International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:1, No:10, 2007

Enhancing the Peer-To-Peer Architecture with a Roaming Service and OWL

Younes Djaghloul, and Zizette Boufaida

***Abstract*—**This paper addresses the problem of building a unified structure to describe a peer-to-peer system. Our approach uses the well-known notations in the P2P area, and provides a global architecture that puts a separation between the platform specific characteristics and the logical ones. In order to enable the navigation of the peer across platforms, a roaming layer is added. The latter provides a capability to define a unique identification of peer and assures the mapping between this identification and those used in each platform. The mapping task is assured by special wrapper. In addition, ontology is proposed to give a clear presentation of the structure of the P2P system without interesting in the content and the resource managed by the peer. The ontology is created according to the web semantic paradigm and using OWL language; so, the structure of the system is considered as a web resource.

***Keywords*—**Peer to peer, ontology, owl.

I. Introduction

PEER to peer is becoming one of the most important areas in computer science, especially in the Internet topic. This is due to the nature of P2P model that have several advantages as self-organization, load balancing, adaptation, and fault tolerance. The P2P system is characterized by the autonomy of each peer and a high degree of decentralization. Each peer acts as client and as server in the same time, so it demands and provides services without using the same typology of a client/Server model.

In P2P systems, peers (nodes) are working to achieve specific needs. The needs can be classified into several areas as: file sharing (music or other), distributed computing, distributed storage, communication…

In practice, one can classify P2P systems in two main architectures: unstructured and structured approaches. Unstructured P2P approach are loosely controlled, there is not special peer to control the execution of query or to maintain a global repository over others peers. This permits to have a high dynamic behavior of the system. So, if a peer joins or leaves the overly network, there are no added tasks to do.

All peers have equal role. Gnutella [3], KaZaa [7] and Freenet [5] belong to this type.

Y. Djaghloul, is a student in doctoral degree at the university Mentouri of Constantine; Algeria (e-mail : djaghloul2008@yahoo.fr). (Laboratoire Lire, Equipe : Systèmes d'information et base de connaissances, Université Mentouri, Constantineà, phone & fax: 00213 31 63 90 10 / 31 61 43 46).

Z. Boufaida, Professor in the university Mentouri of Constantine and member of the LIRE laboratory (e-mail: boufriche@hotmail.com).

In the case of the structured P2P system, the Distributed Hash Tables play a key role to determine exactly the location of the peer and its resources. Many systems are proposed as: Chord [4], CAN [13]. In these systems, a hash function is used to obtain a valid key.

On the other hand, the semantic web is considered as the new vision to the web that try to give semantic to the web resources. Since it is based on a logical foundation, the web semantic and its language OWL provide a good solution to semantic problems and ontology presentation.

One remarks that several P2P systems are proposed; each system is based on a specific platform and appropriate technique. In these systems, a peer is presented with a specific structure and it is influenced by the technical aspect of the node. In our work, we aim to provide a new capability to peer that is the roaming service. As in mobile telephony, the peer can navigate across P2P systems and it maintains a unique profile to enhance the result's quality. The unified profile is assured by our proposed ontology. Special wrappers permit the mapping task between the global profile and the specific ones. In addition, the use of OWL language to describe the structure of the P2P system permits to export the P2P system itself as web semantic resource. So, the structure can be used by others with strength semantic.

The remaining of this paper is organized as follow: in section 2, we give an overview of the P2P technology and the web semantic paradigm. In section 3, we present some important related work in the domain of web semantic and P2P. The fourth section introduces our approach that is base on a new roaming layer and presents the proposed ontology to manage the structure of the system. The paper is ended by a conclusion.

II. Peer-To-Peer System and Web Semantic

The peer to peer paradigm allows giving the capabilities of client server paradigm to each node of the network

In this section, one will give an overview of the Peer-To-Peer paradigm and the web semantic technology.

A. Peer-To-Peer Characteristics

The P2P is characterized by:

Autonomy of peers: Nodes participating in a P2P system operate autonomic entities. Each peer can decide to leave the system, forward or performs a query.

Ad hoc nature: There is no control of any node. This

characteristic may poses a big problem in the organization of the whole system. So, a mechanism is incorporated to ensure a self-organization.

Fault tolerance: In P2P, the system do not crashes if a number of peers cannot provide services (ex. leave the system).

B. Peer-To-Peer Substrates

As shown in Fig 1, the substrate is a key component in any peer architecture. It provides a mechanism for managing peers and resources. For peers, it permits joining and leaving services. For resources, it permits the storing and the locating of every resource in the node.

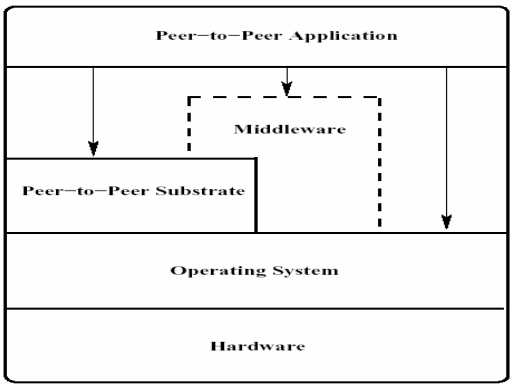

Fig. 1 A simplified model of peer application

P2P systems are divides according to their P2P substrate mechanisms. Therefore, one can divide the P2P substrates into two mains categories: unstructured P2P and structured ones.

Unstructured P2P substrates are loosly-controled. There is not a special peer to organize or to maintain the system; the system is organized as follow.

Two peers that maintain a connection are called *neighbors.* A peer can have more than one neighbor; the number of these neighbors is called *outdegrees.* In an unstructured P2P system, messages (queries or results) are only routed along an open connection. Therefore, if there is not an open connection between two peers. The message must use a path and pass along other peers that are considered as bridge. The length of path is called *hop.*

At the peer level, if a user sends a query, the peer becomes a *source* that sends the query to all its neighbors. Other techniques are proposed to reduce the number of the sent message. They consist to send to a set of neighbors rather than to all ones. Each message is accompanied with a *time to leave* (TTL) values that specify the number of hop. This number decrease on each peer and the sending ends when the TTL =0.

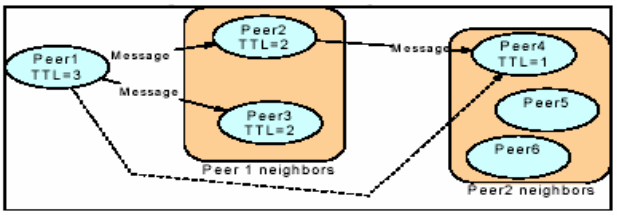

Fig. 2 Message sending in unstructured P2P system

In Fig 2, Peer 1 has two neighbor peers (Peer2, peer3). Peer1 send a message to its neighbors, the latter decrease the TTL value, peer2 do the same task. See Fig 2.

In the follow, we present some important systems in the unstructured peer category.

Gnutella[3] is considered as one of the most widely deployed file sharing system. The Gnutella peer is called *servant* (server/client). As presented before, the peer uses flooding technique to propagate a message; the flooding is controlled by a TTL value whose default value is 7.

Gnutella system provides a high degree of reliability. However, since the peers have equal responsibility, there is no distinguishing between the peers according to their bandwidth or performance. In addition, the use of flooding technique with all the neighbors consumes the bandwidth without guaranty the quality of results.

In order to solve this problem, solutions are proposed. In [2], algorithms to reduce the bandwidth consummation by sending messages to a set of neighbors are proposed. An other solution consists of the introduction of the Super-Peer approach [1].

In a *super-peer* system [1], the network is divided into groups, each one contains one special peer called a supper-peer. All the peers in the group are connected to the supper-peer and the latter is connected to other supper peers of other groups. In the terminology of the supper-peer approach, the group (supper-peer and its peers) is called *cluster*. The main problem in the supper peer approach is the risk for the dysfunction of the supper peer. If the latter is disconnected, the entire cluster is disconnected. In order to resolve the problem a supper-peer redundancy is proposed. Fig 3 shows two clusters, the first with one supper peer, and the second with two supper-peers. The connection between the two clusters is assured by the suppers peers.

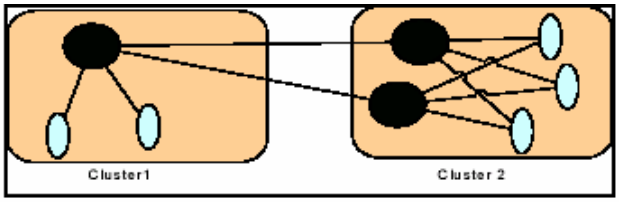

Fig. 3 A supper-peer approach

Structured P2P substrates. In this category, the placement of the resource and the peer is tightly associated to the

network structure. The substrate component uses a Distributed Hash Table (DHT) to locate the peers and the resources. The DHT technique guarantees the locating the file or the peer if they are connected.

DHTs organize the system in different manners. One can uses different geometrical shapes as tree, ring, hypercub. The DHTs are strong technique to lookup for peers resources. However, it suffers of some problems like a) DHTs support an exact match query, so, a little change in one characteristics of the resource causes the fail of the lookup. b) The peers can join or leave much time in a few times, this needs more operations for the management of the DHT.

Many works belong to this category; in (Content Addressable network) CAN the network is considered as virtual d-dimensional Cartesian coordinate. CAN uses dynamic partitions, each ones contain a set of peers. Chord associates to nodes an m-bit identifier using a special hash function as SHA-1. The identifiers values are ordered in a circle manner. Each node maintains a local routed table called the finger table. The system uses the IP addresses and port number to determine the physical location of the node.

C. Semantic Web

The semantic web is a new technology that intends to make web resources more readily from users and machines. The semantic markup extensions permit to obtain this goal. These markup extensions are considered as meta-data annotations to describe their contents.

In the area of the web semantic, ontologies play a key role to describe information sources and permit an efficient share of meta-data. This need prompts the development of several languages as OIL, DAML+OIL, and OWL [11]. The OWL language uses the capabilities of RDFs language as: declaration of classes, organization of classes in subsumption hierarchy. Moreover, OWL gives other extensions for RDFs as: putting that a property is transitive, symmetric, or it is an inverse of another property.

In our work, we aim to use the web semantic to export the P2P system as a web resource.

III. Related Work

Various research projects address the use of semantic in P2P system. Edutela [12] uses the sun JXTA platform to exchange learning resources. P-Grid [6] provides a strong self-organization service in a high-decentralized system. It is based on a virtual distributed search tree. SWAP [9] combines P2P and web semantic. RDFPeer [8] builds the Multi-Attribute Addressable Network (MAAN) that extends Chord.

When analyzing the P2P systems, one remarks that several architecture and technique are proposed to assure the lookup service and the organization task. For the lookup service, techniques as DHT (in structured P2P) provide identifications for the peer and the resources. However, this identification differs according to the used system; one may have the same peer with the same resources but the identification changes.

So, A peer p1 can be identified in P-Grid with ID = "pgrids://01001101" and the same peer can be reference in JXTA with ID="jxta:uuid-150325033CD144F82DED74E". Furthermore, a peer must belong to a unique P2P system in order to be referenced. So the peer cannot navigate between P2P systems because of the characteristics of each system. In addition, each system uses a specific platform and peers are relied to the specific characteristic of its platform.

In order to enhance the capability of a P2P system, we propose a new P2P design that has as objectives:

-To assure a clear separation between the physical and technical aspect of a peer and the logical ones.

-To propose a unique identification of peers unless they use different platforms.

-To permit to the peer to navigate between P2P systems and in the same time to assure a transparency to the user.

-To export the peer structure as web semantic resource by proposing ontology to unify the notation.

IV. Overview of the Proposed Approach

Our approach is based on platform independent identification. A peer has a unique identification that can be mapped into specific one, according to the platform or the system used. In order to provide a unique view of the peer a clear description of the peer structure is needed. The description must be independent from the technical aspect of the used platform. In this case; the ontology and the OWL language provide a good solution to express the semantic of the structure. The OWL language proposes some semantic relations that are needed in our work as Restriction; transitive; symmetric, transitive…

This ontology provides two mains services. One of the most important benefic of this ontology is the roaming service.

A. Peer Roaming Service

Peer roaming is a new service that consists of a few services provided for a peer in order to permit:

-An easy navigation across a P2P system like in the mobile telephony. Therefore, the peer can participate to several P2P systems.

-To export the meta-data to several systems at the same time.

The roaming service is used within other components (cf fig 1). Fig. 4, shows the global architecture of our approach.

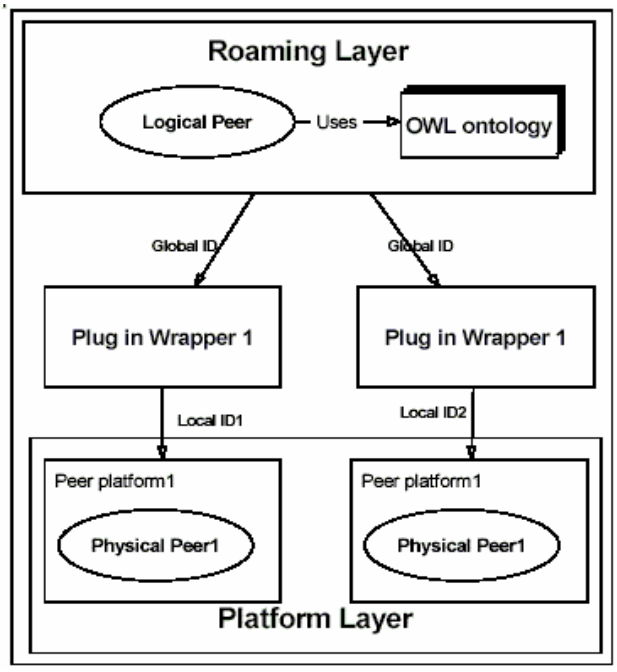

Fig. 4 An overview of the global architecture

The peer's structure in the platform layer is called "physical". However it is called "logical" in the roaming one.

-The roaming layer provides the following services:

-The interception of the *JoinQuery*. This query is sent by other peers to join each platform.

-The interception of the *AddRessource* query. The latter is done by the user in order to publish some resources to share.

-Establishing a connection between the logical peer and the physical one. This is done buy mapping the ID from global to local.

-Storing the profile of the peer as OWL document. This permits to use it with other platform or with the same platform in other time.

For each platform is one associated a special wrapper. The latter uses a correspondence table to map from the logical to the physical representation of the peer.

The platform layer contains the specific platform, techniques to manage the system. The characteristics of each platform are conserved so long as possible.

In the next section, we present the proposed ontology to manage the system.

B. The Proposed Ontology

In order to build ontology, one must use an appropriate methodology or processes. In our case, we use an Ontology Development Process for the Semantic Web [10]. In this process, we follow five phases that are needs specification, conceptualization, formalization, codification and finally verification.

In the following, we directly present the terms dictionary and the codification in OWL language.

TABLE I
THE CONCEPT DICTIONARY

Concept Name	Attributes
PeerCommunity	-PeerComID -PeerComdesignation -Description
PeerCluster	-ClusterID -NumberOfPeers -NumberOfRedundancy -ClusterBandwith
PeerProfile	-OnlineState -PlatformInUse -PeerQuality -IsSupperPeer
PeerPlatformProfile	-PeerUpBandwith -PeerDawnBandwith -PeerPlatformID -PeerTypeMachine -PlatformID
PeerRessource	-PeerRessourceID -PeerRessourceType -PeerRessourceURI
Peer	-PeerID
PeerQuery	-QueryID -PeerQueryContent -QueryType

On each attribute, we define a restriction on its type or the values that can contain. The restrictions are part of the OWL language and are used according to its syntax.

Ex: the concept "PeerCluster" contains an attribute "Number of redundancy". The value of the latter is between 0 and 3. So, by using

<owl:**Restriction**> <owl:maxCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int" >**3**</owl:maxCardinality> <owl:onProperty> <owl:DatatypeProperty rdf:ID="NumberOfRedundancy"/></owl:onProperty> </owl:**Restriction**>an other example of a restriction is the fact that we indicate that some values are allowed for an attribute. The concept "PeerPlatformProfile" contains an attribute "PeerTypeMachine". We determine that a set of values are allowed (PC; PocketPc…). With OWL; we use the "<owl:one of>.

We according to [10], a binary relation table that contains the relation between the concepts must be created. It permits to indicate the source cardinalities (SC), the target cardinalities (TC), the name of the relations and the inverse ones. In addition, we create an attributes table to describe each attribute, logical axioms table and instances table.

TABLE II
RELATION TABLE

Retaliation Name	Concept Source	SC	Target concept	TC	Inverse relation
BelongsToCluster	Peer	0,1	PeerCluster	0..N	ContainsPeer
HasNieghbor	PeerCluster	0..N	PeerCluster	0..N	HasNeighbor
Describes	PeerProfile	1,1	Peer	1,1	HasPeerProfile
SendsTo	PeerQuery	1..N	Peer	1..N	SendsFrom
ExportsRessources	PeerProfile	1..1	PeerRessource	1..N	ExportedByPeer

The OWL language provides many possibilities to enhance the semantic of the relations.

- To indicate that the relation "ContainsPeer" is the inverse of "BelongsToCluster" we use "<owl:inverseOf rdf:resource="#BelongsToCluster"/>".
- To indicate the relation "HasNieghbor" is symmetric (If P1 "Hasneighbor" P2 then P2 "Hasneighbour" P1), we use <owl:ObjectProperty rdf:ID="HasNeighbor"><rdf:type rdf:resource="http://www.w3.org/2002/07/owl#**SymmetricProperty**"/><rdfs:range rdf:resource="#PeerCluster"/> <rdfs:domain rdf:resource="#PeerCluster"/>
- To indicate that the "HaNighbor" is transitive (If P1 "Hasneighbor" P2 and P2 "Hasneighbor" P3 then P1 "Hasneighbor" P3") we use <owl:ObjectProperty rdf:ID="HasNeighbor"> <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#**Transitive Property**"/><rdfs:range rdf:resource="#PeerCluster"/><rdfs:domain rdf:resource="#PeerCluster"/><owl:inverseOf rdf:resource="#HasNeighbor"/></owl:ObjectProperty>

In the following, one directly presents the ontology formalism according to SHIQ DL,

PeerCommunity:= (∃ PeerComID.String)∩(∃ PeerComdesignation.String)∩(∃ Description.String)∩(≤ 1 ContainsCluster PeerCluster)

PeerCluster:= (∃ClusterID.String) ∩ (∃ NumberOfPeers.Integer) ∩(∃NumberOfRedundancy.Integer) ∩ (∃ClusterBandwith.Double)∩(∃ BelongsToCommunity PeerCommunity) ∩(≤ 1 ContainsPeer Peers) ∩(<1 HasNeighbor PeerCluster)

Peer: = (∃ PeerID.String)∩(∃ HasPeerProfile PeerProfile)∩(≤ 1 HasPeerPlatformProfile PeerPlatformProfile)(∃ SendsQuery PeerQuery)

PeerProfile:=(∃ OnlineState.Boolean)∩(∃ PlatformInUse.String)∩(∃ PeerQuality.Integer)∩(∃ IsSupperPeer.Boolean)∩(∃ Describes Peer) ∩(≤ 1 ExportsRessource PeerRessource)

PeerQuery:= (∃ QueryID.String)∩(∃ PeerQueryContent.String)∩(∃ SendedBy Peer) ∩(≤ 1 SendsTo Peers)∩(∃ SendedFrom Peer)

…

In the following, we present a part of the ontology coded in the OWL language.

```
<?xml version="1.0"?><rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns="http://www.owl-ontologies.com/unnamed.owl#"
xml:base="http://www.owl-ontologies.com/unnamed.owl">
<owl:Ontology rdf:about="PeerOntology"/> <owl:Class
rdf:ID="PeerPlatformProfile"/> <owl:Class
rdf:ID="PeerCommunity"> <rdfs:subClassOf
rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
<rdfs:subClassOf> <owl:Restriction> <owl:onProperty>
<owl:InverseFunctionalProperty rdf:ID="ContainsPeer"/>
</owl:onProperty> <owl:minCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
>1</owl:minCardinality> </owl:Restriction>
</rdfs:subClassOf> </owl:Class> <owl:Class rdf:ID="Peer">
<rdfs:subClassOf
rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
<rdfs:subClassOf> <owl:Restriction> <owl:minCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
>1</owl:minCardinality> <owl:onProperty>
<owl:ObjectProperty rdf:ID="HasPeerPlatformProfile"/>
</owl:onProperty> </owl:Restriction> </rdfs:subClassOf>
<rdfs:subClassOf> <owl:Restriction> <owl:onProperty>
<owl:ObjectProperty rdf:ID="BelongsToCluster"/>
</owl:onProperty> <owl:maxCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
>1</owl:maxCardinality> </owl:Restriction>
</rdfs:subClassOf> </owl:Class> <owl:Class
rdf:ID="PeerCluster"> <rdfs:subClassOf> <owl:Restriction>
<owl:maxCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
>3</owl:maxCardinality> <owl:onProperty>
<owl:DatatypeProperty rdf:ID="NumberOfRedundancy"/>
</owl:onProperty> </owl:Restriction> </rdfs:subClassOf>
<rdfs:subClassOf
rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
<rdfs:subClassOf> <owl:Restriction> <owl:someValuesFrom
rdf:resource="#Peer"/> <owl:onProperty>
<owl:ObjectProperty rdf:ID="SupperPeerListe"/>
</owl:onProperty> </owl:Restriction> </rdfs:subClassOf>
</owl:Class>
…
…
<owl:ObjectProperty rdf:ID="BelongsToCommunity">
<owl:inverseOf> <owl:ObjectProperty
rdf:ID="ContainsCluster"/> </owl:inverseOf>
<rdfs:subPropertyOf> <owl:ObjectProperty rdf:ID="Bellongs"/>
</rdfs:subPropertyOf> <rdfs:domain
rdf:resource="#PeerCluster"/> <rdfs:range
rdf:resource="#PeerCommunity"/> </owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="SendedFrom"> <rdfs:range
rdf:resource="#Peer"/> <rdfs:domain rdf:resource="#PeerQuery"/>
```

```
<rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/
> </owl:ObjectProperty> <owl:ObjectProperty
rdf:about="#ExportedByPeer"> <rdfs:range
rdf:resource="#PeerProfile"/> <rdfs:domain
rdf:resource="#PeerRessource"/> <owl:inverseOf
rdf:resource="#ExportsRessources"/> </owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#ContainsCluster">
<rdfs:subPropertyOf> <owl:ObjectProperty rdf:ID="Contains"/>
</rdfs:subPropertyOf> <owl:inverseOf
rdf:resource="#BelongsToCommunity"/> <rdfs:range
rdf:resource="#PeerCluster"/> <rdfs:domain
rdf:resource="#PeerCommunity"/> </owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="HasNeighbor"> <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#SymmetricProperty"/
> <rdfs:range rdf:resource="#PeerCluster"/> <rdfs:domain
rdf:resource="#PeerCluster"/> <owl:inverseOf
rdf:resource="#HasNeighbor"/> <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#TransitiveProperty"/
> </owl:ObjectProperty> <owl:ObjectProperty
rdf:about="#BelongsToCluster"> <rdfs:subPropertyOf>
<owl:ObjectProperty rdf:about="#Bellongs"/>
</rdfs:subPropertyOf> <rdfs:range rdf:resource="#PeerCluster"/>
<rdfs:domain rdf:resource="#Peer"/> <owl:inverseOf>
<owl:InverseFunctionalProperty rdf:about="#ContainsPeer"/>
</owl:inverseOf> </owl:ObjectProperty>…
….
```

V. Conclusion

In this work, we presented our work to enhance P2P architecture with a roaming service. The latter allows the peer to navigate across different platform and to maintain a history of the peer. This is assured by the use of a global profile in the roaming layer; this profile is coded with OWL language. The latter gives strong semantic relations as transitive; symmetric, inverse … The structure of the whole system is exported as a web resource. A special wrapper maps the global ID to local one according to the target platform. The main benefic of such otology is to permit to other platform or user to share a global structure and permits to locate a peer independently to a specific platform.

References

[1] B. Yang and H. Garcia-Molina. Designing a super peer network.
[2] B. Yang and H. Garcia-Molina. Improving efficiency of peer-to-peer search. In *Proc. of the 28th Intl. Conf. on Distributed Computing Systems*, July 2002.
[3] Gnutella home page. www.gnutella.com.
[4] I. Stoica, R. Morris, M. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for Internet applications. In *Proc. of ACM SIGCOMM'01*, San Diego, CA, USA, August 2001.
[5] Ian Clarke, Scott G. Miller, Theodore W. Hong, Oskar Sandberg, and Brandon Wiley. Protecting Free Expression Online with Freenet. *IEEE Internet Computing*, 6(1), Jan./Feb. 2002.
[6] K. Aberer, P. Cudr´e-Mauroux, A. Datta, Z. Despotovic, M. Hauswirth, M. Punceva, and R. Schmidt. P-grid: A self-organizing structured p2p system. *ACM SIGMOD Record*, 32(3), 2003.
[7] Kazaa home page. www.kazaa.com.
[8] M. Cai and M. Frank. RDFPeers: A Scalable Distributed RDF Repository based on A Structured Peer-to-Peer Network. In *International World Wide Web Conference (WWW)*, 2004.
[9] M. Ehrig, P. Haase, R. Siebes, S. Staab, H. Stuckenschmidt, R. Studer, and C. Tempich. The SWAP Data and Metadata Model for Semantics-Based Peer-to-Peer Systems. In *Multiagent System Technologies (MATES)*, 2003.
[10] M. Hemam & Z. Boufaida "*An Ontology Development Process for the Semantic Web*". *EKAW'04* Workshop on Knowledge Management and the Semantic Web, Whittlebury Hall, Northamptonshire, UK. 2004
[11] Mike Dean, Dan Connolly, Frank van Harmelen, James Hendler, Ian Horrocks, Deborah L. McGuinness, Peter F. Patel-Schneider, and Lynn Andrea Stein. Web ontology language (OWL) reference version 1.0. W3CWorking Draft, 2003. Available at http://www.w3.org/TR/2003/WD-owl-ref-20030331.
[12] Nejdl, W., et al.: Super-peer-based routing and clustering strategies for rdf-based peer-topeer networks. In: Proceedings of the Twelfth International World Wide Web Conference (WWW 2003), Budapest, Hungary (2003)
[13] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *Proc. of ACM SIGCOMM'01*, San Diego, CA, USA, August 2001.