Grant Agreement No.: 258378

FIGARO

Future Internet Gateway-based Architecture of Residential Networks

Instrument**:** **Collaborative Project**

Thematic Priority: **THEME [ICT-2009.1.1] The Network of the Future**

Overview of the unified content access

Due date of deliverable: 30.09.2011

Actual submission date: 30.09.2011

Start date of project: October 1st 2010 Duration: 36 months

Project Manager: Henrik Lundgren, Technicolor R&D Paris

Revision: v.1.0

Abstract

This document describes the unified content access system we are proposing in the context of the Figaro project. This system is mainly based on two different layers: the virtual file system and the Digital Asset Management (DAM). Two additional functionalities take advantage of the virtual file system view: the backup and the social-aware caching.

First, we discuss in this document the principle by which content, residing on distinct devices in the home network, can be accessed and shared by any home user at the file system level. Then, we discuss the digital asset management capabilities, such as searching, browsing and indexing. We propose an extension of the DAM to allow networked applications residing on satellite devices in the home to remotely access the content *as if it were on the local file system*. Finally, we shortly present the backup functionalities.

Project co-funded by the European Commission in the 7th Framework Programme (2007-2013)		
Dissemination Level		
PU	Public	✓
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Document Revision History

Version	Date	Description of change	Authors
V 1.0	30.09.2011	Final version submitted to the EC.	Polito (Editor), THRDF, Eurecom, TRDP (technical review), Martel (formal review and quality control)

Table of Contents

1 INTRODUCTION

Today, users enjoy creating and collecting a rapidly growing amount of digital content, such as digital photos, music and videos. New Internet storage services increase this trend. Users expect to share their content with their family, friends or even wider communities making their content available not only over their own devices but also over other devices handled by third parties. Unfortunately, these devices have different capabilities and interfaces. Consequently, many users are discouraged by the complex challenges of accessing, moving, organizing and formatting their large digital content libraries. Indeed, many protocols and technologies can be used to solve the home network content access and sharing issues: popular network storage protocols (NFS, SMB, FTP, HTTP), the established UPnP/DLNA residential networking protocol and the IETF Web DAV (Web based Distributed Authoring and Versioning) HTTP extension protocol. However, although these different protocols are effective in their own domains, none of them covers all the home network requirements.

The most accomplished system in terms of access unification is based on the unified UPnP/DLNA protocol. However, UPnP/DLNA still faces some limitations. First, as new technologies or opportunities appear, the standard needs to evolve, establishing different releases and creating backward compatibility issues. For instance, UPnP/DLNA initially solved the problem of sharing AV content in the home and is currently being extended in order to also support commercial content delivery in the home. Second, the approach requires a device to either natively support the protocol or to provide ways to be upgraded (at least to allow further software installation), which is generally very problematic in many Consumer Electronics (CE) devices as TV sets, mobile phones, etc. In the UPnP/DLNA case for example most TV sets now claim being compliant with UPnP/DLNA, but they generally cannot be upgraded. In order to benefit from the next release of the standard (as commercial content delivery support), users will have to change their TV set.

For these reasons, we choose to explore an alternative solution, not imposing any additional software installation on the devices of the home network. Consequently we propose to use their native communication protocols and, to solve the interoperability problem, we propose to connect each device to a central equipment, to which we devote the role of protocols translator. Software updates are now necessary only on this central equipment. This equipment should be connected and powered on most of the time. It may be under the control of a service provider so as to be upgraded when appropriate, as is typical for a home gateway. Additionally, we propose to implement this translation at the file system level. This choice, driven by content management storage concerns, ensures to keep compatibility with all applications manipulating stored content.

In this document, we propose an innovative solution based on a gateway-centric architecture where the gateway is in charge of translating the native content sharing protocol of any connected device. We present the architecture of the unified content access system of the Figaro project. It firstly describes the gateway architecture. Secondly, the document discusses the unified file system view architecture and the advantages it offers. The document then presents the Digital Asset Management (DAM) subsystem and, more specifically, how local and external software applications can deal with it. We discuss the API plugins that can be provided for interacting remotely with the DAM. Finally, the document provides an overlook of the backup mechanisms that will be addressed by WP4 in the upcoming deliverables.

2 OVERVIEW OF THE UNIFIED CONTENT ARCHITECTURE

In this section, we introduce the gateway file management architecture providing home users (and external gateways) with a unified mechanism to access content, regardless of its location in the home network. The gateway file management, illustrated in *Figure 1*, includes the following components:

- The unified file system view, mapping home network content into a file system representation.
- The DAM achieves access unification at content level, abstracting the location of the content.
- The backup module performs content backup either on local resources or on external, federated resources, leveraging the home users' social contacts.

We first provide an overview of its components, and then proceed to describe each of them in detail.

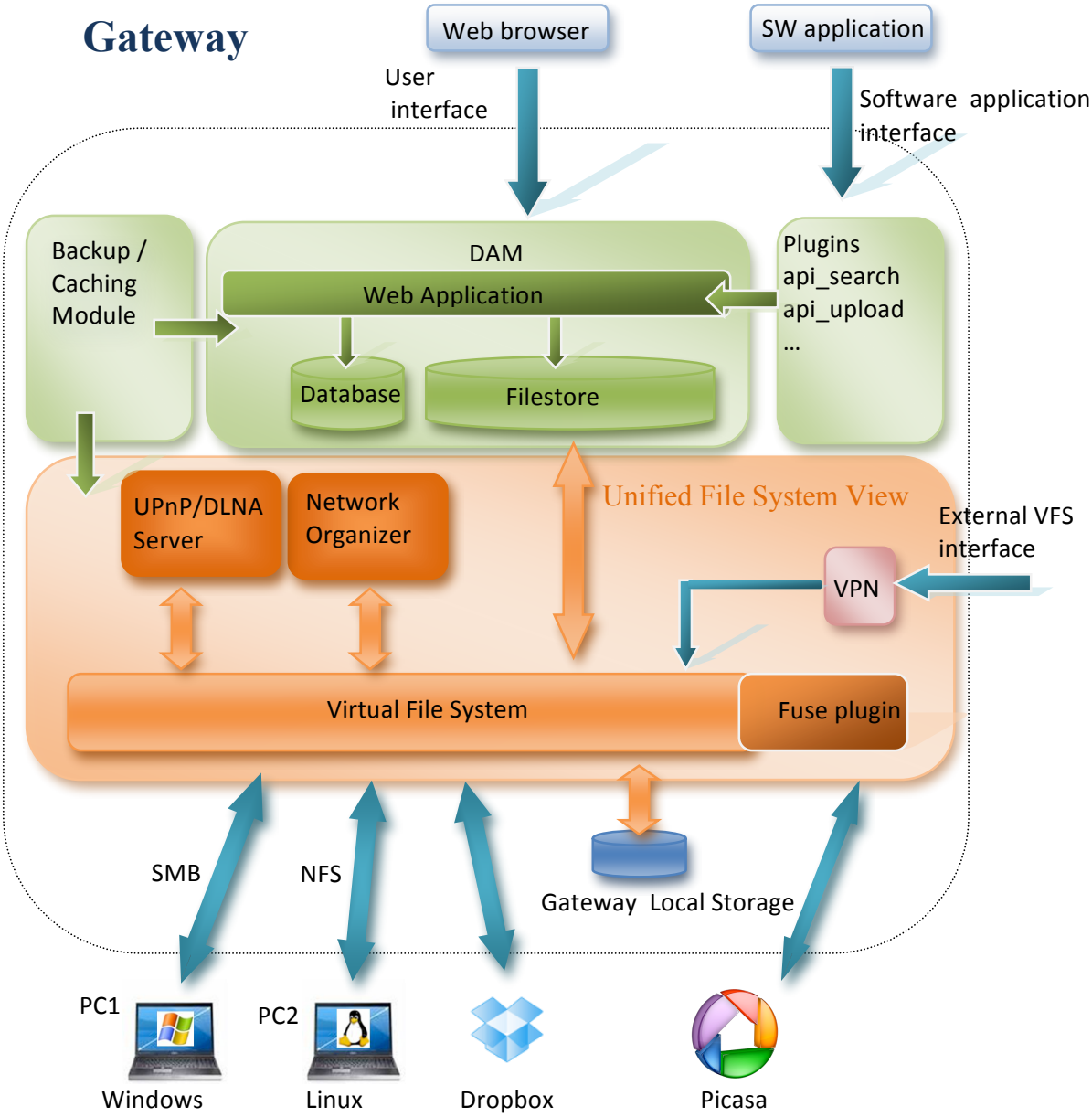

Figure 1: architecture of the content management module in the gateway

2.1 Unified File System View

The Unified File System View unifies the file systems and the network protocols. It offers a file system view of the home network. It is based on the Virtual File System, VFS, provided by the Linux operating system. It therefore represents the unified file system view of the home network content as if it were a local file system interface. It also contains a network organizer module, which is in charge of managing the registration/mounting/unmounting operations and detecting which device leaves the home network. The Unified File System View mainly provides local file system access. As such, mainly local devices (i.e., running on the same network), and properly configured via the Network Organizer can access the gateway at the network file system level. Remote access could also be provided at the same level using a VPN connection.

The Unified File System View layer also includes a UPnP/DLNA media server, running on top of the file system view to expose all the home network content towards UPnP Control Points and Media renderers.

2.2 DAM: Digital Asset Management

The DAM is an application, running on top of the unified file system view. It allows tasks such as indexing and content retrieval. It is based on a LAMP (**L**inux **A**pache **M**ySQL **P**HP) software package [1]. It uses the file system interface provided by the VFS (virtual file system) layer. Each resource has a unique resourceID, which is allocated by the DAM when a new resource is imported. It contains a local storage area, called *filestore* and a metadata database MySQL:

- filestore: a directory tree containing the uploaded content. A file is stored (or a symbolic link is created) and renamed resourceID.ext in the filestore. For example, picture01.jpg is stored in filestore as 152.jpg. These folders contain also icon and preview of the pictures.
- Database: the database contains metadata related to resourceID: file type (mp3, jpg…), tags like id3 or exif tags, the keywords related to the resource that can be used to find the resource.

The DAM also provides a set of API allowing control by remote software applications.

The DAM provides 2 types of access interfaces: the user interface and the software interface, which are described hereby.

2.2.1 User interface

The user has access to the system via a browser. Two types of Front-Ends are available in the DAM: one for PCs/laptops and one for Smartphone (with touch screens). The user access is protected via HTTP Authentication. Therefore, the user needs to be registered in a DAM/Gateway so as to access it. Once authenticated, the user can access the DAM either locally or remotely.

2.2.2 Software application interface

The DAM can be interfaced to another application using an HTTP interface. This is also discussed later. Assuming the application is properly authenticated, it can be interfaced to the DAM whether it runs locally on the same gateway. The application can reside on a remote device like PC or smart phone.

2.3 Backup and caching modules

The backup and caching modules interact with the DAM in order to create copies of home content following users' directions and interests. In this document we will refer to "backup" when talking of reliable, encrypted copies of user content encoded in blocks and uploaded to a storage area (possibly outside the home network). Conversely, we will refer to caching when talking of multiple copies of the same content disseminated among neighboring gateways for the purpose of having the content readily available when needed (though not in a reliable way, i.e., the content may need to be downloaded from other sources if not found in the caching system). Backup and caching modules both exploit the unified content access in order to manage all the resources in a transparent way, saving into the backup (or cached content) all the information about user-generated and file-system-level metadata along with the resources themselves. They also query the DAM database to retrieve the user-generated tags associated to each resource, to be stored and saved as metadata. These metadata will then allow the selection of appropriate destinations for the backed-up content (e.g., DAM-level user-generated tags such as "wildlife pictures" could trigger the backing up of users' photos toward storage facilities shared by federated users with similar interests).

3 THE UNIFIED FILE SYSTEM VIEW

This section provides a detailed description of the distributed file system view, outlining the advantages, from a user's perspective, of accessing home network content through a file system view. Some ideas can also be found in [2].

In the gateway-centric architecture, shown in *Figure 2*, content access and protocol translation are realized at the file system level. To this purpose, we exploit the VFS layer from Linux operating system. The VFS was developed in order to ease the addition of new file systems into the Linux kernel. It is frequently used in Unix-like operating systems to facilitate the integration and the use of several file system types. The basic functionality of the VFS is to hide the file system type used on the system. Furthermore, the VFS allows a local client application to access network storage without this client application noticing the difference. Further details on the VFS can be found in [3] [4].

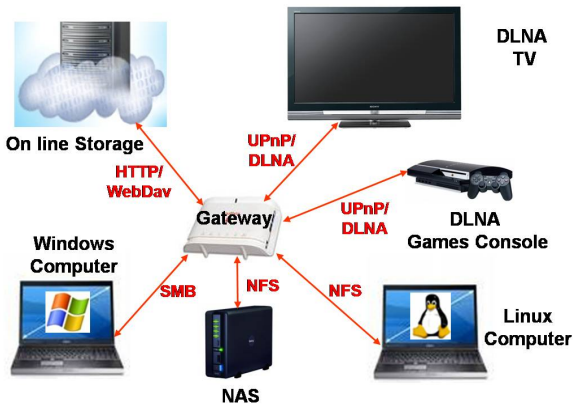

Figure 2: Gateway-centric architecture

Leveraging VFS capabilities, the shared content of every connected device can be represented as a dedicated folder accessible from the file system of the gateway. For example, the machine of a user John will appear in the gateway with a folder name "John_dir", once the machine of John is mapped by using the VFS. Each dedicated folder is the mounting point of the corresponding device in its native network protocol, for each of which VFS offers an adaptation module. However, the implementation of the module, its configuration and the creation of the mounting points require computer skills that not everybody has. The challenge is to make it possible for any user to overcome this obstacle. To this purpose, we developed a piece of software, called "Network Organizer", which automates the creation of the different folders in the gateway dedicated to the different clients. This piece of software runs on the gateway and will be described in the next subsection.

The VFS can be integrated with several local or remote storage systems, as described in *Figure 3*, and further explained below.

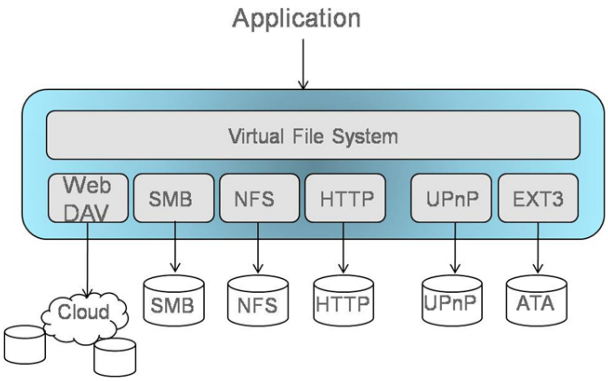

Figure 3: Virtual File System and its integration with local and remote systems

3.1 The Network Organizer

The Network Organizer runs on the gateway and it is in charge of managing the registration of clients, e.g., upon the first connection of a client and the "connection to the gateway" or "disconnection from the gateway" of user devices. All theses actions are done within the home network. The client can execute one of these actions by using the WEB browser of his machine connected to the gateway via HTTP protocol.

In other words, the network organizer is a set of Shell scripts running on the gateway in charge of managing these different actions. Each script has a dedicated role and can be called by an upper layer application. Passing arguments to theses different scripts enables management of different devices' registration/connection/disconnection. These scripts are in charge (i) of registering a new device in the home network and building dedicated folders on the gateway in the file system tree structure, (ii) of connecting the device, and (iii) of disconnecting it. We proceed to describe these different procedures in more detail.

3.1.1 Registration

The new-coming user shall first contact the Network Organizer via its web browser. His first operation is to register himself/herself on the Home Network. The details of the registration procedure, which are shown in *Figure 4* through to *Figure 6* is done only once, when the device joins the home system for the first time. John and Lea are two users who want to join the home network, each with their own client. Thanks to the initial HTTP connection, the Network Organizer collects a certain amount of information about the device and the attached user account, such as operating system, IP address, login and password. The collected information will be used by the Network Organizer first to build export/import folders on its side devoted to the newcomer, and secondly, to generate a run-time script (bash/shell) dedicated to the operating system of the user's device. The script is proposed to the user's device for upload and execution via the web browser. When executed, the script builds on the user's device appropriate folders (import and export). At this point in time, everything has been prepared so that the new user/device can join (connect to) the home network.

- Build folders in the gateway
- Build script for clients
- Entry point created on gateway

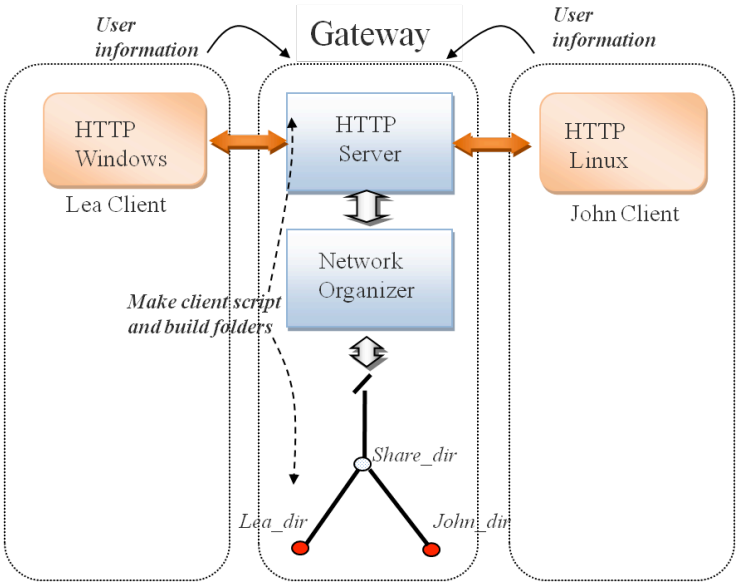

Figure 4: Registration operations (i)

- Script built by the gateway is downloaded to the client.

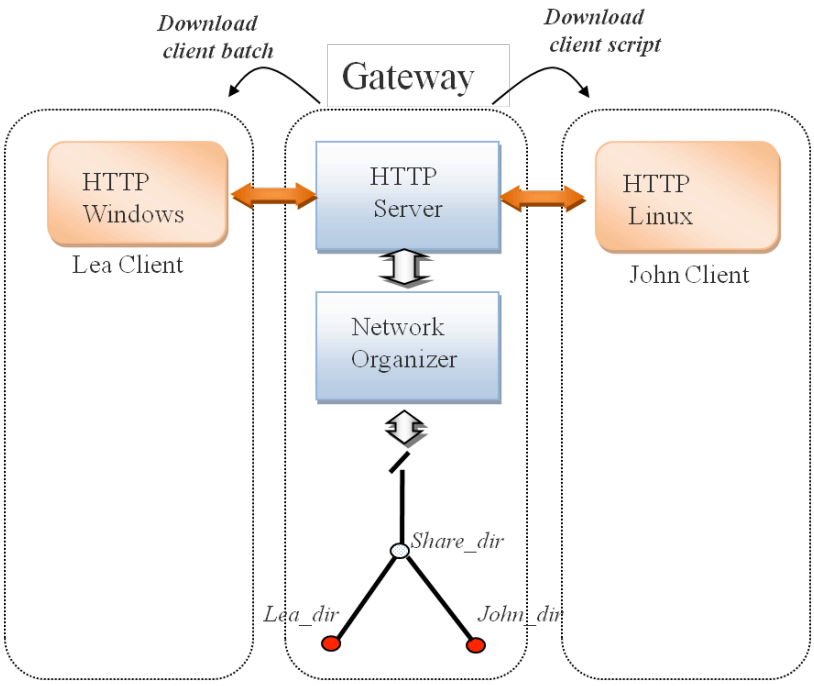

Figure 5: Registration operations (ii)

- Downloaded script is executed
- Build folders on the clients
- Folders are ready to be mounted/exported

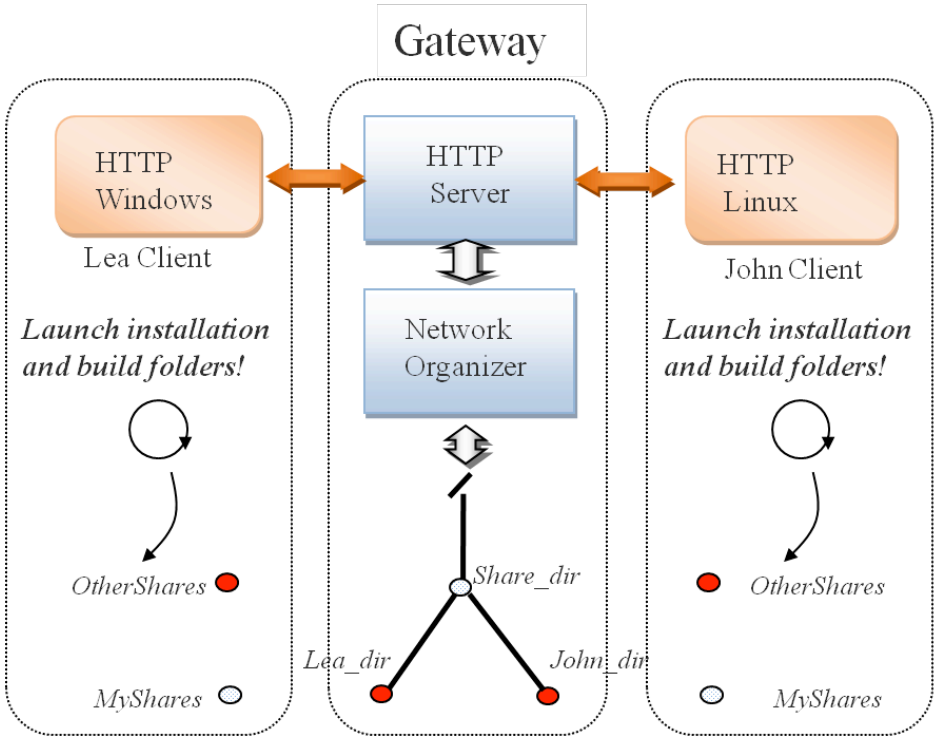

Figure 6: Registration operation (iii)

3.1.2 Connection

The connection procedure, shown in *Figure 7*, may be done just after the registration and has to be done each time the user/device joins the home network, for instance at each power-on of the user's device or when a user returns to home carrying a mobile device.

The connection procedure activates the mounting points prepared by the registration procedure. After this step the content of an exported folder of the user's device is now available into the tree folders of the file system structure of the gateway, as it is part of the local gateway file system. This content is exported from the gateway to the rest of the network. The content is now visible by every connected device. Similarly, in the other direction, the content shared by the other devices of the home network appears as a mounted folder in the tree structures of the local file system of the device. Note that, for the sake of security, the content appears as read only, therefore the different clients will access other clients' content only in read mode.

- Folders are exported/mounted
- Client can access to the content shared
- Client can share its local content

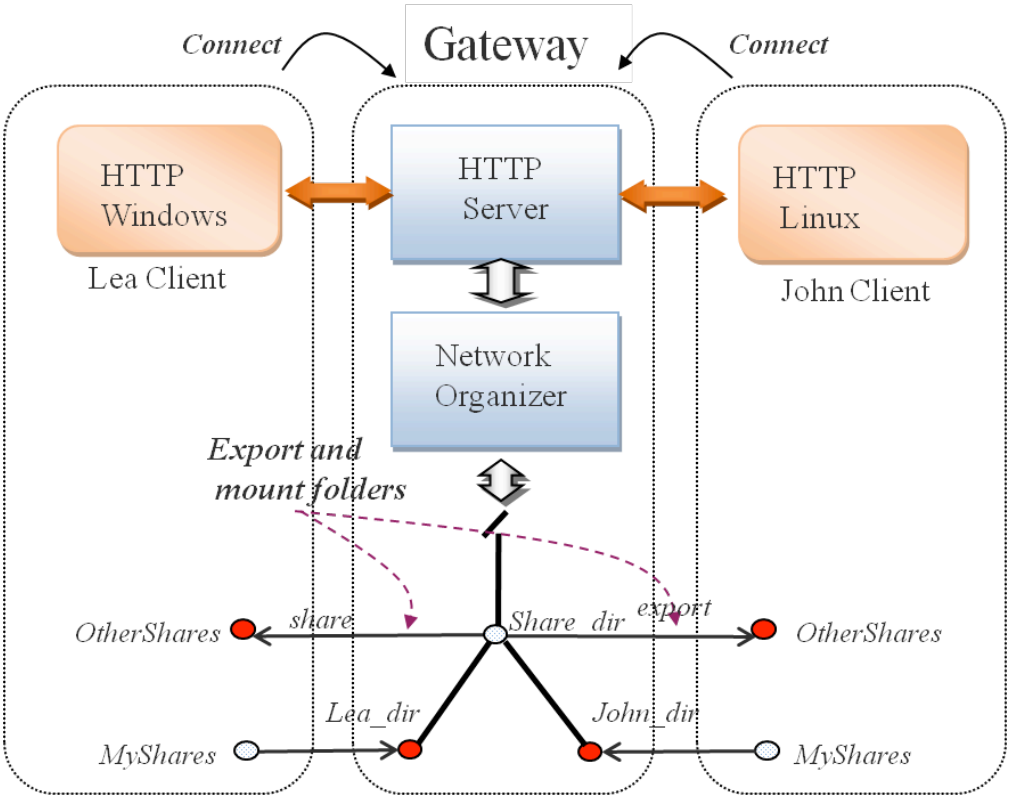

Figure 7: Connection operations

3.1.3 Disconnection

The disconnection procedure in *Figure 8* shall take place when the user/device leaves the network. It consists in unmounting the user's exported folders in the gateway. The user's device will not be part of the network anymore. The other devices still connected to the home network will not be able to access the shared content of the disconnected device, which now appears as "disconnected" to them. When a user, for example Lea, is disconnected, several options are possible: i) hide the name of the disconnected user (remove "Lea_dir"), ii) display the name with no changes (Lea_dir), or iii) display it as disconnected ("Lea_dir_disconnected"). Removing the name of the disconnected user can be misunderstood by the rest of the users still connected: they may be under the impression that the content has disappeared. Leaving the "Lea_dir" folder with no files available inside can give the impression that the content has been deleted. Thus, we propose to replace the original connected name ("Lea_dir") with "Lea_dir_disconnected". This solution suggests to the users still connected that there may no content available under "Lea_dir_disconnected" for the time being, but that it will reappear when Lea connects again.

The disconnection procedure can be initiated by the device itself (as part of the shutdown/standby procedure). Other, more abrupt disconnection events may occur, though, as devices may be disconnected from the network simply by cable removal or by a user leaving the home carrying her mobile device. Consequently, the Network Organizer has to monitor the device presence, for instance by regularly pinging such devices.

- Folders are not unmounted by the gateway
- Unmounted folder appears as disconnected

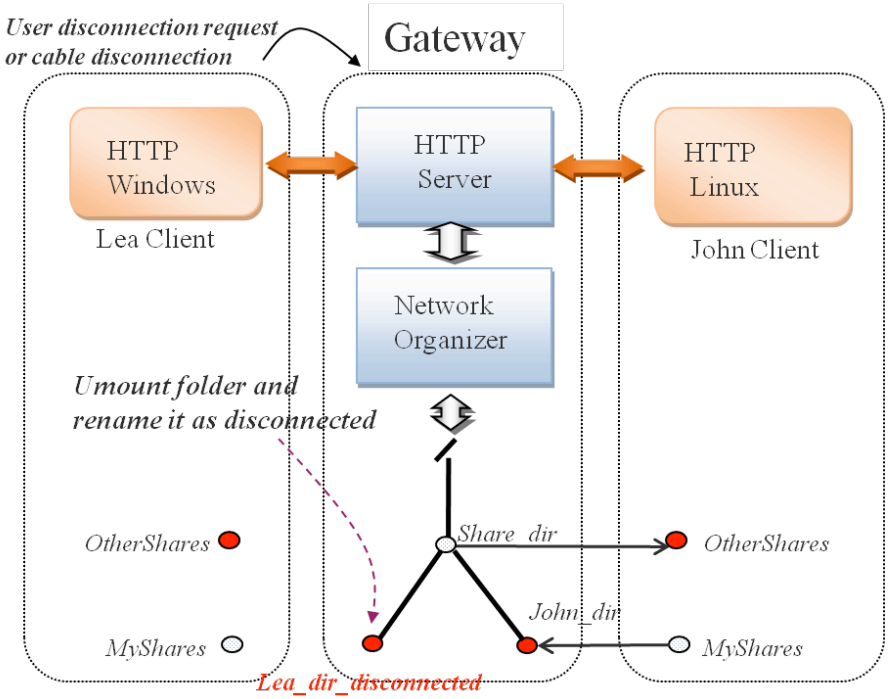

Figure 8: Disconnection operation

3.1.4 User Right Management

When mounting folders to the VFS, basic Unix file system right management is used. There is at least one user account in every device of the home network. This user account is exploited by the network organizer to create the mounting point on the VFS. By default it is assumed that every user of every device has read access to the unified file system view. Only the Gateway has write access on the overall file system view. Each user of a device is granted read/write access on the folders that are created for her by the Network Organizer. This basic right management policy is considered sufficient for sharing data within the home network and at that layer. Additional, more sophisticated right management features are available at the DAM layer.

3.1.5 Multi users / Multi machines mode

A user can be part of the home network by logging onto different machines or using different portable devices. For instance, user John can use his personal computer to access the home network content, but John can also use Lea's computer to access the content. The system also allows several users to

use a single machine. For instance, users John and Lea can have their personal accounts on the same machine. John will be able to access and share content and Lea will be able to access and share the content too. In the sample scenarios below, we explain the different possible folder representations supported by the architecture.

Scenario 1:

John uses two different machines to share and access the content.

Figure 9 illustrates the folder representation in the gateway, when John is connected with two different machines. The names PC1 and PC2 are the hostnames of the machines. In the example, user John uses PC1 and/or PC2 to access and share the content. So the content is located under folders "PC1" and "PC2" in order to differentiate its origin.

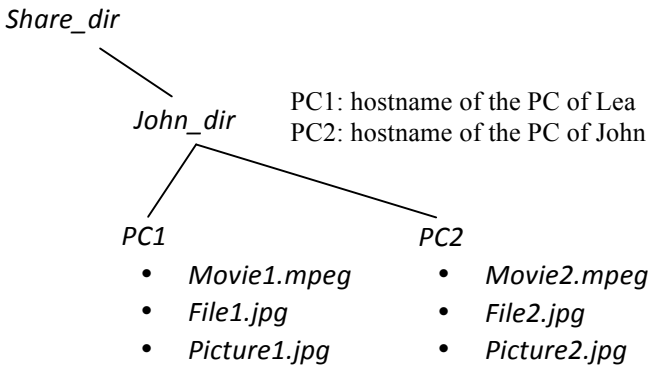

Figure 9: Scenario 1 - single user connected onto two machines.

Scenario 2:

Lea and John are connected onto the same machine to share and access the content.

Figure 10 illustrates the folder representation in the gateway, when John and Lea are connected onto the same machine. The name PC1 is the hostname of the machine. John and Lea use the same machine PC1, so the folder name "PC1" is created under "John_Dir" and "Lea_Dir".

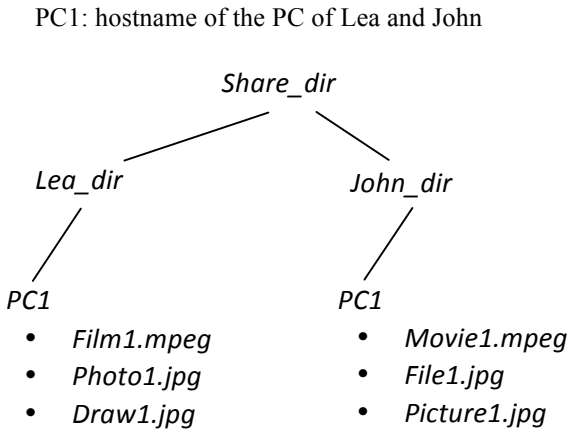

Figure 10: Scenario 2 - two users connected onto the same machine

Scenario 3:

Lea is connected onto her machine (whose hostname is PC1) and John uses his machine (hostname: PC2) and also Lea's machine, PC1, to share and access the content.

Figure 11 illustrates the folder representation in the gateway, when John is connected onto two different machines and Lea is connected onto one of the machine that John uses.

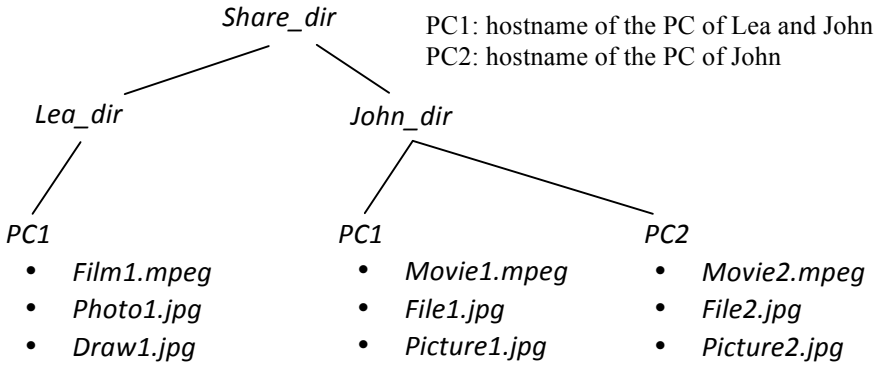

Figure 11: Scenario 3 - two users connected onto two machines

Scenario 4:

Lea and John are connected onto the same machine, PC1. Lea and John are in the same group, so they have the same rights access to the files stored on the machine PC1, typically read/write mode. John can access the Lea's files, and Lea can access John's in read/write mode. John is also connected onto another machine, PC2. When mounting the different folders of the different machines onto the gateway, the access rights are fixed by the gateway in read-only mode. Every user will access the content in read-only mode. The file owner will only be able to access his content in read/write mode by modifying his files located on his machine and not through the gateway. *Figure 12* illustrates the folder representation in the gateway, when John is logged onto two different machines and Lea is connected to one of the machines that John uses. Even if Lea and John are in the same group on PC1 (a Linux machine, for example), they can access the content in read-only mode through the gateway.

PC1: hostname of the PC of Lea and John. Both users are in the same group.
PC2: hostname of the PC of John

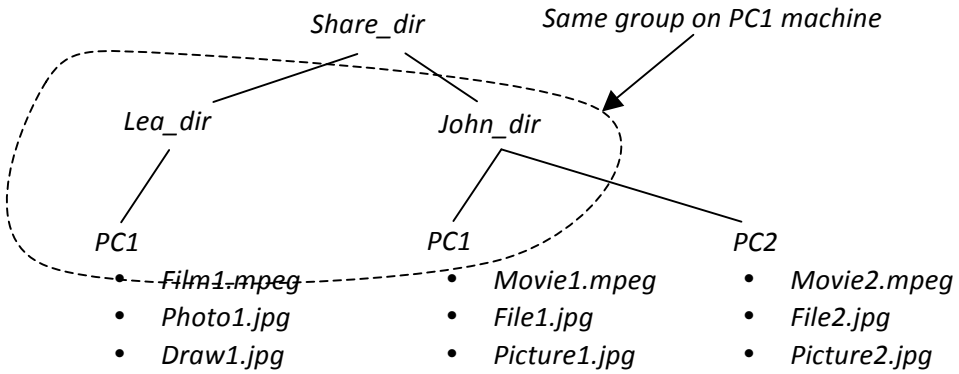

Figure 12: Scenario 4 - two users in the same group connected onto two machines

3.2 Integrating UPnP/DLNA devices

UPnP/DLNA protocols can be interfaced to the VFS at two levels. A UPnP/DLNA Media Server running on the gateway on top of the VFS is in a position to offer to UPnP/DLNA clients the content shared by the users connected. Therefore, the installation of a UPnP Media Server allows the files available locally on the gateway (the folders structure of the gateway with the different mounted/exported client's folders), to be exposed towards any UPnP/DLNA Player and/or Controller. In the opposite direction, a UPnP FUSE (File system in UserSpace) module like *DJmount* will map under the VFS all the available UPnP/DLNA Media Servers on the network and their attached offered content. Local applications running on the gateway can therefore see this content (present on remote UPnP/DLNA Media Servers) as if they were locally available from the local file system interface. It can also be exported towards remote client using other network file system specific export functions (NFS, SMB) as described later.

UPnP/DLNA natively includes a discovery protocol so that no specific action is required when UPnP/DLNA devices enter or leave the home network. This implies the devices are DLNA-compliant, which is not always the case. In our solution, there is no discovery protocol, but instead the user declares his machine during the registration operation whatever the operating system is. The Network Organizer performs the UPnP/DLNA installation on the VFS.

3.3 Integrating Cloud storage systems

Nowadays, an increasing number of providers offer storage services to Internet users, supporting a wide range of applications from raw data storage (Dropbox, Box.net, Carbonite, etc) to photo (Picasa, Flickr, Facebook, etc) or video sharing (YouTube, Vimeo, Dailymotion, etc). Most of these servers provide FUSE-compliant software that can be installed at the VFS level so as to seamlessly integrate home and cloud storage. When they do not provide the code themselves, they generally provide an open interface so that third parties can develop applications, easing the integration of those cloud storage services with other applications. Additionally, the Open Source community often develops FUSE-compliant software.

The integration of Picasa and Youtube has been tested by adding GdataFS to the VFS. Raw data storage service has been tested too by integrating Dropbox software to the VFS. Such software installations need to be done only once.

3.4 Integrating SMB/NFS devices

A last important part of home network devices consists in the personal computers themselves. As file system level network protocol, they natively support SMB (for Microsoft Windows systems) and NFS (for Unix-like systems).

In such devices, the Network Organizer creates dedicated folders, basically one devoted to shared content exported by the device and one devoted to import content shared by other devices. At the same time, the shared content of each device appears in the gateway as a part of its own folder tree.

The network protocols (SMB/NFS) expect the devices to go through a procedure when entering (connecting) or leaving (disconnecting) the system. This raises the issue of sorting out these complex manipulations without any additional software installation on every user's devices.

4 DIGITAL ASSET MANAGEMENT

The Digital Asset Management (DAM) is an application running on the gateway. It exploits the virtual file system facilities. It requires only the support of a web browser on an end device. The DAM is derived from the Resourcespace application [5] which is an open source solution which does fit completely with our requirements.

The term DAM is used here to refer to all the different tasks and applications used to ingest, to index, to catalogue and retrieve digital assets (content) such as documents, digital photos, music and videos based on metadata embedded in the digital assets or at least file names. Whole home network content can be indexed by the DAM application and the results (i.e., the name of the content, the location of the content, and, possibly, a preview or a jacket) stored in a database on the gateway. Users can browse this database from their PCs, their Smartphones, or any device equipped with a web browser, the DAM being in charge of building the web pages adapted to the device screen size and resolution. The support of query services by the DAM, using keyword search is also straightforward and enables more powerful content retrieval by the end user. Lastly, to play content, users will transparently run the local media players or take advantage of the ones supported by the DAM itself.

The basic function to import content to the DAM is to use the upload function. This function is available via a form and can be filled within a browser. The upload will physically copy the data from the original device (running the browser) into the filestore area. Access to the data can be finely managed using the concept of *collections* (see [5] [6]).

The design of the original Resourcespace source code has been modified so as to be able to import new content without requiring a physical copy into the filestore. To do so, a new function has been provided allowing the DAM to parse a part of the file system view (rooted at some folder). When parsing the folders, the DAM will populate the database with metadata.

4.1 DAM Front end towards the user

The DAM requires users to have an account in the DAM, and then uses HTTP authentication. As such it can be used inside or outside the home.

The browser has to be pointed to http://GatewayIPAddress/resourcespace/. GatewayIPAddress must be set to the local home network IP address or to the external public IP address depending whether it is accessed from the home or outside the home.

The DAM natively provides functions to upload data, and then search data (see [5] for more information). These functions are available both inside and outside the home network.

The front end has been extended to add additional functions (buttons) to use the file system view as created by the Network organizer. As such, these buttons will be working only when connected inside the home network. These buttons are:

- Share: will extract metadata from the exported folder (as exported by the Network Organizer), and generate a symbolic links towards the physical files.
- Share&Sync: Same as before, but in addition it creates a copy of the files into a central location, so that files are always available
- IndexingPicasa: same as Share but on the Picasa folder view
- IndexingDropbox: same as Share but on the Dropbox folder view

4.2 DAM Front end towards Smartphones or Tablets

We have developed a front end for touch screen devices, allowing the interaction in *Figure 13*. However, only a reduced set of functions are available, mainly login, search and play. There is no function to upload content from that interface.

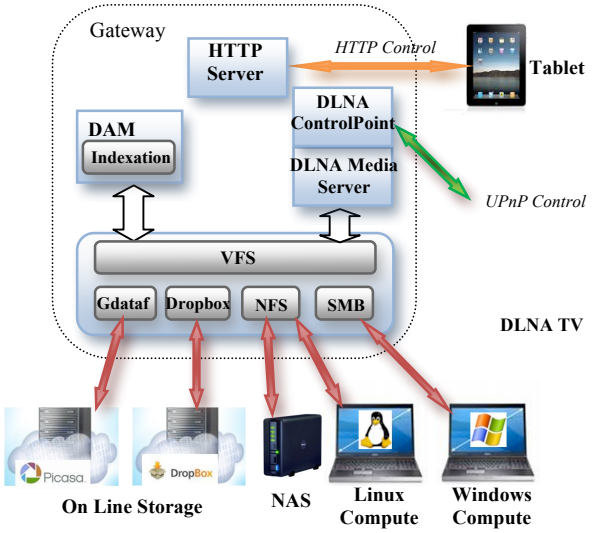

Figure 13: Tablet as a remote controller

4.3 DAM FrontEnd towards Software application

Resourcespace plugins have been extended from version Release 4.0.2429 (28 February 2011) to include remote APIs. For a complete list, see [7].

Additional information on Resourcespace remote APIs can be found in the Resourcespace Wiki pages [8].

Since this is a working document and we are still developing the remote access to the DAM, its APIs will be extended according to future needs. Here we give a simple example with search and upload functions to understand the principle.

Current DAM version supports a set of APIs, among which two are of particular interest for the Figaro project :

- api_search - allows API access to search resourcespace (requires api_core)
- api_upload - allows API access to upload new resources (requires api_core)

api_search example:

The search operation (like search operation using the UI) can be launched by a software application by making an HTTP request with arguments. This HTTP request can be launched manually in a URL, or can be launched by a software application.

Its format is: `http://url/plugins/api_search/?key=[authkey]&[optional parameters]`

The detailed information on the search operation on Resourcespace can be found in [9].

We extend these APIs so as to provide additional functionalities.

The current version of Resourcespace includes a MD5 computation that can be used for various purposes (deduplication), and can be used as a unique file identifier shared by the DAM and other apps.

The following modifications will be introduced:

Add MD5 parameter as an output of api_upload()

Create some new apis, for instance:

- Api_add_tag (MD5, tag)
- Api_delete_tag (MD5, tag)

4.4 DAM Right Management: collections

Resourcespace defines the concept of collections that allow a user to share part of his/her resources with some other users in a very flexible way. By adding resources to a collection, users can build up their own personal selection of images, stories, videos and saved searches. ResourceSpace manages and remembers collections so they are available whenever a user returns and from any computer from which a user accesses ResourceSpace. Collections act in a similar way to 'shopping basket' found on many e-commerce websites. Collection management is ensured by Resourcespace for any data located in the filestore area. Specifically, it is available for any data that has been added to Resourcespace using the upload functionality.

We propose to use Resourcespace collection to manage access right management between users for content stored on GWs.

5 OVERVIEW OF BACKUP AND CACHING MODULES

In this section we provide a brief overview of the solutions that Figaro will adopt to address the need for content caching and backup in households equipped with a home gateway that stores, tags and manages the data collected by the home users through the interface of DAM application which is already introduced in the above section. Both modules exploit the unified content access in order to manage all the resources in a transparent way and rely on a file transfer module that copies the content selected for backup/caching onto remote resources.

5.1 Backup module

As explained in the previous sections, each connected device results in a mounted subtree of the gateway filesystem. Once the folder is mounted, the backup module performs synchronization with the GW-attached storage, in order to have a first local backup inside the Home Network. As soon as new device and new folders are mounted, the corresponding synchronized copy is saved on the GW storage.

Specific resources' access rights are maintained as in the original device both at the file system level (in the synchronized copy) and as additional metadata to be scheduled for backup in the federation, as well as all the information coming from the VFS level (position, size, ownership, etc.). In this way, as soon as a resource changes access rights policy (i.e., from read/write to read-only for a certain user), the modification is propagated the same on the gateway's storage and, by consequence, on the backup.

Furthermore (see *Figure 1*) the Backup module will use the DAM level, by querying the DAM database to retrieve the user-generated tags associated to each resource, to be stored and saved as metadata as well.

After the synchronization step and the extraction of the resources' metadata, files are scheduled for the backup on the federated network. Each gateway in the federation offers a fixed part of its own physical resources for storing fragments computed out of the synchronized files.

From this time on, the Backup process will act as follows. Resources scheduled for the backup will be encoded into blocks and spread into the Federation, resulting in other gateways to store those blocks. By exploiting erasure coding techniques, it will be assured that a subset of the stored blocks can be further downloaded into the owner's gateway for restoring after a crash. Whenever, e.g., for other gateways' crashes, the number of encoded blocks reaches a lower bound, a *maintenance* procedure will take place computing new parity blocks to be stored in other locations (see also *Section 5.3*).

5.2 Socially-aware Caching

The Socially-aware Caching leverages the interaction between remote home gateways in a social way, i.e., by exploiting the users' social networking information, so that caching recipients are those gateways whose users are most likely to be interested in accessing the shared content, as explained in [10].

The caching module operates on top of the DAM layer and is composed of a caching decision algorithm module which runs a background caching procedure by uploading selected content owned by home users to selected remote friend gateways, and a social information-gathering module.

In the caching decision module, a low-complexity, distributed heuristic algorithm will be included, to match remote friends' interests with the content to cache by leveraging typical social networks indicators, such as interests, hobbies and preferences of home users, and by having all personal digital

data appropriately tagged.

The module takes two different viewpoints: that of content owners who have data to share or cache and that of remote gateways which provides their own storage space for their social friends. From the viewpoint of content owners, not only do they wish to cache or restore the data as fast as possible, but, in the long run, they also wish that the remote gateway keeps the content for as long as possible. Therefore, content owners are naturally disposed to choose friends from whose gateways they can retrieve the cached items more quickly. Also, they would like friends to be interested in the content they upload, because such friends are more inclined to store it for a long time. At the receiving end, the remote gateway will devote its storage space for friends to maximize the whole benefit accounting for both its user's interest and the bandwidth toward the content owner's gateway.

Although the caching decision algorithm module is running in the background, it still requires the home users to tag which content needs to be cached through the DAM interface. The module must also be able to gather the knowledge of content type tags by DAM APIs (such as an API_read_tag in resourceSpace).

The other module used by the socially-aware caching functions is the social information-gathering module, which allows the gateway to interact with social networks. Clearly, given the specific characteristics of each social network, more than one module will have to be designed and implemented so as to be tailored to Online Social Networks.

5.3 Interactions between Backup and Caching

The Backup module will exploit the socially aware caching functionalities in the *maintenance* step of the process. Cached replicas of a specific content can be used for generating new parity blocks in case the *owner* of the content (i.e., the Backup module running on the content owner's gateway) is offline, and, at the same time, the number of parity blocks decreases below a lower bound.

6 CONCLUSION

In this document, we have presented FIGARO's unified content architecture overview. We have highlighted the simplifications that are introduced by the unified content modules. From a bottom up viewpoint, the VFS module allows the gateway to be connected to remote devices and mount remote folders as part of the local file system, using the native protocols of each device. No software has to be installed on remote devices and no particular skills are required from the user to establish the connections.

This architecture is flexible enough so that applications, for instance a DAM, may effortlessly take advantage of the underlying virtual file system. The DAM can index content on the different connected devices whose folder are mounted by the gateway and it supports various retrieve operations. Additionally, appropriate API plugins can be provided to generic software applications and portable devices to remotely interact with the DAM. The combination of the infrastructure working at the VFS level and the services offered by the DAM running on the gateway provide users with a user-friendly experience of the content and features offered by the home network.

Finally, the document provided an outlook of the backup and caching mechanisms that will be addressed by WP4 in the upcoming deliverables, outlining how these modules can access the DAM and widen their operational capabilities, paving the way for new functionalities.

LIST OF ACRONYMS

API:	Application Programming Interface
CE:	Consumer Electronics
DAM:	Digital Asset Management
DLNA:	Digital Living Network Alliance
FTP:	File Transfer Protocol
HTTP:	Hypertext Transfer Protocol
LAMP:	Linux Apache MySQL PHP
FUSE:	File system in User Space
GW:	Gateway
NFS:	Network File System
SMB:	Server Message Block
UPnP:	Universal Plug And Play
VFS:	Virtual File System
VPN:	Virtual Private Network
Web DAV	Web based Distributed Authoring and Versioning

BIBLIOGRAPHY

1. *http://fr.wikipedia.org/wiki/LAMP.*
2. *S. Defrance, R. Gendrot, J. Le Roux, G. Straub, and T. Tapie, Home Networking as a Distributed File System View, HomeNets 2011, Toronto, Canada, August 2011.*
3. *http://en.wikipedia.org/wiki/Virtual_file_system.*
4. *Moshe Bar, "Linux File Systems", McGraw-Hill, 2001, ISBN 0-07-212955-7.*
5. *http://wiki.resourcespace.org/index.php.*
6. *http://wiki.resourcespace.org/index.php/Working_with_Collections.*
7. *http://wiki.resourcespace.org/index.php/Plugin_list.*
8. *http://wiki.resourcespace.org/index.php/Pluggable_Remote_API_Architecture.*
9. *http://wiki.resourcespace.org/index.php/Api_search.*
10. *J. Jiang, C. Casetti, Socially-aware Gateway-based Content Sharing and Backup, HomeNets 2011, Toronto, Canada, August 2011.*
11. *http://wiki.resourcespace.org/index.php/Plugin_list.*