SemanticHIFI

IST-507913

Public Report of WP7 "Sharing"

Covering period: December 2003 – November 2006

Report version: **1.1**

Report preparation date: November 29, 2006

Writers: **Jens Hasselbach, Patrick Aichroth (FhG), Jerome Barthelemy (IRCAM)**

Control: **Hugues Vinet, Francis Rousseaux (IRCAM)**

Classification: **Public**

Contract start date: **December, 1st 2003**

Duration: **36 months**

Project co-ordinator: **Hugues Vinet, IRCAM**

Involved Partners: Fraunhofer IDMT (WP Coordinator), Ircam-SEL, Native Instruments, Sony EuTec

Project funded by the European Community under the "Information Society Technology" Programme

Table of contents

1 WP Overview

1.1 Objectives

The "Sharing" work package aims at implementing a peer-to-peer (P2P) sharing system that manages all metadata transfers between various Hifi Systems and Authoring Applications. Peer-to-peer describes a communication model in which each party has the same capabilities and either party can initiate a communication session. The SemanticHIFI P2P sharing system provides functionalities for instant messaging, and for sharing and searching user-generated data within defined user groups. Furthermore the sharing system provides means for controlling the shared data in such a way that only user-created data (such as mix files, play lists or metadata files describing music tracks the user has produced using either the Authoring Application or the Hifi System) can be shared. WP7 integrates peer-to-peer functionalities into the Hifi system and the PC authoring application.

1.2 Partners' roles

(1) Fraunhofer IDMT (WP leader): responsible for the development and integration support of the P2P sharing system core libraries (including appropriate mechanisms for managing secure user groups and controlling data to be shared) and for providing required client/server infrastructure

(2) Ircam SEL: responsible for the development of middleware components between the sharing system core libraries and the local metadata database on the Hifi System, described below as "metadata conversion libraries", the shared metadata model and the related publication formats, as well as for the user testing and feedback.

(3) Native Instruments: responsible for the integration of sharing system core libraries into the Authoring Application

(4) Sony EuTec: responsible for the integration of the provided middleware components into the Hifi system

1.3 Synthesis of main achievements

Shared Metadata Model

Due to specific needs and requirements of the industrial partners of the SemanticHIFI consortium, a specific data model has been developed for metadata to be shared. This data model merges several existing models from different initiatives, including Dublin Core, UPnP, DIDL Lite and ID3.

Embedded Sharing Client

The sharing system core libraries can be accessed either via the Hifi System or the Authoring Application. The Authoring Application utilizes a Java Applet in order to provide a graphical user interface for the sharing system.

The sharing system core libraries provide the functionalities for sharing, searching, messaging, and supports management of different kinds of user groups. Users can create and join their own groups using either an open membership policy (unprotected) or a closed membership policy (password protected – the weak algorithm of the original implementation was replaced by a more secure one (SHA1)). Additionally users can register and join to centrally-managed secure user groups.

In order to control data sharing, different mechanisms are provided: a W3C XML-signature based license mechanism, and a content filter mechanism. In order to link audio files to related metadata, the sharing client supports the Fraunhofer IDMT AudioID system. The core libraries can be accessed in a programming language agnostic way via the integrated socket interface.

Sharing Server

The "Sharing Server" consists of several server components.

The "shf-sharing-server" peer: acts as a bootstrapping rendezvous peer in order to support the discovery process of other dynamic rendezvous peers. Rendezvous peers (RDV) are like any other peers, but maintain a cache of so called advertisements (language-neutral metadata structures represented as XML documents, used to describe and publish the existence of a peer resource) and other peers query them for that list. Additionally this peer acts as relay peer which sends dynamic requests across different networks.

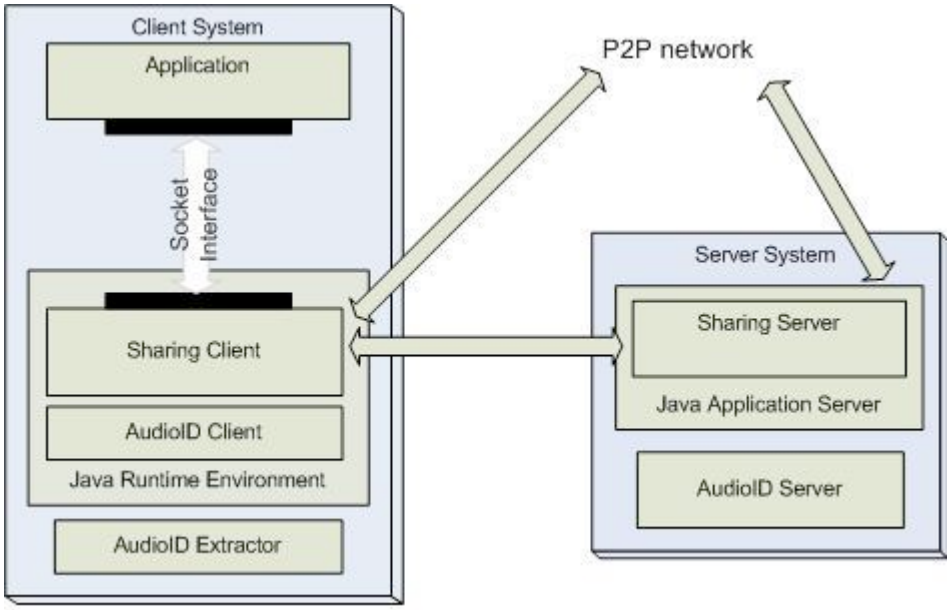
The User & Group Management web service: The management of centrally administered secure user groups is done via the user group management web service. Only the sharing server can create such secure groups. Internally this membership policy uses public key cryptography mechanism to verify the authenticity of users. X.509 certificates used within the system are issued by the User & Group Management web service.

2 Sharing System Core Libraries and Infrastructure

Responsible partner: Fraunhofer IDMT

2.1 Functional description

The following figure shows how the sharing system components are connected to each other and more generally how they are connected to the other components of the system:



Sharing metadata files (example: audio file related metadata file)	
Purpose	user wants to share an audio file related metadata file
Actors	user
Assumption	user has extracted the AudioID fingerprint of the audio file
Input data	o the metadata file describing the audio file[1] o searchable metadata for the metadata file[1] o AudioID fingerprint
Output data	advertisement for the metadata file containing searchable metadata
Steps	o user chooses searchable metadata[1] o user chooses metadata file o user chooses fingerprint[2] o user chooses the group in which the advertisement should be published
Specific issues	[1] according to the "Shared Metadata Model" specification [2] the fingerprint is not used directly, instead a track unique id (tuid) is written to the advertisement, the tuid is retrieved as response of a classification query to the AudioID server

Sharing data files (example: playlist)	
Purpose	user wants to share a playlist
Actors	user
Assumption	-
Input data	o the playlist file o the metadata file describing the playlist[1] o searchable metadata for the metadata file[1] o searchable metadata for the playlist[1]
Output data	o advertisement for the metadata file containing searchable metadata o advertisement for the playlist
Steps	o user chooses searchable metadata for the playlist[1] o user chooses searchable metadata for the metadata file[1] o user chooses playlist o user chooses metadata file for the playlist o user chooses the group in which the advertisements should be published
Specific issues	[1] according to the "Shared Metadata Model" specification

Messaging	
Purpose	user wants to send a message to another user
Actors	user a, user b
Assumption	-
Input data	o a text message
Output data	-
Steps	o user a chooses user b as destination for his message o user a sends message
Specific issues	-

Group creation (example: password protected group)	
Purpose	user wants to create a new password protected group
Actors	user
Assumption	-
Input data	login and password, group name
Output data	advertisement for the group
Steps	o user chooses login & password[1] and group name
Specific issues	[1] Every user who wants to join this group needs to know the login and password.

Join a secure group	
Purpose	user wants to join a secure group
Actors	User
Assumption	The user is registered for the group he wants to join.
Input data	-
Output data	-
Steps	o user chooses the group to join o depending on the response of the server, the user becomes a member of the group[1]
Specific issues	[1] The sharing server verifies the signed request (signed with the user's private key) and checks the database if the user is registered for the requested group.

Retrieving metadata from the p2p network	
Purpose	User wants to retrieve metadata from the p2p network for an audio file
Actors	User
Assumption	Metadata are available on the p2p network
Input data	Audio file
Output data	Metadata (title, artist's name, genre...)
Steps	o user inserts a new audio file in the HIFI system. o the Audio fingerprint is extracted from the audio file. o a Track Unique ID is assigned to the audio file by the central ID server, on the basis of the audio fingerprint. o a request is launched on the p2p network for metadata

Steps	○ user inserts a new audio file in the HIFI system. ○ the Audio fingerprint is extracted from the audio file. ○ a Track Unique ID is assigned to the audio file by the central ID server, on the basis of the audio fingerprint.

2.2 Position over state-of-the-art

In order to meet the demands of the project, WP7 built P2P sharing components on top of the JXTA framework which are provided with some – to our knowledge - unique features like certificate based secure user group management, the use of control mechanisms (licenses, filtering), the ability of metadata sharing, and the use of semantic fingerprints for the search of audio file related metadata via a central matching.

The JXTA software architecture is divided into three layers: the core layer, the services layer and the applications layer. All development activities in the context of the Semantic Hifi project are placed in the services layer and the applications layer. The JXTA core libraries, as provided by the JXTA project, remained unmodified.

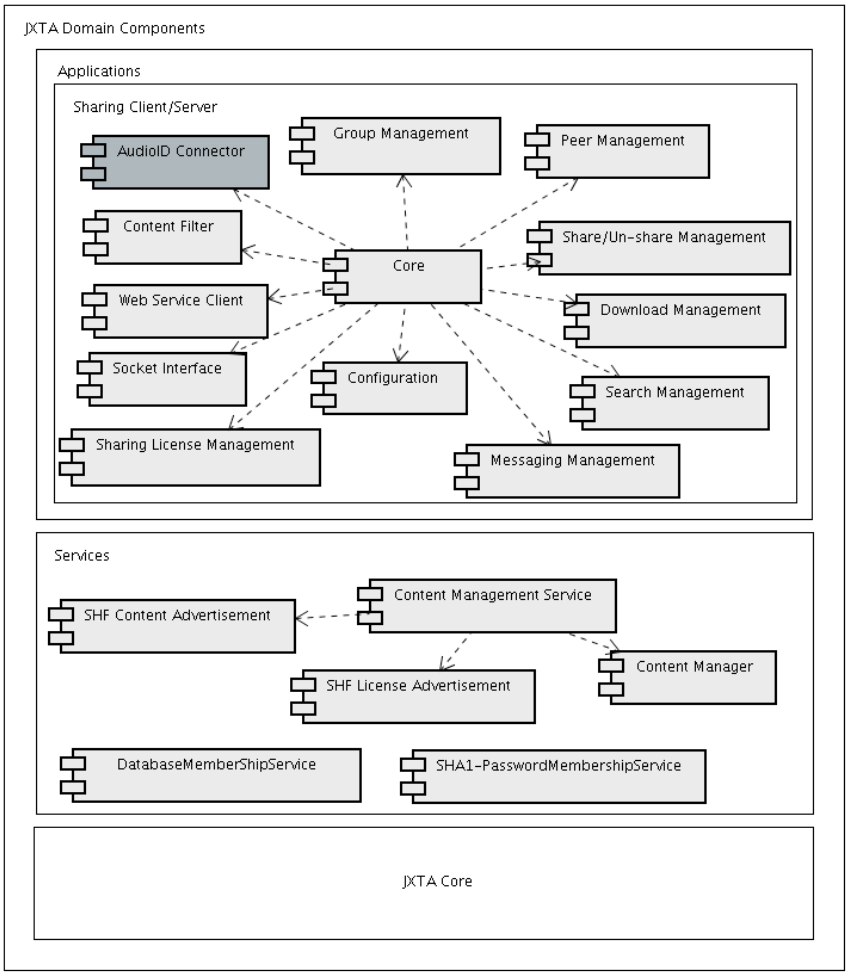


Figure 1: Developed components within the JXTA architecture model

Services Layer (Membership Service)

In JXTA, a Membership Service is used to apply for peer group membership, join a peer group, and resign from a peer group. The membership service allows a peer to establish an identity within a peer group. Once an identity has been established, a credential is available which allows the peer to prove that it rightfully has that identity. Identities are used by services to determine the capabilities which should be offered to peers.
The "JXTA Core" offers two membership services:

- "Null Membership Service": provides a Membership Service implementation which is intended to be used with peer groups which require no real authentication.
- "Password Membership Service": provides a Membership Service implementation which is based on a password scheme similar to the unix' `/etc/passwd` system. It is intended mostly as an example of a simple Membership Service service and *not* as a practical secure Membership Service, since "the current algorithm has been breakable since ancient times".

In order to meet the demands of the Semantic Hifi project for secure group membership services, the following new membership services were implemented:

- "**SHA1 Password Membership Service**": provides a Membership Service implementation which is based on a password scheme, using the secure algorithm SHA1 for storing passwords. Once a peer creates a new group, he has to choose a login and password for this group. Only peers who know those credentials are allowed to join the group.
- "**Database Membership Service**": provides a Membership Service implementation which is based on PKCS compliant digital certificates. This membership service involves a dedicated web service for issuing certificates and user management, developed also by Fraunhofer IDMT (see "User & Group Management Web Service" for details). Since the "Database Membership Service" is designed to be centrally managed, this type of group can be created only by a single instance - the Semantic Hifi bootstrapping rendezvous peer (using the "admin gui application").

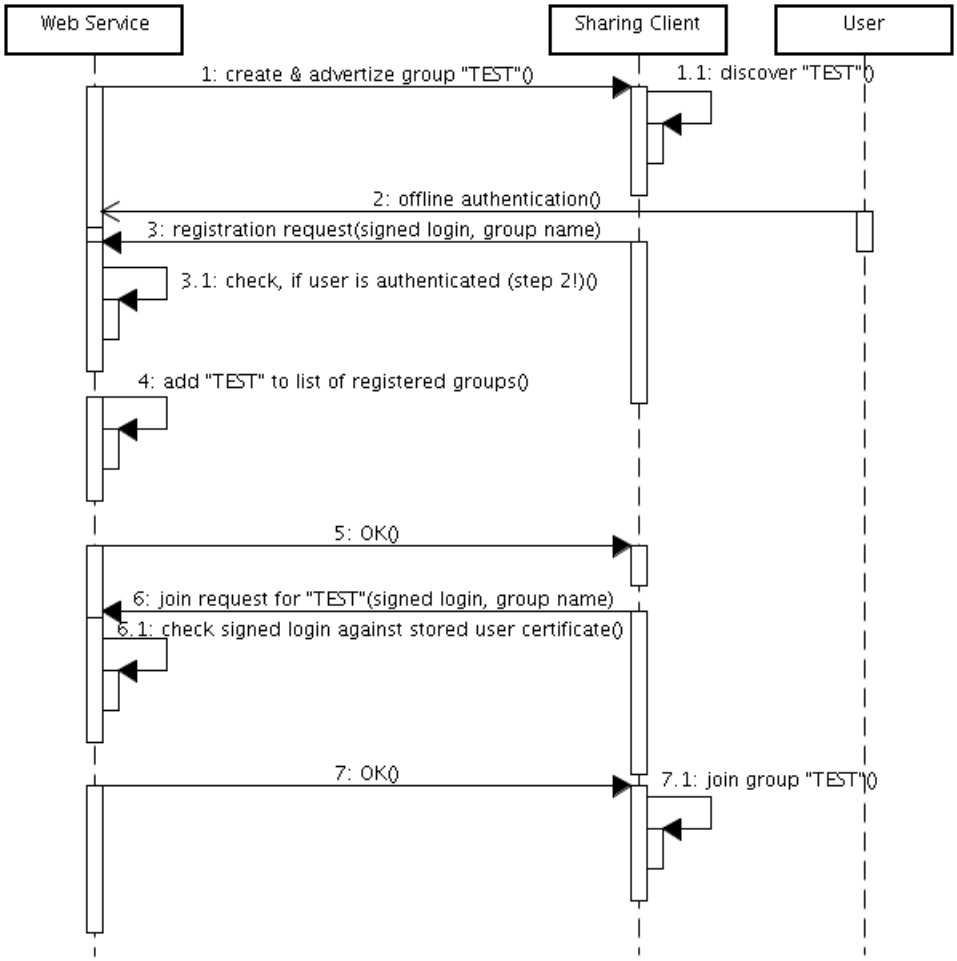


Figure 2: The DBMembershipService registration process

Service Layer (Content Management Service)

The Content Manager Service (CMS - provided by the JXTA Community) allows JXTA applications to share and retrieve content within a peer group. Each item of shared content is represented by unique content id and a content advertisement which provides meta-information about the content, such as its name, length, mime type, and description. The CMS also provides a protocol based on JXTA pipes for transferring content between peers. Each shared content item has an associated content advertisement which provides meta-information describing the content. Content advertisements are stored as XML documents. The standard CMS implementation only supports content name, length, mime type, id, and description as meta-information.
In order to meet the demand of the Semantic Hifi project for a generic way to support the metadata model (provided by IRCAM-SEL) within the sharing system, the JXTA CMS has been extended and modified. The resulting **SHF CMS** and **SHF Content Advertisement** provide support for 21 generic meta-information fields (including a field for AudioID fingerprints/TUID). Moreover, a complete new advertisement, the **SHF License Advertisement** has been introduced. This advertisement is part of the sharing license mechanism and stores W3C XML-Signature compliant meta-information about the sharing process (see Sharing License Management for details).

Applications Layer (IPR components, Metadata Linking)

The following components are main units of the Fraunhofer IDMT sharing client/server software developed for the Semantic Hifi project, adding functionality above state-of-the-art:

- **Content Filter**: is part of the IPR management system. This component verifies whether a file to be shared is allowed for sharing. Within the context of the Semantic Hifi project there are three valid types of files:
    - metadata files (xml, containing properties like specified in the Semantic Hifi metadata model: Since the sharing system is integrated into the Sony CSL MCM (Multimedia Content Management) system, the metadata model tries to be compliant to the MCM model as much as possible. The main entities of the metadata model are classes (MCM item types) and properties (MCM fields).)
    - play lists (xml, Native Instruments ”Traktor” compliant)
    - mix-files (binary, Native Instruments ”Traktor” compliant)

  Depending on the type there are different tests which have to be passed successfully in order to be recognized as valid file. For example the validity check of a mix-file includes the following steps:
    1. Parse and verify the binary part of the file (e.g. check if all fields are present and correctly filled)
    2. Validate the XML part of the file
    3. Manipulation detection, e.g. check if the file contains audio data (An attacker could try to store audio data within the mix-file in order to misuse the sharing system for un-authorized purposes.)

- **Sharing License Management**: is part of the IPR management system. The purposes of the license component are to complement the JXTA secure group concept by preventing ”leaking out” content to other groups than the specified ones (feature A) and on the other hand to establish traceability of shared content by including the digital signature of the peer who shared the content (feature B). In order to fulfil these tasks, the following premises have to be met: only files with a valid license are allowed to be shared and downloaded and every user is provided with a digital certificate (PKCS#12). The W3C XML Signature standard is used for the generation of licenses. Signed fields (signed with the private key of the user) are:
    - user id (unique identifier for the user)
    - group id (unique identifier for the P2P group, specifies in which group the file is allowed to be shared)
    - hash (SHA-1 hash value of the associated file)
    - usage (public: available in the Base Group, restricted: restricted to user group specified by group id, private: not shareable at all)
    - date

  In case of infringement, e.g. an attacker has managed to bypass the filter component and has shared unauthorized files, the signature within the license can be used for obtaining his identity. Based on this information the user could be banned from the system. Licenses are shared within the ”Base Group”, thus they are available for every user of the Semantic Hifi system. Moreover, in order to improve the availability of licenses, the system can be configured to provide all licenses also by the ”Sharing Server”. With respect to the need of support for batch publishing of files, the time consuming license mechanism is configurable. The basic level provides support for feature (A) by uploading each license to the server. Because this level doesn’t involve

publishing of licenses, the performance is around 5 times better compared to the extended level. The basic level is the default level. In contrast the extended level provides support for both features (A) and (B). However, in case of use of cryptographic hashes instead of semantic fingerprint based links between files and licenses, the effectiveness of feature (B) is low. The performance related to publishing and downloading is significantly lower than for the basic level.

- **AudioID Connector**: provides an API for requesting fingerprints and adding new fingerprints to the AudioID database. This component uses other components developed in WP2 by Fraunhofer IDMT.

Other components are:

- Core: central component, which manages the interaction of all other components.
- Group Management: manages group creation, group joining, and group re-signing, and group searching related processes.
- Peer Management: manages peer searching processes.
- Share/Un-share management: manages content sharing and un-sharing processes.
- Search Management: manages contents searching related processes.
- Download Management: manages download processes.
- Messaging Management: manages instant messaging processes.
- Socket Interface: provides a programming language agnostic way to connect the sharing client to gui applications.
- Web Service Client: provides the client functionality for the "User & Group Management" Web Service
- Configuration: manages all JXTA and non-JXTA related configuration issues, e.g. providing functionality for pre-configured clients.

2.3 Benchmarks

The performance of JXTA has been compared a number of times to various other P2P systems. Most published papers on this topic have focused on the widely-used communication layer of JXTA-J2SE: the pipe service. However, these studies are primarily based on JXTA 1.0 or even older versions. Newer versions, starting with JXTA 2.0, have been claimed to introduce significant design enhancements making these results obsolete. To our knowledge, only two papers have published results about JXTA 2.0.

Finally, one particular project worth noting with respect to JXTA performance evaluation is the JXTA Bench project (http://bench.jxta.org/), whose goal is to collect and report information about the different aspects of JXTA performance. The project site proposes a plan for the benchmarking of JXTA and the integration of tests into the project.

Relevant publications and resources:

[1] *Performance Evaluation of JXTA Communication Layers*, Gabriel Antoniu, Phil Hatcher, Mathieu Jan and David A. Noblet, IRISA/INRIA, Campus de Beaulieu 35042 Rennes Cedex, France, University of New Hampshire Department of Computer Science Durham, New Hampshire 03824-3591, U.S.A, gforge.inria.fr/docman/view.php/29/186/2005-05-09_GP2PC.pdf

[2] E. Halepovic and R. Deters, *The Costs of Using JXTA*. The Third IEEE International Conference on Peer-to-Peer Computing, Linköping, Sweden, 2003, http://bistrica.usask.ca/madmuc/Pubs/emir880.pdf

[3] http://bench.jxta.org/loadtest_results/stable/emir.html

[4] http://jdf.jxta.org/juxtest/Overall/Fast-Ethernet/index.html

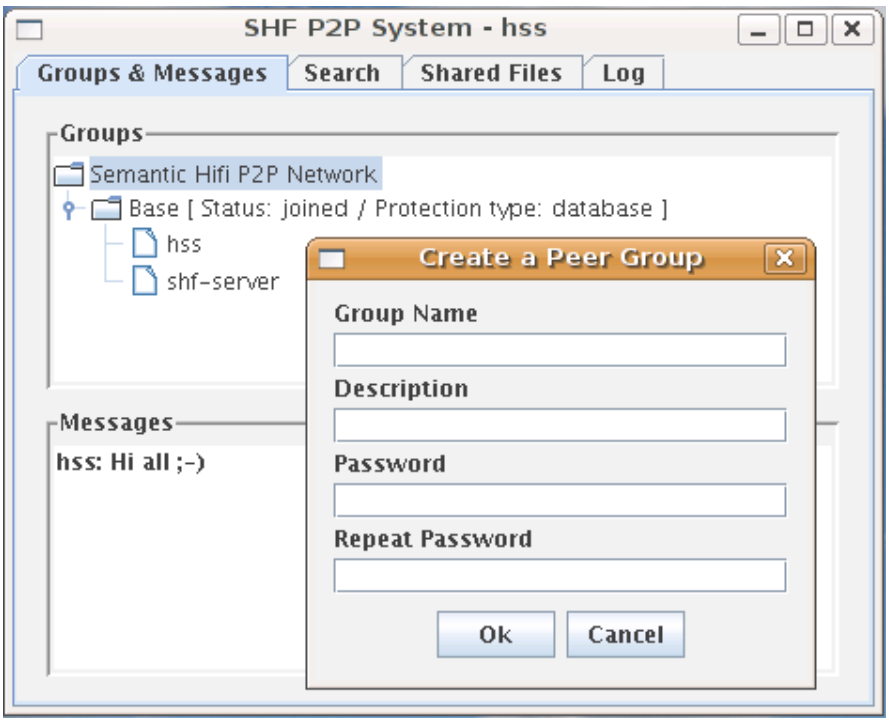
2.4 Implementation

The sharing system core libraries are implemented using JAVA 5.0. The socket API which is used to control the core libraries is programming language agnostic. During the project the socket API was adapted to the needs of the different partners.

First Version

The socket API of the prototype defines messages for providing the following functionalities:

- Create a new peer group (one type is supported: certificate protected)
- Join a specific peer group (providing the necessary credentials for authentication)
- Resign from a specific peer group
- Send text messages to specific peers
- Share files
- Un-Share files



Screenshot showing the demonstrator gui of the sharing client

- Search for files
- Stop search
- Search groups by name
- Search peers by name
- Start download

Final Version

The socket API of the final version defines messages for providing the following functionalities:

- Create a new peer group (three types are supported: unprotected, password protected, certificate protected)
- Join a specific peer group (providing the necessary credentials for authentication if needed)
- Resign from a specific peer group
- Send text messages to specific peers
- Send text messages to specific peer groups
- Share files (up to 22 parameters can be used for the description of published files, including AudioID track unique ids)
- Un-Share files
- Search for files (using up to 22 parameters in parallel, including AudioID track unique ids), multiple searches in parallel possible
- Stop specific searches, automatic search stop after time-out
- Automatic peer group discovery (each time a new peer group is discovered, a notification is sent)
- Automatic peer discovery (each time a new peer is discovered, a notification is sent)
- Search groups by name
- Search peers by name
- Start download (multiple downloads in parallel possible)
- Cancel a specific download process

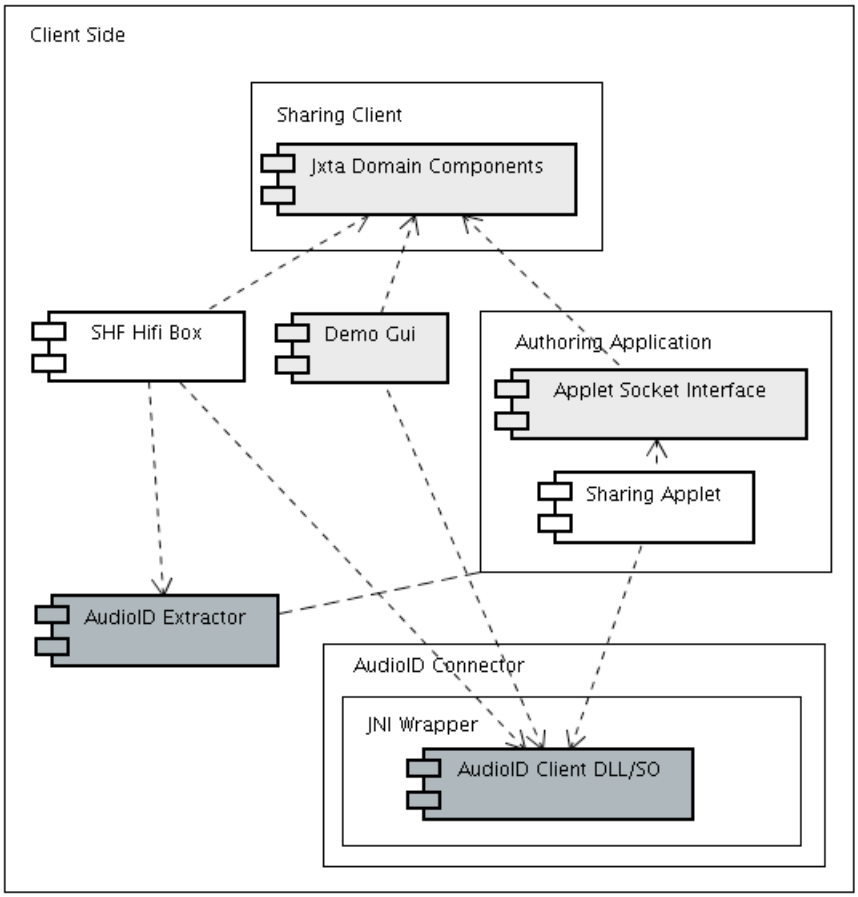


Figure 3: Client side architecture (final version)

User & Group Management Web Service

The management of centrally administered secure user groups is done via the user group management web service. Internally the centrally managed "DatabaseMembershipService" policy uses public key cryptography mechanism to verify the authenticity of users. X.509 certificates used within the system are issued (automatically) by the User & Group Management web service during the registration process of a SHF user. This web service is also responsible for uploading and storing sharing.

Bootstrapping Rendezvous Peer

The bootstrapping rendezvous peer supports the discovery process of other dynamic rendezvous peers. Rendezvous peers (RDV) are like any other peers, but maintain a cache of so called advertisements (language-neutral metadata structures represented as XML documents, used to describe and publish the existence of a peer resource) and other peers query them for that list. Additionally this peer acts as relay peer which sends dynamic requests across different networks. This component uses the same classes as the sharing client, with the exception of a specific "core" component and several configuration related classes (see also figure 1).

AudioID Server

In order to link audio files to related metadata, the sharing client supports the AudioID system. These components are developed within WP2 (by Fraunhofer IDMT) and they are described therefore within the appropriate section of this document.

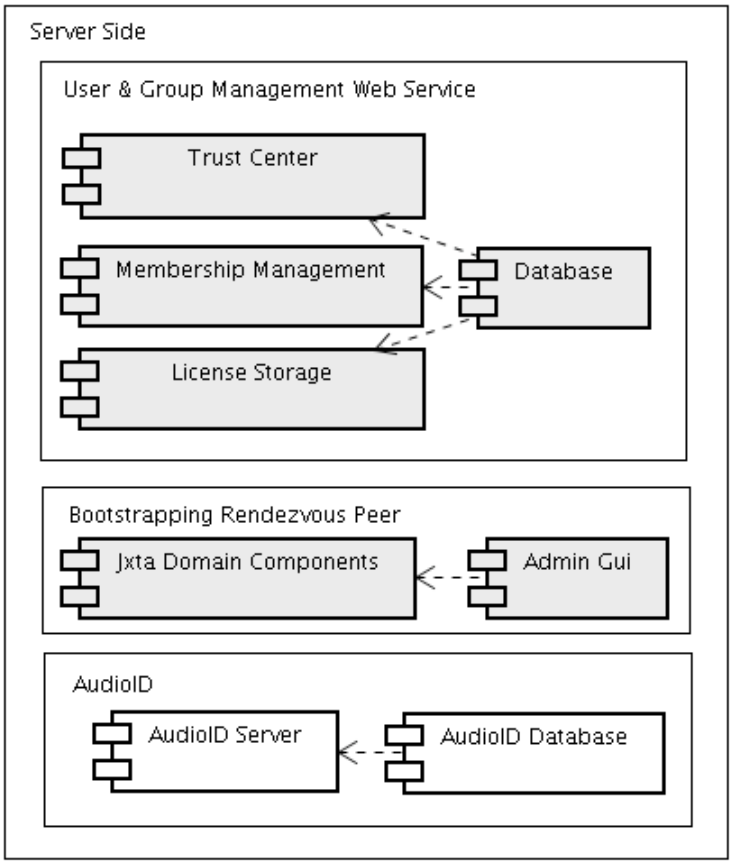
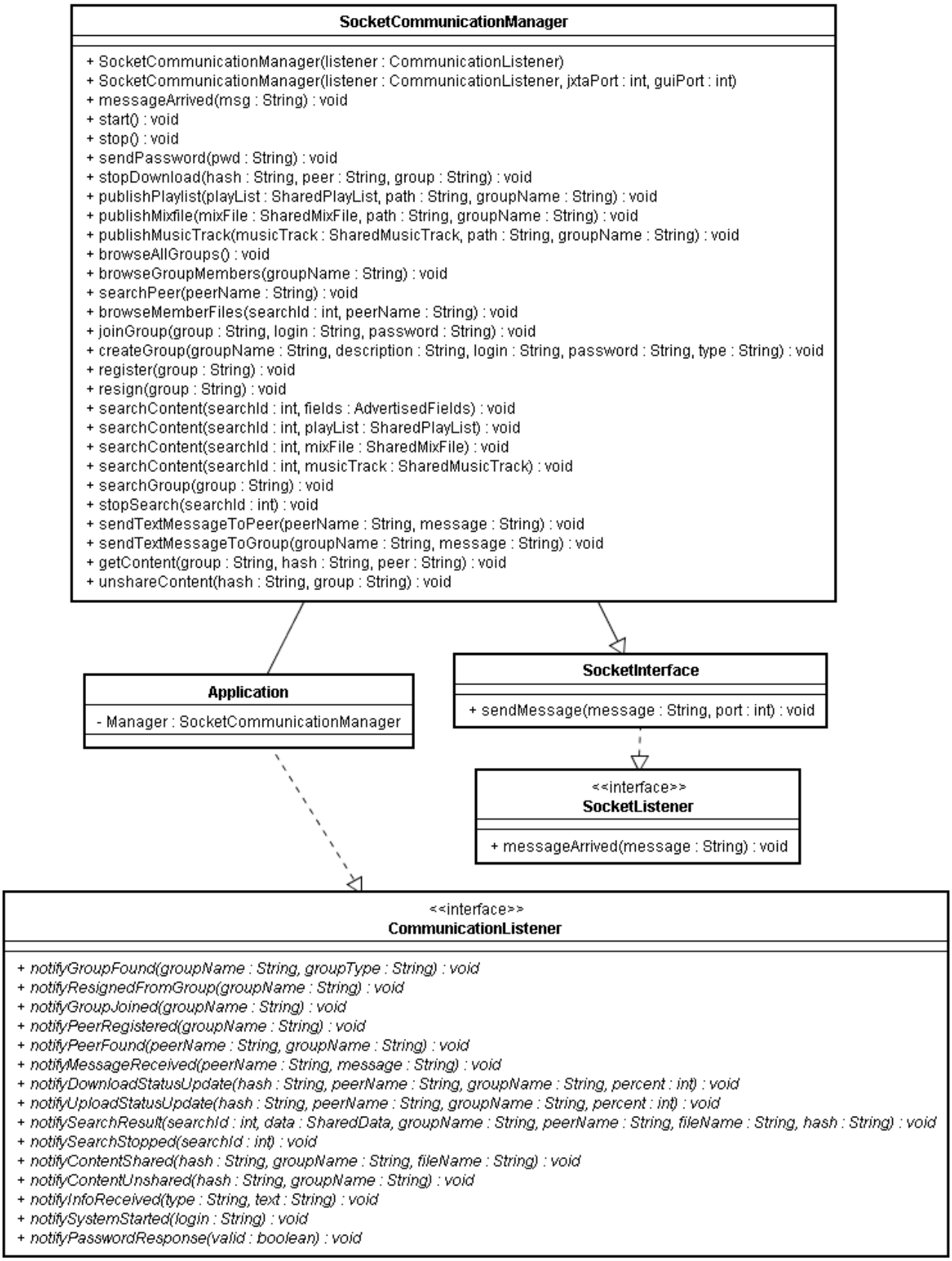


Figure 4: Server side architecture (final version)


SocketCommunicationManager
+ SocketCommunicationManager(listener : CommunicationListener)
+ SocketCommunicationManager(listener : CommunicationListener, jxtaPort : int, guiPort : int)
+ messageArrived(msg : String) : void
+ start() : void
+ stop() : void
+ sendPassword(pwd : String) : void
+ stopDownload(hash : String, peer : String, group : String) : void
+ publishPlaylist(playList : SharedPlayList, path : String, groupName : String) : void
+ publishMixfile(mixFile : SharedMixFile, path : String, groupName : String) : void
+ publishMusicTrack(musicTrack : SharedMusicTrack, path : String, groupName : String) : void
+ browseAllGroups() : void
+ browseGroupMembers(groupName : String) : void
+ searchPeer(peerName : String) : void
+ browseMemberFiles(searchId : int, peerName : String) : void
+ joinGroup(group : String, login : String, password : String) : void
+ createGroup(groupName : String, description : String, login : String, password : String, type : String) : void
+ register(group : String) : void
+ resign(group : String) : void
+ searchContent(searchId : int, fields : AdvertisedFields) : void
+ searchContent(searchId : int, playList : SharedPlayList) : void
+ searchContent(searchId : int, mixFile : SharedMixFile) : void
+ searchContent(searchId : int, musicTrack : SharedMusicTrack) : void
+ searchGroup(group : String) : void
+ stopSearch(searchId : int) : void
+ sendTextMessageToPeer(peerName : String, message : String) : void
+ sendTextMessageToGroup(groupName : String, message : String) : void
+ getContent(group : String, hash : String, peer : String) : void
+ unshareContent(hash : String, group : String) : void
Application
- Manager : SocketCommunicationManager
SocketInterface
+ sendMessage(message : String, port : int) : void
<<interface>>
SocketListener
+ messageArrived(message : String) : void
<<interface>>
CommunicationListener
+ notifyGroupFound(groupName : String, groupType : String) : void
+ notifyResignedFromGroup(groupName : String) : void
+ notifyGroupJoined(groupName : String) : void
+ notifyPeerRegistered(groupName : String) : void
+ notifyPeerFound(peerName : String, groupName : String) : void
+ notifyMessageReceived(peerName : String, message : String) : void
+ notifyDownloadStatusUpdate(hash : String, peerName : String, groupName : String, percent : int) : void
+ notifyUploadStatusUpdate(hash : String, peerName : String, groupName : String, percent : int) : void
+ notifySearchResult(searchId : int, data : SharedData, groupName : String, peerName : String, fileName : String, hash : String) : void
+ notifySearchStopped(searchId : int) : void
+ notifyContentShared(hash : String, groupName : String, fileName : String) : void
+ notifyContentUnshared(hash : String, groupName : String) : void
+ notifyInfoReceived(type : String, text : String) : void
+ notifySystemStarted(login : String) : void
+ notifyPasswordResponse(valid : boolean) : void

3 Shared Metadata Model

Responsible partner: IRCAM SEL

3.1 Metadata model

For the shared metadata, models of different initiatives were merged, particularly:

- The Dublin Core
  The Dublin Core metadata element set is a standard for cross-domain information resource description. It has been developed by the Dublin Core Metadata Initiative, which is an open forum engaged in the development of interoperable online metadata standards.
- UPnP
  The UPnP Forum is an industry initiative designed to enable simple and robust connectivity among stand-alone devices and PCs from many different vendors. UPnP is the basis for most networked home equipment currently in the market and utilizing the corresponding parameters will facilitate Home-AV interoperability.
  UPnP defines a metadata element set, which also includes element from the Dublin Core and DIDL Lite.
- DIDL Lite
  DIDL (Digital Item Declaration Language) is a descriptive language based on part 2 of the MPEG-21 standard; the Digital Item Declaration. DIDL Lite is itself based on a subset of DIDL.
- ID3
  Even if not supported by any standardization body nor any organization, ID3 can be considered as a de facto standard for embedding information in MPEG audio format layer I, II or III (MP3).

The main entities of the meta-model are **classes** and **properties**. The properties can be text properties or numerical (float and int). This distinction might not be implemented in the sharing library though. Properties are defined as being optional or mandatory.

3.2 Sharing on the p2p network

For each object, what can be shared are a metadata file and an optional data file. The metadata file should contain as much information as possible, allowing knowledge exchange between users. This document tries to collect a set of agreed properties, but a metadata file is not limited to contain just the properties listed here. It can contain any property (see Shared Metadata Syntax).

When an object is shared on the p2p network it has to be *advertised* by a set of properties. These properties will be available for searching by the user. Not all properties are advertised, since this would lead to a useless network traffic and RDV peer charge. For this reason beneath each property in the data model you will find a column *Adv*. that tells whether or not the property has to be advertised.

3.3 Namespaces

Classes and properties are identified by URIs. The following namespaces are used:

dc:	http://purl.org/elements/1.1/	Dublin Core
shf:	http://shf.ircam.fr/	Semantic HIFI
didl:	urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/	DIDL lite
upnp:	urn:schemas-upnp-org:metadata-1-0/upnp/	UPnP
rdf:	http://www.w3.org/1999/02/22-rdf-syntax-ns#	RDF
id3:	http://www.id3.org/id3v2/	ID3 version 2.

The data model defines the following entities:

- MusicTrack (shf:MusicTrack)
- Play list (shf:PlayList)
- MixFile (shf:MixFile)
- MusicClassification (shf :MusicClassification)

3.4 Shared metadata syntax

The shared properties of the music track are encoded in a simple XML file compliant with the RDF standard. This is an abstract format that can be used to generate also metadata files for other classes. In this way, the shared metadata model can be extended to other objects or properties, and the objects or properties can be exchanged or advertized by the mean of the sharing system, provided they comply with the advertisement of core properties described below.

3.5 Advertisements

3.5.1 Core properties

Some properties are in common between all (or almost all) classes. These properties are always advertised as the first and always in the same order.

Prop. name	***Description***

Prop. name	*Description*
rdf:type	The class the object is an instance of. Its value is the full class URI of the object, for example for a music track it will be “http://shf.ircam.fr/MusicTrack”.
shf:md5	The md5 check sum of the advertised (metadata) file.
dc:title	A label that can be shown to the user.
didl:size	The size of the advertised (metadata) file.
shf:dataMd5	The md5 sum of the corresponding data file or an empty string if there is no data file.

3.5.2 Data files

When an object is advertised on the p2p network, a metadata file is always generated and advertised, as has been seen till now. But the data file has also to be advertised in order to be downloadable (if this is allowed).

A data file is always advertised as being of of class shf:DataFile (rdf:type = “http://shf.ircam.fr/DataFile”).Data files are handled in the following way:

- An advertisement is found that tells something about an object the user is interested in (a music track with a given title, a play list containing a given TUID and so on)..
- The application can download the full metadata file of the object and do something with it (for example store the information in the database).

The application can check if there is a data file available and download it. To do this it will check the shf:dataMd5 property and, if not empty, search for an advertisement having rdf:type = “http://shf.ircam.fr/DataFile“ and having the same value on the shf:dataMd5 property. Once this new advertisement has been got, the application can download the data file.

4 Metadata Conversion Libraries

Responsible partner: IRCAM SEL

4.1 Functional Description

IRCAM implemented two libraries, the shf-ircam-sharing library relying on Fraunhofer sharing system core library and the mcm-sharing-library on top of the previous one.

The overall purpose of the first one is to implement the sharing metadata model defined for music track metadata, playlist, music classification and mixfiles. This implies two features:

- The mapping of advertisable metadata to the publication and retrieval process of the core sharing library and a protocol specification on top of the P2P in order to match these data.

- The file implementation (writing-parsing) of the whole metadata model for the file types handled by the system.

The purpose of the second one is to map the sharing system to the database management system, to furnish a simple access to the sharing system provided features and to implement a daemon service that will handle automatically the P2P publishing.

Matching MCM ids with AUIDs, the library allows the HiFi system to interoperate with the sharing system, publishing and retrieving metadata from the network.

A simplified connection interface has been implemented that deals with network and group management so that implementation on top of this library is easy and fast.

An automatic auid retrieval field has been implemented so that the database management system can access this information according to MCM retrieval standard methods.

A customizable publishing daemon was implemented so that the HiFi system deals only with two Boolean flags for each database item, one flag for the data refresh and one flag for the shareability of the data. The library will generate publisheable files according to database metadata, and publish them according to sharing groups specifications and dealing whenever it's needed with fingerprint generation and auid retrieval.

4.2 Implementation

The Ircam Sharing System and MCM Sharing libraries were implemented using JAVA 5.0. During the project JNI combined with C++ code was used in order to implement fingerprint generation on top of Fraunhofer fingerprint SDK.

This was later removed, in order to handle mp3 files as well as WAV files, so that the system retrieves the fingerprint generation output directly from a process call.

5 User testing

Responsible partner: Ircam SEL

A user's experimentation phase has been settled up in order to obtain an user's feedback on the Hifi system

One full week was dedicated to this HiFi System experimentations, from Tuesday 20th June 2006 to Sunday 25th June 2006, in cooperation with the Carrefour Numérique of the Cité des Sciences et de l'Industrie in Paris, taking advantage of the national Music feast. Monday 19th June 2006 was dedicated to the settlement of our different devices (the Cité is closed on Mondays). More information on the organization of these experiments can be found in the SemanticHIFI internal deliverables "User Feedback Report (D9.2.3) & Specification Update (D1.2.4)".

The main conclusions of these experimentations are just summarized below, according to following suggestions made by users:

- Exchanges of enhanced song structures by the means of the sharing system, containing qualification of segments (names of segments, like "Refrain", "Verses", "Main Theme"). These qualifications should be made by qualified persons (authors

themselves, or musicologists...). The implementation of this feature is strongly related to the availability of an authoring feature, letting the user able to qualify himself the segments of that structure. This

- Sharing of time-consuming descriptions. This feature has been suggested by users in order to cope with the problem of availability of songs during the computation of descriptions (like Query by Humming, song structure).

5.1 Features of interest to be developed

5.1.1 Sharing editorial metadata

It has been judged relevant to share tracks internal structure qualifying metadata on the P2P. The implementation of this feature needs the addition of the following elements in the system:

- A tool for authoring enhanced structures – this tool can be kept simple in a first implementation, and give the user the ability to give a name to each element of the structure.
- A new item in the shared metadata model (see below), “SongStructure”
- A function for searching qualified structures on the sharing system. This tool can be very simple and can be automatically launched when an item is inserted in the system, or be triggered by a simpel tiem in the User Interface.

The definition of the structure itself (as an XML file) should be enhanced in order to accept qualifiers.

It shall be noticed that this feature implies complex developments and thus is not to be implemented in the remaining time of the project.

5.1.2 Sharing of time-consuming descriptors

In order to be able to exchange time-consuming descriptors, it could be sufficient to add such descriptors to the MusicTrack data model. These descriptors could be exchanged in the same way as simple descriptors (like key, bpm and so on). There is no need of additional tool, the actual system will automatically download these descriptors.

Implementation of that feature will remain extremely simple.

On the other hand, it should be noticed that exchange of such material via p2p mechanisms can be subject to copyright restrictions, like for the melody itself. Thius issue has to be examined more in depth.

5.2 SHF Shared metadata model update

In order to be able to exchange qualified music structures, the following structure has to be added to the metadata model:

5.2.1 MusicStructure (shf:MusicStructure)

Prop. name	***type***	***Description***	***M/O***	***Adv.***
shf:tuid	text	The track unique id.	M	Y
dc:title	text	The title of the structure.	M	Y
shf:dataMd5	text	The md5 checksum of the structure file in hexadecimal digits.	M	Y
shf:dataSize	int	The structure file size in bytes.	M	Y
dc:date	text	The date the structure file was created.	O	Y
dc:creator	text	The name of the creator	O	Y
xml:lang	text	The language of these metadata or of the user that shared them.	O	Y

5.2.2 MusicTrack (shf:MusicTrack)

In order to be able to exchange time-consuming descriptors, like song structure automatically extracted from the track and query by humming descriptors, the following items are to be added to the shf:MusicTrack structure:

Prop. name	***type***	***Description***	***M/O***	***Adv.***
shf:qbhMelody	text	The melody of the track for Query By Humming	O	N
shf:automaticStructure	text	The structure as extracted from the song by automatic computation	O	N

6 Dissemination materials

6.1 Scientific publications

[1] Nützel, J., Kaufmann, M., „Sharing Systems for Future HiFi Systems", Wedelmusic 2004 Conference, Barcelona, Spain, 13th -14th September 2004

[2] Hasselbach, J., "P2P sharing within the Semantic Hifi European Project", Axmedis 2006, Leeds, UK, December 12th -15th 2006

6.2 Expert workshops

[3] Hasselbach, J., „Semantic Hifi: XMLSignature-basierte P2P-Sharing-Lizenzen für das JXTA Framework", 4. XML-Signatur Workshop (XML Uni), Koblenz, Germany, May 11th - 12th 2006

[4] Boutard, G., Goldszmidt, S., Peeters, G., "Browsing inside a Music Track, the Experimentation Case Study", submitted to 1st Workshop on Learning the Semantics of Audio Signals (LSAS) 2006 part of 1st intl. conference on Semantics And digital Media Technology (SAMT), Athens, Greece, December 6th 2006