Project Number:	**FP7-257123**
Project Title:	**CONVERGENCE**
Deliverable Type:	**Report**
Dissemination Level	**Public**
Deliverable Number:	**D3.1**
Contractual Date of Delivery to the CEC:	**31.10.2010**
Actual Date of Delivery to the CEC:	**02.11.2010**
Title of Deliverable:	**Definition of the system services and functional components**
Workpackage contributing to the Deliverable:	**3**
Nature of the Deliverable:	**Report**
Editors:	**Mihai Tanase, Richard Walker, Nicola Blefari Melazzi**
Authors:	**Angelos-Christos Anadiotis, Maria Teresa Andrade, Nicola Blefari Melazzi, Helder Castro, Leonardo Chiariglione, Andrea Detti, Panagiotis Gkonis, Kostas Kalaboukas, Georgios Lioudakis, Aziz Mousas, Stelios Papadopoulos, Charalampos Z. Patrikakis, Petros Salichos, Stefano Salsano, Mihai Tanase, Giuseppe Tropea, Iakovos S. Venieris**
Keyword List:	**Architecture, functional components, applications, middleware, network, walkthroughs.**

Executive Summary

This document provides the functional architecture and an initial description of technical concepts for the CONVERGENCE framework. It is based on the use cases and functional requirements deliverables, defined in Work Package 2 (WP2). The document describes:

- The CONVERGENCE architecture: the decomposition of the architecture into abstract levels according to functional criteria. These levels (Application, Content, Network) contain related components. The deliverable provides a functional description of each of these components and details the services exposed and the corresponding APIs.
- The Implementation strategy: a description of the different phases in the project's strategy for implementing the CONVERGENCE Framework.
- Walkthroughs: a set of walkthroughs, illustrating the behaviour of different levels and services and the relations and interactions between them.

It is to be noted that:

i) the project considers this deliverable as work in progress; it includes a number of results which were originally scheduled to appear in D3.2.

ii) terminology is not yet stable; it will be finalized in the next deliverable D 3.2 (System architecture), due in month 12.

iii) several implementation decisions and APIs are also not stable; these too will be completed and finalized in deliverable D 3.2.

INDEX

1 Introduction

The CONVERGENCE project aims to enhance the Internet with a content-centric [8], publish-subscribe service model [5], based on a common container for any kind of digital data, including representations of people and Real World Objects. This document provides the functional architecture and an initial description of technical concepts for the CONVERGENCE framework. It is based on the use cases and functional requirements deliverables, defined in Work Package 2 (WP2). The document describes:

- The CONVERGENCE architecture: the decomposition of the architecture into abstract levels according to functional criteria. These levels (Application, Content, Network) contain related components. The deliverable provides a functional description of each of these components and details the services exposed and the corresponding APIs.
- The Implementation strategy: a description of the different phases in the project's strategy for implementing the CONVERGENCE Framework.
- Walkthroughs: a set of walkthroughs, illustrating the behaviour of different levels and services and the relations and interactions between them.

2 Terms and Definitions

Term	**Access Rights**
Definition	Criteria defining who can access a VDI under what conditions.
Note	
Term	**User**
Definition	An entity outside the CONVERGENCE system, which interacts with the system and uses what the system delivers.
Note	
Term	**Application**
Definition	Software, designed for a specific purpose that exploits the capabilities of the CONVERGENCE System.
Note	
Term	**Clean-slate architecture**
Definition	The CONVERGENCE implementation of the Network functional level, totally replacing existing IP functionality.
Note	
Term	**CoMid Resource**
Definition	A virtual or physical object or service, referenced by a VDI, e.g. a video file, a Real World Object, a person, an Internet service, etc.
Note	See “VDI”. A resource can be virtually packaged as a VDI, on its own or together with other resources and metadata. The packaging operation includes the creation of an XML file describing the structure of the VDI and the links among its components. N.B. within the chapter devoted to CoMid or when there is no ambiguity a **CoMid Resource** will be referred to simply as **resource**
Term	**Community Dictionary Service (CDS)**
Definition	A functional block belonging to the Content level of CONVERGENCE that provides CoMid with all the matching concepts in a users subscription, search request and publication.
Note	The CDS parses and interprets the ontologies created by the CONVERGENCE users. It thus enables to understand users subscriptions or search requests as

	well as publications. See “Functional block” and “Functional level”.
Term	**CoNet Resource**
Definition	A resource of the CoNet that can be identified by means of a name; it can be either a **Named data** or a **Named service access point**
Note	N.B. within the chapter devoted to CoNet or when there is no ambiguity a **CoNet Resource** will be referred to simply as **resource**
Term	**Content-based Subscription**
Definition	A subscription based on a specification of user’s preferences or interests, (rather than a specific event or topic). In other terms, the subscription is based on the actual content of the considered events, which are not classified according to some predefined external criterion (e.g., topic name), but according to the properties of the events themselves [5].
Note	See “Subscription” and “Publish-subscribe model”.
Term	**Content-centric**
Definition	A network paradigm in which the network directly provides users with contents, and is aware of which content is actually transported, instead of limiting itself to providing communication channels between hosts.
Note	
Term	**Content-based resource discovery**
Definition	A user request for resources, either through a subscription to the CONVERGENCE System or a search request to the CONVERGENCE system. The CONVERGENCE System will then return a list of VDIs compatible with the search criteria.
Note	See “subscription” and “search”.
Term	**CONVERGENCE Applications level**
Definition	A functional level of CONVERGENCE that establishes the interaction with CONVERGENCE users. The Applications Layer interacts with the other CONVERGENCE functional levels on behalf of the user. The applications level is split into two layers: a Tools layer and a User Applications layer.
Note	See “Functional level”.
Term	**CONVERGENCE Content level**
Definition	A functional level of CONVERGENCE that provides the means to handle resources on the basis of “what” they contain and offer. These functionalities are implemented using a set of technologies that we call CONVERGENCE Middleware (CoMid) and the Community Dictionary Service (CDS).
Note	

Term	**CONVERGENCE Core Ontology (CCO)**
Definition	A semantic representation of the CoReST taxonomy.
Note	See “CONVERGENCE Resource Semantic Type (CoReST)”
Term	**CONVERGENCE Device**
Definition	A device deploys functionality implemented by a functional block.
Note	See “Functional block”.
Term	**CONVERGENCE Middleware (CoMid)**
Definition	A functional block of the CONVERGENCE system, implementing functionality belonging to the Content level.
Note	See “Functional block and level”.
Term	**CONVERGENCE Network (CoNet)**
Definition	A functional block of the CONVERGENCE system, implementing functionality belonging to the Network level.
Note	See “Functional block” and “Functional level”.
Term	**CONVERGENCE Network level**
Definition	A functional level of CONVERGENCE that provides access to *named-resources* on a public or private network infrastructure. A named-resource is any resource that can be identified by means of a name; named-resources may be either data or service-access-points. Examples of named-resources include: a VDI; an electronic document, an image, a source of information with a consistent purpose, the point of access to a service, and a collection of other resources.
Note	
Term	**CONVERGENCE System**
Definition	A system built by using the technologies specified or adopted by the CONVERGENCE specification.
Note	The CONVERGENCE system consists of a set of interconnected CONVERGENCE devices - peers and nodes - connected to each other. See “node” and “peer”.
Term	**CONVERGENCE Resource Semantic Type (CoReST)**
Definition	A list of concepts or terms that makes it possible to categorize CONVERGENCE resources, establishing a connection with the resource’s semantic metadata.
Note	
Term	**Describe, Discover, Distribute (DDD)**

Definition	The DDD paradigm uses three different axes to aggregate the pillar functionalities offered by the CoMid functional block of CONVERGENCE.
Note	
Term	**Digital Item (DI)**
Definition	A structured digital object with a standard representation, identification and metadata. A DI consists of content, content and context related metadata, and structure. The structure is given by a Digital Item Declaration (DID) that links content and metadata.
Note	See "MPEG-21 standards".
Term	**Domain ontology**
Definition	An ontology, dedicated to a specific knowledge domain or application
Note	In CONVERGENCE, domain ontologies will be used to establish associations between VDIs and their semantic metadata. Examples of domain ontologies include the W3C Time Ontology and the GeoNames ontology.
Term	**Elementary Service (ES)**
Definition	A concept imported from the MPEG-M emerging standard. Refers to the most basic unit of functionality offered by the CoMid. ES provide CoMid's main functionality.
Note	Elementary Services can be used on their own or in combination with CONVERGENCE entities to implement a series of actions.
Term	**Entity**
Definition	An object which an Elementary Service can act upon or with which it can interact with.
Note	Entities can be VDIs, resources, devices, events, group, licenses/contracts, services and users.
Term	**Fractal**
Definition	A semantically defined virtual cluster in a distributed overlay network composed of all CONVERGENCE peers running CoMid.
Note	
Term	**Functional block**
Definition	Partial or complete implementations of the functionality required by a specific level of the CONVERGENCE architecture.
Note	The CONVERGENCE architecture comprises four functional blocks: the CONVERGENCE Network (CoNet), the CONVERGENCE Middleware (CoMid), the Community Dictionary Service (CDS) and CONVERGENCE

	Applications.
Term	**Functional level**
Definition	An aggregated set of conceptually similar functions.
Note	The CONVERGENCE architecture defines three functional levels: Application, Content and Network.
Term	**License**
Definition	A machine-readable expression of Operations that may be executed by a Principal
Note	
Term	**Local named resource**
Definition	A named-resource made available to the CONVERGENCE users through a local device, permanently connected to the network.
Note	Users have two options to make named-resources available to other users: 1) store the resource in a device, with a permanent connection to the network; 2) use a hosting service. In the event she chooses the former option, the resource is referred to as a local named-resource.
Term	**Metadata**
Definition	Data describing a resource referenced by a VDI, including but not limited to provenance, classification, expiry date etc.
Note	
Term	**MPEG-M**
Definition	An emerging standard proposed by MPEG as an extension of the MXM standard.
Note	
Term	**MPEG eXtensible Middleware (MXM)**
Definition	A standard Middleware specifying a set of Application Programming Interfaces (APIs) so that MXM Applications executing on an MXM Device can access the standard multimedia technologies contained in the Middleware as MXM Engines
Note	
Term	**Named data**
Definition	A named-resource, consisting of data.
Note	"Data" should be understood as referring to any kind of binary or textual information, including documents, video, images, VDIs, etc.
Term	**Named resource**

Definition	A CoNet resource that can be identified by means of a name. Named-resources may be either data (in the following referred to as “named-data”) or service-access-points (“named-service-access-points”)
Note	Examples of named-resources include VDIs, electronic documents and points of access to a service.
Term	**Named service access point**
Definition	A kind of named-resource, consisting of a service access point identified by a name.
Note	A named-service-access-point is a network endpoint identified by its name rather than by the Internet port numbering mechanism.
Term	**Network identifier (NID)**
Definition	The name of a resource for use by CoNet
Note	A NID resembles an anycast address, in that it contains no reference to the location where the resource can be accessed.
Term	**Node**
Definition	A CONVERGENCE device that deploys CoNet functionality only.
Note	Typically, a node is owned by a Telco or Service Provider.
Term	**Overlay architecture**
Definition	An implementation of CoNet as an overlay over IP.
Note	
Term	**Parallel architecture**
Definition	An implementation of CoNet as a new networking layer that can be used in parallel to IP.
Note	
Term	**Peer**
Definition	A CONVERGENCE device that implements CoMid and CoNet functionality.
Note	A node will typically be owned by an end-user or by a Service Provider or Telco.
Term	**Principal**
Definition	The entity associated with a VDI as represented in the CCO. We adopt here the quite generic, yet representative, term principal, actually following the DONA [7] paradigm.
Note	For instance, this entity can be the provider of a service, the publisher or the author of a book, the controller of a traffic lights infrastructure, etc. Alternatively, this entity can be the publisher of a VDI. Each VDI is directly

	associated with a principal.
Term	**Publisher**
Definition	A user of CONVERGENCE who advertises resources on the CONVERGENCE system, thus making them available to subscribers.
Note	
Term	**Publish-subscribe model**
Definition	A service model based on an *asynchronous* exchange of messages or events. The CONVERGENCE publish subscribe model, encompasses a set of clients that publish VDIs, which are then forwarded to clients who have expressed interest in receiving them.
Note	There are basically two approaches to implementing a publish-subscribe system: 1) topic-based; and 2) content-based. CONVERGENCE adopts the latter. The topic-based approach has 2 variants: list-based and broadcast-based. Both use pre-identified topics to send messages, but their implementations differ. In the list-based approach, the system maintains lists of published topics (available information data, classified by subject) and of subscribers (clients) and notifies subscribers individually as relevant events occur. In the broadcast-based approach the system broadcasts messages to all subscribers in a group whenever someone publishes information relevant to the group. This means that publishers and subscribers have to know the list of pre-defined topics and explicitly specify a topic when publishing or subscribing. The Content-based approach does not require previous knowledge of topics. Subscribers indicate their preferences and the system dynamically inspects published content to decide which subscribers should be notified. In other terms, the subscription is based on the actual content of the considered events, which are not classified according to some predefined external criterion (e.g., topic name), but according to the properties of the events themselves [5]. This approach is harder to implement - it requires efficient, scalable algorithms to inspect content and users' interests, and to provide good matches.
Term	**Real World Object**
Definition	An object existing in the real (as opposed to the virtual) world.
Note	
Term	**Search**
Definition	The act of a User submitting a query to the Content level of CONVERGENCE, using keywords, free text, similarity with a resource or a specific VDI
Note	On receiving a query, the CoMid inspects published VDIs (in cooperation with

	CDS) and returns a ranked list of VDIs that match the query.
Term	**Subscriber**
Definition	A user of CONVERGENCE who declares his/her interest in receiving VDIs.
Note	A subscriber can declare her interest using keywords, free-text or by indicating specific VDIs.
Term	**Subscription**
Definition	The act of a User indicating his/her interests to the Content level of CONVERGENCE using keywords, free text or similarity with a resource.
Note	On receiving a subscription, the CoMid dynamically and periodically inspects published VDIs, notifying users whose subscriptions match published VDIs.
Term	**Unnamed data**
Definition	A data resource with no NID.
Note	
Term	**User ontology**
Definition	An ontology (a set of concepts and their relationships), created by users of CONVERGENCE when publishing a VDI or subscribing to a VDI.
Note	
Term	**Versatile Digital Item (VDI)**
Definition	A structured, hierarchically organized, digital object containing one or more resources and metadata, including a declaration of the parts that make up the VDI and the links between them.
Note	VDIs can be seen as an extension of the MPEG-21 concept of a Digital Item (DI)

3 Bird's eye view

The CONVERGENCE project aims to enhance the Internet with a content-centric [8], publish-subscribe service model [5], based on a common container for any kind of digital data, including representations of people and Real World Objects. Figure 1 depicts the main players in the CONVERGENCE framework. Publishers advertise their *resources* (both data and service-access-points) on the CONVERGENCE system; subscribers express their interest in specific resources on the CONVERGENCE system. The CONVERGENCE system notifies subscribers when the resources they are interested in are available.

If the subscriber is interested in retrieving data stored in the network, the CONVERGENCE system provides her with the possibility of downloading it. If the subscriber is interested in exploiting an application or middleware service, then the CONVERGENCE system provides the involved application or middleware entities with the possibility of exchanging data.

The CONVERGENCE system supports *content-based* subscriptions [5]; subscribers can use subscriptions to express interest in specific resources. To support content-based subscription, a resource (or a structured set of resources) is digitally represented by a data structure called a *Versatile Digital Item* (VDI). This includes descriptors of the resource(s) that the VDI refers to. Consequently, each resource published in the CONVERGENCE system is associated with a VDI. Subscriptions express criteria that can be verified by inspecting VDI information.

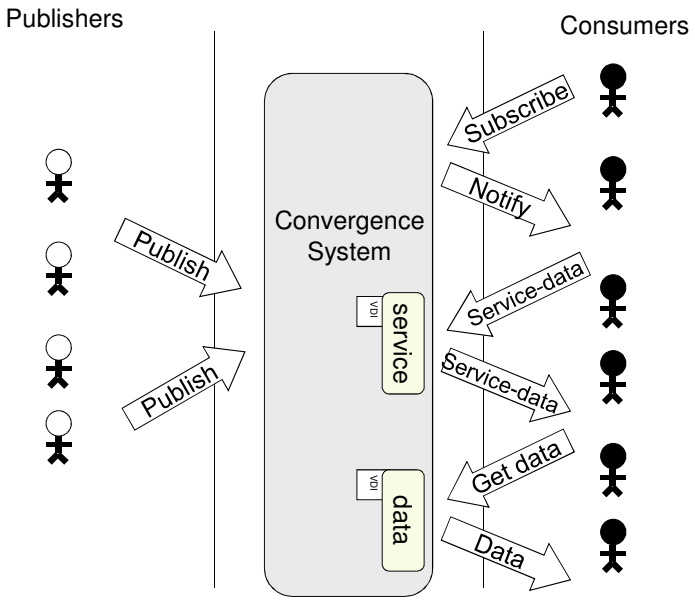

Figure 1: service model

4 CONVERGENCE Architecture

Figure 2 depicts the three functional levels (Application, Content and Network) of the CONVERGENCE system. A functional level is an aggregate set of conceptually similar functionality[1]. The functionality of a functional level is realized by means of *functional blocks*. The architecture comprises four functional blocks: the CONVERGENCE Network (CoNet), the CONVERGENCE Middleware (CoMid), the Community Dictionary Service (CDS), and CONVERGENCE Applications. Figure 2 shows their interactions.

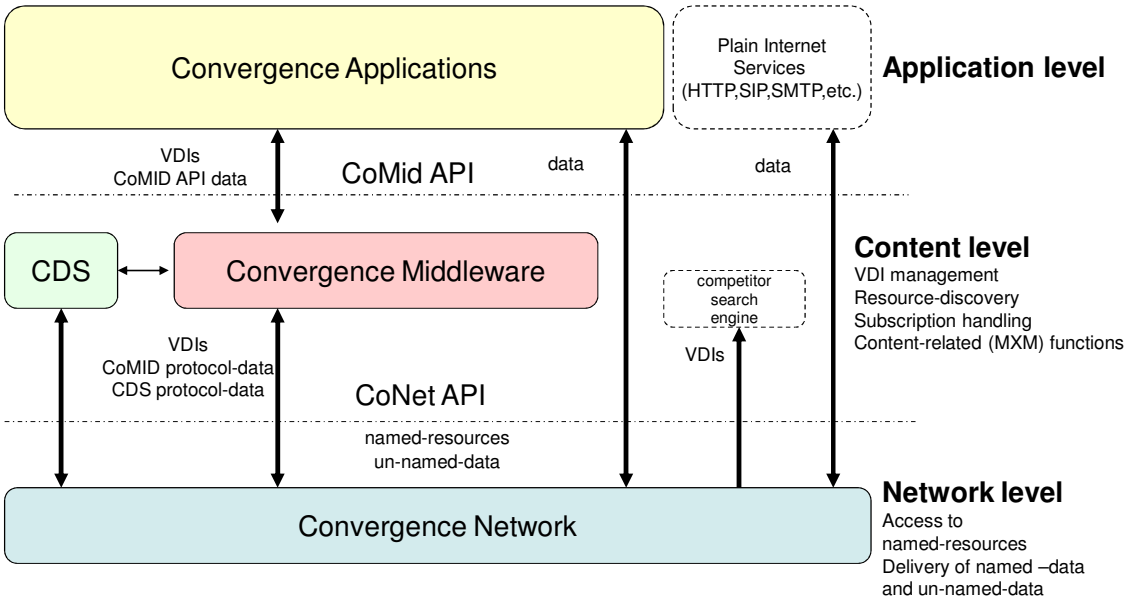

Figure 2: functional levels and technologies

The CONVERGENCE system is formed by a set of networked devices that implement the functionalities introduced above. As depicted in Figure 3, there are two kinds of CONVERGENCE device: *nodes* and *peers*. A CONVERGENCE peer is a device executing Applications on top of CoMid and CoNet functionality and may be owned either by an end-user or by a Telco, or by a service provider. A CONVERGENCE node is a device that only supports CoNet functionality. CONVERGENCE nodes provide the CONVERGENCE system, with services similar to those that a router provides for the Internet. A typical owner of a node would be a Telco, or a service provider.

[1] We use the term "level" instead of (OSI) "layer", since the functions of a single level in CONVERGENCE may be accomplished by different protocol layers.

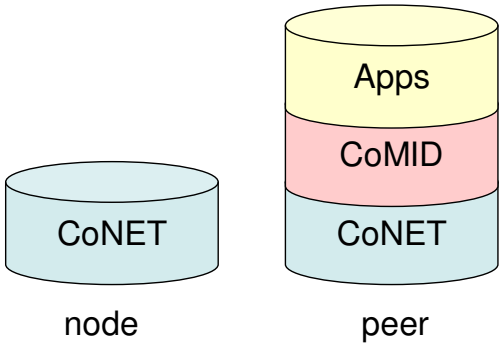

Figure 3: type of CONVERGENCE devices

As depicted in Figure 4, the CONVERGENCE system consists of a set of interconnected peer and nodes. Network level functionalities are provided in a distributed way by the whole set of CONVERGENCE nodes. Content level functionalities (and related security) are provided in a distributed way by CONVERGENCE peers.

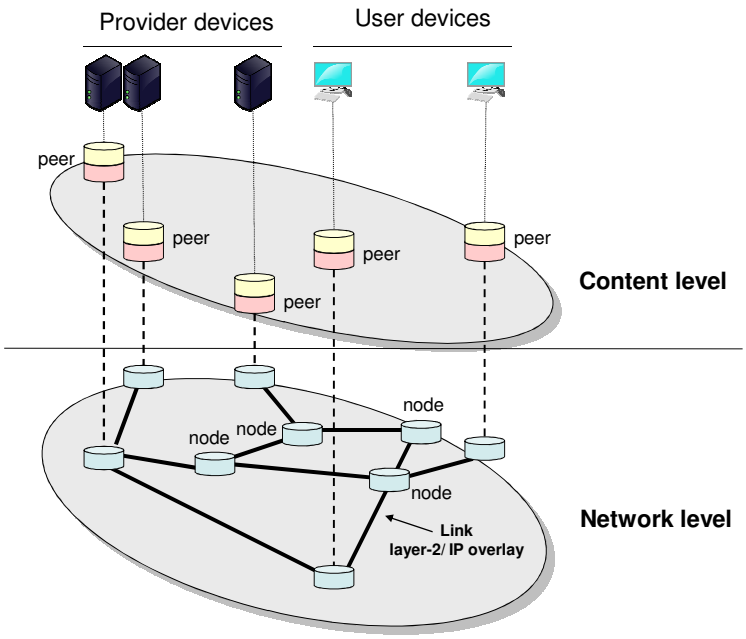

Figure 4: sketch of CONVERGENCE system

4.1 Network level (CoNet)

4.1.1 Description

Network level functionalities are provided by the CONVERGENCE Network (CoNet). The network level provides access to *named-resources* on a public or private network infrastructure [6][7][8]. A named-resource is any resource that can be identified by means of a

name; named-resources may be either data[2] (in the following referred to as "named-data") or service-access-points[3] ("named-service-access-points"). Examples of named-resources include: a VDI; an electronic document, an image, a source of information with a consistent purpose, the point of access to a service, and a collection of other resources. The name of a resource is its *network-identifier* (NID).

Unlike Internet URLs, which include information about "where" a resource is located, CONVERGENCE NIDs do not contain any reference to resource's location. A network-identifier is an *anycast* address: a system may contain multiple replicas of the same named-resource. Network functionality will connect a user to the "best" replica (e.g., the closest one). For example:

- a PDF copy of the TIMES newspaper for Sept-30-2010 could be a resource identified by "the times: Sept-30-2010";
- the service-access-point for an SQL database service provided by the Foo company could be a resource identified by the name "Foo: sql-database";
- the service-access-point for an MXM Content-provider device owned by the Foo company could be a resource identified by the name "Foo: mxm-content-provider".

The handling of named-resources through their network identifiers, rather than IP addresses, means that the network infrastructure is aware of the resources it is handling. This awareness can be exploited to support anycast routing, resource replication, and in-network caching [9] (if resources are data and not points of access to a service). These mechanisms are very needed but not supported by current IP networks[4]. The possibility of accessing resources without having to specify their location, simplifies mobility issues. It also means that resource names can be portable and do not depend on the service provider providing the resource.

The CoNet allows users (e.g., end-users and providers) to advertise and revoke their named-resources. A resource can be replicated in different geographical locations by adopting the same network-identifier. Replication enables users to exploit resources that are closer to their location and improves reliability of access. The CoNet provides *secure distribution* of resources by assuring that the network-identifier cannot be forged and that receivers can verify the validity and provenance of named-resources [7][10]. Network functionalities may autonomously cache transiting resources [9], if this is allowed by security rules and by the nature of the resource[5].

---

[2] Named-Data include: documents, video, images, structured information, VDIs. The Network level is a general purpose one, as it can handle any kind of data and not only data generated by the CONVERGENCE system.

[3] A named-service-access-point is a network endpoint through which an upper layer entity (e.g, a server or a client) sends and receives data. In the actual Internet, for instance, TCP port n. 80 is the default service-access-point for HTTP servers.

[4] Some partial support is provided by proprietary systems, such as Content Delivery Networks (e.g. Akamai).

[5] It is possible to cache data but not points of access to a service.

When a named resource is a named-data, the CoNet provides the means to deliver it to intended recipients. It can also facilitate the delivery of the named-data through replication/caching.

When a named resource is a named-service-access-point, the CoNet provides the means to exchange data between a requesting upper layer entity and the upper layer entity addressed by such named-service-access-point. This answers to a different need with respect to the distribution of named-data mentioned above. In the present case upper layer entities want to exchange data that do not need to be named and do not need to be made accessible and identified in the network, per se. This functionality is needed by "traditional" client/server services (e.g. HTTP, POP, SMTP) and middleware services (e.g. transport of middleware service data), which do not assign names to their data. To support this need we introduce the concept of "*un-named-data*", i.e. data not identified by a NID.

The CoNet thus offers two mechanisms for the delivery of data.

1. When users need to access named-resources that are data (e.g., documents, VDIs, files, etc.), the CoNet allows them to download actual content. In this case the user asks the network to provide a resource with a given network-identifier and the network provides her with the actual content, without involving any other upper layer functionality. This modus operandi follows the *content-centric networking* paradigm recently proposed in the literature.
2. When it is necessary to support service sessions for upper layer entities (e.g. a client-server couple), the CoNet couples the local upper layer entity with the named-service-access-point for the "best" remote upper layer entity providing the service (via anycast routing) and goes on to support an interactive exchange of un-named-data between these two upper layer entities. In the case of a client-server service session, for example, the un-named-data is upper layer data (e.g. HTTP, SMTP, POP, SQL, MXM messages) exchanged between a local client and a remote server (e.g. an HTTP server, a middleware device etc.). Thanks to this functionality, the CoNet can natively support most current Internet services, existing middleware protocols and any service that requires point-to-point bidirectional interaction. This *extension* to the capabilities of a "plain" content-centric network [8] allows the CoNet to support not only content retrieval but also more traditional services.

Access to a named-resource involves two steps: i) anycast routing of a user "request" to the best CoNet node providing the named-resource and ii) unicast routing of the "response" from the selected CoNet node toward the requesting CoNet user. Anycast routing uses the network-identifier of the named-resource, as previously discussed. In unicast routing, endpoints are identified by a *location-identifier* (LID) – the equivalent of the IP address and port number on the current Internet. The use of location-identifiers makes it possible to support sessions that require interactive exchange of data between a "client" and a "server". After the first phase of anycast routing, when the CoNet selects the best server, further communications between the client and the server can be routed using their location-identifiers; in fact, if the client

application tried to reuse the (anycast) network-identifier, the CoNet might select a different server, thus interrupting the session [6.7].

4.1.2 Preliminary description of the CoNet Application Program Interface

This section describes the Application-programming interface provided by the CoNet. Table 1 reports a preliminary set of possible primitives. Thanks to these primitives, the CoNet allows its users to:

- advertise named-resources, i.e. resources identified by a network-identifier (NID);
- access named-resources;
- deliver named-data and un-named-data (i.e., upper layer data).

The scope of applicability of some primitives depends on the type of the named-resource they deal with. We thus distinguish two classes of named-resources: named-data (i.e., documents, video, images, structured information, VDIs, etc.) and named-service-access-points.

For examples of the use of these primitives see the following section.

Table 1 – Service primitives offered by the CoNet API

Service Primitive	Applicability	Origin	Output	Description
Advertise (NID, resource-type, [resource /port], expiry, auth,…)	Named-data and named-service-access-points	CoNet user (e.g., application or middleware)	success/failure	This primitive is used by a CoNet user to make a local named-resource accessible to other CoNet users. NID is the network-identifier of the named-resource. resource-type specifies if the resource is named-data or a named-service-access-point. In the case of named-data, resource is actual content (e.g., a document, an image, media, file, a VDI). In the case of a named-service-access-point, port is a *local entity port* where an entity expects to receive data through an Indication

[6] One way to implement the location-identifier (without exploiting IP) would be to adopt a *link-layer source-routing* approach. In this case, the structure of the location-identifier would represent the sequence of link-layer interfaces to be followed from the CoNet node (Server) to the requesting CoNet user, i.e. on the reverse path. This sequence is built during the preceding anycast routing phase. The location-identifier has a *temporary* meaning; it is assigned in a distributed way, and has no impact on the routing-plane of the CoNet, which is concerned only with network-identifiers. *Link-layer source-routing* is the approach adopted by the PSIRP FP7 project (http://www.psirp.org/) (see [11][12]).

[7] It has not yet been decided whether the CONVERGENCE system will support also service sessions between two NIDs, when it happens that NIDs are unicast addresses and thus univocally identifies a service access point.

				primitive (in the IP networks it corresponds to the transport port that is used as local address in a host to deliver the data to a given entity). expiry defines a date after which the advertisement will be automatically revoked. auth is a set of information that authenticates the user advertising a named-resource with a specific NID. This prevents the forging of NIDs. Failures may occur if Auth is not valid, a resource type is not consistent, etc. After Advertise, a new named resource is introduced into the network. The resource will be reachable by name by any entity in the CONVERGENCE network. Given that CoNet supports routes-by-NID, this information has to be introduced into the network to routes requests for the new named resource towards the appropriate network nodes Thus, Advertise changes routing information on CoNet
Update (NID, resource-type, resource/port, expiry, auth,…)	Named-data and named-service-access-points	CoNet user (e.g., application or middleware)	success/failure	This primitive is used by a CoNet user to update the resource associated with a previously advertised NID. The meaning of the parameters is the same as in Advertise. However, in this case it is not necessary to introduce or modify routing information.
Revoke (NID, auth,…)	Named-data and named-service-access-points	CoNet user	success/failure	This primitive is used by a CoNet user to revoke (and delete) a local named-resource, identified by NID. auth is a set of information that authenticates the user seeking to revoke the named-resource. A failure occurs when the named-resource is not available. Like Advertise , Revoke modifies CoNet routing information
Get (NID)	Named-Data	CoNet user	data / failure	This primitive is used by a CoNet user to retrieve a named-resource identified by NID. A failure occurs if the named-resource is not available.
Send2Name (NID, data, port)	Named-service-access-points	CoNet user	success / failure	This primitive is used by a CoNet user to send un-named-data (i.e. upper layer data) toward a named-service-access-point, identified by NID. If the CoNet user waits for a response from the remote endpoint, the parameter port is the local entity port where

				the CoNet user expects the Indication primitive.
Indication (LID,data)	Service access point	CoNet		When un-named-data (i.e. upper layer data) is received by a service-access-point, this primitive is invoked by the CoNet on the entity which is listening to the local entity port associated with the service-access-point (see advertise). The primitive gives the un-named-data and the temporary identifier LID of the location of the remote entity that has sent this un-named-data to the service entity listening to the port.
Send2Location (LID,data,port)	Service access point	CoNet user		This primitive is used by a CoNet user to send un-named-data (i.e. upper layer data), via unicast, to a specific entity whose location is temporary identified by LID. If the CoNet user expects a response from the remote endpoint, the parameter port is the local entity port where the CoNet user expects the Indication primitive.

4.1.3 Examples of use of the CoNet Application Program Interface

In this section, we show how the CoNet API could be exploited to support typical operations. Details are preliminary. The final specifications will be completed at a later stage in our work.

4.1.3.1 Advertising local data

Figure 5 depicts the case of a CoNet user (e.g. a provider application), that wishes to make available to other CoNet users the local file "times09-30-10.pdf", with network-identifier (NID) "the times: Sept-30-2010". The provider calls **Advertise**. The local CoNet node stores the file locally, updates an internal database that maps "the times: Sept-30-2010" to the file "times09-30-10.pdf," and updates the routing plane of the whole CoNet, so that the local CoNet node becomes a provider (perhaps together with others) of the named-resource "the times: Sept-30-2010".

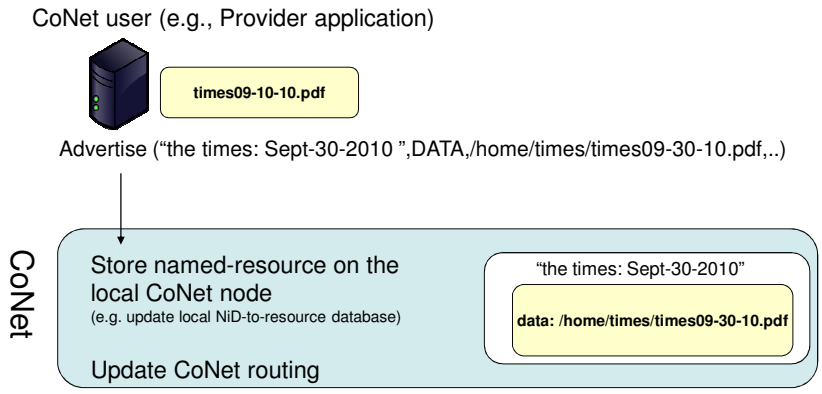

Figure 5: Advertising a local named-resource (named-data)

4.1.3.2 Advertising local services

Figure 6 depicts the case of a CoNet user (e.g. a provider application) that wishes to make available to other CoNet users a local SQL database service with network-identifier (NID) "Foo: sql-database". The provider runs the relevant server application, which is configured to receive its service-data (un-named-data for the CoNet) through the local entity port 0x067497. Then the provider application uses the Advertise primitive. Subsequently, the local CoNet node updates an internal database that maps "Foo: sql-database" to the local entity port 0x067497 and updates the routing plane of the whole CoNet, so that the local CoNet node becomes a provider of the named-resource "Foo: sql-database". To improve the reliability of database access, the Foo company may replicate the database by repeating the same advertising operation on different CoNet nodes.

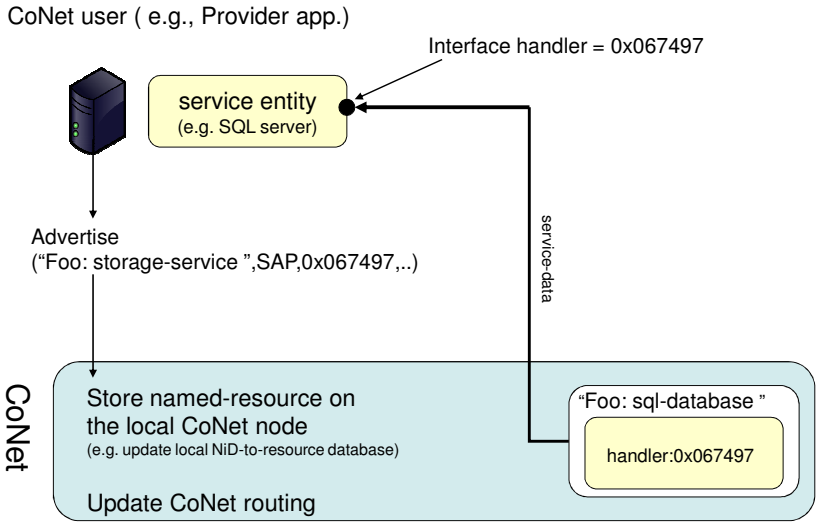

Figure 6: Advertising a local named-resource (named-service-access-point)

4.1.3.3 Downloading data

Figure 7 depicts the case of a CoNet user (e.g. a end-user application) that wishes to download the document identified by NID "the times: Sept-30-2010". The user application uses the GET primitive. The local CoNet node then identifies the specific instance of the GET primitive with a Location Identifier (LID). Then the CoNet node sends the network a data unit containing (at least) the LID and the NID. The CoNet *routes-by-NID* this data unit to the best remote node that is advertising "the times: Sept-30-10". When the remote node is reached, the CoNet functionality directly replies by sending-back a data-unit, containing the file "times09-10-10.pdf". This data-unit is routed using the LID and contains the NID ("the times: Sept-30-10"). The presence of the NID in the reply allows intermediate nodes to identify the traversing content and possibly to perform caching.

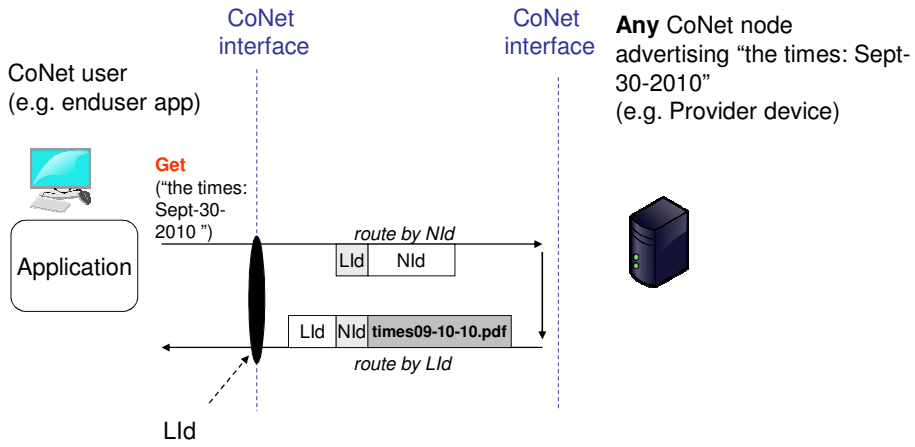

Figure 7: Downloading a named-resource (named-data)

4.1.3.4 Request-response service session

Figure 8 depicts the case of a CoNet user (e.g. an end-user client application), that wishes to send *un-named-data* (e.g. a SQL query)), to a remote server with NID "Foo: sql-database". The user application uses Send2Name, specifying the local entity port 0x0076 where the response is expected. The local CoNet node associates the local entity port with a Location Identifier, LID. Then the CoNet node sends the network a data unit-containing the LID and the NID. The CoNet routes-by-NID the data-unit to the best remote node that is advertising "Foo: sql-database". When the remote node is reached, the remote CoNet functionality invokes Indication on the entity listening to the local entity port of the server application, and transfers the LID and the un-named-data, i.e. the SQL query. Subsequently, the server processes the request and replies to the client using Send2Location and including the response (un-named-data), the client LID and the local entity port 0x067497 of the local interface where further replies from the client are expected. The CoNet adds a local-identifier LID* addressing the local entity port and routes by LID the resulting data-units. When the *un-named-data* reaches the client, the CoNet functionality of the client device sends an Indication to the client application, which includes both the server response and the LID* of the remote server.

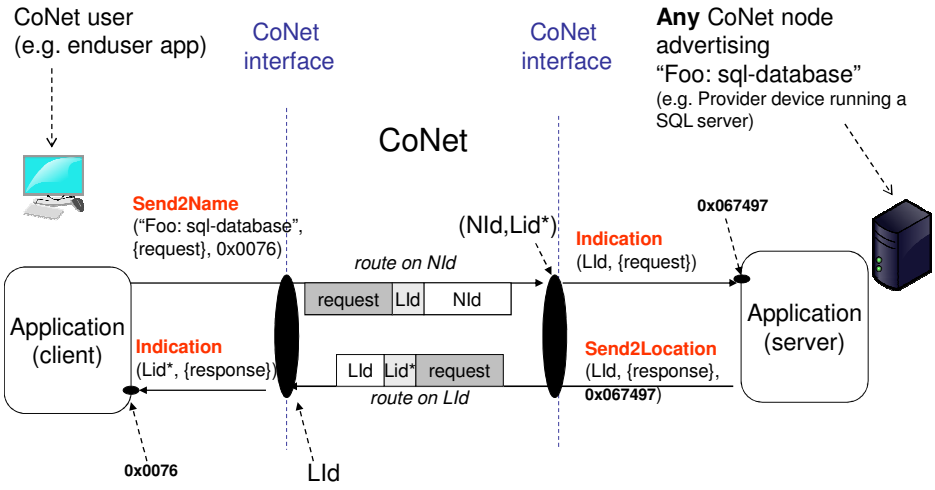

Figure 8: Request-response service session

4.1.3.5 Interactive service session

Figure 9 depicts the case of a CoNet user (e.g. an end-user CoMid entity) that wishes to interactively exchange un-named-data (i.e., middleware messages) with a remote CoMid entity identified by NID “Foo: mxm-content-provider”. This case is equivalent to the case of the request-response service session described in Section 4.1.3.4. However, after the first interaction, the endpoints continue to exchange un-named-data using Send2Location.

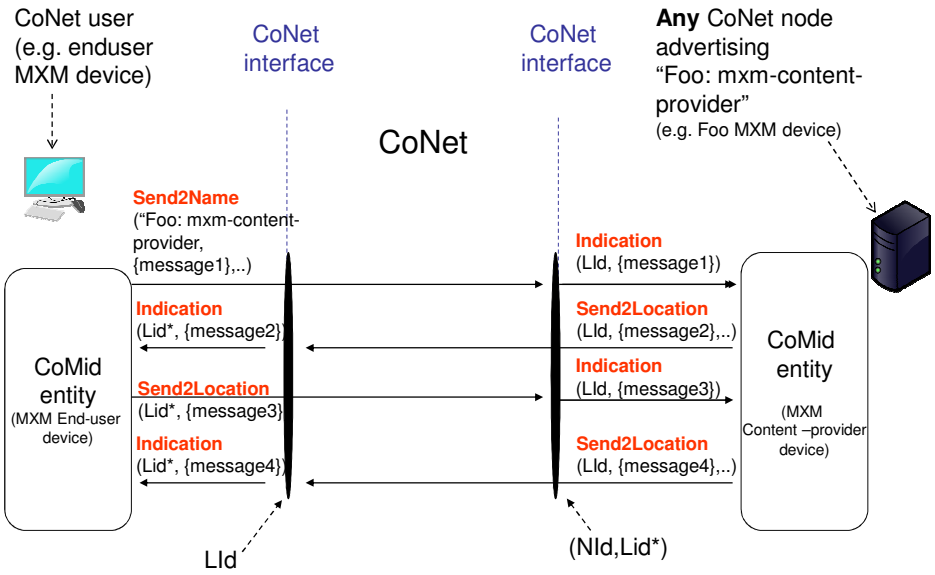

Figure 9: Interactive service-session

4.1.3.6 Remote advertising

The advertising examples presented in sections 4.1.3.1 and 4.1.3.2 describe cases in which the named-resource is provided locally. This means that a user that wishes to provide a named-resource should use a device that is always connected with the network. However, the user may not want to manage such a device and prefers to use a hosting provider who will take care of advertising her named resources. This can be achieved by using an application service, rather than a CoNet service. Thus, a Foo hosting provider may advertise a service named “Foo: hosting-service” to which the user can “tunnel” her named-resource (i.e., the named-data and the relevant NID). The hosting-service will then advertise (and store) the named-resource locally.

4.2 Content level (CoMid, CDS)

4.2.1 Description

Whereas CONVERGENCE network functionality handles named-resources (named-data or named-service-access points) only in terms of their names, content functionality provides the means to handle resources on the basis of “what” they contain and offer. These functionalities

are implemented using of a set of technologies that we call CONVERGENCE Middleware (CoMid) and the Community Dictionary Service (CDS)[8].

To understand the content of named-data and the service offered by a named-service-access-point, the CONVERGENCE system digitally describes *data* and *service-access-points* using metadata in a structure called a VDI. The content level provides functions for the *management of VDIs*, i.e. to create/edit, publish and un-publish VDIs.

The separation between the Content Level and Network Level is based on the conceptual separation between metadata and data. If metadata needs to be moved around, stored, searched, interpreted, updated, this is a task for the CoMid. When data has to be stored/retrieved/replaced/distributed, this is a task for the CoNet. This distinction clearly separates functionality that needs to "understand" VDIs (implemented in the CoMid) from functionality that only needs to know the name of the resources it is handling (implemented in the CoNet). Some VDIs may only contain metadata describing given data, and information (e.g. a NID) to retrieve the data (e.g. a picture). Others may also contain the data itself. In this latter case, the resource (e.g. a picture) is embedded within the VDI.

The CDS can be regarded as the CONVERGENCE knowledge base. The CDS will contain the ontologies used to perform semantic search. When the user subscribes to a set of VDIs satisfying certain search criteria, the CDS will provide CoMid with all matching concepts. CoMid will then perform matching with the VDIs.

The Content Level supports *content-based resource-discovery*. A user expresses interest in certain kinds of resources by specifying a set of criteria for resource attributes. The resource-discovery functionality will then return a list of VDIs compatible with the search criteria. Some resource attributes may contain semantic information. By interacting with the Community Dictionary Service infrastructure, the content level provides *semantic* and *content-based resource-discovery*. Resource-discovery is carried out by analyzing the set of VDIs distributed inside the CoMid.

A user can perform (semantic) content-based resource-discovery either by making a search request or through a subscription. In the case of a search request, the system answers the request and then forgets everything about the request. In the case of a subscription, the content level provides a *content-based subscription* that makes the discovery instance permanent. Every time the system finds a new resource fitting subscriber search criteria, it delivers a notification to the subscriber.

In addition to resource-discovery and subscription functionality, the Content Level also includes VDI management services (for example, creating, packaging, storing, rights management, and delivery). This functionality is either directly derived from the MPEG-M (MPEG eXtensible Middleware) framework being developed by MPEG, or offered as

[8] We propose not to include the CDS in the CoMid. In this way the CDS can be used even by devices that do not implement the CoMid. Given that CoMid plans to extend the MPEG M middleware, the inclusion of CDS functionality within this framework could jeopardize the standardization process.

additional building blocks compatible with MPEG-M. The CoMid functionality and architecture, detailed in the following sections, can thus be considered as a CONVERGENCE-specific part of a wider, standard, MPEG framework of services.

4.2.2 Semantic foundations of the Content Level

CONVERGENCE is intended to support a large number of user needs, involving different kinds of resources. In many cases, users will not refer directly to these resources (using their name – network identifier) but indirectly, by performing semantic search. Bearing in mind, that the VDIs representing these resources (and their metadata) are distributed across the network in ways that have no relationship to semantics, the search space is unbounded. It is thus necessary to narrow the number of queries that need to be executed. For this purpose, the Content Level introduces the concept of resource categorization and organizes the search space around this categorization. Furthermore, the Content Level supports a semantic *ranking* mechanism which, by using knowledge from the CDS, is able to provide results that match users request with a greater accuracy.

The proposed approach revolves around what we call the CONVERGENCE Resource Semantic Type (CoReST). CoReST is meant to represent resource categories, based on a core type that the user sets upon publication. CoReST is different from semantic metadata; however, intuitively, there has to be a connection between the two concepts. Consider, for example, a research paper; its semantic type is “research paper”. However it may also have a variety of metadata, such as authors, title, keywords, date, etc.

CONVERGENCE will provide a semantic taxonomy covering all supported CoReSTs. This taxonomy will be hierarchical. The root concept will be “Resource”. The first level children will be: “Digital Resource”, “Service”, “Real World Object” and “Principal”. In this scheme, “Video” is a child of “Digital Resource” and “Movie” is a child of Video. There are a number of indications that such ontology is feasible. The Common Procurement Vocabulary (CPV) [16] constitutes an exhaustive common vocabulary shared by all European stakeholders as the ultimate standard for references for material products and services in procurement contracts. The IST DISCREET project has also defined such a taxonomy [13].

CONVERGENCE will incorporate the following three types of ontologies:

- CONVERGENCE Core Ontology (CCO)
- Domain Ontologies
- User Ontologies

The CCO provides a semantic representation of the CoReST taxonomy. Figure 10 illustrates this concept.

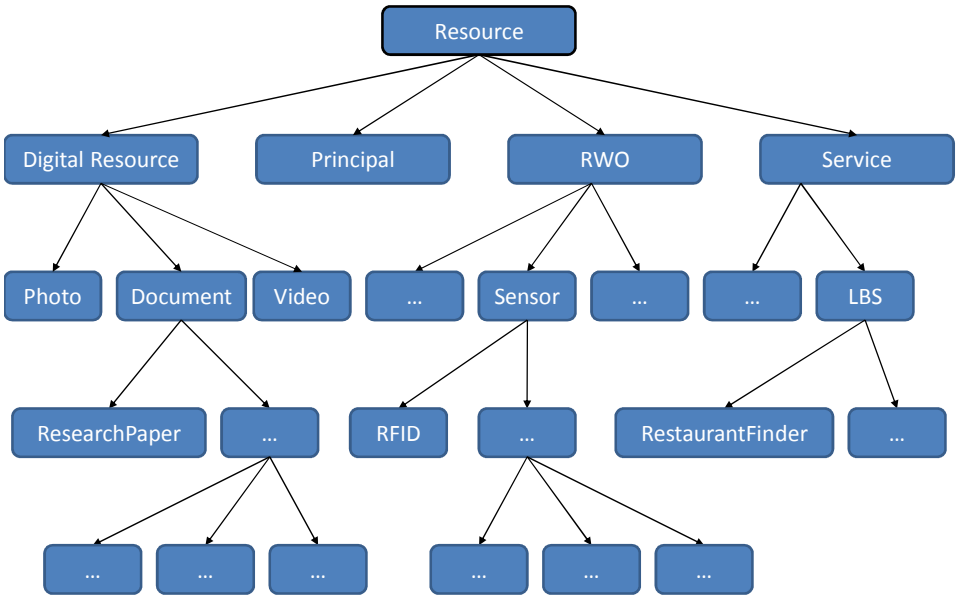

Figure 10: CONVERGENCE Core Ontology.

Domain and User Ontologies are used to associate VDIs with their semantic metadata. The difference between the two types of ontology is that domain ontologies are made available by operators or dedicated authorities, while user ontologies are created by users when they publish or subscribe to VDIs. Examples of domain ontologies include the Time Ontology [14] for describing time concepts, the GeoNames Ontology [15], and the thematic ontology of the IEEE Communications Society, which is used to classify concepts in communications and networking[9].

4.2.3 CONVERGENCE Middleware

4.2.3.1 Description

The CoMid architecture is based on concepts developed in the MPEG-M (MPEG eXtensible Middleware) standard, an extension of the former MXM MPEG platform proposed by MPEG. CoMid functionality is implemented using Elementary Services in MPEG-M [see working drafts of ISO/IEC 23006 at http://mpeg.chiariglione.org/working_documents.htm#MPEG-M], and, when MPEG-M does not provide the necessary functionality, by new CONVERGENCE-specific Elementary Services.

CoMid is based on a DDD (Describe, Discover, Distribute) paradigm. Its main purpose is to offer a set of standardized building blocks, APIs and protocols to describe, discover and

[9] IEEE Communications Society Digital Library: http://dl.comsoc.org/comsocdl/.

distribute resources. Higher-level Tools and applications are built by assembling CoMid blocks into custom software, manipulated by users.

4.2.3.2 Overview of key CoMid functionality

Describe functionality

Whenever a user produces a piece of multimedia content, or wants to market a good or service which is going to be governed and mediated through the CONVERGENCE system, the resource has to be described in a VDI. The VDI is a structured repository for all metadata and context information describing a specific resource. CoMid's "Describe" functionality is used to: i) access ontologies and directory services needed to tag resources; ii) create, identify and parse VDIs; iii) generate licensing information and contractual obligations which accompany the resource.

Discover functionality

Discovery of resources is crucial in large distributed networks. CoMid supports semantic searches over the VDI space[10]. CoMid Discover functionality uses semantic criteria to search for VDIs and to manage the topology of the VDI space, making it possible to perform queries efficiently.

Distribute functionalities

Users interact with the CONVERGENCE system through a publish/subscribe asynchronous paradigm. CoMid Distribute functionality implements pub/sub abstraction end-points towards the user, supports un-publish operations and implements content-based subscription to resources.

Security Services

The DDD functions of the middleware are based on a Security Services horizontal ecosystem of Elementary Services dealing with verification and authentication of resources and transactional orchestration of workflows between DDD services. They are part of the MPEG-M standard (Verify, Transact, Authenticate, Authorize, Check With elementary services), and will contribute to the Security Components area of the CONVERGENCE framework.

Content-Network Interaction

These services will reside at the interface between the CoMid and the CoNet. The current version of the CoMid architecture provides for at least one service in this functional area. This is the Map VDI ID-NID service. The job of this service will be to map VDI identifiers to their associated network-identifiers, and vice versa. In future versions of the architecture, it may be necessary to implement other services at this functional level.

[10] By semantic search, we mean a structured and multi-criteria search over the semantic part of VDIs. The result is a ranked list of resources, matching the search criteria.

4.2.3.3 CoMid technical architecture

The CoMid is organized around the concept of Elementary Services (ESs) assembled in a Service Oriented Architecture (SOA) and is largely based on the MPEG-M standard being developed by MPEG. Some of the Services needed by CoMid are directly available in the MPEG-M standard. Others will have to be specifically implemented within the CONVERGENCE framework. These services constitute a suite of building blocks facilitating the implementation of CONVERGENCE value chains (value chains is defined in MPEG-M as part of a business scenario). In such value chains, all devices are based on the same set of technologies, which they access via CoMid. These will include but not be limited to technologies standardised by MPEG. The use of a common set of technologies guarantees that all devices along the value chain can interoperate.

Figure 11 shows the set of ESs making up CoMid and the way they map onto functional areas of the CONVERGENCE middleware.

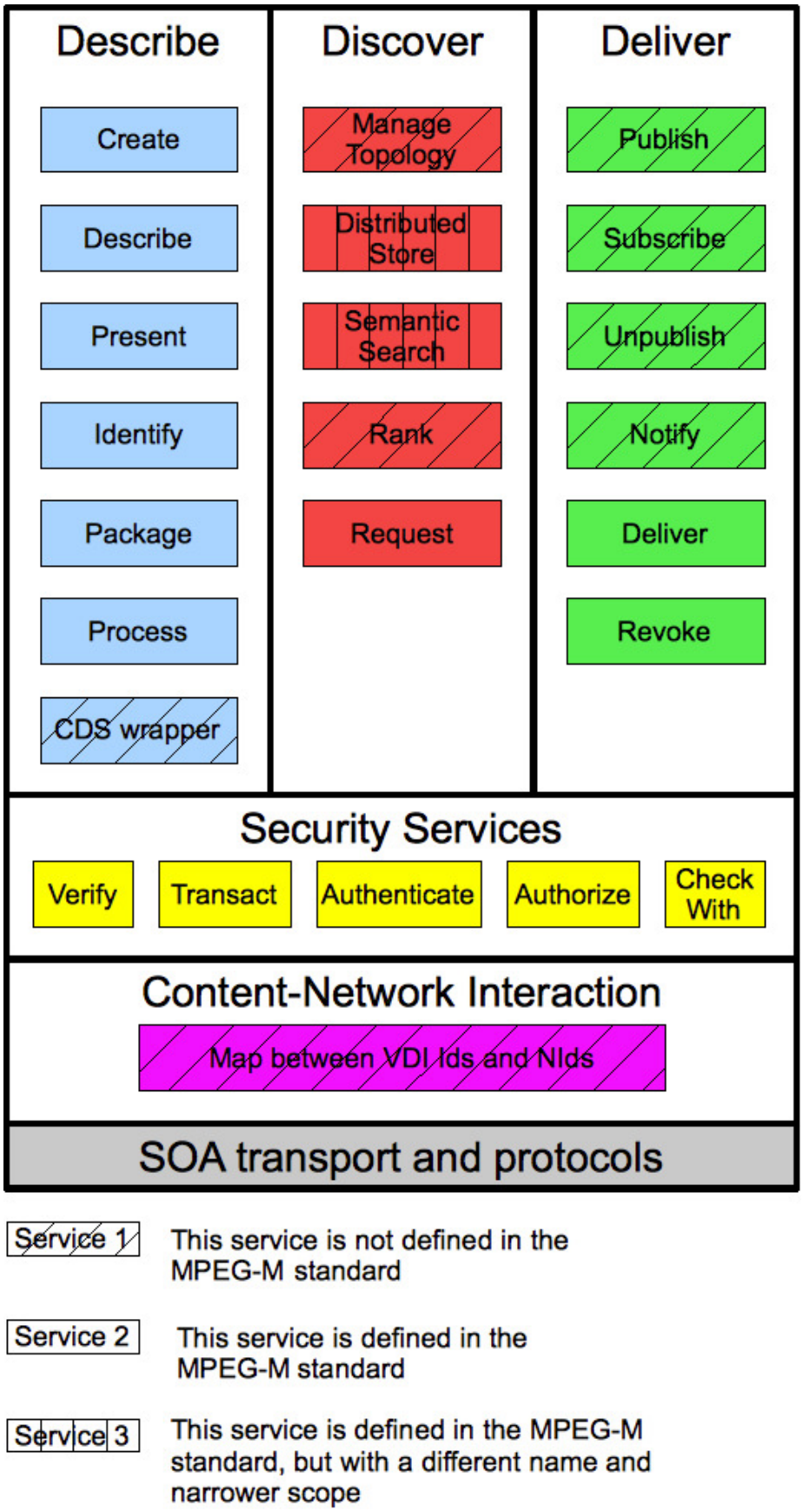

Figure 11: set of ES that compose the CoMid

New Elementary Services to be developed within the CONVERGENCE framework are marked with a hatched background. The "Search" and "Store" services already defined in the MPEG-M standard will be enhanced and in some cases modified. In what follows, we refer to these services as "Semantic Search" and "Distributed Store".

CoMid also introduces the concept of Entities, meaning the objects on which ESs act: VDIs (equivalent to Content in MPEG-M nomenclature), Devices, Events, Groups, Licences/Contracts, Services, and Users. By combining ESs and Entities, it is possible to describe a series of Actions. These are the basic building blocks in the CoMid SOA. The following Table 2 lists some possible CoMid Actions and provides a short explanation for each. The list is non-exhaustive. Also, readers should note that in MPEG-M nomenclature, the name of the standard MPEG-M actions reported in the first column of Table 2 have the term "Content" instead that "VDI" (so one should read e.g. "Authenticate Content"). As the terminology of CONVERGENCE is still not stable we defer the issue of finalizing these terms in the next deliverable D 3.2.

Table 2 –Some possible CoMid Actions

Action	**Description**	**Action Group**
Authenticate VDI	Allows Users to confirm the identity of a VDI	VDI Validation
Verify VDI	Allows Users to check the integrity of VDI	
Describe VDI	Allows Users to associate metadata with a VDI	VDI Creation
Identify VDI	Allows Users to assign Identifiers to a VDI	
Package VDI	Allows Users to prepare a VDI, ready for delivery	
Process VDI	Allows Users to perform operations on a VDI	
Semantic Search VDI	Allows users to search for a VDI according to specified criteria	VDI Usage
Subscribe VDI	Allows users to subscribe to a VDI according to specified criteria	VDI Usage
Deliver VDI	Allows a User to transfer a VDI to another User	VDI Transfer
Distributed Store VDI	Allows Users to save a VDI for later use on other Devices	VDI Storage
Authenticate Contract	Allows Users to confirm the identity and signers of a Contract	Rights Management
Check With Contract	Allows Users to verify if a usage request matches with the content (e.g., obligations, prohibitions)	

	expressed in a Contract	Rights Management
Create Contract	Allows Users to generate a Contract	
Deliver Contract	Allows Contract transfers between Users	
Identify Contract	Allows Users to assign Identifiers to a Contract	
Negotiate Contract	Allows Users to conclude an agreement on terms and conditions of use e.g., obligations and prohibitions, with respect to a VDI item, a Device or a Service	
Request Contract	Allows Users to request a Contract	
Revoke Contract	Allows Users to discontinue the validity of a Contract	
Authorise User	Allows Users to obtain authorization of some usage(s).	Rights Management
Authenticate Device	Allows Users to confirm the identity of a Device	Device Validation
Report Event	Allows Users to report an Event	Event Reporting
Request Event	Allows Users to request the creation of an Event Report for one or more Events that have occurred or are going to occur	
Authenticate Group	Allows Users to confirm the identity of a Group	Group Management
Create Group	Allows Users to create groups of entities such as Users, Devices, VDIs, Groups and Services	
Describe Group	Allows Users to associate metadata with a Group	
Identify Group	Allows Users to assign unique Identifiers to Groups	
Process Group	Allows Users to change the attributes of a Group	
Authenticate Service	Allows Users to confirm the identity of a Service	Service Validation
Describe Service	Allows Users to associate metadata with Services	Service Management
Identify Service	Allows Users to assign Identifiers to Services	
Process Service	Allows Users to change the scope and configuration of an Elementary	

	Service or compose an Aggregated Service	
Authenticate User	Allows Users to Authenticate Users	User Identity Validation
Describe User	Allows Users to Describe Users	User Characterization
Identify User	Allows Users to assign unique Identifiers to Users	
Search Entity	Allows Users to locate Entities such as User, Services, Contracts, Licences, VDIs, Devices and Groups, satisfying some criteria	General Search Action
Transact Entity	Allows Users to interface with Payment and Cashing systems with regard to Entities such as Services, Contracts, Licences, VDIs, Devices and Groups	Economic Action

These Actions define a corresponding set of protocols and APIs that enable any user of the CONVERGENCE system to access those services in an interoperable fashion. The CoMid architecture only defines the payload format, which is XML-based. This payload can be delivered using any suitable transport protocol, such as HTTP or SOAP. Underlying CoNet functionality will support this kind of interactive service session.

Each CONVERGENCE peer will be augmented with the DDD, Security and Content-Network Interaction functionality of the CoMid. Some parts of Content-Network Interaction functionality may also need to be deployed on CONVERGENCE nodes, at the Network Level, within the CoNet.

The CoMid is based on open APIs and allows third parties to provide applications to those APIs.

4.2.3.4 Orchestration and Aggregation of CoMid Elementary Services

ESs can be orchestrated to provide powerful chains of operations. As an example of service orchestration, consider a scenario in which an End User uses the “Subscribe VDI” Action to obtain a sequence of songs satisfying certain VDI and User Descriptions (i.e. metadata associated to the VDI entity and the User entity). The following Figure 12 depicts how a chain of Actions could be set up to respond to the User’s request.

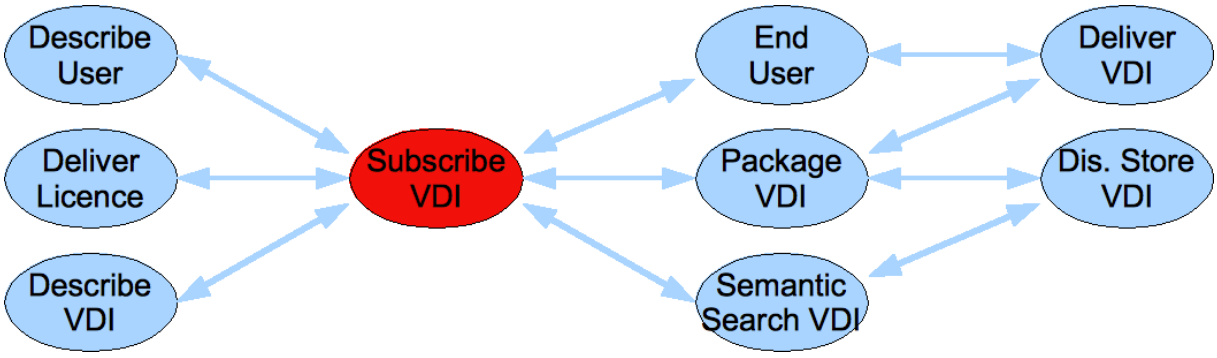

Figure 12: **A possible chain of Actions centred on the Subscribe VDI Action**

In Figure 12, an End User invokes "Subscribe VDI" to obtain appropriate information from "Describe VDI" and "Describe User". She then uses this information to invoke Semantic Search VDI actions and thus to prepare a sequence of songs. She then obtains the necessary licences from "Deliver Licence". The sequence of songs is handed over to Package VDI, in the form of a set of VDIs and associated resources. Package VDI obtains the Resources from "Distributed Store VDI" and hands over the Packaged VDI to "Deliver VDI" which streams the Packaged VDI to the End User.

4.2.3.5 Elementary Services specific to CONVERGENCE

These Services perform analysis and harmonization of the semantic metadata headers for each VDI they cache, replicate or route. Proper orchestration of these Services ensures that all VDIs that pass through the CONVERGENCE peer or are stored on it are parsed and indexed. This makes it possible to maintain a large distributed database of all known fields in the VDI metadata.

The Semantic Search service will offer an interface making it possible to retrieve the searchable fields in metadata space. In this way, a semantic query in metadata space would consist of a vector of target couples {property, instance}. The system could support exact matches, range matches and proximity/fuzzy string match.

These different matching operators present different degrees of difficulty for practical implementation. Whether or not they are supported natively will depend on the underlying substrate running the distributed metadata database.

4.2.3.6 CDS wrapper ES

Orchestration and aggregation of elementary Describe Services by the CONVERGENCE middleware will allow distributed applications to exchange information related to VDI items and their parts, including rights and protection information. The CDS will make it possible to parse and interpret community-developed ontologies. The entry point for access to the CDS is the CDS wrapper ES in CoMid. The CDS wrapper service plays a key role in:

- The VDI publication procedure, where the CONVERGENCE Tools exploit CDS services for the semantic annotation of resources, and to fetch the vocabulary available to describe them.
- The VDI search procedure, where the Semantic Query ES exploits the ontology reasoning capabilities of CDS, as described later in the paragraph on 'Semantic Query ES'.

4.2.3.7 Manage Topology ES

CoMid is responsible, among other things, for managing the metadata distributed across the CONVERGENCE system. To this end it will provide a semantic toolbox offering semantic search as part of its functionality. As explained in the introductory section about CoReST and the semantic foundations of CONVERGENCE Content Level, there are serious scalability issues when performing semantic queries over VDIs.

In this Section, we show how we can use CoReST to organize the metadata used by semantic services and improve system performance.

In order to restrict the search space, the metadata circulated in the CONVERGENCE system will be stored in a distributed overlay network composed of all CONVERGENCE peers running CoMid. The topology of the overlay network will be managed by the Topology Management Service, in such a way as to facilitate and support efficient semantic search. This "discovery" overlay network will be divided into virtual clusters, called *fractals*[11]. Each fractal will be composed of peers containing metadata for resources of a given semantic type, as shown in Figure 13. By narrowing the search space, we can improve search efficiency, without imposing limitations on the locations where CoNet stores named-resources.

11 The idea of applying clustering methods for efficient information retrieval has already been proposed in the literature (Ming 2010).

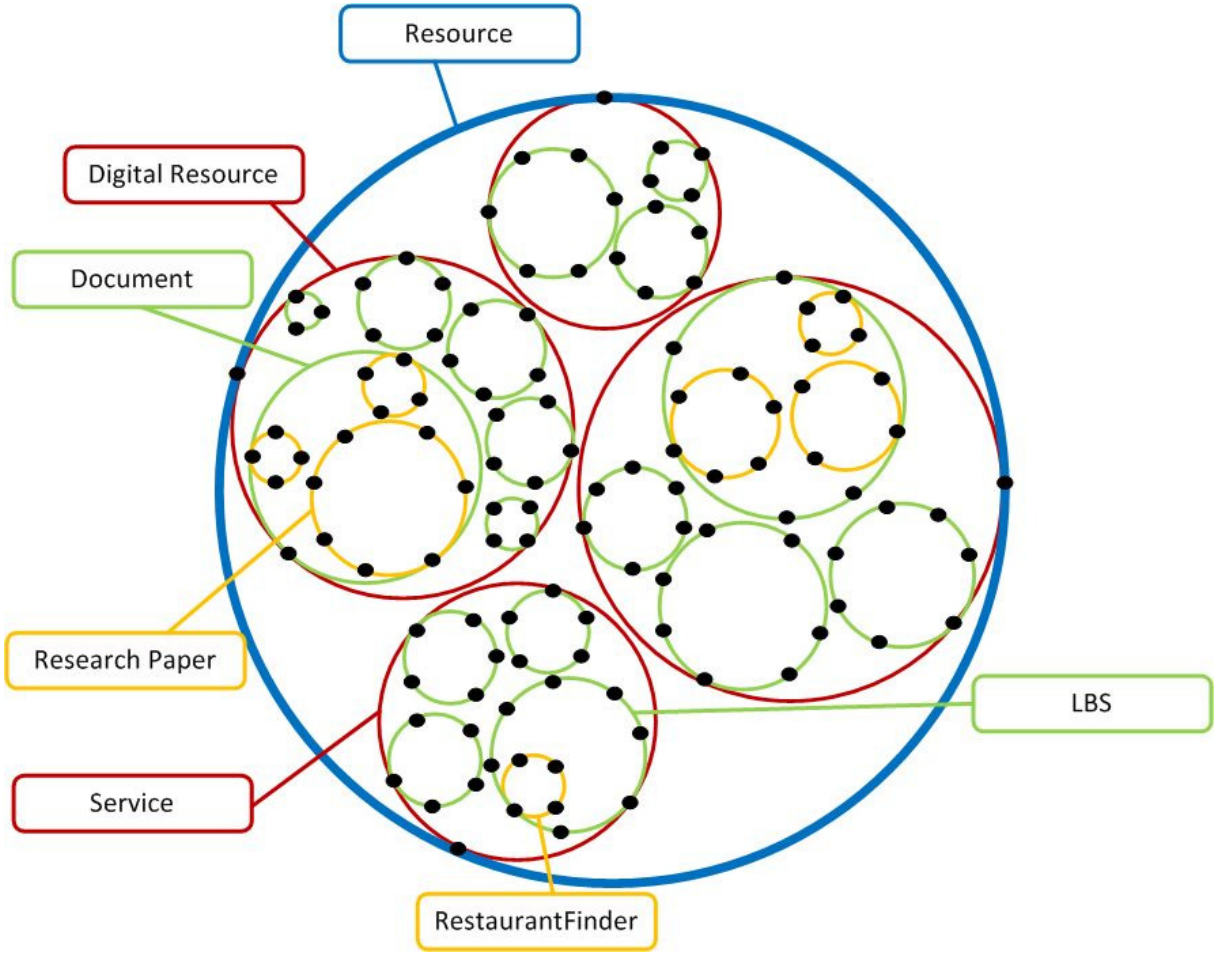

Figure 13: Topology management of the Discovery Network

When a user tries to publish a VDI containing certain resources, she selects the resource semantic type that better describes them. When the middleware receives the VDI, it places relevant metadata, extracted from the VDI, in the appropriate fractal in the discovery network. This way, when a user searches for a VDI with a given resource semantic type, the CoMid will optimize the search inside the respective fractal.

The Manage Topology ES uses gossiping techniques to distribute publications and subscriptions across network fractals. The use of these techniques in CONVERGENCE is inspired by research in the field of epidemics and gossiping in distributed systems. Gossip algorithms have been widely used for data dissemination in large-scale distributed systems where they are a popular solution due to their simplicity, robustness and flexibility [4]. Work by A.-M. Kermerrec et al. suggests that group communication is scalable and fault tolerant [3]. S. Voulgaris and M. Van Steen in [1] argue that gossip algorithms, combined with semantics, can provide very good results in terms of speed and accuracy. K. Sripanidkulchai et al. [2] have proven that using clusters of hosts with similar content (in their case the same files) improves the time needed to discover a resource. In the case they studied (Gnutella) it also improved download speed.

Underlying System Model

Each node executing Manage Topology exposes a set of internal sub-services needed for the correct operation of distributed gossiping.

1. *publication gossiping*: invoked by other nodes, to notify the executing node that there has been a publication regarding its resource semantic type and to send the node the relevant metadata extracted from the original publication VDI.
2. *subscription gossiping:* invoked by other nodes, to notify the executing node that a user has made a subscription regarding its resource semantic type.
3. *subscription matching*: invoked by other nodes to return VDIs which match a previous subscription gossiped by the executing node.

Each node executing Manage Topology is assigned one or more resource semantic types. Given that each CONVERGENCE node is also a Real World Object (RWO), each node will itself be described in a VDI. This VDI is used to access the above-mentioned sub-services and to hold the CoReSTs for the node. It contains:

1. The resource semantic types, this node provides information on.
2. The named-service-access-point (i.e. the CoNet network-identifier to be used to physically access the sub-service on the node) of this node's *publication gossiping*, *subscription gossiping* and *subscription matching* sub-services. These are called *publication gossiping endpoint*, *subscription gossiping endpoint* and *subscription matching endpoint.*

As is usual in gossiping techniques, each node maintains a partial view of the Discovery Network, by keeping a list specifying <CoReST, *publication gossiping endpoint*, *subscription gossiping endpoint*> for a number of other nodes, extracted from their respective VDIs.

Topology Management operations during Publish actions

Whenever a Publish action is executed, Manage Topology will perform the following steps (see Figure 14):

1. Check the Published Entity's resource semantic type.
2. Based on this type, choose a number of nodes, and use their *publication gossiping endpoint*, to send them a message containing metadata extracted from the VDI, including the VDI identifier. Each node receiving this message stores it and forwards it to another set of nodes, of the same or related resource semantic type[12].

[12] The technique used to choose these nodes has not yet been decided. It may be either hierarchical or random. The time-to-live (TTL) for this packet will be defined when the system has been set up and we will be able to run some tests.

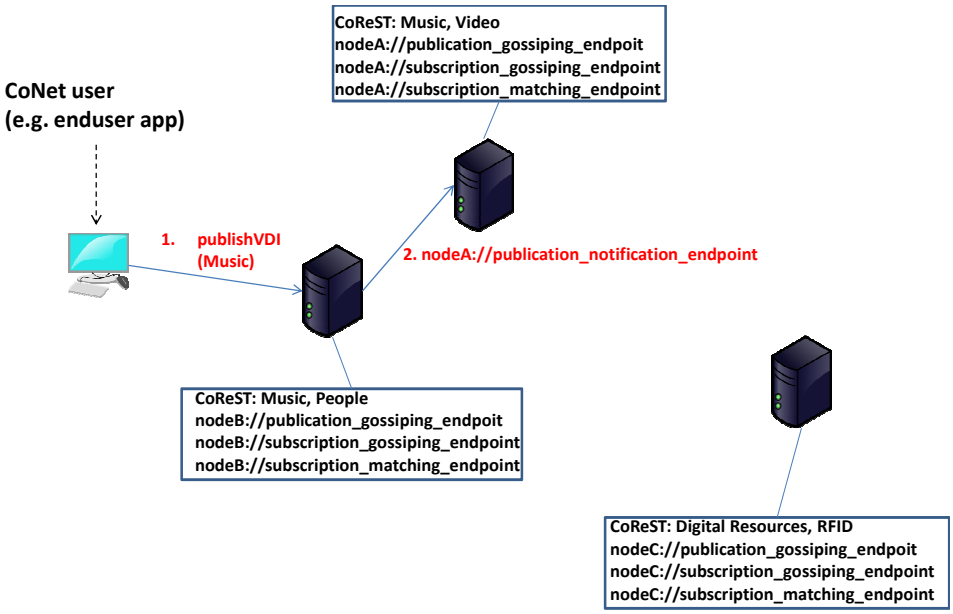

Figure 14: Use of publication gossiping endpoint

Topology Management operations during Subscribe actions

In the case of a subscription expressed in terms of semantic criteria Manage Topology will perform the following actions (see Figure 15):

1. Check the Subscribed Entity's resource semantic type.
2. Based on this type, perform the following actions:
    a. Check local storage for VDIs matching the required concept.
    b. Checks the received metadata (received via gossiping messages) for a matching concept, using the VDI identifier to fetch VDIs for which a match has been found.
    c. Use the subscription gossiping endpoint to notify a set of nodes (which may or may not be the same as the ones used for the publication gossiping endpoint) of the request for that specific CoReST.

When a node receives an interest from a gossip round, it performs steps 1 to 2-b. Then, based on the gossip packet TTL, it decides whether or not to perform step 2-c. If the node finds a VDI matching the subscription criteria, it uses the subscription matching endpoint to send the VDI back to the node that started the gossip round.

Each VDI received by the initial node is returned to the caller ES.

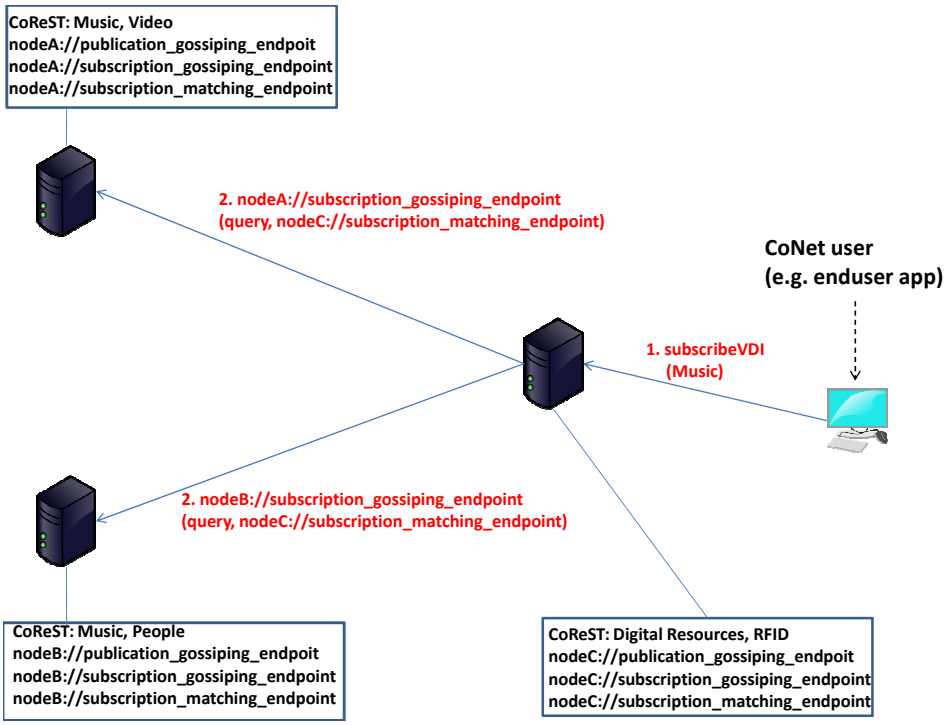

Figure 15: Use of subscription gossiping and subscription matching endpoints

4.2.3.8 Distributed Store ES

Distributed Store allows storage of CoMid's entities by other CONVERGENCE peers and nodes. From the Network Level point of view, VDIs are just like any other generic data, and can thus be stored in the network like any other named-resource. The Distributed Store ES exploits this approach. Thus each VDI published by Publish VDI actions will be advertised, by the Distributed Store, at the Network Level. In this way, the network behaves as a storage facility for metadata produced at the Content Level: VDIs are identified by a network-identifier (NID) at the Network Level. This means that the network level primitives (e.g. advertise, revoke) defined in the previous chapter can be applied to VDIs. Thus, Network Level operation and primitives support VDI publish and un-publish operations at the middleware level.

The storage of VDIs in the CoNet allows the CONVERGENCE system to support competitor "search engine" providers. Such providers could exploit the power of VDIs in describing resources, but use alternative discovery algorithms. It also allows CoMid to retain only those VDI fragments that are actually used by the search/indexing service. This could simplify the development of middleware storage services, exploiting CoNet functionality to store and retrieve VDIs.

A VDI references its resources by using the NIDS assigned to them.

A *VDI references other VDIs*, by using their VDI identifiers; however to access referenced VDIs it is necessary to map VDI identifiers to NIDs (assuming that VDIs have been previously advertised in the CoNet with a NID).

Additionally, by accessing the NID for a VDI, users can download its metadata directly.

4.2.3.9 Semantic Search ES

When a user does not have a specific resource in mind, she can use Semantic Search VDI actions to find resources matching her needs. The Semantic Search ES orchestrates the VDI matching process, in close cooperation with the CDS, effectively supporting the *semantic and content-based subscription* functionality of the CoMid.

In this scenario, the user provides the service with a semantic description of the resources she is interested in, in the form of a query. The service returns a list of VDIs satisfying the query. This procedure is performed in two sequential steps.

A simple user query will consist of a set of {property, instance} criteria connected via logical operators. A more complex query may have nested criteria in the form {property1, {property2, instance}}. For example a user may be interested in "photos of cities in Europe" i.e. {CoReST, Photos} AND {pictureTakenIn, {isCityIn, Europe}}.

Step 1. Query expansion

In this step the CoMid interacts with the Community Dictionary Service seeding it with the user's query. The CDS reduces complex queries to a series of simple queries from which it generates additional simple queries. In the end, each query consists of a set of contains {property, instance} pairs that cannot be inferred from other pairs. In cases where there are many instances satisfying a query this kind of complete expansion may not be possible.

In this case the CoMid will first perform the expanded semantic queries over VDI metadata, in order to narrow down the search space, and then follows an inverse procedure by trying to match VDI metadata with CDS concepts, before considering a VDI match. For example, consider a user looking for papers published in 2002 in a conference that took place in Europe. If we fully expand the query, then we will have to incorporate any town or village of Europe. To come around this, we narrow down the search space by first getting all VDIs referring to papers published in conferences in 2002 and, then, for these papers, we get their location and check whether it is in Europe.

Step 2. VDI matching

The Semantic Search ES performs the search by projecting it onto the Discovery Network via the Manage Topology actions detailed in the previous paragraph.

If the queries are fully expanded then a VDI is considered a match if and only if it satisfies at least one of them. If the resources' metadata for a certain VDI do not contain the property of a given {property, instance} pair of a simple query, the query is considered to be unsatisfied. In this way the system adopts the closed world assumption (i.e., the presumption that what is not currently known to be true, is false).

If queries are not fully expanded, that is if they contain {property, instance} pairs from which it would be possible to infer other {property, instance} pairs, the CDS will verify the matching. VDIs that are matched by the above procedure are returned to the user.

4.2.3.10 Ranking ES

The Ranking Service sorts the results produced by the CDS Wrapper Service for use in describing resources and for the VDI matching procedure (managed by the Semantic Query Service).

When the service is used to rank the results used to describe a resource, the result will be a list of properties or instances of CDS ontologies. When it is used in VDI matching, the result will be a list of VDIs. Sorting will be based on the frequency with which the different concepts of the same VDI properties are used. For example, if there are two concepts matching the same property, then the VDI referring to the most popular one will have a better rank. This method will promote the development of a widely accepted common vocabulary for describing resources, thus increasing the discoverability of both popular and unpopular resources.

4.2.3.11 Publish ES

The CONVERGENCE middleware Publish Service is the CoMid front-end service that receives a VDI before it is published. Authoring tools used to create VDIs will make calls to CoMid's Create VDI Action. A User wishing to publish a VDI will invoke the Publish elementary service on a VDI entity. The Publish Service will then use the Distributed Storage Service to disseminate the necessary metadata.

The Publish Service will also be responsible for assigning a unique VDI identifier, which is returned to the user. If the user decides to change metadata, resources or access rights for a VDI, the publish service will provide a transparent process, in which each update creates a new version of the initial VDI, while the system maintains a VDI version history (unless the user chooses otherwise).

4.2.3.12 Subscribe ES

The CONVERGENCE middleware Subscribe Service enables users to express their interests by specifying either VDI identifiers or semantic metadata (through the Semantic Query Service). The system will then automatically and periodically deliver all relevant VDIs to them. This possibility frees users from routinely performing repetitive search tasks and eases the burden on the CoMid, by eliminating a considerable amount of redundant VDI search activity.

As mentioned earlier, users may directly subscribe to an individual VDI, to a set of VDIs (specified by their identifiers) or to an abstract set of VDIs, composed of all VDIs that match a set of semantic search criteria. In all these cases, the Subscription Service takes the following actions:

1. The CoMid instance of the peer where the user makes her subscription ("local CoMid") locally registers the user specified search criteria, the user associated with the subscription and its expiry date.
2. In the next step:

a. If VDI ID was given, the service invokes a Distributed Store VDI to directly fetch the given VDI.
b. If semantic parameters were given, the service invokes a Semantic Search ES.

The local CoMid periodically performs searches for VDIs that satisfy the registered (step 1) subscription criteria (that is, it periodically repeats steps 2 or 3 until the expiry date for the subscription). The expiry date for the subscription is either specified by the user or set to a default value.

CoMid keeps track of the users subscribed to a VDI, so that the owner may contact them via CONVERGENCE. However, the users' credentials are not necessarily visible to the owner.

Where the owner's privacy settings allow, the subscription service maintains a VDI history. For example, in case of a medical record, updated every time the patient goes to the doctor, the doctor will be able to see the previous versions of the record.

Every time a users tries to browse a VDI or its resources, the CoMid Subscribe Service will evaluate the user's access rights.

4.2.3.13 Unpublish ES

The objective of the Unpublish ES is to support digital forgetting and garbage collection. Digital forgetting is a complex task, and the exact details of the whole Unpublish process will be better defined in upcoming project deliverables. Unpublish requires strict collaboration between CoMid and CoNet, given that the CONVERGENCE middleware handles metadata separately from the network and that metadata is spread throughout the system using probabilistic gossiping techniques. The unpublish process is guided by the CoMid and by the Application level, but it is based on the *revoke* primitive of the CoNet.

Operatively, unpublishing always starts from the CoNet and is propagated to the middleware. Whenever CoNet signals that a requested resource has been revoked, the matching VDI and its metadata also get unpublished by the CoMid at the Content Level.

4.2.3.14 Notify ES

The Notify ES is strongly connected with the Subscribe ES, described above. As indicated previously, the Subscribe ES is asynchronous; that is, the user may express her interest on a VDI and ask to be notified when resources are found at a later time. Notify ES is triggered by the subscribe service each time a matching VDI is found. The action taken depends on user preferences. For instance the service may send an email every 5 VDIs or every hour.

4.2.3.15 Map VDI ID-NID ES

This service is invoked whenever other ESs or the CoNet need to retrieve a VDI identifier given its associated network-identifier or vice versa. The service resides on the border between CoMid and CoNet and can be invoked by both.

Associations are created whenever a new VDI is created by a Create VDI Action (performed by the CoMid) and are subsequently stored inside the CoNet. This service is also able to generate and assign new network identifiers to VDIs that are passed to it.

4.2.3.16 Preliminary description of CoMid Application Program Interface

Service Primitive	Which ES	Applicability	Origin	Output	Description
getVDI_ID()	Identify	Data, services	User	VDI ID	Generates and delivers a system wide unique VDI ID.
createVDI (VDI ID, CoReST, contents, author, restrictions)	Create	Data, services	User	VDI	Generates a new VDI. The arguments are the VDI ID (mandatory), its contents (mandatory), its resource semantic type (optional), its author (optional) and restrictions (if any). The system sets the default access rights to the VDI.
getVDI (VDI ID, userCredentials)	Get	Data, services	User	VDI	The CoMid function to retrieve a VDI from the system. The system evaluates the user's right to retrieve said VDI.
publishVDI (VDI)	Publish	Data, services	User	Success / error	At publishing time, the CoMid is given the VDI. It extracts the necessary information from the VDI (which it then gossips to other peers), and "injects" the VDI into the CoNet.
subscribeToVDIs ([VDI id1, VDI id2, ... OR criteria], CoReST, TTL, userCredentials, notificationType)	Subscribe	Data, services	User	VDI ids (if there are instant matches)	A user can subscribe to a VDI using its identifier or indicating search criteria. Optionally, the user can provide the TTL of the subscription and specify how she would like to be notified. The system needs the user's credentials to check whether she has access rights to the VDI.

searchVDIs (criteria, CoReST, TTL, userCredentials, notificationType)	Search	Data, services	User	ID[], metadata[], /failure	Same as the previous one, except that in this case the user does not want to be notified of matching VDIs. The result consists of the IDs for the matching VDI.
sendMsgToSubscribers (VDI ID, fromDate, toDate)	Manage Topology	Data, services	User	Success/failure	The author can send a message to all VDI subscribers who have subscribed in the given date range.
signVDI (VDI, fragment, userCredentials, method)	Create	Data, services	User	Processed VDI/failure	The author or a privileged user may want to sign the VDI or a fragment using a specific method, according to her credentials.
markVDI (VDI, fragment, userCredentials, method)	Create	Data, services	User	Processed VDI/failure	Same as the previous function, except that this time the user applies a watermark.
encryptVDI (VDI, fragment, userCredentials, method)	Create/Process	Data, services	User	Processed VDI/failure	Same as the previous function, except that this time the user encrypts a VDI fragment.
unpublishVDI (VDI ID, userCredentials)	Unpublish	Data, services	User	Success/failure	A privileged user can unpublish a VDI if she has the proper credentials.
notifyUser (VDI ID[], metadata[], method, arguments)	Notify	Data, services	CONVERGENCE	Success/failure	CONVERGENCE uses this operation to notify subscription results to the user.

4.2.4 Community Dictionary Service

4.2.4.1 Description

The Community Dictionary Service is the distributed knowledge base of the CONVERGENCE Framework. One can see it as the semantic broker of the system providing a means for annotating resources with metadata based on common vocabularies. These vocabularies take the form of ontologies developed and maintained by community projects and CONVERGENCE users.

Appending metadata to resources is of great importance for organizing, sharing and discovering resources. But this is not a trivial task. The CDS will provide a means to overcome common issues arising in this task, including word disambiguation and problems arising from the use of different vocabularies to describe resources.

To this end the CDS uses:

- Wide spread community ontologies like Dublin Core, DBpedia, SKOS, FOAF.
- Knowledge mapping, in the form of inter-ontology equivalence statements.
- Ontology reasoning, for discovering intra-ontology facts.

The CDS provides the CoMid with:

- Query extraction and evaluation services powered by its ontology reasoning capabilities.
- An ontology properties and instances search service, powered by information retrieval techniques, like searching for synonyms and words in other languages or finding various morphological forms of words by stemming them to their root.

4.2.4.2 Preliminary description of CDS Application Program Interface

Service Primitive	Applicability	Origin	Output	Description
getProperties (description)	VDI publication - VDI search	User	List of ontology properties	This service provides the user with vocabulary for describing a resource. The user provides a description of the property she wants to annotate her resource with and the CDS returns a list of already available ontology properties. For example the user wants to state that the 'author' of her pdf is 'Stephen King'. When calling this primitive with 'author', the CDS will return a list of ontology properties along with descriptions that possibly cover her needs, e.g. <dc:author, dc:creator …>
getInstances (description)	VDI publication – VDI search	User	List of ontology instances	This service provides the user with vocabulary for describing her resource. The user provides a description of the instance she wants to annotate her resource with and the CDS returns a list of instances already available in the ontology. For example the user wants to state that the 'author' of her pdf is 'Stephen King'. When calling this primitive with

				'king' the CDS will return her a list of ontology instances along with their descriptions, e.g. <dbpedia:Stephen_King, dbpedia:Coretta_Scott_King …>
expandQuery (semantic query)	VDI search	CONVERGENCE	Semantic query / error	This service performs semantic query expansion. The CDS parses the query appropriately into its building blocks (ontology properties, ontology instances, and logical operators). It then starts the reasoning process by identifying equivalent properties and instances from available ontologies. Inferred knowledge is returned to the CoMid in the form of an expanded semantic query. If inferred knowledge extracted from the query provided approaches a continuous set (e.g. a geographical region, see example in paragraph 4.2.3.9), the CDS returns an error, stating that the query cannot be expanded.
evaluateQuery (semantic query)	VDI search	CONVERGENCE	True or false	This service runs a query against available ontologies in the CDS and returns a value stating whether the query can be satisfied. This service is extremely useful when a semantic query cannot be fully expanded, since it checks whether a VDI property has a match on a CDS concept, whereas the normal search process checks whether a CDS concept has a match on a VDI's semantic metadata.

4.2.5 Examples of the CoMid Application Program Interface in use

4.2.5.1 VDI Publication

A registered user, Alice, logs into the CONVERGENCE platform, which retrieves her credentials. Through the application user interface, Alice defines resources which will be referenced by a VDI and its associated metadata. She defines the metadata with the help of CONVERGENCE application tools, which use the CDS.

Then, Alice asks the application to publish her VDI. The application tools call publishVDI using the content provided by Alice and her credentials. The middleware then calls the

Advertise network operation to publish the VDI in CoNet. Finally, the metadata is distributed to the CONVERGENCE system, as illustrated in Section 3.2.4.8. The middleware returns the VDI identifier to the tools, which in turn passes it to Alice through the user interface.

4.2.5.2 Subscribe to VDI using VDI ID.

Alice logs into the system in order to find the VDI containing her CV. She has noted down the VDI identifier that she got back when publishing the VDI, so she decides to request the VDI directly, using its identifier. She also indicates that she would like to receive an email with the contents of the VDI. Her email is in a RWO VDI describing Alice.

The tools call the subscribeToVDI operation using the VDI ID as a parameter and setting "email" as the notification type. The middleware then retrieves both the CV VDI and Alice's VDI using its name, which is contained in her CONVERGENCE account information. Finally, they notifyUser with the CV VDI. This sends the email to the address contained in Alice's VDI.

4.2.5.3 Subscribe to VDIs using Semantic Search

Alice logs into the system in order to find books written by Stephen King. She really enjoys his books but she doesn't have a lot of free time, so she would like to read a book with less than 200 pages.

Through the application user interface and the CONVERGENCE application tools, Alice defines her search criteria. The application tools invoke the CDS and end up with the query 'hasAuthor' - > 'Stephen_King' and 'hasPages' - > '< 200'. The CoMid begins by expanding the query and then performs metadata matching with VDIs that have already been published. The CoMid finds 3 VDIs matching Alice's search criteria. However, for licensing reasons, one of the discovered VDI is not available to Alice. The CoMid thus returns just 2 VDIs. Alice reads the description of each book and chooses to subscribe to the 2nd one using its VDI ID.

4.3 Applications level

The applications level provides the interface between users and CONVERGENCE and is the top level from the user point of view.

In order to give more flexibility to CONVERGENCE users, the applications level is split into two layers: a Tools layer and a User Applications layer, (see Figure 16). The Tools Layer contains functional components that combine a subset of CoMid functionalities and can be reused by many applications but not all of them. Typical examples might include a Video Content Creation tool or a Service Catalogue Management tool.

The User Application Layer allows people to use the CONVERGENCE platform and, for the typical Converge user, is the only layer of interest. The CONVERGENCE user only needs to know how to use the applications available in this layer.

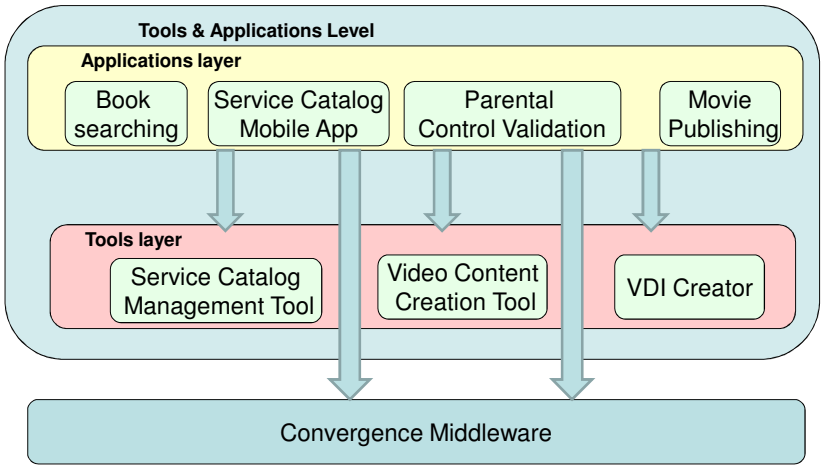

Figure 16: Applications level

The tools provided are components that are neither sufficiently generic to be considered part of the middleware nor sufficiently specific to be considered applications. The CONVERGENCE framework will offer an open environment in which CONVERGENCE developers will be able to deploy their own tools and publish them to other users. Based on their popularity and usefulness and after a security and consistency audit, these new tools could become CoMid components and be incorporated in the middleware.

When building their applications, the developers can use these (new and old) tools or can access the middleware directly, using the CoMid API.

4.3.1 CONVERGENCE Application Tools

CONVERGENCE Application Tools stand between the middleware and applications. Their main responsibility is to provide the user interface with the data needed to assist the user. The tools layer is strongly connected both with the middleware and the Community Dictionary Service.

Using the CDS, the tools can provide the user with information helping her to describe her resources. The CDS contains all the concepts available in the CONVERGENCE framework. Hence, when a user tries to describe a resource (either for publication or for semantic search) there has to be an interconnection between the user interface and the CDS.

Finally, CONVERGENCE application tools are responsible for integrity and consistency checks on the VDIs, before the end up on the network. It is the Tools layer that receives VDIs from users and deliver them to the middleware to be published.

4.3.2 CONVERGENCE Human Machine Interaction

One of the major innovations of CONVERGENCE is that it introduces the concept of Versatile Digital Items (VDIs) with the ability to represent almost any kind of resource. To exploit the potential of VDIs, users need a sophisticated user interface, allowing them to define the VDIs they want to publish or to retrieve.

The CONVERGENCE human-machine interface aims to facilitate VDI search and retrieval as well as VDI creation and modification. The interface will provide a user friendly environment where convergence users can define and extract all requested information about particular categories of VDI. Procedures such as adding private or public comments to a VDI will be supported with the necessary security and privacy constraints.

Another important feature of the human-machine interface will be the ability to visualize different VDI components in an effective manner that facilitates VDI navigation and information extraction. The human machine interface will allow CONVERGENCE users to create and publish new VDIs, or browse through different VDIs belonging to the same category.

Finally, human machine interface will allow interoperability with context-aware applications, such as GPS navigators. In this case, the procedure will facilitate the retrieval of VDIs directly or indirectly associated with specific geographic locations (e.g. a VDI representing sales locations for a specified product).

5 Implementation strategy

This section presents an implementation strategy for the system architecture (see Figure 2). It is related to the test of the CONVERGENCE system described in the Description of Work (DoW) (see DoW section B.1.1.7 - Testing and refining the solution).

We envisage an implementation in three steps: *centralized* system, *distributed overlay* architecture, *distributed network layer* architecture.

5.1 Centralized system

This case corresponds to Track 1 of DoW section B.1.1.7 (Single centralized system): users and applications will access CONVERGENCE via **a single centralized system** - all VDIs will be stored and handled in a single system that will provide CoMid and CoNet APIs.

5.2 Distributed Overlay architecture

This case corresponds to Track 2 Phase 1 of DoW section B.1.1.7.

The overlay-architecture is depicted in Figure 17. In this architecture CoMid is a distributed entity, based on an overlay over IP. The middleware uses CoNet services through an API. The same occurs for the CDS.

The CoNet is used to store/deliver/retrieve VDIs and any other type of named-resources. The CoNet is therefore a general-purpose network, which is not restricted to VDIs and the CONVERGENCE system.

At this stage in the implementation, the CoNet will be realized as an overlay over IP. In this case CONVERGENCE will have two overlays: the CoMid overlay, which will deal with VDI metadata and semantic content, and the CoNet overlay, which will deal with storing/retrieving/distributing generic resources, including VDIs.

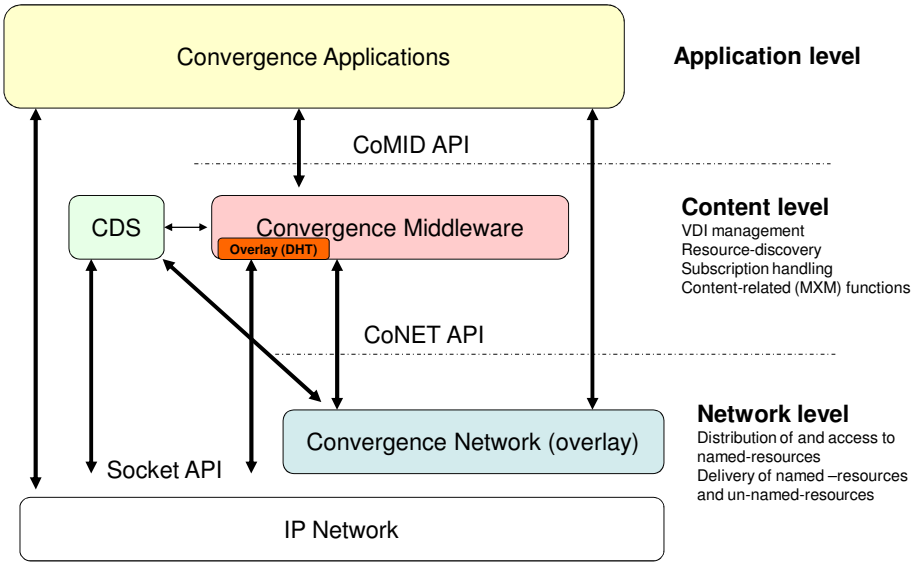

Figure 17: Overlay architecture

5.3 Distributed network layer architecture

This case corresponds to Track 2 Phase 2 of DoW section B.1.1.7.

We can think of further subdividing this case in two steps. The first one is that envisaged in the DoW and reported in the following subsection (Parallel Architecture). The second step was not envisaged in the DoW; it is reported in 5.3.2 (Clean Slate Architecture) and its effective implementation by the project is still to be confirmed.

5.3.1 Parallel Architecture

The parallel architecture is shown in Figure 18. In this stage of the implementation, CoNet is realized as a new networking layer that can be used “in parallel” with IP.

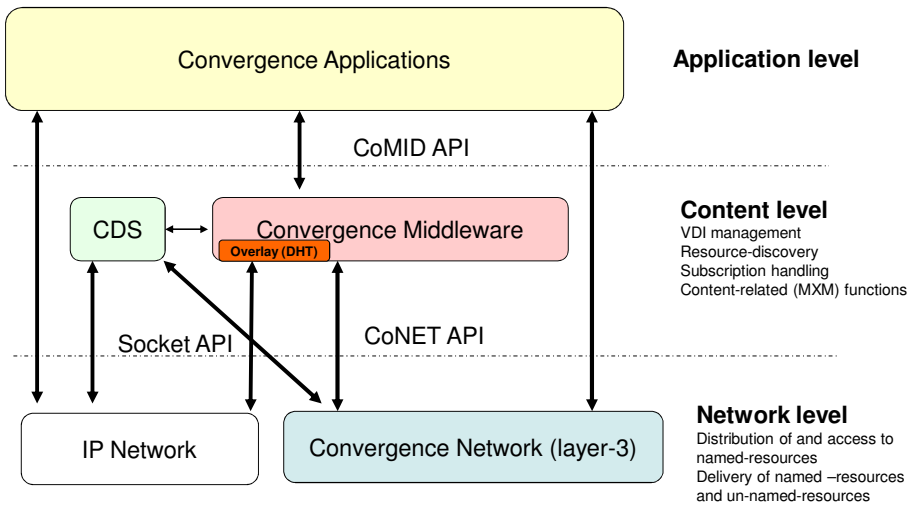

Figure 18: Parallel architecture

5.3.2 Clean-slate architecture

Figure 19 shows the clean-slate architecture. In this case IP is no longer present and networking functionality is provided by CoNet.

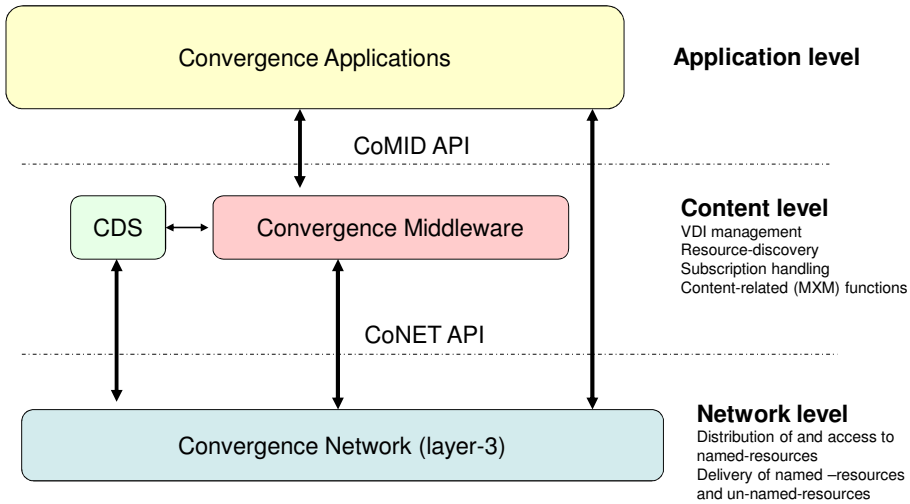

Figure 19: Clean-slate architecture

6 Walkthroughs

This section presents use cases for the CONVERGENCE system and describes how the CoMid and the CoNet interact with each other and with the application layer to offer complex services.

6.1 Walkthrough #1

With a number of colleagues Bob has written a paper about publish/subscribe. Once he has completed the paper, he stores it locally and creates a VDI for the paper. The VDI is "published" on the CONVERGENCE system. The middleware offers a *publishVDI* primitive.

Alice is looking for Bob's publish/subscribe paper. A CONVERGENCE application lets Alice express her request. The CONVERGENCE application uses the *subscribeToVDIs* primitive in the CONVERGENCE middleware. The CONVERGENCE middleware returns the matching VDI. Then the CONVERGENCE application retrieves the paper. In this example, there is no need to use a fully-fledged publish/subscribe approach, either at the CoMid or at the CoNet layer.

In this walkthrough we assume that Bob has an application (which we call Publish Content APP) that is responsible for creating new VDIs. In the same way, we assume that an Alice's system contains an application (Search VDI APP), which enables her to search and retrieve desired VDIs and resources. The description of the interfaces between these applications and the users lies outside the scope of this document.

We assume that the Publish Content APP can assist Bob in generating a unique NID for the resource that he wants to publish (the paper). The NID for the corresponding VDI is generated by the *map VDI ID - NID* elementary service provided by the CoMid.

By	To	Action
Bob	Publish Content APP	Delivers his paper on publish/subscribe, requests that it should be stored on the network and creates a VDI for this purpose
Publish Content APP	CoNet	Requests the storage of Bob's paper by using the `Advertise` primitive. The data will be assigned a NID.
Publish Content APP		Creates a VDI containing: 1. a link to the paper; 2. a set of metadata according to a specific metadata scheme; 3. a licence possibly containing keys to decrypt the resource; 4. an event report request; 5. any other data element useful for distributing the paper; 6. a URI identifier compliant with the MPEG-21 DI standard

		for the resource, the VDI and possibly other data sets in the VDI.
	CoMid	Asks to publish the VDI through the `PublishVDI` primitive
CoMid		Extracts the metadata from the VDI and runs the map *VDI ID – NID* elementary service to generate the NID for the VDI.
CoMid		Stores the metadata in a distributed database, by associating it with the NID
		Uses the `searchVDIs` primitive to identify VDI subscriptions matching the metadata for the new VDI
		Doesn't find any matching subscription.
	CoNet	Requests the local storage of the VDI by using the `Advertise` primitive
Alice	Search VDI APP	Searches for Bob's paper on publish/subscribe
Search VDI APP	CoMid	Asks for VDIs containing Bob's paper by using the *subscribeTo*`VDIs` primitive
CoMid		Finds the reference to Bob's paper with the associated NID in its database
	CoNet	Asks to obtain the VDI by using the `Get` primitive
CoNet	CoMid	Returns the requested VDI
CoMid	Search VDI APP	Returns the VDI related to Bob's paper
Search VDI APP		Extracts the NID identifier of the paper from the VDI
	CoNet	Asks to retrieve Bob's paper with the `Get` primitive
CoNet	Alice	Returns Bob's paper

6.2 Walkthrough #2

Alice wants to be notified when VDIs related to the forthcoming U2 concert in Rome will be published. She is interested in any concert in the next 6 months. The query will be handled using the CoMid's semantic capabilities. On her subscription, Alice will receive a list of current VDIs matching the query.

The day before the concert, Bob, another U2 fan, takes a video of Bono Vox eating in a fancy restaurant in the centre of Rome. He creates a VDI containing the video and publishes it.

According to the semantic layer of CoMid, this publication matches Alice's subscription. Therefore, Alice's application will be notified of the new VDI referring to the video.

We assume that Alice has a CONVERGENCE application that allows her to generate VDIs. We call this application, Subscribe VDI APP. A detailed description of the application lies outside the scope of this document.

Analogously, we assume that Bob can use a CONVERGENCE application that allows him to publish new content. As in the previous case, we call this application the Publish Content APP.

In this walkthrough, we also assume that the Publish Content APP can assist Bob in generating a unique NID for the resource he wants to publish. The NID for the corresponding VDI is generated by the *map VDI ID - NID* elementary service provided by the CoMid.

In the following walkthrough, we will skip the steps carried out by Bob for the publication of the video. These are exactly the same as in the Walkthrough 1.

By	To	Action
Alice		Provides parameters that describe the type of VDIs she wishes to receive.
	Subscribe VDI APP	The Subscribe VDI APP collects the information needed by the CoMid to generate a VDI.
Subscribe VDI APP	CoMid	Uses the `PublishVDI` primitive provided by the CoMid to create the VDI.
CoMid		Prepares a message for the NID Generation Server with a request for a valid NID that will be assigned to the VDI.
	CoNet	The above request is sent to the NID Generation Server using the `Send2Name` primitive offered by the CoNet.
NID Generation Server	CoNet	Returns the NID corresponding to the VDI to the CoMid, using the `Send2Location` primitive.
CoNet	CoMid	Returns the NID to Alice using an `Indication` primitive.
CoMid		Stores the metadata in a distributed database, associating it with the NID. In this case the middleware will store the fact that the VDI will elapse in 6 months. The CoMid will use the `SearchVDIs` primitive to identify VDIs matching the criteria established by Alice and will use an appropriate primitive to provide the list of these VDIs to the Subcribe VDI APP.

	CoNet	The VDI is then published in the network, using the *Advertise* primitive offered by the CoNet. In this way CoNet will store the VDI for a certain time.
…	…	...
Bob creates a new VDI with Bono's video and publishes it. This is achieved in the same way as in Walkthrough 1 (see the previous section).		
…	…	...
CoMid		Uses the *searchVDIs* primitive to identify VDIs matching the metadata characterizing the new VDI. In this case, CoMid will identify the VDI issued by Alice and generate, a message containing the VDI created by Bob.
	CoNet	CoMid will send the message to the Alice's system exploiting the *Send2Name* primitive offered by CoNet.
CoNet	CoMid	The message will be passed to the CoMid through an *Indication* primitive.
CoMid	Subscribe VDI APP	CoMid will notify the subscribe VDI APP about the VDI using the *NotifyUser* primitive.
CoMid		Periodically, the CoMid will check whether there are obsolete VDIs on the network.
	CoNet	In this case it will call the CoNet *Revoke* primitive that contains the NID of the VDIs to be revoked. Thus, in this example, VDI generated by Alice will be revoked after 6 months.

6.3 Walkthrough #3

Alice is on a three weeks vacation in Tokyo and her BlackBerry mobile phone's loudspeaker has just broken. Fortunately, she has previously published a VDI for her BlackBerry product, and also has a VDI of the warranty. Using the warranty VDI and a Catalogue Service application, she searches for warranty service providers in Tokyo but discovers that her warranty is not valid in Japan.

Therefore, she uses the product VDI to search in the Catalogue Service application for companies located in Tokyo that can offer repair services for her product. Ideally, the service can also provide a replacement for her mobile phone while they are working on it. So she narrows the search to a few candidate providers choosing the one nearest to her hotel.

By	To	Action
Alice	Catalogue Service application	Browses through the services ontology for mobile phone service providers, using the CONVERGENCE Core Ontology. Enters filters for location (Tokyo) and service (Warranty) for her BlackBerry warranty VDI using the available Domain Ontologies and searches for VDI's of service providers.
	CDS API	Semantic search
	CoMid API	SearchVDI
CoMid API	Catalogue Service Application	Returns "no VDIs matching the searching criteria"
Alice	Catalogue Service application	Browses up through the ontology leaving only "repairing" service and uses the details of the VDI of her BlackBerry product as search criteria. Filters the results using "replacement" service and "response time, 0-48 hours" as additional filters
	CDS API	ExpandQuery
	CoMid API	SearchVDI
CoMid API	Catalogue Service application	Returns a list of VDIs from matched providers.
Alice	Catalogue Service application / Location application	Alice imports the list of the VDIs into her Location Application and uses the VDI for her hotel to determine the nearest provider to the hotel, and to retrieve the instructions to get there.
Location application	Alice	Instructions how to get to the service provider.

7 Bibliography

[1] *Epidemic-style Management of Semantic Overlays for Content-Based Searching.* **Steen, Spyros Voulgaris and Maarten Van Steen** s.l. : Springer, 2005. pp. 1143-1152.

[2] *Efficient content location using interest-based locality in peer-to-peer systems.* **Sripanidkulchai, K., Maggs, B., Zhang, H.** s.l. : IEEE, 2003. INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies. Vol. 3, pp. 2166- 2176.

[3] *Probabilistic reliable dissemination in large-scale systems.* **Kermarrec, A.-M. and Massoulie, L. and Ganesh, A.J.** s.l. : IEEE, mar. 2003, Parallel and Distributed Systems, IEEE Transactions on, Vol. 14, pp. 248 - 258. 1045-9219.

[4] *Gossiping in distributed systems.* **Anne-Marie Kermarrec, Maarten van Steen.** New York, NY, USA : ACM, 2007, Vol. 41. 0163-5980.

[5] *The Many Faces of Publish-Subscribe*, **P.T. Eugster, P.A. Felber, R. Guerraoui, A. Kermarrec**, in ACM Computing Surveys (CSUR), Volume 35, Issue 2, 2003

[6] *Network layer solutions for a content-centric Internet*. **A. Detti, N. Blefari-Melazzi** in "Trustworthy Internet", Springer 2010.

[7] *A data-oriented (and beyond) network architecture*. **T. Koponen, M. Chawla, B.G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica** in procs. of ACM SIGCOMM'07, 2007

[8] *Networking named content*. **V. Jacobson, D. K. Smetters, et al.**, Fifth ACM International Conference on emerging Networking EXperiments and Technologies (CoNEXT), 2009

[9] *On the Cache-and-Forward Network Architecture* **L. Dong, H. Liu, Y. Zhang, S. Paul, D. Raychudhuri** in procs. of IEEE International Conference on Communications 2009, ICC 2009

[10] *An Introduction to Petname Systems*, **Mark Steigler**, available at http://www.skyhunter.com/marcs/petnames/IntroPetNames.html

[11] *LIPSIN: Line Speed Publish/Subscribe Inter-Networking*, **P. Jokela, A. Zahemszky, C. Esteve, S. Arianfar, and P. Nikander**, in proc. of ACM SIGCOMM'09, 2009

[12] *ROFL: Routing on flat labels*. **M. Caesar, T. Condie, J. Kannan, K. Lakshminarayanan, I. Stoica, and S. Shenker** in procs. of ACM SIGCOMM'06, 2006

[13] *A Semantic Framework for Privacy-Aware Access Control.* **G. V. Lioudakis, E. A. Koutsoloukas, N. Dellas, G. M. Kapitsaki, D. I. Kaklamani, I. S. Venieris** in procs. of the 3rd International Workshop on Secure Information Systems (SIS'08), Wisla, Poland, 20 – 22 October 2008.

[14] *Time Ontology in OWL.* **The World Wide Web Consortium (W3C)**. http://www.w3.org/TR/owl-time/

[15] *Geonames Ontology*. http://www.geonames.org/ontology/

[16] *Regulation 2195/2002/EC of the European Parliament and of the Council on the Common Procurement Vocabulary (CPV).* **European Parliament and Council,** official Journal of the European Communities, No. L 340, pp. 1-562, December 2002.