences in data representation and how a resource is accessed</td>
</tr>
<tr>
<td>Location</td>
<td>Hide where a resource is located</td>
</tr>
<tr>
<td>Migration</td>
<td>Hide that a resource may move to another location</td>
</tr>
<tr>
<td>Relocation</td>
<td>Hide that a resource may be moved to another location while in use</td>
</tr>
<tr>
<td>Replication</td>
<td>Hide that a resource is replicated</td>
</tr>
<tr>
<td>Concurrency</td>
<td>Hide that several competitive users may share a resource</td>
</tr>
<tr>
<td>Failure</td>
<td>Hide the failure and recovery of a resource</td>
</tr>
<tr>
<td>Persistence</td>
<td>Hide whether a (software) resource is in memory or on disk</td>
</tr>
</tbody>
</table>

The reasoning till now focused mainly on distributed (computer oriented) systems. This still covers a large area. SPATION doesn’t look at distributed storage in PC oriented environments but focuses on Consumer (Electronics) environments. This narrows the broad scope a lot. Within these Consumer Electronics environments the graduation assignment is limited to content navigation. Thus, the scope of the graduation assignment is reduced to:

**Retrieval of distributed stored (multimedia) content in Consumer Electronics home networks**

In the graduation assignment the requirements should be investigated of a home network system that makes it easy to develop applications that support the user in dealing with digital multimedia content that is distributed through the house. In such a system it’s very important to have a network middleware solution. Without such a
middleware layer, solutions would need to be reinvented for each device and application. Moreover, systems from different vendors would not be able to communicate with each other.

In Figure 1 an overview of network stacks is given. On the left the OSI and TCP/IP protocol stack are shown. Compared to the OSI stack the TCP/IP stack is simpler, i.e. the presentation and session layer are not used. Originally the OSI and TCP/IP application layer consisted of protocols like TELNET, FTP, etc. Currently applications moved to a higher level and make use of an extra layer called middleware. The right side of the picture makes clear that middleware in this case is built on top of the used network stack and includes functionality that applications have in common. Examples of applications that can be placed on top of the middleware layer are video streaming, browsing remote databases, or showing a user interface on any display in the house. Functions that middleware stacks typically take care of are for example, easily adding of new devices with new services. Depending on the combination of the physical network connection, the network stack and the middleware stack, the home network can handle adding of this new devices and services.

<table>
<thead>
<tr>
<th>OSI model</th>
<th>TCP/IP model</th>
<th>Middleware model</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Application</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Presentation</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Session</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Transport</td>
<td>Transport (TCP/UDP)</td>
</tr>
<tr>
<td>3</td>
<td>Network</td>
<td>Internet (IP)</td>
</tr>
<tr>
<td>2</td>
<td>Data link</td>
<td>Host-to-network (Ethernet)</td>
</tr>
<tr>
<td>1</td>
<td>Physical</td>
<td></td>
</tr>
</tbody>
</table>

Figure 1: Comparison between the OSI model and the TCP/IP reference model.

In the SPATION case the home network consists of Consumer Electronics devices, which poses additional boundary conditions caused by the difference in processing power and display capabilities between devices. When with these devices a distributed system is built, four goals should be met to make building a distributed system worth the effort. A distributed system should:

- Easily connect users to resources
- Hide the fact that resources are distributed across a network
- Be open
- Be scalable
To meet these four goals, the SPATION project made the decision to focus on UPnP as a middleware stack. The goals of UPnP are to allow devices to connect seamlessly and to simplify the implementation of networks in the home and corporate environments. UPnP achieves this by defining and publishing UPnP device control protocols built upon open, Internet-based communication standards.

The main reasons for choosing UPnP as middleware standard for the SPATION system are:

- It is an open standard based on widely known Internet technology.
- It offers a good balance between standardization and implementation freedom.
- It has relatively low implementation complexity and cost.

UPnP is a good trade-off between the ideal middleware standard and a practical implementation that is based on existing protocols. Moreover, UPnP is the only middleware standard likely to be adopted by CE vendors. The argument that UPnP has the biggest chance of being successful in the market outweighs technical arguments. UPnP is far from being perfect. Some aspects are well thought through and are improvements over other standards. For some aspects other standards offer more advanced solutions.

In this chapter it became clear that there are several important aspects where the graduation assignment should focus on. The most important aspects are: distribution of storage, heterogeneity, scalability and transparency. By addressing these important aspects, insights where gained about the boundary conditions of the graduation assignment. These operation boundaries bring focus in the graduation assignment and help to solve the main problem of how to deal with distributed storage in a heterogeneous home network.
3. Analyzing user benefits and requirements

The user benefits and user requirements should be analyzed to get insights what users expect when they access content in a home network of connected devices. This analysis has been conducted on the basis of the application scenarios used in the SPATION project.

3.1. The SPATION project

SPATION (Services Platforms and Applications for Transparent Information management in an in-hOme Network) is a European project, which focuses on the technologies, infrastructure and user interfaces for networked home environments. The objective is to find novel solutions for content management tasks such as moving, organizing and retrieving information in a heterogeneous home system. This heterogeneous home system consists of a number of interconnected stationary devices and various handheld devices; all these devices communicate with each other and have their own storage capacity that allows storing large amounts of content.

The SPATION project investigates the requirements of a home network system that makes it easy to develop applications that support the user in dealing with digital content that is distributed through the house. In such a system a network middleware solution is important. Without this middleware layer, solutions would be reinvented for each device and application. Common tasks in network middleware stacks therefore are;

- Assignment of addresses
- Announcement of presence of a device
- Giving devices access to functions of other devices
- Standardizing commands to allow control of (future) devices
- Transporting user interfaces over the network.
- Allowing devices to share content
- Allowing devices to stream content

Various middleware stacks are currently available, offering most of the above-mentioned solutions. However, there are major differences in how they handle certain problems. For example, the middleware stack JINI [7] supports control of devices over the network by sending code that will create a user interface of the controlled device on the controlling device. UPnP handles control of devices by announcing services in the network. In this case a controlling device, contacts the device it wants to control, requests a list of services, and asks the controlled device for an action on a specific service. As explained in chapter 2 SPATION made the decision to focus on UPnP because the assumption is that UPnP has the biggest chance of being successful in the market.
Scenario: A New Personal Home Recorder

Dad enters the house with a newly bought Personal Video Recorder. The old personal video recorder just allowed him to record, watch and pause programs. The new DVD hard-disk recorder combo has a bigger storage capacity and has more features. He bought it because it allowed him to record TV programs on a DVD such that he can make his own archive.

When he connects the new Personal Video Recorder, automatically all relevant information from the network such as personal preferences and content are found.

The children will use the old personal video recorder now. He puts it upstairs in the bedroom of the eldest child. The new device has an option to automatically create a summary of stored content to quickly check the content of a program. The children were already familiar with the old PVR in the living room, but they quickly discovered that the old PVR also could present summaries now the new device is present.

Dad starts playing with his new device and discovers that the new device shows an improved overview of all the content. It even allows him to see what is stored in other devices and shows what movie the device of the children is currently playing.

Mom is playing with the system too; she's using the handheld to access it. She discovers that it's easy to watch a summary on the handheld. She finds a nice movie and they decide to watch it together.

Figure 2: Example of a SPATION scenario.
3.2. Application scenarios

Within the SPATION project six user scenarios were written to get a better understanding of issues and requirements related to the problem of dealing with distributed content in the home. In Figure 2 an abbreviated version of one of these scenarios is shown. The main focus of this scenario is on the use of a personal video recorder to manage content, and in particular to offer a new video summarization function as a service to other devices.

From this scenario, it becomes clear how a user wants to transparently use the applications and resources over the network. In practice this means that the user, while sitting on the couch, can use a handheld device to control devices in the home. The handheld in this case is used as a combination between the lean-backward way of watching television and the lean-forward way of using a PC. Using the handheld for these types of system interaction, users can search and browse content, but also simple annotation and editing are possible. An example of such a handheld (advanced remote control and internet browser) is the Philips iPronto (see Figure 5). This device is used in the SPATION project to control all devices in the home network. In the project additional software was developed to support browsing video content stored in the home network using the iPronto. When the user wants to watch a particular program, found during browsing, the user selects it and selects the device where she/he wants to display it. The handheld is also used for video browsing functions like a key-frame browser (see section 3.4.2), a video preview (see section 3.4.3) and a pictorial overview of a video (see section 3.4.4). Metadata such as key-frames are automatically generated on more powerful devices such as PVR's connected to the network. Because a PVR has sufficient processing power, it can support the content analysis algorithms used to enable the video browsing functions on the handheld device. In the next section the developed applications that allow users to use these previously described video browse functions are described.

3.3. SPATION system setup

From the previous sections the following system requirements can be summarized:

The assumption is that the home system will consist of multiple devices that are interconnected. There are multiple solutions available for this interconnection. This will lead to the situation that the network in the home will be a mixture of solutions, such as wired network connections, wireless network connections (802.11), bluetooth etc. A lot of these devices are equipped with hard-disk drives and thus have large storage capacity.

Typical devices that will be present in the house are:

- TVs that can receive content via a network interface.
- Audio devices that can playback audio via the network like portable audio players.
- Personal Digital recorders that can store large amounts of music, video and photographs.
- A handheld device that can be used to control other devices and search for content.
- A PC (is not required, music, video and photographs can be enjoyed without it).
Combining these devices will create a network that allows users to control devices in the house and search for content stored on any of these devices while sitting on the couch with a handheld device.

A prototype of the system described above is built (see Figure 3). This system will allow the validation of the UPnP standard for its use in Consumer Electronics based home networks as opposed to a PC centric solution. Additionally, this prototype system will allow the investigation what is needed to create a system that can be managed by end users, and which features are important to users.

![Figure 3: SPATION system: Streamium, iPronto, PC, and TV.](image)

For creating such a prototype system a lot of software development needs to take place. To reduce the development time open software solutions are used where possible because it can be easily extended. Common PC’s are used to develop prototypes of the devices. All devices run the open source Linux operating system and an implementation of a UPnP stack. Each device hosts an implementation of a UPnP root device (see chapter 4) for the basic UPnP functions.

The network is a combination of wired and wireless network connections, and provides the possibility to use IEEE1394, bluetooth and IRDA. A number of small PCs act as prototypes for Personal video recorders, TVs with storage, and audio jukeboxes. A connection to the Internet is available through the gateway, which shields the UPnP network from the Internet. The iPAQ and the Philips iPronto serve as handheld devices that allow controlling devices in the network using their touch screens. In Figure 4 an overview of the current prototype system is shown.
3.4. Content navigation

As described previously, access of stored content and control over the home system is made possible via handheld devices. An important aspect is the organization and retrieval of data in the home system. In the SPATION prototyping system users should be allowed to control devices and quickly access stored content. The assumption is that when a user picks up the handheld from the table she/he already decided what to do. For example, the user intends to listen to music, to watch a movie or to turn down the volume of the TV. Therefore, the main screen provides access to music, photos, video and the devices separately, instead of providing an integrated browser (Figure 5).

Pressing the device button gives users access to all devices in the home. This screen also provides the control of all the available devices like setting the volume of the TV. To allow users to quickly search for a video, the following functionality is provided:

- View a list of all available video content on all devices.
- Browse a video based on automatically selected key-frames.
- Watch a pictorial overview of a video (automatically generated).
- Watch a short preview of a video (automatically generated).
3.4.1. Video browser

In Figure 6 the video browser screen that gives access to these functions is shown. The list shows all video content available on all devices. It is not necessary for a user to, for example select PVR1 to see its content, and then PVR2 etc. When the user wants to playback a movie the system will stream the content from the storage device to a display, hence the user does not even need to know where a particular program is stored.

Figure 6: Overview of stored video programs.
3.4.2. Key-frame browsing

The Browse button gives access to browse a particular piece of stored content within the home network (e.g. a recording of a TV show). Using video content analysis, the personal video recorder generates automatically content descriptors that can be used to enable content-based browsing and retrieval functionality. For the user this means that she/he can browse through thumbnails of key-frames representing meaningful scenes of a selected program on a handheld device. This key-frame selection is based on clustering the results of a simple but effective shot-cut detection algorithm that runs during video recording (see Figure 7).

![Figure 7: Video browser interface.](image)

3.4.3. Video preview

When a large amount of video content is available it is difficult for a user to find something interesting. A method that helps the users to decide whether a certain recorded program is interesting enough to watch is provided by automatically generated video previews [5].

A video preview (or video skim) is a short video clip, for example 30 seconds, which provides an overview of a recorded program. During recording, content analysis algorithms compute low-level descriptions of the content such as scene boundaries, scene complexity, motion activity, human presence, audio volume, etc. When the recording is complete and all the content descriptions are available, a number of heuristic rules are applied to select the scenes that are most relevant for users that want to preview the content without actually spoiling it. Examples of such rules are non-disclosure of the end of a program, inclusion of action scenes and dialogues, etc.

Concatenating the most relevant scenes generates video previews that convey key-aspects of a video program and helps users judge whether or not they would like to see the entire content.

The actual video preview is stored as metadata in the form of list of pointers to the parts of the original content that are included in the preview. Users can watch directly video previews on TV sets or stream transcoded versions to handheld devices.
3.4.4. Pictorial overview of a video

Using a pictorial overview of a recorded video program, users can quickly see what a program is about, remember whether she or he has seen it before or decide whether or not to watch it or to download it to a portable device.

The pictorial overview is a set of the 15 most relevant pictures extracted from the original content. An automatic summarization algorithm [5][6] selects a set of images that best represents a particular piece of content. An example of a video preview of a home video is shown in Figure 8.

![Figure 8: Pictorial overview of a program.](image)

3.5. User benefits and requirements

As explained, in the future users need to deal with large amounts of distributed content in the home. To provide users an easy way to access this distributed content, users need to have one central place where they can access all their content like the video browser explained in chapter 3.4.1. In this way users can benefit from the fact that Consumer Electronics devices, PCs and handheld devices become interconnected. Other facts are that future Consumer Electronics devices will have large processing power and will have their own storage capacity that allows storing large amounts of content. To make the combination of these facts advantageous for users the following user requirements should be considered in detail:

1. Distributed content should be accessed at one central place instead of searching every device separately to find the content of interest.
2. Information should be obtained about content stored on devices even if the device is not accessible.
3. When content is browsed and/or searched responses should be returned quickly.

3.5.1. Access content in one central place

Users don’t want to be bothered with the need to search every device separately to find the content of interest but want to have one central place where they can access their content. The user can access in this central place all the content that is distributed
over a lot of devices in the home. This central place should shield the actual storage locations for the user. When the access to the content is centralized and the real storage location of the content is shielded, users get the idea that they access one virtual storage place.

To make sure that all the content in the home network can be accessed in this virtual storage place, searching for content descriptions is needed because the content can be stored at different locations in the network. These content descriptions usually referred to as metadata, need to be combined, stored and retrieved. This prevents that users need to interact directly with the device containing the content.

3.5.2. Obtain information about stored content

Users want to know which content is stored in the complete home network. When at a certain moment content is not accessible, users still want to know that this content exists. There are several examples when accessing of content is not possible, e.g. a Consumer Electronics device or PC can be turned off or a mobile device can be outside the home. Moreover, the user even wants to access the content of their CD or DVD collection stored in e.g. a rack in the living room. It should be taken into account that when data is not accessible, the system, in some way, must make clear to users that content is temporarily not accessible. Users can also connect new devices with storage to the network. These newly connected devices contain content, which also should become accessible for users.

3.5.3. Quickly access content of interest

For browsing and searching content quick response times are important. In a distributed system particularly there is the need to take care that not too many delays are introduced. Though not direct visible architectural choices of the system implementation have a large impact on response times. For example central access to content descriptions is likely to be faster than accessing each device separately.
4. UPnP Basics

As stated in chapter 2 without a standardized middleware stack every application needs to solve the same problems over and over again. Moreover, systems from different vendors would not be able to communicate with each other. A middleware stack takes care that e.g. adding new devices with new services is easy because this is standardized. How well a home network can handle the adding of new devices and services depends on the combination of the physical network connection, the network stack and the middleware stack. In the SPATION case the home network consists of CE devices, which poses additional boundary conditions caused by the difference in processing power and display capabilities between devices.

The UPnP Forum [8] is, as stated on their website http://www.upnp.org, a group of companies and individuals across multiple industries that intend to play a leading role in the authoring of specifications for UPnP devices and services. Formed in June 1999, the Forum is a non-profit association of more than 592 consumer electronics, computing, home automation, home security, appliances, printing, photography, computer networking, mobile products and other leading companies working together in an open process to design schema and protocol standards for the UPnP initiative.

The goals of the Forum are to allow devices to connect seamlessly and to simplify the implementation of networks in the home and corporate environments. The Forum will achieve this by defining and publishing UPnP device control protocols built upon open, Internet-based communication standards.

The aim of UPnP is to create a plug and play network where no configuration is needed; in this way users do not need to have any network knowledge. Just plug in your device and use it. The device tells the network what it can do and it learns from the other available devices what they can do. This makes it is possible for devices to use each other’s functionality. As easy as it is to plug in a device, it can also be removed from the network without causing major trouble in the network.

The SPATION project investigated whether UPnP can serve as a basis in a system that allows users to easily search, browse and manage content. As already stated in Chapter 2, the main reasons for choosing UPnP as middleware standard for the SPATION system are:

- It is an open standard based on widely known Internet technology.
- It offers a good balance between standardization and implementation freedom.
- It has relatively low implementation complexity and cost.

The opinion in the project is that UPnP is a good trade-off between the ideal middleware standard and a practical implementation that is based on existing protocols. Moreover, UPnP is the only middleware standard likely to be adopted by CE vendors.
UPnP defines a way for multiple machines to interact with each other over a network. It is a stack of protocols that starts at the IP level, where it defines how to receive an IP address. Most of UPnP is based on existing standards such as TCP/IP, HTTP, XML, SSDP, SOAP etc. In a UPnP based network two main device categories are distinguished, devices and control points. A control point can be implemented on a device, such as a handheld device, from where control needs to take place. The control point can search for devices in the network that it can control. The control point might look for a certain type of device, such as a CD player. When it finds a device it is interested in, it can query which services the device offers. One of these services could be a rendering control service, which contains functions like setting the volume, muting the sound etc. The control point could query the current volume setting and display it in the user interface. When the user changes the volume, the control point can send a command to the device to set the volume to the new value. When multiple control points are used, these control points can reflect the change in their user interfaces, since UPnP allows to send event notifications for this purpose. The control point coordinates these devices using UPnP *action commands* to initialise, configure and to make devices ready to transfer content from amongst each other. The devices do not interact directly with each other and use a non-UPnP "out-of-band" communication protocol. The control point is not directly involved in the actual transfer of the content. In Figure 9 an example of UPnP interaction between standardized devices is shown. The control point initiates the transfer from *MediaServer* acting as a source to a *MediaRenderer* acting as a sink.

![Diagram](image)

*Figure 9: Interaction between devices in UPnP.*

The previous small explanation introduces the basic functionality of UPnP. In the following sections each aspect will be described in detail.

### 4.1. UPnP Architecture

UPnP is set up in different levels, stacked on top of each other. All levels make use of existing technology as far as possible (Figure 10). The levels start at the network level, where it defines how to get an IP address. From there UPnP defines how to
discover devices and services, then how to describe services and how to interact with them. The UPnP architecture specifies six interaction phases as follows:

- **Addressing**, devices and control points obtain a IP addresses using DHCP (Dynamic Host Configuration Protocol) or AUTO-IP to participate in the network.
- **Discovery**, when a device is connected to the network it sends an advertisement to inform the control points its availability in the network. When a control point is connected to the network it can search for devices of interest. Both use SSDP (Simple Service Discovery Protocol).
- **Description**, control points learn details about devices and their services using HTTP.
- **Control**, control points invoke service actions on devices using HTTP and SOAP (Simple Object Access Protocol).
- **Eventing**, devices notify control points of changes in their state using HTTP and GENA (General Event Notification Architecture).
- **Presentation**, Control points control devices and/or view device status using an HTML UI.

![Figure 10: UPnP protocol stack.](image)

4.1.1. **Addressing**

The first phase that starts is the addressing phase. It starts when a new device or control point is connected to a home network. This addressing phase takes care that the new device or control point gets an IP-address by using DHCP or AUTO-IP. DHCP is a protocol that is widely used in business networks. In that case one DHCP server is present in the network. This server hands out IP addresses. When a device contacts the DHCP server, the server looks in its list for the next free IP address. It returns the IP address possibly along with other network settings. In home networks this solution is used less frequently, since it requires a central service that should always be available. It could, however, be implemented in a gateway or broadband cable modem. When the DHCP server is not found UPnP makes use of AUTO-IP. The principle of AUTO-IP is to choose an IP-address randomly. The device that
wants to access the network uses its AUTO-IP algorithm to check whether there are any devices in the network that react to that address, i.e. it is already in use by another device. When the address is already used, a new IP-address is tested till one is found that is not already in use. When users currently connect 2 or 3 PCs at home they often assign IP-addresses manually. This option is not desired in a home network with many CE-devices, in that case users should not be aware of the fact that network addresses exist.

After the addressing phase all the devices and control points available in the network have an IP-address and can participate in the network.

4.1.2. Discovery

When IP-addresses have been assigned, devices can announce their presence on the network. When a device is connected to the network (e.g. when it is turned on), it sends a multicast advertisement. To ensure that all advertisements are received properly they can be sent several times since multicast based on UDP is not guaranteed to arrive. An advertisement provides a device name as well as a link to additional information, such as embedded devices and services. Discovery advertisement messages have limited lifetime. Therefore, the advertisement needs to be re-sent before the lifetime of the advertisement expires. The advertisements are based on existing protocols such as the so-called SSDP (see [9]) and a multicast HTTP variant that has been extended using GENA NOTIFY (see [10]) message.

When a device wants to leave the network it should send a message to explicitly tell the network it is going away before going offline. The device sends a “bye bye” message to the multicast address in a way similar to the advertisement messages. When a device does not send such a message before leaving the network, for example, when the user disconnects the network cable, the absence needs to be detected by waiting till the advertisement messages are expired. Since advertisement lifetimes are generally quite long to prevent the system from flooding the network with these messages, it can take quite a while before the absence of a device is detected.

In UPnP the devices will not monitor each other’s advertisements, i.e. devices are not aware of each other. Control points, however, can detect the presence of devices, and control these devices. When content is streamed from a server device to a device that can render the content, the control point is needed to setup this transaction.

When a control point is connected to the network it searches for devices of interest also using the discover protocol based on SSDP. The control point sends a SSDP M-SEARCH message containing the type of device or service of interest to a standard multicast address. When a device is of the right type or has a service of interest it responds to the control point’s IP-address (direct unicast) with a message containing the discovery advertisement information. When a control point has located a number of devices it is interested in, it can request further information. This is referred to as the description phase.
4.1.3. Description

After a control point has discovered devices of interest the description phase starts. In this phase the control point requests additional information about the device using HTTP-GET on the URL obtained from the discovery message. Similarly the control point can request information about services a device offers by using the service description URL from the discovery message. The device or the device service reply with a device description and/or service descriptions (in XML format) in the body of an HTTP response.

After the description phase, the control point can start one of the three following phases: control, eventing or presentation.

4.1.4. Control

In the control phase the control point sends a control message (in XML using SOAP). The message is sent via HTTP requests using the control URL for the service (provided in the device description). The device service responds to the action with specific values or fault codes. The response is sent via HTTP. A control message can be compared to a remote procedure call. The requested action is sent to the device, the device performs the action, and informs the control point of its result. Typically the actions are initiated by the user from a user interface. Details about specific services and actions that are specific for AV devices will be presented in section (4.2).

4.1.5. Eventing

When, for example, multiple control points can change the volume setting of a certain audio device it would be nice if the current volume setting is displayed properly on all control points. To implement this behavior all control points need to subscribe to the service that contains the volume setting. When one control point changes the volume setting, all control points will be notified when the volume is changed. This notification mechanism is called eventing in UPnP. The eventing phase consists of two parts, the subscription to certain events, and the reception of event notifications.

The subscription consists of an initial subscription and a “keep subscription active” phase. During the initialization phase the control point sends a subscription message, containing a delivery URL of the control point, to the subscription URL that was specified in the device description. To confirm the subscription, the device service sends the accepted subscription to the specified URL. (The control point needs an HTTP server for this). This includes: a unique subscription identifier and the duration of the subscription. In the “keep subscription active” phase the control point sends a subscription message including a subscription identifier to the device before expiration.

When an event occurs, i.e. when a state variable changes, notifications are sent to the control points that subscribed to that event. The event message (in XML) contains the names of one or more state variables. The implementation is based on GENA NOTIFY messages that are sent using HTTP. When the control point receives such a
message it replies with an HTTP OK to acknowledge that the event message was received correctly. This mechanism also supports multiple control points. Note that not all variables return an event, the UPnP standard describes which variables are evented and which not. All evented variables of a certain service are sent when one of them changes.

4.1.6. Presentation

Devices can be controlled by implementing a graphical user interface in the control point. In this case different implementations need to be made for different devices. Alternatively a device can present an HTML page with information and controls of the device. In this way any control point can control any device that presents its controls using an HTML page. The drawback, however, is that different devices in the home can have a completely different user interface. When a handheld device is used to control devices in the house, the look and feel of the user interface can vary drastically depending on the devices in the home.

4.2. UPnP devices and services

In the previous sections it was already mentioned that UPnP distinguishes between devices and control points. Control points are used to control devices and are always in control, devices cannot directly communicate to each other without an action from a control point. The following sections describe in more detail how devices can be further subdivided and which services these devices can offer. In UPnP, devices such as printers, scanners and AV devices are standardized. In the remainder of this section the focus will be on the AV devices.

The AV devices are divided into media servers and media renderers. The servers supply content over the network to renderers such as TVs, CD players etc. It is important to note that a UPnP device does not need to match exactly with a physical device. Often physical devices are described as multiple UPnP devices. A TV, for example, could be a pure media renderer device. The TV might have a digital input, such as network connection or IEEE1394 that is connected to the PC. The PC in that case should implement the media server. When the TV has storage (e.g. a hard disk), it can implement a media renderer device as well as a media server device, to also allow other devices in the network to make use of the content stored (on the hard disk) in the TV. The control point can be implemented on the PC, the TV, or a third device (for example a handheld). Multiple control points are possible as well. When a control point is implemented on the PC, it allows the user to select a video file to play on the TV, change the volume, the brightness etc. When a control point is implemented in the TV, it allows the user to select e.g. a video file from the PC using the TVs normal remote control selecting from a list of titles that are displayed via an on screen display.

Each UPnP device can offer an arbitrary number of services. These services offer one or more functions. Functions are associated to so-called state variables, and most functions are intended to set or get the value of a state variable. To structure the
The amount of services a UPnP device offers, a device can contain embedded devices with their services. The device containing the embedded devices is called the root device.

For control points it is possible to search for services on the network, allowing them to search for a specific service. Additionally, control points can subscribe to receive event notification when a certain state-variable changes. Figure 11 shows the possibilities of structuring UPnP functions in the network.

![Figure 11: Structure of UPnP.](image)

### 4.2.1. Media renderer device

The Media renderer device is a standardized UPnP device (see [11]). It supports exchange of information of file formats and protocols to negotiate with a server how content should be transferred. It offers functions to alter rendering characteristics and it offers functions to control the flow of content. Three services with accompanying functions and state-variables have been standardized to support this functionality.

These services are called:
- Rendering Control
- Connection Manager
- AVTransport Service (optional)

**Rendering Control:**

The rendering control (see [12]) offers functions to control how content is rendered. This includes functions such as changing the volume, checking the current volume setting, muting the audio, adapting contrast and brightness etc. In total a list of 35 functions have been described in the media renderer standard. These functions are associated with state variables, and in general the functions can be divided into functions to set values, and functions to get values.
One state-variable called “last change” reflects the last change of any of the other state-variables. When the user changes the volume of a device the last-change state variable will describe in XML the name of the variable that changed along with its current properties. An example is shown in Figure 12 (The instance ID is an identifier that supports having multiple instances of the rendering control service. E.g. when a device has multiple audio sources of which the settings can be modified. The main control is given ID 0)

```
<InstanceID val=0>
  <Volume channel="Left" val= 24 />
</InstanceID>
```

Figure 12: Example of last change state-variable.

The “last change” state variable is the only variable in the Rendering Control service that can generate events. When multiple control points allow the user to change, e.g. the audio volume, control points can subscribe to the “last change”-variable events to reflect the current volume setting in the user interface.

**Connection Manager:**

The Connection manager service (see [13]) is responsible for setting up and destroying connections between devices. When a control point wants to setup a connection between a media server and a media renderer, it searches for the connection managers of the server and renderer. One of the functions of the connection manager is to supply a list of supported protocols and formats. The control point needs to match the protocols and formats of the server and the renderer with each other and select an appropriate one. Then it can inform each device, and from thereon the server and the renderer can send and receive data without any intervention from a control point. Additionally the connection manager service offers an action to allow a control point to terminate a connection and free the used resources.

Evented variables of the connection manager allow control points to be notified when the list of supported protocols changes (perhaps a user plugs in an IEEE1394 cable) and when the current number of active connections changes.

**AV Transport Service:**

The AV-Transport Service (see [14]) enables control over the transport of audio and video streams. It is used to control streams from CD players, mp3 players, DVD recorders etc. It includes controls such as starting, pausing, and stopping playback, and setting the playback speed. Additionally, it allows for requesting information about the medium type, the number of tracks, current track, etc.

For eventing in the AV transport Service, a “last-change” state variable exists. Similarly to the case of the connection manager, control points can use this state variable to be notified when, for example, the current track changes.
4.2.2. Media Server device

The media server standardizes how devices like CD players, DVD players, mp3 servers, Personal video recorders etc. acting as media servers in the network can be addressed (see [15]). Additionally, the PC can also implement this service allowing the PC to act as a service that can supply content to, for example, a TV or an mp3 player.

The media server can contain three services:

- Content Directory Service
- ConnectionManager
- AVTransport Service (optional)

The connection manager and the AV Transport Service have been described in the previous section. The content directory service makes it possible to search for content. The content directory service is basically a service, which provides various views of the stored content. Using detailed descriptions of the content a search query can return a set of content items. Additionally, organization of content is supported via containers, which can be used to cluster content similar to directories or folders. The content directory service is one of the most important services in the SPATION home network, where most consumer electronics devices offer a substantial storage capacity. Therefore the Content Directory Service and its metadata descriptions is described in more detail in the next section.

4.3. The Content Directory Service

The Content Directory Service provides standard functionality to search for content stored on a media server (see [16]). The primary actions that the content directory service provides are browse and search. These actions allow control points to obtain detailed information about content. This information includes metadata such as name, artist, date created, file size, etc. Stored content is represented by the content directory service as objects that are divided into classes of different types of content.

To improve interoperability a number of objects have been standardized. Two main classes of objects are distinguished; items and containers. Items represent a single piece of data while containers represent collections of objects. As shown in Figure 13 items are divided into classes like audioItem and videoItem, which in their turn are divided into classes like photo, musicTrack, and movie. It is allowed to add vendor specific subclasses. Similarly to items, several classes of containers have been defined as well (see Figure 14).

For each class of objects a set of required and optional properties have been defined. The object class defines properties like ID, parentID, title, creator etc. Each subclass inherits these properties and adds a set of specific properties. This means that, for example, for a musicTrack the properties of the parent classes audioItem, item and object apply as well. The properties are based on definitions from the UPnP forum, the Dublin core [17], and DIDL-lite namespace. (DIDL-Lite is derived from a subset
of DIDL, a Digital Item Declaration Language developed within ISO/MPEG21 (see [18]). Additionally it is allowed to specify additional properties using any XML namespace.

![Diagram of Item definition](image1)

![Diagram of Container definition](image2)

4.3.1. Browsing and Searching

There are two ways of accessing the content stored in the Content Directory Service, by a browse and a search query. A browse allows the caller to incrementally browse the hierarchy structure of the objects exposed by the Content Directory Service, including information listing the classes of objects available in any particular object container. A browse action typically starts with requesting meta-data of the root structure that is exposed by the Content Directory Service. Next descriptions of the items and containers of the root object can be requested. This does not include contents of any of the containers. These containers can be browsed by using the browse command with the identifier of any of these returned containers as a parameter. Consider the following example; the root directory contains two
storageFolder containers called My_Music with identifier “1” and My_Pictures with identifier “2”. (A storageFolder is a subclass of container that is used to describe folders on a storage volume). To browse the content of the My_Music storageFolder the identifier “1” should be passed as a parameter of the browse command. The result of this browse action will be the content of the My_Music storageFolder container that consists of an additional container with identifier “3” and an item with identifier “4”. The container with identifier “3” is a musicAlbum which contains three musicTracks (childCount="3"). Figure 15 shows the logical directory hierarchy and the browse request and response.

Logical directory hierarchy:
- Name="Content"
  - Name="My_Music"
    - Name="Brand New Day"
      - Name="A Thousand Years – Sting.wma", Size="100000"
      - Name="Desert Rose – Sting.wma", Size="50000"
      - Name="Big Lie Small World – Sting.mp3", Size="80000"
    - Name="State Of Love and Trust – Pearl Jam.wma", Size="70000"
  - Name="My_Pictures"

Request: Browse("1", "BrowseDirectChildren", "*", 0, "")

Response: Browse(
  "<DIDL-Lite xmlns:dc='http://purl.org/dc/elements/1.1/'"
  xmlns:upnp='urn:schemas-upnp-org:metadata-1-0/upnp/'
  xmlns='urn:schemas-upnp-org:metadata-1-0/DIDL-Lite'>
  <container id="3" parentID="1" childCount="3" restricted="false">
    <dc:title>Brand New Day</dc:title>
    <dc:creator>Sting</dc:creator>
    <upnp:class>object.container.album.musicAlbum</upnp:class>
    <upnp:searchClass includeDerived="false">
      <upnp:class>object.item.audioltem.musicTrack</upnp:class>
    </upnp:searchClass>
    <upnp:createClass includeDerived="false">
      <upnp:class>object.item.audioltem.musicTrack</upnp:class>
    </upnp:createClass>
  </container>
  <item id="4" parentID="1" restricted="false">
    <dc:title>State Of Love And Trust</dc:title>
    <dc:creator>Pearl Jam</dc:creator>
    <upnp:class>object.item.audioltem.musicTrack</upnp:class>
    <res protocolInfo="http-get::audio/x-ms-wma:" size="70000">
      http://10.0.0.1/getcontent.asp?id=5
    </res>
  </item>
</DIDL-Lite>"), 2, 2, 21)

Figure 15: Browse request and response.
Apart from exposing a folder-based representation, the Content Directory Service can also expose additional views. The root could, for example, contain several musicGenre containers that allow browsing the music based on genre even though this structure is not physically present on the hard disk. This is possible because multiple containers can contain the same single item.

Additional functionality supplied by the browse command is to filter or sort the result. For example, the objects in the My_Music folder might be sorted based on their title. The UPnP standard allows for sorting based on any object property, however, not all Content Directory Service implementations will support this because not all object properties are required. Using the getSortCapabilities command the Content Directory Service will return the list of object properties for which it supports sorting. Filtering means restricting the output of a browse command to object properties of interest. For example, if the user likes to know the title of the musicTrack objects, but not the genre this can be indicated. The filter is a list of all object properties that are interesting. To obtain all available meta-data "*" can be used. When properties that are required by the standard are omitted from the filter list they are added automatically. I.e. the result will always contain the object properties that are required by the standard. Only optional object properties can be filtered.

A search, which is an optional action, allows the control point to search the content directory for objects that match some search criteria. The search criteria are specified as a query string operating on properties with comparison and logical operators. For example, after the query Search("O", "dc:creator = "Sting"", "+dc:title") the Content Directory Service returns all content created by Sting and sorts the result in ascending order by title. To let the control point know which searching capabilities are supported by the device, the getSearchCapabilities action can be used. This action returns a list of property names that can be used in the search queries.

4.3.2. Object creation and file transfer

In addition to browsing and searching content, the Content Directory Service can also be used to add new content, delete content, move content etc. Using CreateObject the Content Directory Service can store the metadata for a new object. The real data such as an MP3 file is not passed via the CreateObject command, instead a link is specified where the content can be obtained. In this case it is even possible to create objects in the Content Directory Service that refer to content on the Internet. This link can be used to listen to music from an Internet Radio station. By querying the Content Directory Service the control point can show a list of available content including live content. When the user selects an Internet Radio station, the control point will pass this link to the media renderer device. Normally content is stored in the Content Directory Service. In this case the content needs to be sent to the Content Directory Service, or retrieved by the Content Directory Service. This can be done using the importResource and exportResource commands. Using these commands a control point can submit content to a Content Directory Service or initiate transfer of content from one Content Directory Service to another. When content is being transferred a control point can ask which transfers are in progress and make inquiries about the status of a transfer. The GetTransferProgress command returns the total size that is to be transferred, as well as the currently-transferred amount. By using StopTransferResource the transfer of content can be stopped.
It is important to note that since the Content Directory Service manages both the storage of content and the representation of the content during search and browsing, any mapping can be used between these two. I.e., the way content is stored on a hard disk does not need to be the same as the way it is represented during browsing. In an extreme case all content could be stored in just one directory, while the Content Directory Service presents a nice structure of music albums and photo albums to the outside world.

4.4. UPnP system set-up examples

In Figure 16 three devices are shown, a PC, a Streamium and an iPronto. The Streamium is an audio system with mp3 playback option and a network connection. The iPronto is a handheld device that can be used as a remote control and Internet browser. A UPnP implementation on each device will allow communication between these devices.

On the PC a Media Server device can be running, to allow streaming of mp3 songs to the Streamium. The Streamium should implement a media renderer. A nice graphical user interface can be displayed on the iPronto, which should implement the control point. The iPronto, in that case, will search the network for media renderers, or more specifically for the Streamium. Additionally, it needs to search for a media server. When it finds a media server, it will allow the user to search for interesting content. The control point queries the media server’s content directory service, and displays the results in an appealing way to the user. The user can then select a number of songs to listen to. Next, the control point sets up a connection between the PC and the Streamium by first identifying a common protocol by calling the GetProtocolInfo functions from both devices. When the connection has been set up, playback can be controlled via an AV transport service. The AV Transport service allows the control point to control playback. This way functions like play, pause, stop, next song, jump to time index, etc. can be offered to the user.
Many extensions and variations to UPnP software setup are possible:

- The PC could implement a control point as well. This allows the user to select songs and direct them to the Streamium.
- The Streamium mp3 player can also implement a control point. Just pressing the “next”-button on the device will instruct the PC to jump to the next song.
- When multiple control points are used at the same time, each can reflect the current state by subscribing to the appropriate events. When the user presses the “next”-button on the Streamium mp3 player, the iPronto can immediately change the current song title in its user interface.
- The Streamium could also implement a media server, allowing clients to receive content from the current CD in the Streamium.
- A media renderer implementation on the iPronto can allow the user to listen to music streamed wirelessly from either the PC server, or the Streamium (if it implements a server). In the user interface on the iPronto these options need to be presented.
- The Streamium can supply an HTML page to allow controlling it using an arbitrary Internet browser. This could, for example, offer the user the possibility to change settings like audio volume. More advanced control requires the presence of a (simple) control point, for example, to discover servers and renderers, to find content, and to setup a connection.

The first mentioned extension could be seen as follows: The PC can implement a control point and a media server at the same time. On top of the control point a graphical PC application can run. The media server also implemented on the PC provides the available songs on the PC. In the graphical PC application the user can select these songs and if the user wants to listen to a song on the Streamium, the integrated control point in the PC sets up the communication between the PC and the Streamium. In this way the mentioned extension can be used to allow the user to select in the graphical PC application songs and direct them to the Streamium (See Figure 17).

![Figure 17: Control point and Media Server in one physical device](image-url)
5. UPnP based distributed storage: The "Super" CDS

By defining the user requirements in chapter 3.5 it became clear that users want to: access their distributed content at one central place instead of searching every device separately, obtain information about content stored on devices even if the device is not accessible and have quick responses when content is browsed and/or searched. To offer the user a home system that can fulfill these requirements a metadata aggregator service is needed. The metadata aggregation service must provide a uniform mechanism to browse content on one server. The task of this server is to obtain detailed information about individual content stored on devices in the home network. There are several possible metadata aggregator approaches that need to be investigated theoretically. Based on these investigations the metadata aggregator requirements can be defined. Finally, a practical technical solution is introduced to build a metadata aggregator based on the UPnP Content Directory Service. This "Super" CDS can act as an aggregator of metadata for different UPnP Content Directory Services available in the home.

5.1. Metadata aggregator

To make it possible for users to: access content at one central place, obtain information about stored content, and get quick access to content, the overall concept of a metadata aggregator is introduced.

5.1.1. Overall concept

The metadata aggregator is a centralized place where aggregated metadata can be accessed (metadata service). This metadata is related to content distributed on different devices in the network. The metadata aggregator should be independent of the storage location of the content in the distributed home network.

Let's consider the two extremes of possible metadata aggregator implementations:

1. A metadata aggregator that has no storage at all.
2. A metadata aggregator that stores all metadata.

In the first case the metadata aggregator merely passes on queries to all devices in the network, asking devices which content they have available. Figure 18 shows this process. The user enters the queries in the user interface (UI), which passes the queries to the query distribution. This function distributes the query to all devices in the home network. The devices each query their database (DB) with the user query and return the query results. The query results can contain metadata or can be empty when the device doesn't contain the searched content. All the query results (including the empty query results) are passed on to the merge metadata function. The results of the user queries are combined and sorted in a central device in the network. This in contrast to the functions that get the metadata, which are located in separate devices in the network and thus operate de-centralized. Finally, the merged metadata is sent to the user interface.
There is a difference between the previous described searching for information on content and storing new content and its related metadata. The difference is that when new content and its related metadata are stored only the device on which the content will be stored needs to be contacted instead of the need to contact all devices. This procedure is shown in Figure 19. When the user wants to store content and its metadata, the user interface (UI) is responsible for storing the content to the hard-disk (HDD) of the selected device, and for storing the metadata in the database (DB). For example, a user wants to store a digital picture and its metadata (e.g. the date of creation, etc.) is at the same time stored in the database. To store the actual storage location of the stored content, a content reference is made in the database. The content reference is a URL that points to the exact storage location on the hard-disk.

Figure 18: User query on a metadata aggregator without storage

Figure 19: Storing content and its metadata on a metadata aggregator without storage
Let's now consider the second case where the metadata aggregator stores all metadata. In Figure 20 this process is shown. The user queries are passed from the user interface (UI) to device that provides the get metadata function. Since in this case information on all content is available it is not necessary to contact other devices. The get metadata function queries the aggregator database with the user query and passes the results from the database on to the user interface. Also in this case the query results can contain the queried metadata or can be empty when the database doesn't contain the user-queried metadata but this situation differs from the one described in the metadata aggregator that has no storage at all. In the situation that the metadata aggregator has no storage at all, the get metadata function of a device can send an empty query result to the merge metadata function, but that doesn't mean that the query result sent to the user interface is also empty. This is due to the fact that another device can contain the queried metadata and after merging the query results by the merge metadata function the final query result contains the queried metadata. When, in the second situation where the metadata aggregator stores all metadata, an empty query result is returned also an empty query result is delivered to the user interface.

![Figure 20: User query on a metadata aggregator that stores all metadata](image)

When the metadata aggregator that stores all metadata needs to store new content and its related metadata, the device on which the content will be stored needs to be contacted and the aggregator database needs to be informed. Figure 21 shows this procedure. Using the user interface (UI) the user can store content and its metadata. The user interface handles both the storing of the content on the hard-disk (HDD) of the selected device and the storing of the content related metadata in the database (DB). When the database is contacted to store the metadata also the aggregator database is informed. This aggregator database needs to store exactly the same metadata and content reference as the database. This content reference is again a URL that points to the exact storage location on the hard-disk of a device.
There are of coarse many intermediate cases. Two of these intermediate cases are shown in Figure 22 and Figure 23. The intermediate case shown in Figure 22 shows one of the possible places where a cache can be used. A cache is a store of recently used data objects that is closer than the objects themselves. Caches are only used to improve access times to data. The cache in this case can store the queries and their results to help to speed up the query answering by e.g. combining previous queries and query results etc. The main issue is to keep these caches up to date.
Also for the metadata aggregator that stores all metadata, the most important question that arises is: Can a consistent aggregator database be built and kept consistent? For example, the network plug of the device that provides the aggregator database is pulled out of the device. In this case the aggregator database isn’t informed about the metadata that is stored in a database on a device. When the network plug is put back the aggregator database should check if it is still consistent. One of the solutions that can be used is the use of a checksum per database. When the checksum of a database on a device is different from the checksum the aggregator database has of that device, something is changed on the device and the aggregator database isn’t consistent anymore. The aggregator database should take action to become consistent again. In the worst case e.g. when too much is changed; the aggregator database must completely reconstruct itself. For example, this reconstruction can be done by browsing and/or searching all metadata stored in the Content Directory Services available in the network. The previous explained intermediate case where the aggregator database uses a checksum to check if it is still consistent is shown in Figure 23.

![Figure 23: Solution to try to keep the aggregator database consistent](image)

To conclude, the metadata aggregator that has no storage at all can be seen as a decentralized storage solution and the metadata aggregator that stores all metadata can be seen as a centralized storage solution. The main challenge for the metadata aggregator that has no storage at all is to quickly deliver the merged metadata to the user when she/he initiates a query. For the metadata aggregator that stores all metadata the main challenge is to keep the aggregator database consistent. The ideal metadata aggregator is probably an intermediate case instead of one of the extreme cases. This depends for the most on: how easy it is to keep the metadata aggregator cache consistent, how worse it is if the cache isn’t consistent anymore and how much time is involved to reconstructed a consistent cache when a cache became inconsistent.
5.1.2. Metadata aggregator requirements

From the previous sections the following important requirements for the metadata aggregator can be written down:

- The metadata aggregator has to be consistent (what is shown is available and what is available is shown).
- The metadata aggregator may not slow down the user-system interaction. This means that the query results need to be assembled quickly. For example, when a new device is connected to the network or when the metadata aggregator needs to get up to date after it was disconnected.
- It should be possible to reconstruct the metadata aggregator e.g. after it was disconnected for a while.
- The metadata aggregator has to show all content in the network regardless if a device is available on the network or not.
- It is useful to move aggregation and sorting to a powerful device because this can take a lot of processing power.

5.2. "Practical" technical solution based on UPnP Content Directory Service

Now the requirements of a metadata aggregator are known they can be used as guidelines for the verification of the possible implementations of a metadata aggregator. For the implementation of a metadata aggregator the UPnP Content Directory Service is used. As explained in chapter 4.3, every media server device that contains content in a UPnP network has its own UPnP Content Directory Service. The Content Directory Service provides standard functionality to search for content stored on a media server device. A control point can contact the Content Directory Service of all media server devices in the network to obtain detailed information about its content. As explained in chapter 3.5.1, a user requirement is to access content in one central place. Thus, it must be prevented that a control point needs to interact separately with all the media server devices containing the content. In order to enable this, a service needs to provide a uniform mechanism for user interface devices to browse the content on one server that can obtain detailed information about individual content. The proposed solution is to upgrade the UPnP Content Directory Service to a "Super" CDS, which can act as an aggregator of metadata for different UPnP Content Directory Services available in the home. This introduced concept of the "Super" CDS should also be independent of the storage location of the content in the distributed home network, as stated in chapter 5.1.1. To make sure that all the content in the home network can be accessed in the "Super" CDS, metadata descriptions of the content are needed. As stated earlier, this metadata need to be combined, stored and retrieved. The rest of this section discusses the hooks that UPnP offers to build a metadata aggregator.
5.2.1. UPnP Content Directory Service

The UPnP Forum standardized the Content Directory Service as described in chapter 4.3. The Content Directory Service provides a set of actions that allows the control point to enumerate the content that a media server can provide. The primary actions of this service are browse and search. These actions allow control points to obtain detailed information about each content item. This information includes metadata such as name, artist, date created, file size, etc. The control point can collect all content within the network by browsing the content directory service in every media server. The control point can show the results of the browse actions directly to the user in a user interface (see Figure 24).

![Figure 24: Straightforward way of browsing multiple UPnP Content Directory Services](image)

When the Content Directory Service is used in a straightforward way the user needs to browse every media server device separately. The user must make the decision which media server device to browse, and needs to scan through the returned results if the content (the user is searching for) is included. When the content is not included in the result the user must start browsing another media server. A search can also be used instead of the browse. This allows the caller to search the content directory for metadata that matches some search criteria like title, date, descriptor etc. The search criteria are specified as a query string operating on properties with comparison and logical operators. The search makes it a bit easier for the user to find more directly the content she/he is search for, but this doesn’t change the fact that the user must start searching on another media server device when the content isn’t found.

The results of the explained browse and/or search actions can be merged (and sorted) in the control point. In this way, the control point is the metadata aggregator. This could solve that the user needs to search every media server device separately. The main drawbacks are that the metadata aggregator code needs to be re-implemented on every control point and that the control point might not be the most powerful device to do the merging and the sorting.
What can be used from the UPnP Content Directory Service standard, in the case that a user wants to store metadata and allows access to the content directly, is the fact that the Content Directory Service on a device doesn't need to point to content on that device. Pointing to content is important because this makes a unique coupling between the metadata stored in the Content Directory Service and the stored content on a certain media server device. One of the metadata fields in the Content Directory Service is the res field (where res stand for resource). In this res field a URL is stored, containing the actual storage location of the content. This URL can also point to a storage location on another media server device. In the extreme case there can be media server devices with both a Content Directory Service describing the content of the other media server device, although obviously this is not the intention of the UPnP standard. Figure 25 shows this extreme case.

Figure 25: Extreme Content Directory Service example

5.2.2. Announcing that a Content Directory Service is a “Super” CDS

To define a non-standard service like the “Super” CDS, UPnP vendors may add actions and services to standard devices. The “Super” CDS should be backward compatible. This means that control points that do not know about the presence of a “Super” CDS still can use the normal Content Directory Service. Defining a new service that’s called SuperContentDirectory instead of ContentDirectory is an option but control points of other vendors would not understand the service at all.

Alternatively, to announce that a Content Directory Service is a “Super” CDS, a UPnP vendor specific action could be added. It is possible to use non-standard actions because control points already need to know of the presence of a “Super” CDS to make use of it and therefore also know about the presence of non-standard actions. The UPnP standardization commission defined that the name of a UPnP vendor specific action must start with a X_. A standard control point ignores all the actions that start with a X_. Thus, by adding UPnP vendor specific actions to the regular Content Directory Service actions, the Content Directory Service can still be used by control points that don't know about the existence of the “Super” CDS actions. The action X_isSuperCDS, which only returns true, is already enough to let a control point discover that a Content Directory Service is a “Super” CDS. Instead of adding the X_isSuperCDS action, the action X_getSuperCDSCapabilities could be added. This action could be more useful because is gives all the capabilities of the “Super” CDS such as which Content Directory Services are included in the “Super” CDS etc. In the previous described approach the Content Directory Service is compliant to the UPnP standard but the control points and the “Super” CDS are not.
Another option could be to have only the “Super” CDS accessible in the network. All the other Content Directory Services in the network should become only accessible by the “Super” CDS and not from control points anymore. The behavior of this “Super” CDS should become the same as a normal Content Directory Service. In this way all control points can access the “Super” CDS without the need to know that it is a “Super” CDS. In this approach the control points are compliant to the UPnP standard but the Content Directory Service and the “Super” CDS are not.

5.2.3. Keeping “Super” CDS up to date using UPnP Eventing

When a metadata aggregator has storage included, the issue of keeping the cached metadata consistent arises. There are solutions to keep a cache up to date, like a check during startup of the device and/or scanning every now and then to verify if the cached metadata is still valid. It should be advantageous if the cache could be updated immediately after something is changed. In this way the user isn’t bothered with long boot time and/or the need to process to keep the cache consistent. UPnP’s eventing mechanism (see chapter 4.1.5) can be used to keep the “Super” CDS informed immediately after something is changed. To receive events from the Content Directory Service the control point can subscribe as client to the eventing mechanism of the media server device. When an event occurs, i.e. when a state variable changes, notifications are sent to the subscribed control points. The event message contains the names of one or more state variables. The evented state variables for the Content Directory Service are stated in Table 1.

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Required or Optional</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransferIDs</td>
<td>Optional</td>
<td>String</td>
</tr>
<tr>
<td>SystemUpdateID</td>
<td>Required</td>
<td>Unsigned Integer (ui4)</td>
</tr>
<tr>
<td>ContainerUpdateIDs</td>
<td>Optional</td>
<td>String</td>
</tr>
</tbody>
</table>

Table 1: Evented state variables for the Content Directory Service

TransferIDs contains a list that is evented to notify clients when file transfers initiated by ImportResource or ExportResource started or finished. When a file transfer starts, its transfer id is added to the TransferIDs list. When the transfer ends, its id is removed from TransferIDs. This state variable is used for eventing only.

The SystemUpdateID number changes whenever anything in the Content Directory of the media server changes. A change could be a new or removed object, or a change in the metadata of an object. Clients should only check for equality with previous values of SystemUpdateID. However, some state variable values may change too rapidly for eventing to be useful. Event notifications for such state variables are only send when the degree of change exceeds a limit or when a specified amount of time has elapsed since the last notification (maximum event rate is once every 2 seconds).

Each time a container is modified the ContainerUpdateIDs change. This state variable is an unordered list of ordered pairs. Each pair consists of a ContainerID and a ContainerUpdateID, in that order. There can be at most one occurrence in ContainerUpdateIDs of an ordered pair with any specific ContainerID. When a container is modified, its ContainerUpdateID is incremented and the ordered pair of the ContainerID and ContainerUpdateID are concatenated to the list.
ContainerUpdateIDs. If the ContainerID already appears in ContainerUpdateIDs, the new ordered pair is not added to the list. Instead, the corresponding ContainerUpdateID that is already in ContainerUpdateIDs is replaced by the new ContainerUpdateID value. A subscribing control point only sees the last value of ContainerUpdateID prior to the event. ContainerUpdateIDs is an evented state variable and is only used for eventing. This means that there is no action like getContainerUpdateIDs that could be used to query the value of ContainerUpdateIDs. Event notifications for state variables are only send when the degree of change exceeds a limit or when the last notification was send more than two seconds ago. Table 2 shows a time-ordered sequence of activities on a Content Directory Service for container modifications.

<table>
<thead>
<tr>
<th>Activity on Content Directory Service</th>
<th>ContainerID</th>
<th>New value of ContainerUpdateID</th>
<th>New value of ContainerUpdateIDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialization</td>
<td>-</td>
<td>-</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>Container modified</td>
<td>MusicAlbum15</td>
<td>53</td>
<td>&quot;musicAlbum15,53&quot;</td>
</tr>
<tr>
<td>Container modified</td>
<td>photoAlbum28</td>
<td>427</td>
<td>&quot;musicAlbum15,53,photoAlbum28,427&quot;</td>
</tr>
<tr>
<td>Container modified</td>
<td>musicAlbum15</td>
<td>54</td>
<td>&quot;musicAlbum15,54,photoAlbum28,427&quot;</td>
</tr>
<tr>
<td>Container modified</td>
<td>musicAlbum11</td>
<td>12</td>
<td>&quot;musicAlbum15,54,photoAlbum28,427, musicAlbum11,12&quot;</td>
</tr>
<tr>
<td>ContainerUpdateIDs is evented</td>
<td>-</td>
<td>-</td>
<td>(value not changed)</td>
</tr>
<tr>
<td>New control point signs up for events</td>
<td>-</td>
<td>-</td>
<td>(no change to value, the special event value unicast to new control point includes the full set of 3 pairs)</td>
</tr>
<tr>
<td>Container modified</td>
<td>musicAlbum01</td>
<td>97</td>
<td>&quot;musicAlbum01,97&quot;</td>
</tr>
</tbody>
</table>

Table 2: ContainerUpdateIDs Example

With respect to the SystemUpdateID, the ContainerUpdateIDs variable provides more information about the scope of the change, since it takes advantage of the ContainerUpdateIDs values maintained for each container. The drawback is that the ContainerUpdateIDs is optional. If a certain device does not support this optional variable, a control point may use the browse or search actions to query the ContainerUpdateID for an individual container. For indications of Content Directory Service wide change, the evented variable SystemUpdateID, should be used.

All the described UPnP Content Directory Service evented state variables are based on GENA NOTIFY messages that are sent using HTTP. When the control point receives such a message it replies with an HTTP OK to acknowledge that the event message was received correctly. Event notifications also contain sequence numbers to allow detection of lost or out-of-order messages (which should not occur, since notifications are sent using TCP). The HTTP OK acknowledgement, the sequence number and the evented state variables itself can help with keeping caches in the "Super" CDS consistent.
To acknowledge receipt of this event message, a subscriber must respond within 30 seconds, including expected transmission time. If a subscriber does not respond within 30 seconds, the publisher should abandon sending this message to the subscriber but should keep the subscription active and send future event messages the subscriber until the subscription expires or is cancelled. The subscriber must send a response in the following format.

5.2.4. Merging metadata

In order not to slow down the user-system interaction it is important to quickly merge metadata. How quick metadata can be merged depends on the difference of processing power per devices. Merging metadata means that the Content Directory Service replied XML fields need to be combined and sorted in one XML field by the “Super” CDS. For example, a user searches for all videos of the actor Tom Cruise. The “Super” CDS consults for this all devices in the home network or its own database. The replies contain videos with Tom Cruise. These replies need to be sorted and merged in one XML field that should be returned to the user. The main profit from merging data in the “Super” CDS, implemented on a device with a lot of processing power, is that the same merging and sorting code doesn’t need to be implemented on every control point. In this way, complex processing intensive tasks are taken away by the control point, which most of the time has small processing power.

5.2.5. Storing self defined metadata

To make the “Super” CDS work, it could be needed to store extra content related metadata like when is the metadata obtained. Storing metadata in the Content Directory Service can be done by using the specified extended properties or by using the descriptor namespace. The AV working Committee has specified extended properties, which can store metadata in the Content Directory Service. Examples of specified properties are base properties like title, creator, etc. and people involved like artist, actor, etc. One of these extended properties is the userAnnotation. This is a general-purpose tag defined by the committee, where a user can annotate an object with some user-specific information. The userAnnotation property can be used to store metadata, which cannot be stored in the other specified properties.

Another way to store the metadata is to add a descriptor tag to the DIDL-Lite description. Using the descriptor tag gives the vendors the possibility to extend the specified properties by placing blocks of vendor-specific metadata into the descriptor. A descriptor is employed to associate blocks of other XML-based metadata with a given Content Directory Service object. The contents of each descriptor must be associated with only one namespace. But there can be more than one descriptor in the DIDL-Lite description.

The previous explained methods to store the metadata are completely vendor specific solutions. The consequence of storing the most of these metadata in a proprietary way is that the retrieval of this metadata is a problem. When a control point searches for e.g. a descriptor of a video, the complete descriptor is returned, including all metadata it contains. Afterwards the included metadata needs to be parsed on the non-powerful control point to reconstruct the required metadata.
5.3. Analysis of UPnP based “Super” CDS

UPnP offers enough hooks and possibilities to build a metadata aggregator (“Super” CDS). In this way, the UPnP based “Super” CDS provides a solution to get access to a distributed storage system and serves it as a central place where information of all stored content can be obtained.

The hooks and possibilities that UPnP offers to build the “Super” CDS can help to take care of:

- Keeping the metadata aggregator consistent by using UPnP eventing.
- Announcing that a Content Directory Service is a “Super” CDS.
- Reconstructing the metadata aggregator by using the Content Directory Service browse and/or search.
- Storing self defined metadata.

For users this means that:

- All stored distributed metadata can be accessed in a central place and the actual storage locations of the content are shielded.
- There is a quick response on a browse or search action because the “Super” CDS provides fast searches through a centralized index.
- If the metadata aggregator has storage it can show all content in the network regardless if a device is available on the network or not. This provides even the possibility to access e.g. DVD collections stored in a rack in the house or handheld devices that are not in the home.
6. "Super" CDS implementation approaches

There are a lot of implementation possibilities to verify the two extreme cases and their intermediate cases described in chapter 5.1.1 using the UPnP Content Directory Service. First alternatives for implementing the "Super" CDS without storage are considered. Note: in all the suggested implementation possibilities, the browse action can be seamlessly be replaced by a search action.

6.1. One Content Directory Service for all media server devices in the network

One possibility to make a "Super" CDS implementation is to use only one Content Directory Service in the network for all media server devices (see Figure 26). What is used for this implementation is that the res field can point to storage location on other media server devices (see chapter 5.2.1). By using this property of the res field only one Content Directory Service is needed in the whole network. In the network this Content Directory Service is then the "Super" CDS. This implementation not only conflicts with the UPnP thoughts that every media server must contain a Content Directory Service but also has a single point of failure. For example, when this "Super" CDS is not accessible (e.g. when the device with the "Super" CDS is disconnected from the network), content cannot be found anymore. Another problem of this implementation is that content and its related metadata only can be stored via the media server device that offers the "Super" CDS and not directly at the media server device that should contain the content. This proposed solution of not having a Content Directory Service per media server device even isn't a distributed solution anymore. Thus, this implementation is not a satisfying solution and will not be analyzed any further.

![Figure 26: One Content Directory Service for all media server devices in the network.](image)
6.2. Simple "Super" CDS implementation

The first simple implementation of a "Super" CDS is when a software layer is included in the control point that can merge the browse or search results (see Figure 27).

![Diagram of Simple "Super" CDS implementation]

Figure 27: Simple "Super" CDS implementation.

The advantages and disadvantages of this implementation with respect to the metadata aggregator requirements stated in 5.1.2 are:

Advantages:
- The complete architecture can be implemented using the UPnP standard without modifications.
- Because the data is asked on the fly to the devices and there is no caching at all, there is no consistency and reconstruction problem.

Disadvantages:
- Time consuming because every time a question is asked, the control point needs to search/browse all devices and afterwards the results need to be merged.
- No information about devices that are off.
- The control point has to deliver the processing power for merging and sorting the metadata when e.g. a difficult search is done on a media server.
- Every control point needs to implement the same code.
6.3. Simple “Super” CDS implementation with local caching

An extension to the simple “Super” CDS implementation approach could be to add a cache in the control point that stores the queries and its (merged) results (See Figure 28). This can speed up the response on a query because often the same queries are asked in a short time. For example, a user selects the video browser to see a list of the available videos in the home network. From this list she/he selects a certain video to see its pictorial overview. After the user has seen this overview she/he goes back to the list of available videos. The cache stores the merged list of videos the first time the user enters the video browser. When the user goes back from the overview to the video browser, the cached video list can be replied immediately to the user interface.

![Diagram of Simple “Super” CDS implementation with local caching](image)

Figure 28: Simple “Super” CDS implementation with local caching.

The advantages and disadvantages of this implementation with respect to the metadata aggregator requirements stated in 5.1.2 are:

**Advantages:**
- The complete architecture can be implemented using the UPnP standard without modifications.
- Browse queries and its results can be returned very quickly when they are already stored in the cache. In this way the control points doesn’t need to ask all the devices on the network on the fly.
- There can be information in the cache about devices that are turned off.
- The control point can subscribe to Content Directory Service eventing and use this to keep the cache up to date.
- Reconstruction is not needed because the cache stores the user-initiated queries and its returned results. Moreover, it is even not possible to reconstruct the previous user-initiated queries.
Disadvantages:

- When a difficult search is done on the devices the control point has to deliver the processing power for merging and sorting this search.
- Every control point needs to implement the same code. This means that every control point that enters the network has to build up its own cache. The control point can only access its own build cache because the control point cannot offer the local cache as a service. This means that the cache is not accessible for other devices and/or control points within the network.
- The cache can become inconsistent.

6.4. "Super" CDS implementation on a media server

Another "Super" CDS implementation possibility is to build up a central "Super" CDS in a media server, which can collect and store the metadata of content stored on all media servers. To collect all the metadata of the media servers, an embedded control point needs to be in the device that provides the "Super" CDS. The embedded control point is needed because only control points know of the presence of devices and their services in the network. The user can send a browse request to the "Super" CDS by using a control point that, by using e.g. the action X_getSuperCDSCapabilities, knows about the existence of this "Super" CDS. The "Super" CDS passes on the browse request to the embedded control point. This embedded control point knows which Content Directory Services are available in the network and can ask on the fly every Content Directory Service (including itself) their metadata describing its content. On the media server these browse results are combined and stored in the "Super" CDS. Then the "Super" CDS can return the result of the user initiated browse query.

![Diagram of media server with embedded control point](image)

**Figure 29: Media server with embedded control point**
The advantages and disadvantages of this implementation with respect to the metadata aggregator requirements stated in 5.1.2 are:

Advantages:

- The “Super” CDS is based on the standardized UPnP Content Directory Service with some proprietary actions included for the operation of the “Super” CDS. This means that the “Super” CDS is backward compatible and can be accessed as a normal Content Directory Service by a control point that doesn’t know about the existence of a “Super” CDS.
- Because the data is asked on the fly to the devices and there is no caching at all, there is no consistency and reconstruction problem (Found content is really available).
- This service will transfer the processing and communication demands associated with interaction with multiple media servers from the control point to a more powerful media server device.
- The control point can still see all the Content Directory Services of all media servers in the network. This makes direct access to metadata possible when the storage location of the metadata on a device is already known.
- Only the media server that offers the “Super” CDS has to implement the content aggregation code.

Disadvantages:

- On the fly asking can be time consuming because every time when a question is asked, the control point needs to search/browse all devices and afterwards the results need to be merged.
- A non-standard control point and “Super” CDS is needed. The control point needs to know about the existence of the “Super” CDS and its UPnP vendor specific actions like X_getSuperCDSCapabilities to make use of the “Super” CDS. Otherwise, a control point that is not aware of the “Super” CDS might show the user all content twice because the control point can still see all the Content Directory Services of all media servers in the network.
- There is no information available about devices that are turned off.

6.5. “Super” CDS implementation on a media server using a cache

An extension of the previous implementation could be to aggregate metadata from the multiple Content Directory Services into a centralized cache (see Figure 30). The cache can use the large amount of storage space in the future CE devices to store its information. In this approach the embedded control point is responsible for keeping the “Super” CDS cache consistent. The embedded control point can subscribe to the Content Directory evented state variables to try to keep the “Super” CDS consistent. These evented state variables could be used to discover that there is a change in the metadata in a certain media server.
Figure 30: “Super” CDS implementation on a media server using a cache.

If the user accesses the “Super” CDS she/he should use a control point that knows about the existence of this “Super” CDS. This means that the control point needs to know about the UPnP vendor specific actions like e.g. X_getSuperCDSCapabilities. Otherwise, all content might be showed twice to the user because in this implementation the control point can still see all the Content Directory Services of the other media servers. An advantage of seeing all Content Directory Services is that the control point can still access all the Content Directory Services separately, which makes direct access to metadata possible when the storage location of the metadata on a device is already known. The embedded control point must keep the “Super” CDS consistent for example by using the eventing mechanism. In this way, only the cache needs to be searched for the answer of the user query. This makes it possible to immediately return the result of the query.

The advantages and disadvantages of this implementation with respect to the metadata aggregator requirements stated in 5.1.2 are:

Advantages:
- Quick reply to query.
- There is information available about devices that are turned off.
• The "Super" CDS is based on the standardized UPnP Content Directory Service with some proprietary actions included for the operation of the "Super" CDS. This means that the "Super" CDS is backward compatible and can be accessed as a normal Content Directory Service by a control point that doesn’t know about the existence of a "Super" CDS.

• The control point can subscribe to Content Directory Service eventing and use this to keep the cache up to date.

• The embedded control point could reconstruct the cache if that’s needed.

• This service will transfer the processing and communication demands associated with interaction with multiple media servers from the control point to a more powerful media server device.

• The control point can still see all the Content Directory Services of all media servers in the network. This makes direct access to metadata possible when the storage location of the metadata on a device is already known.

• Only the media server that offers the "Super" CDS has to implement the content aggregator code.

Disadvantages:

• The cache should be kept consistent.

• A non-standard control point and "Super" CDS is needed. The control point needs to know about the existence of the "Super" CDS and its UPnP vendor specific actions like X_getSuperCDSCapabilities to make use of the "Super" CDS. Otherwise, a control point that is not aware of the "Super" CDS might show the user all content twice because the control point can still see all the Content Directory Services of all media servers in the network.

6.6. "Super" CDS implementation using two Content Directory Services in one media server

The "Super" CDS implementation shown in Figure 31 uses two Content Directory Services, one is an empty "pass through" Content Directory Service and one is the "Super" CDS. The empty Content Directory Service is used as an entrance to ask the queries to the media server. The empty Content Directory Service literally passes on these queries to the embedded control point. This embedded control point can use these queries on the "Super" CDS and can return the results via the empty Content Directory Service to the querying control point. The second Content Directory Service stores the metadata of its own media server and of the other media servers in the network. The embedded control point is again responsible to keep the cache up to date. A solution is to subscribe the embedded control point to the Content Directory evented state variables to keep in touch with changes in the metadata in the Content Directory Service of a certain media server. In this way, only the cache needs to be searched for the answer of the user query. This makes it possible to immediately return the result of the query.
The “pass through” Content Directory Service is acting as a normal Content Directory Service. The user accesses this Content Directory Service by using a normal control point that doesn’t need to know about the existing of the “Super” CDS. The embedded control point should set up a point-to-point communication between the Content Directory Services and the embedded control point. After setting up the point-to-point communication the Content Directory services should stop to broadcast their presence to the whole network. To set up the point-to-point communication and stop the broadcast of the presence of the Content Directory Services, new UPnP vendor specific actions should be defined for the Content Directory Service. Thus, in this implementation the control point cannot see all Content Directory Services anymore but only sees one Content Directory Service. This only Content Directory Service needs to take care of the storing of content and related metadata on all the devices in the network. When another device that also contains a “Super” CDS accesses the network, the two devices containing the “Super” CDS can set up point-to-point communication as described above. In this way one of the two devices that contain a “Super” CDS is stopped to broadcast. After this action there is again only one accessible Content Directory Service available in the network (recursion). This implementation really shields the storage locations for the user and offers really one virtual place to access distributed content. The embedded control point is responsible to keep the cache up to date. In terms of implementation complexity this is an easy solution. However, it has the problem of a single point of failure and possible slow access times because there is only one point where the multiple control points in the network can access the media server devices. Thus, if these multiple control points query all at the same time the “Super” CDS the requests and its results need to be scheduled.
The advantages and disadvantages of this second implementation with respect to the metadata aggregator requirements stated in 5.1.2 are:

**Advantages:**

- Possible quick reply to query because of the caching, but this depends on the number of control points that are simultaneous accessing the “Super” CDS device.
- There is information available about devices that are turned off.
- To the outside world a normal Content Directory Service is offered, this means that standard control points can be used to access the “Super” CDS.
- The control point can subscribe to Content Directory Service eventing and use this to keep the cache up to date.
- The embedded control point could reconstruct the cache if that’s needed.
- This service will transfer the processing and communication demands associated with interaction with multiple media servers from the control point to a more powerful media server device.
- Only the media server that offers the “Super” CDS has to implement the code.

**Disadvantages:**

- Single point of failure and possible slow access times because there is one central place where the media server devices can be accessed.
- The cache should be kept consistent.
- Non-standard Content Directory Service is needed. The Content Directory Service needs to have UPnP vendor specific actions to establish the point-to-point communication and to stop the broadcasts.
<table>
<thead>
<tr>
<th>Implementation</th>
<th>Consistency</th>
<th>User-system interaction speed</th>
<th>Reconstruction</th>
<th>Show all content, even when a device is switched off.</th>
<th>Merging and sorting on device with big processing</th>
<th>Compatible with UPnP standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple “Super” CDS</td>
<td>Not an issue</td>
<td>Slow</td>
<td>Not needed</td>
<td>Not possible</td>
<td>Not true</td>
<td>Yes</td>
</tr>
<tr>
<td>Simple “Super” CDS with local caching</td>
<td>The cache can become inconsistent. Content Directory Service eventing can be used</td>
<td>Can be quick if the query and its result is stored in the cache</td>
<td>Not needed</td>
<td>Possible if the metadata is already queried earlier and stored in the cache</td>
<td>Not true</td>
<td>Yes</td>
</tr>
<tr>
<td>“Super” CDS on a media server</td>
<td>Not an issue</td>
<td>Slow</td>
<td>Not needed</td>
<td>Not possible</td>
<td>True</td>
<td>Yes, “Super” CDS is backwards compatible but needs vendor specific actions, not standard Control Point needed</td>
</tr>
<tr>
<td>“Super” CDS on a media server using a cache</td>
<td>The cache can become inconsistent. Content Directory Service eventing can be used</td>
<td>Can be quick</td>
<td>Possible</td>
<td>Possible</td>
<td>True</td>
<td>Yes, “Super” CDS is backwards compatible but needs vendor specific actions, not standard Control Point needed</td>
</tr>
<tr>
<td>“Super” CDS using two Content Directory Services in one media server</td>
<td>The cache can become inconsistent. Content Directory Service eventing can be used</td>
<td>Can be quick</td>
<td>Possible</td>
<td>Possible</td>
<td>True</td>
<td>Yes, standard Control Points can access the only available Content Directory Service, not standard Content Directory Services needed in the other media server devices</td>
</tr>
</tbody>
</table>

In Table 3 the possible implementation approaches are summarized with respect to the metadata aggregator requirements stated in 5.1.2.

6.7. Summarizing the possible implementation approaches...
6.8. Useable extra possibilities for the building of a “Super” CDS

There are some extra possibilities that can be used for the building of the above-summarized implementations. These extra possibilities are written here and not in the chapters about the specific implementations because they are not specifically coupled to a certain implementation. For example these possibilities can be used to speed up the system. In this chapter these extra possibilities that can be to use for the building of a “Super” CDS are clarified.

6.8.1. Cache strategies

There are many possible cache strategies, which could be used to speed up the user-system interaction. An example of such cache strategies can be the caching of subsets of the metadata. In this case, only the metadata most used is cached, like content name, actors, duration, etc. When the user wants to access metadata that is not cached the “Super” CDS could get it on the fly by a browse or search action from the media server device that contains the original metadata. This can be done very quickly because the “Super” CDS knows exactly where the original metadata is stored in the Content Directory Services on the media server devices.

Another possible cache strategy that can be used is to cache the knowledge about the accessibility of metadata on a media server in the home network. This metadata could be stored in self-defined metadata fields as explained in chapter 5.2.5. The metadata that could be stored is e.g. the dynamically leaving and entering pattern of a mobile media server in the home network. This information could help to calculate the statistical change that a device becomes available and its metadata is accessible.

6.8.2. Keeping cache up to date using UPnP eventing

Using cached results would introduce the possibility that the result is not completely up-to-date. For example, the metadata that becomes available when a media server device is added or metadata that isn’t available anymore when a device is removed. Validity confirmation and availability checking functions are therefore needed to ensure that the displayed metadata is completely and fully available. There are a lot of issues that make it difficult to implement consistent caching. For example, device A changes the name of a video; device B changes the same name at the same time. Both actions are executed, but depending on which action is received first there becomes an inconsistency. To avoid this kind of problems the Content Directory Service evented state variables as explained in 5.2.3 could be used. The required evented variable SystemUpdateID could be used for indicating changes in the Content Directory Service on the media server devices. The main issue when using this variable is that the "Super” CDS still needs to search in the Content Directory Service of the media server what exactly is changed. The evented state variable ContainerUpdateIDs provides more information about the scope of the change, since it takes advantage of the ContainerUpdateIDs values maintained for each container. The drawback is that the ContainerUpdateIDs is optional.
6.8.3. All media server devices provide a “Super” CDS

In a UPnP network devices can dynamically appear and disappear. Of course, this is also valid for the media server device that provides the “Super” CDS. To avoid that there is no “Super” CDS anymore, when the device that provides the “Super” CDS is leaving the network, another media server device should start providing the “Super” CDS. A solution could be to let every media server device provide a “Super” CDS. The method of stopping the broadcast of a Content Directory Service as described in chapter 6.6 could be used to provide only one “Super” CDS in the network. By using known master slave negotiations, the media server device that provides the master “Super” CDS must be chosen. In this master-slave negotiation it should be taken into account that the media server device with the highest processing power and highest uptime in the network is chosen. The advantage of this approach is that when the media server that contains the master “Super” CDS is leaving the network another device could take over its task and can become the master “Super” CDS.

6.9. Verification of “Super” CDS implementations approaches

At the end of the graduation assignment a beginning is made with the integration of the previous explained implementation approaches into the SPATION demonstrator. The goal of integrating these implementations is to verify and to show the working of a distributed storage network with the SPATION demonstrator. The first chosen implementation of the “Super” CDS to integrate, is the one described in chapter 6.5. This implementation is chosen because it has a lot of advantages and only some disadvantages. Another advantage of implementing this one first is that it can serve as a basis for the “Super” CDS implementation described in chapter 6.6, which approaches the optimal solution to access content transparently at one central place. The implementation is still going on during the writing of this report.
7. Conclusions

CE devices become more and more powerful, have large amounts of storage space and become interconnected. When users have more storage space they collect more and more content and its related metadata and in this way they get enormous content and metadata collections in the home. The large amount of content and metadata, and the fact that it becomes distributed over the home network makes retrieval a challenging task.

The graduation started with a very broad assignment, this assignment was how to deal with distributed storage in a heterogeneous home network. To bring more focus in the assignment important aspects were investigated like, distribution of storage, heterogeneity, scalability and transparency. These important aspects helped with addressing the boundary conditions of the graduation assignment.

The key issue in distributed home storage using Consumer Electronics devices is single point access to content and its related metadata. To solve this issue abstract solutions of metadata aggregators are introduced. A metadata aggregator is a centralized place where metadata is aggregated. This metadata is related to content distributed on different devices in the network. The metadata aggregator must be independent of the storage location of the content in the distributed home network. The introduced abstract solutions help the user with transparently accessing their distributed metadata related to stored content. From these solutions important requirements for the metadata aggregator were found that are used as guidelines for the verification of the suggested implementations of a metadata aggregator.

In the SPATION project the decision is made to focus on UPnP as a middleware stack. This UPnP network middleware standard can be used to support storing, accessing and searching for content and metadata in the home network. To share metadata between devices in the network UPnP standardized the Content Directory Service. This service offers a starting-point to really build the abstract suggested solutions of the metadata aggregators. The metadata aggregator service needs to provide a uniform mechanism for user interface devices to browse the content on one server that can obtain detailed information about individual content. The concrete introduced solution is to build a “Super” CDS (based on the UPnP Content Directory Service), which can act as an aggregator of metadata from different UPnP Content Directory Services available in the home.

Various possible “Super” CDS implementations are elaborated with all their own advantages and disadvantages with respect to the metadata aggregator requirements. The “Super” CDS implementation using two Content Directory Services in one media server approaches the optimal solution to access content transparently at one central place. However, the “Super” CDS implementation on a media server using a cache is chosen as first to integrate into the SPATION demonstrator because it has a lot of advantages and only some disadvantages. An advantage of implementing this one first is that it can serve as a basis for the “Super” CDS implementation using two Content Directory Services in one media server.
8. Future work

The research done for this graduation assignment is already a step in the direction of the creation of a heterogeneous home network where metadata and content can be transparently distributed. Research wouldn’t be research if it wouldn’t result in many new research questions that could be investigated in the future. After this graduation assignment still a lot of questions are open and left untouched. The rest of this section provides possible future research topics that could be investigated.

In this assignment investigation is done in the area of distribution of storage. This assignment focuses on the following three key challenges defined by Coulouris (See chapter 2): heterogeneity, scalability and transparency. These are only three out of the seven key challenges. The four left key challenges: openness, security, failure handling and concurrency of components should also be investigated. From this investigation it should become clear what their relation and contribution could be to realize a heterogeneous home network where metadata and content can be transparently distributed.

In chapter 4 it became clear that UPnP is a good standard but does not solve everything. It is a good trade-off but still some standardization needs to be done to make it possible to transparently distribute metadata and content in an interoperable way between different vendors.

The line of argumentation that is started in chapter 5.1 should be continued to find more possible intermediate cases that could approach the real optimal solution. For these intermediate cases several new caching variants should be investigated, reasoned through and tested to speed up the system and the user-system interaction. The most important aspects in this future research are the several consistency questions stated in this report. To answer these questions the implementations using a cache should be analyzed in-depth theoretically. In this analysis all the cases when the cache can become inconsistent must be reasoned through.

The approach that every media server device in the network is a “Super” CDS as substantiated in chapter 6.8.3 could be more investigated.

After choosing the best implementation approaches they should be really tested. After implementing the chosen implementation approach a question could be answered like, what is an acceptable time to wait for the response on a search/browse action? When, for example this response time is too long the next step that could be investigated is at what level additional caching is needed.
References


[12] Rendering Control V1.0 http://www.UPnP.org/download/RenderingControl%201.0.pdf


[18] MPEG21 overview
   http://www.chiariglione.org/mpeg/working_documents.htm#MPEG-21

List of figures and tables

List of figures:

Figure 1: Comparison between the OSI model and the TCP/IP reference model........................................... 6
Figure 2: Example of a SPATION scenario........................................................................................................ 10
Figure 3: SPATION system: Streamium, iPronto, PC, and TV........................................................................... 12
Figure 4: SPATION prototyping system.............................................................................................................. 13
Figure 5: Start screen of the SPATION user interface on the iPronto................................................................. 14
Figure 6: Overview of stored video programs..................................................................................................... 14
Figure 7: Video browser interface..................................................................................................................... 15
Figure 8: Pictorial overview of a program........................................................................................................... 16
Figure 9: Interaction between devices in UPnP.................................................................................................. 20
Figure 10: UPnP protocol stack......................................................................................................................... 21
Figure 11: Structure of UPnP............................................................................................................................. 25
Figure 12: Example of last change state-variable.............................................................................................. 26
Figure 13: Item definition.................................................................................................................................... 28
Figure 14: Container definition............................................................................................................................ 28
Figure 15: Browse request and response............................................................................................................ 29
Figure 16: System setup: Left: device structure, Right: UPnP structure............................................................. 31
Figure 17: Control point and Media Server in one physical device..................................................................... 32
Figure 18: User query on a metadata aggregator without storage................................................................. 34
Figure 19: Storing content and its metadata on a metadata aggregator without storage............................... 34
Figure 20: User query on a metadata aggregator that stores all metadata......................................................... 35
Figure 21: Storing content and its metadata on a metadata aggregator that stores all metadata................. 36
Figure 22: Query and query result caching......................................................................................................... 36
Figure 23: Solution to try to keep the aggregator database consistent............................................................ 37
Figure 24: Straightforward way of browsing multiple UPnP Content Directory Services............................ 39
Figure 25: Extreme Content Directory Service example.................................................................................. 40
Figure 26: One Content Directory Service for all media server devices in the network............................. 45
Figure 27: Simple “Super” CDS implementation............................................................................................... 46
Figure 28: Simple “Super” CDS implementation with local caching.............................................................. 47
Figure 29: Media server with embedded control point...................................................................................... 48
Figure 30: “Super” CDS implementation on a media server using a cache.................................................... 50
Figure 31: “Super” CDS implementation using two Content Directory Services............................................ 52

List of tables:

Table 1: Evented state variables for the Content Directory Service................................................................. 41
Table 2: ContainerUpdateIDs Exam.p1e........................................................................................................... 42
Table 3: Conclusions of implementation approaches......................................................................................... 54
Abbreviations and Acronyms

AV       Audio Video
CD       Compact Disc
CE       Consumer Electronics
CDS      Content Directory Service
DB       DataBase
DC       Dublin Core
DHCP     Dynamic Host Configuration Protocol
DVD      Digital Versatile Disc
GENA     General Event Notification Architecture
HDD      HardDisk Drive
HTML     HyperText Markup Language
HTTP     HyperText Transfer Protocol
id       identifier
IP       Internet Protocol
PC       Personal Computer
PVR      Personal Video Recorder
res      Resource
SOAP     Simple Object Access Protocol
SPATION  Services Platforms and Applications for Transparent Information management in an in-hOme Network
SSA      Storage Systems and Applications
SSDP     Simple Service Discovery Protocol
TCP      Transmission Control Protocol
TV       Television
UI       User Interface
UPnP     Universal Plug and Play
URL      Uniform Resource Locator
val      value
XML      eXtensible Markup Language
xmlns    XML Name Space
Appendix A

ABSTRACT

Consumer electronic devices with large storage capacity become more and more interconnected in home networks. This gives users the opportunity to distribute huge amounts of content over multiple devices. In this situation information retrieval becomes a challenge and content-based navigation and search tools represent a possible solution. The SPATION project investigates how content-based organization and retrieval can be integrated in a distributed home system built using existing middleware standards such as Universal Plug and Play (UPnP). In this paper we will provide an overview of the SPATION system and applications, and we will illustrate how the UPnP standard can be used to support the integration of content-based access tools in home consumer networks.

1. INTRODUCTION

Consumer electronics devices will soon have a huge storage capacity, and enormous processing power, which will create the possibility to offer a wide range of functions. Additionally high speed wired or wireless networks provide the opportunity to interconnect these devices. This results in a home network with tremendous possibilities: connected devices can share both control (distribution of functionality) and content (distribution of storage). From the end users' point of view, sharing control means that control functions of devices are available on other devices. Content sharing means that the stored content is transparently accessible on every device even if it is spread over the whole home network and it is not at a single physical location anymore. The huge amount of data a user can store in the house becomes very difficult to manage due to the fact that retrieving data is more difficult in a distributed situation. Without an integrated storage solution, the users would have to search each device separately. To help users find the content they want, we propose to shield the actual storage location from the user and use metadata to support advanced content-based retrieval queries. When metadata obtained from service providers is combined with content analysis algorithms running on devices attached to the home network, devices can offer content analysis services to other devices in the network. For example, a Personal Video Recorder (PVR) that has enough processing power to generate a visual table of content can offer this service to a less powerful handheld device used as an advanced video browsing remote control.

SPATION (Services Platforms and Applications for Transparent Information management in an in-hOme Network) is a European project that focuses on the technologies, infrastructure and user interfaces for networked home environments. The objective is to find novel solutions for content management tasks such as moving, organizing and retrieving information in a heterogeneous home system. This heterogeneous home system consists of a number of interconnected stationary devices and various handheld devices; all these devices communicate with each other and have their own storage capacity that allows storing large amounts of content.

At this moment a large amount of literature on home networking and network protocols is available ([1], [2]) and obviously there is also a large amount of information available on content analysis [3]. However, the combination of using content analysis and home networking is rare. This paper gives an overview of the SPATION project with a focus on the combination of content analysis techniques with the Universal Plug and Play (UPnP) standard to provide transparent content management in home networks. To reach this goal the following structure in the paper is used: In section 2 a typical SPATION application scenario will be illustrated. Section 3 will address the navigation of content supported by content analysis. Why UPnP is chosen as network
2. APPLICATION SCENARIO

Within the SPATION project we wrote six user scenarios to get a better understanding of specific issues and requirements related to the problem of dealing with distributed content in the home. In Figure 1 an abbreviated version of one of these scenarios is shown. The main focus of this scenario is on the use of a personal video recorder to manage content, and in particular to offer a new video summarization function as a service to other devices. From this scenario, it becomes clear how a user wants to transparently use the applications and resources over the network. In practice this means that the user, while sitting on the couch, can use a handheld device to control devices in the home. The handheld in this case is used as a combination between the lean-backward way of watching television and the lean-forward way of using a PC. Using the handheld for these types of system interaction, users can search and browse content, but also simple annotation and editing are possible. An example of such a handheld (advanced remote control and internet browser) is the Philips iPronto. This device is used in the SPATION project to control all devices in the home network. In the project additional software was developed to support browsing video content stored in the home network using the iPronto. When the user wants to watch a particular program, found during browsing, the user selects it and selects the device where she/he wants to display it. The handheld is also used for video browsing functions like a video browser (see section 3.1), a video preview (see section 3.2) and a pictorial overview of a video (see section 3.3). Metadata such as key-frames are automatically generated on more powerful devices such as PVR’s connected to the network. Because a PVR has sufficient processing power, it can support the content analysis algorithms used to enable the video browsing functions on the handheld device. In the next section we describe the developed applications that allow users to use these previously described video browse functions.

3. CONTENT NAVIGATION SUPPORTED BY CONTENT ANALYSIS

As described previously, access of stored content and control over the home system is made possible via handheld devices. An important aspect is the organization and retrieval of data in the home system. In the current implementation we wanted to allow users quick access to middleware standard is substantiated in section 4. In section 5 the basics of UPnP are explained. How to store and retrieve metadata using UPnP’s Content Directory Service is the topic of section 6. Conclusions and acknowledgements are presented in section 7 and 8.

Figure 1: Example of a SPATION scenario.
stored content and control of devices. We assume that when a user picks up the handheld from the table she/he already decided what to do. For example, the user intends to listen to music, to watch a movie or to turn down the volume of the TV. Therefore, the main screen provides access to music, photos and video separately, instead of providing an integrated browser (Figure 2).

Pressing the device button gives users access to all devices in the home. To allow users to quickly search for a video, the following functionality is provided:

- View a list of all available video content on all devices.
- Browse a video based on automatically selected key-frames.
- Watch a pictorial overview of a video (automatically generated).
- Watch a short preview of a video (automatically generated).

In Figure 3 the video browser screen that gives access to these functions is shown. The list shows all video content available on all devices. It is not necessary for a user to, for example select PVR1 to see its content, and then PVR2 etc. When the user wants to playback a movie the system will stream the content from the storage device to a display, hence the user does not even need to know where a particular program is stored.

3.1. Video browser

The Browse button gives access to browse a particular piece of stored content within the home network (e.g. a recording of a TV show). Using video content analysis, the personal video recorder generates automatically content descriptors that can be used to enable content-based browsing and retrieval functionality. For the user this means she/he can browse through thumbnails of key-frames representing meaningful scenes of a selected program on a handheld device (Figure 4). Key-frame selection is based on clustering the results of a simple but effective shot-cut detection algorithm that runs during video recording.

3.2. Video preview

When a large amount of video content is available it is difficult for a user to find something interesting. A method that helps the users to decide whether a certain recorded program is interesting enough to watch is provided by automatically generated video previews [4].

3.1. Video browser

The Browse button gives access to browse a particular piece of stored content within the home network (e.g. a recording of a TV show). Using video content analysis, the personal video recorder generates automatically content descriptors that can be used to enable content-based browsing and retrieval functionality. For the user this means she/he can browse through thumbnails of key-frames representing meaningful scenes of a selected program on a handheld device (Figure 4). Key-frame selection is based on clustering the results of a simple but effective shot-cut detection algorithm that runs during video recording.

3.2. Video preview

When a large amount of video content is available it is difficult for a user to find something interesting. A method that helps the users to decide whether a certain recorded program is interesting enough to watch is provided by automatically generated video previews [4].
A video preview (or video skim) is a short video clip, for example 30 seconds, that provides an overview of a recorded program. During recording, content analysis algorithms compute low-level descriptions of the content such as scene boundaries, scene complexity, motion activity, human presence, audio volume, etc. When the recording is complete and all the content descriptions are available, a number of heuristic rules are applied to select the scenes that are most relevant for users that want to preview the content without actually spoiling it. Examples of such rules are non-disclosure of the end of a program, inclusion of action scenes and dialogues, etc. Concatenating the most relevant scenes generates video previews that convey key-aspects of a video program and helps users judge whether or not they would like to see the entire content.

The actual video preview is stored as metadata in the form of list of pointers to the parts of the original content that are included in the preview. Users can watch directly video previews on TV sets or stream transcoded versions to handheld devices.

3.3. Pictorial overview of a video

Using a pictorial overview of a recorded video program, users can quickly see what a program is about, remember whether she or he has seen it before or decide whether or not to watch it or to download it to a portable device.

The pictorial overview is a set of the 15 most relevant pictures extracted from the original content. An automatic summarization algorithm [4][5] selects a set of images that best represents a particular piece of content. An example of a video preview of a home video is shown in Figure 5.

Figure 5: Pictorial overview of a program.

The content navigation functions described in the previous sections are part of an experimental system developed for demonstration purposes. The next section provides an overview of the hardware and software architecture that is currently under development as part of the SPATION research activities.

3.4. SPATION system architecture

The SPATION system consists of several prototypes of stationary CE devices (e.g. TV, PVR, Audio jukebox, photo browser, etc.) and handheld devices. Each device is equipped with hard-disk drives and thus has large storage capacity. To implement prototypes, most of the devices are built using common PC's augmented with extra hardware boards such as tuners and mpeg decoders. All devices run the open source Linux operating system and an implementation of a UPnP stack. Each device hosts an implementation of a UPnP root device (see section 5) for the basic UPnP functions. Although new advanced services such as the content-based navigation functions are implemented with proprietary solutions, the overall software architecture is designed taking into consideration features, limitations and future developments of the UPnP standard.

Two of the PC’s are used as UPnP PVR’s and one simulates a UPnP TV. This UPnP television uses a Digital Video Broadcast (DVB) satellite receiver card to record DVB streams. The UPnP PVR’s typically have a huge amount of storage space (hundreds of gigabytes) to store these recorded streams. The PVR’s do the content analysis described in the previous sections and store their metadata in a proprietary harmonized directory structure to keep the content and the metadata together. In the future we want to store the content and the metadata together in a database like for example described in section 6.

Figure 6: Interaction amongst devices in the SPATION system.
Figure 6 illustrates some of the logical steps involved during the interaction amongst SPATION devices for a common operation of video browsing initiated by a handheld device. In the next section, after providing a short description of the main principles of UPnP, we will discuss how these logical steps are standardized and how the middleware supports the content-based navigation functions presented in section 3.

4. NETWORK MIDDLEWARE STANDARDS

Without a standardized middleware stack every application needs to solve the same problems over and over again. Moreover, systems from different vendors would not be able to communicate with each other. A middleware stack takes care that e.g. adding new devices with new services is easy. How well a home network can handle the adding of new devices and services depends on the combination of the physical network connection, the network stack and the middleware stack. In our case the home network consists of CE devices, which poses additional boundary conditions caused by the difference in processing power and display capabilities between devices.

Network middleware standard that are currently available are HAVi [6], Jini [7], UPnP [8] etc. We investigated whether UPnP can serve as a basis in a system that allows users to easily search, browse and manage content. To do these tasks it would be advantageous if UPnP could be used to offer content analysis algorithms and content analysis results as a service to the network. The main reasons for choosing UPnP as middleware standard for the SPATION system are:

- It is an open standard based on widely known Internet technology.
- It offers a good balance between standardization and implementation freedom.
- It has relatively low implementation complexity and cost.

In our opinion UPnP is a good trade-off between the ideal middleware standard and a practical implementation that is based on existing protocols. Moreover, UPnP is the only middleware standard likely to be adopted by CE vendors.

5. UPnP BASICS

In the following section we will briefly introduce the main technical solutions standardized by UPnP to get a better understanding of the SPATION system architecture described in section 3.4 with respect to the requirements mentioned in section 2 and to enable the application described in section 3.

UPnP defines a way for multiple machines to interact with each other over a network. It is a stack of protocols and starts at the IP level, where it defines how to obtain an IP address. After that it defines how to discover other devices and services on the network, and how to interact with these services. At its highest-level UPnP currently defines standard services for devices like: VCR, CD player, and DVD player.

UPnP defines two types of functionality: server function, which offers the services on the network and the client function that make use of the services. In UPnP the server is called a device, inside a device there can be one or more services. It is possible for a device to contain more embedded devices, which in turn contain services. The device containing the embedded devices is called the root device. The equivalent of a client is the control point. A control point typically has provision to control a few types of services, and can control multiple of these services at once (see Figure 7).

The control point coordinates (root) devices using UPnP action commands to initialize, configure and to make devices ready to transfer content from amongst each other. The devices do not interact directly with each other and use a non-UPnP "out-of-band" communication protocol. The control point is not directly involved in the actual transfer of the content. In Figure 8 an example of UPnP interaction between standardized devices is shown. The control point initiates the transfer from MediaServer acting as a source to a MediaRenderer acting as a sink.
6. STORING AND RETRIEVING METADATA USING UPNP’S CONTENT DIRECTORY SERVICE

The content analysis based functions described in section 3 generate metadata that need to be stored and retrieved. For these purposes it must be prevented that users need to interact directly with the device containing the content (e.g. the user should not have to walk over to the server device). In order to enable this capability, services need to provide a uniform mechanism for UI devices to browse the content on the server and to obtain detailed information about individual content. The UPnP Forum standardized the content directory service offered by the UPnP MediaServer. The Content Directory Service (CDS) provides a set of actions that allows the control point to enumerate the content that the server can provide. The primary actions of this service are browse and search. These actions allow control points to obtain detailed information about each content item that the server can provide. This information includes metadata such as name, artist, date created, file size, etc.

The CDS defines a hierarchical class system to represent the different types of objects that are managed by the CDS. The base class, from which all other classes are derived, is named object. An object is a data entity that can be returned by the CDS from a browsing or searching action initiated by the control point. A class identifies the minimum required and optional set of properties (expressed in XML) that must be present on that object. There are two sorts of first-level classes derived directly from object: item and container. An item represents a single piece of AV data; a container represents a collection of objects (see Figure 9). Containers can represent the physical organization of objects (storage containers) or logical collections.

A CDS object is represented by a property. Three metadata namespaces are used to describe such a property: DIDL-Lite, Dublin Core (dc) [9] and UPnP (upnp). (see Figure 10). DIDL-Lite is derived from a subset of DiDL, a Digital Item Declaration Language recently developed within ISO/MPEG21 [10].

An advantage of distributing metadata in the network is that content analysis algorithms on other devices can make use of this metadata and can benefit from it. Another advantage is that metadata generated by different content analysis algorithms can be combined using another content analysis algorithm. In this way it is possible to get better results, e.g. combining metadata generated by an audio analysis algorithm (e.g. to find loud music/speech) and metadata generated by a video analysis algorithm (e.g. action scenes) to make an automatic video preview.

6.1. Storing metadata

Storing metadata in the CDS can be done by using the specified extended properties or by using the descriptor namespace. The AV working Committee has specified extended properties, which can store metadata (some generated by content analysis algorithms or manual annotation) in the CDS. Examples of specified properties are base properties like title, creator, etc. and people

```xml
  <item id="7" parentID="2" restricted="false">
    <dc:title>SPATION Team</dc:title>
    <dc:date>2003-63-27</dc:date>
    <upnp:class>object.item.imageItem.photo</upnp:class>
    <res protocolInfo="bttp-get:*:image/jpeg:*" size="20000">
      bttp://110.0.6.1/getcontent.asp?id=7
    </res>
  </item>
</DIDL-Lite>
```

Figure 10: DIDL-Lite example returned by a browse action.
involved like artist, actor, etc. One of these extended properties is the userAnnotation. This is a general-purpose tag defined by the committee, where a user can annotate an object with some user-specific information. The userAnnotation property can be used to store content analysis generated metadata, which cannot be stored in the other specified properties.

Another way to store the metadata is to add a descriptor tag to the DIDL-Lite description. Using the descriptor tag gives the vendors the possibility to extend the specified properties by placing blocks of vendor-specific metadata into the descriptor. A descriptor is employed to associate blocks of other XML-based metadata with a given CDS object. Examples of other XML-based metadata include DIG35 [11], MPEG7 [10], RDF [12] or XrML [13]. Descriptor blocks could also be employed to contain vendor-specific content ratings information, digitally signed rights descriptions, etc. The contents of each descriptor must be associated with only one namespace. But there can be more than one descriptor in the DIDL-Lite description.

The big disadvantage of the previous explained methods to store the metadata is that they are completely vendor specific solutions. A trade-off must be found between total implementation freedom and completely standardizing output of content analysis algorithms. We use the prototype system developed in the SPATION project to explore this trade-off and recommend extensions of the standard where needed.

6.2. Retrieving metadata

When the content is stored in the CDS there are two ways of retrieving it, by a browse or a search query. A browse allows the client to incrementally browse the hierarchy of the Content Directory objects exposed by the Content Directory Service. A search allows the caller to search the content directory for objects that match some search criteria like title, date, descriptor etc. The search criteria are specified as a query string operating on properties with comparison and logical operators.

The consequence of storing the most of the metadata in a proprietary way described in section 6.1 is that the retrieval of this metadata is a problem. When a control point searches for e.g. a descriptor of a video, the complete descriptor is returned, including all metadata it contains. Afterwards the included metadata needs to be parsed on the non-powerful control point to reconstruct the wanted metadata.

In the previous sections it became clear that UPnP is a good standard but does not solve everything. In our opinion overall it is a good trade-off. But still some standardization needs to be done to store and retrieve the metadata in an interoperable way between different vendors.

7. CONCLUSIONS

CE devices become more and more powerful, have large amounts of storage space and become interconnected. When users have more storage space they collect more and more content and in this way they get enormous content collections in the home. The large amount of content and the fact that it becomes distributed over the home network makes retrieval a challenging task. Content analysis algorithms used for content navigation and retrieval help the user with this task. The UPnP network middleware standard can be used to support storing, accessing and searching for content in the home network. When metadata needs to be shared between devices in the network, standard descriptions are needed. The UPnP standard partly standardizes these descriptions and allows for making proprietary extensions. For a generic standard it is important to keep a good balance between simplicity and completeness of the standard. At the moment UPnP offers limited support for content-based search and retrieval. Ultimately, descriptors that are often used need to be added to the standard while less general ones can be used in proprietary services. In the SPATION project we have developed a prototype system to support the investigation of these issues. In the future we will extend the system with new functions as well as evaluate it by performing user tests.

8. ACKNOWLEDGEMENTS

SPATION is funded by the European Union under the IST FP5 program. We would like to thank all the partners of the SPATION consortium (Institut Eurecom, France, Philips Research, The Netherlands, Technical University of Denmark, Denmark, Tomorrow Focus AG, Germany, University of Brescia, Italy) for their contribution and work. More details about the project can be found on the SPATION project website:
http://www.extra.research.philips.com/euprojects/spation/

9. REFERENCES


http://www.havi.org/


[8] UPnP, The Universal Plug and Play Forum,
http://www.upnp.org/


http://www.w3.org/RDF/

[13] XrML extensible rights Markup Language
http://www.xrml.org/