

---

# Peer to Peer solutions in the Semantic Web context: an overview

---

Ronny Siebes (Vrije Universiteit Amsterdam)

**Executive Summary.**

SWAP EU IST-2001-34103 Project Deliverable D1.1

This Report tries to provide an overview of Peer-to-Peer systems where the focus lies on the ability of the systems to integrate techniques from the Semantic Web community. First the report shows some different models and viewpoints of Peer-to-Peer (P2P) systems. Next it takes a look on the common characteristics when dealing with P2P systems and it also look at the specific aspects of combining P2P with the Semantic Web, like variation in ontologies and ontological drift. After having discussed these points and after have looked at some P2P protocols it finally reaches the point of describing existing implementations of P2P systems. Only those systems are chosen that have the ability (or are easily to extended) of using semantic meta-knowledge on the data (e.g. ontologies) for peer selection and/or routing of messages through the network.

**Document Id.** SWAP/2002/D1.1/v1.0  
**Project** SWAP EU IST-2001-34103  
**Date** October 30<sup>st</sup>, 2002  
**Distribution** Resticted

---

Copyright © 2002, Vrije Universiteit Amsterdam

---

## SWAP Consortium

This document is part of a research project partially funded by the IST Programme of the Commission of the European Communities as project number IST-2001-34103. The partners in this project are: Institute AIFB / University of Karlsruhe (coordinator, Germany), Vrije Universiteit Amsterdam VUA (Netherlands), Meta4 (Spain), empolis UK Ltd. (UK), empolis Polska (Poland), Dresdner Bank AG (Germany), and IBIT (Spain).

### University of Karlsruhe

Institute AIFB  
Englerstr. 28  
D-76128 Karlsruhe  
Germany  
Tel: +49 721 608 3923, Fax: +49 721 608 6580  
Contactperson: Steffen Staab  
E-mail: [staab@aifb.uni-karlsruhe.de](mailto:staab@aifb.uni-karlsruhe.de)

### Meta4 Spain S.A.

Rozabella, 8 Centro Europa Empresarial  
28230 Las Rozas (Madrid)  
Spain  
Tel: +34 91 634 85 00, Fax: +34 91 634 86 86  
Contactperson: Adelma Stolbun  
E-mail: [adelmas@meta4.com](mailto:adelmas@meta4.com)

### Empolis Polska Sp. Z o. o.

Plocka 5a  
01-231 Warsaw  
Poland  
Tel: +48 22 535 88 06, Fax: +48 22 535 88 14  
Contactperson: Mariusz Olko  
E-mail: [mariusz.olko@empolis.pl](mailto:mariusz.olko@empolis.pl)

### Fundación IBIT

Reverend Francesc Sitjar 1  
07010 Palma de Mallorca  
Spain  
Tel: +34 971 177 271, Fax: +34 971 177 279  
Contactperson: Esteve Lladó Martí  
E-mail: [esteve@ibit.org](mailto:esteve@ibit.org)

### Vrije Universiteit Amsterdam (VUA)

Division of Mathematics and Informatics W&I  
De Boelelaan 1081a  
1081 HV Amsterdam  
The Netherlands  
Tel: +31 20 4447786, Fax: +31 20 4447653  
Contactperson: Hans Akkermans  
E-mail: [hansakkermans@cs.vu.nl](mailto:hansakkermans@cs.vu.nl)

### Empolis UK Ltd.

Unit B, The Dorcan Complex, Faraday Road  
SN3 5HQ Swindon  
United Kingdom  
Tel: +44 77 99694621, Fax: +44 17 93485451  
Contactperson: Graham Moore  
E-mail: [gdm@empolis.co.uk](mailto:gdm@empolis.co.uk)

### Dresdner Bank AG

Jürgen-Pronto-Platz 1  
60301 Frankfurt am Main  
Germany  
Tel: +49 69 263 58265, Fax: +49 69 263 81385  
Contactperson: Martin Lorenz  
E-mail: [martin.lorenz@dresdner-bank.com](mailto:martin.lorenz@dresdner-bank.com)

---

---

---

1.1.1.1.1 Changes

<i>Version</i>	<i>Date</i>	<i>Author</i>	<i>Changes</i>
1	10/02	Ronny	

# Contents

<u>2</u>	<u><a href="#">Introduction</a></u> .....	7
<u>3</u>	<u><a href="#">“P2P” models</a></u> .....	8
<u>3.1</u>	<u><a href="#">Broker mediated model</a></u> .....	8
<u>3.2</u>	<u><a href="#">direct P2P model</a></u> .....	9
<u>3.3</u>	<u><a href="#">Resource sharing model</a></u> .....	10
<u>4</u>	<u><a href="#">Different peer metaphors</a></u> .....	10
<u>4.1</u>	<u><a href="#">Remote procedure calls</a></u> .....	11
<u>4.2</u>	<u><a href="#">Peers as web services</a></u> .....	11
<u>4.3</u>	<u><a href="#">Peers as agents</a></u> .....	12
<u>5</u>	<u><a href="#">Overview of main characteristics in the P2P area</a></u> .....	13
<u>5.1</u>	<u><a href="#">Interoperability: protocols vs. APIs</a></u> .....	13
<u>5.2</u>	<u><a href="#">Scalability and network efficiency</a></u> .....	13
<u>5.3</u>	<u><a href="#">Authenticity, confidentiality and security of information</a></u> .....	15
<u>5.4</u>	<u><a href="#">Privacy/anonymity for users</a></u> .....	15
<u>5.5</u>	<u><a href="#">Digital Rights management</a></u> .....	17
<u>5.6</u>	<u><a href="#">popularity (follow the users, what is already out there)</a></u> .....	18
<u>6</u>	<u><a href="#">Specific P2P+SW issues</a></u> .....	18
<u>6.1</u>	<u><a href="#">Peer selection service</a></u> .....	19
<u>6.2</u>	<u><a href="#">Variation of ontologies and lack of ontological precision</a></u> .....	19
<u>6.3</u>	<u><a href="#">Ontological drift</a></u> .....	20
<u>7</u>	<u><a href="#">Four different P2P protocols</a></u> .....	20
<u>7.1</u>	<u><a href="#">JXTA Protocol suite</a></u> .....	22
<u>7.2</u>	<u><a href="#">Fipa</a></u> .....	24
<u>7.3</u>	<u><a href="#">Pastry</a></u> .....	28
<u>8</u>	<u><a href="#">Existing systems</a></u> .....	30
<u>8.1</u>	<u><a href="#">Neurogrid</a></u> .....	30
<u>8.1.1</u>	<u><a href="#">Model</a></u> .....	31
<u>8.1.2</u>	<u><a href="#">Methaphor</a></u> .....	31
<u>8.1.3</u>	<u><a href="#">P2P characteristics</a></u> .....	31
<u>8.1.4</u>	<u><a href="#">P2P+SW characteristics</a></u> .....	33
<u>8.1.5</u>	<u><a href="#">Protocol</a></u> .....	35
<u>8.2</u>	<u><a href="#">JXTA implementation by SUN</a></u> .....	35
<u>8.2.1</u>	<u><a href="#">Model</a></u> .....	36
<u>8.2.2</u>	<u><a href="#">Metaphor</a></u> .....	36
<u>8.2.3</u>	<u><a href="#">P2P characteristics</a></u> .....	36
<u>8.2.4</u>	<u><a href="#">P2P + SW characteristics</a></u> .....	40
<u>8.2.5</u>	<u><a href="#">Protocol</a></u> .....	40
<u>8.3</u>	<u><a href="#">Poblano</a></u> .....	41
<u>8.3.1</u>	<u><a href="#">Model</a></u> .....	42
<u>8.3.2</u>	<u><a href="#">Metaphor</a></u> .....	43
<u>8.3.3</u>	<u><a href="#">P2P characteristics</a></u> .....	43
<u>8.3.4</u>	<u><a href="#">P2P + SW characteristics</a></u> .....	45
<u>8.3.5</u>	<u><a href="#">Protocol</a></u> .....	45
<u>8.4</u>	<u><a href="#">Edutella</a></u> .....	46
<u>8.4.1</u>	<u><a href="#">Model</a></u> .....	47
<u>8.4.2</u>	<u><a href="#">Metaphor</a></u> .....	47
<u>8.4.3</u>	<u><a href="#">P2P characteristics</a></u> .....	47
<u>8.4.4</u>	<u><a href="#">P2P + SW characteristics</a></u> .....	48
<u>8.4.5</u>	<u><a href="#">Protocol</a></u> .....	49

<a href="#">8.5</a>	<a href="#">InfoQuilt/PSW</a> .....	50
<a href="#">8.5.1</a>	<a href="#">Model</a> .....	50
<a href="#">8.5.2</a>	<a href="#">Metaphor</a> .....	51
<a href="#">8.5.3</a>	<a href="#">P2P characteristics</a> .....	51
<a href="#">8.5.4</a>	<a href="#">P2P + SW characteristics</a> .....	53
<a href="#">8.5.5</a>	<a href="#">Protocol</a> .....	54
<a href="#">8.6</a>	<a href="#">Jade</a> .....	55
<a href="#">8.6.1</a>	<a href="#">Model</a> .....	55
<a href="#">8.6.2</a>	<a href="#">Metaphor</a> .....	56
<a href="#">8.6.3</a>	<a href="#">P2P characteristics</a> .....	56
<a href="#">8.6.4</a>	<a href="#">P2P + SW characteristics</a> .....	58
<a href="#">8.6.5</a>	<a href="#">Protocol</a> .....	59
<a href="#">9</a>	<a href="#">Summary</a> .....	60
<a href="#">10</a>	<a href="#">Literature</a> .....	62

## 2 Introduction

In today’s knowledge-based economy, the competitiveness of enterprises and the quality of work life are directly tied to the ability to effectively create and share knowledge both within and across organizations. Emerging peer-to-peer solutions are particularly well suited to the increasingly decentralized nature of today’s organizations, be it a single enterprise or a dynamic network of organizations. They make it possible for different participants (organizations, individuals, or departments within an organization) to maintain different views of the world while exchanging information. They also circumvent the bottlenecks associated with more traditional solutions, which rely on one or a small number of centralized servers. At the same time, because they rely on keyword search and rather simple knowledge representation techniques, today’s peer-to-peer solutions are extremely limited. They cannot easily support the introduction of new concepts, make it difficult to determine whether two terms are equivalent, and generally can only support very limited levels of automation – all types of functionality, which Semantic Web (SW) technologies have been shown to support.

This paper looks at existing peer-to-peer solutions and how suitable they are at including SW technologies. We provide a brief overview on existing issues in the P2P area like security, trust, scaling and network (in)efficiency. Also on new issues specific to the SW in combination with P2P are tackled like dealing with ontological drift, autonomy of peers, peer selection, variation of ontologies and lack of ontological precision. Next, chapter two provides three different “P2P” models namely the broker-mediated model, the direct P2P model and the resource-sharing model. We make this distinction to make a categorization of ‘P2P’ systems where we leave the discussion open if, or if not, systems fit in the P2P paradigm. Chapter four shows three metaphors on peers, namely the agent-, web service- and RPC (remote procedure call) metaphor. We use these metaphors, because these three are also used in the agent, peer-to-peer and web service community. They all share the characteristic of P2P, however each metaphor lightens a peer from a different viewpoint. Chapter three provides an overview of general issues in the P2P area.

Chapter four examines the main topics when integrating SW technologies with P2P solutions. Chapter five shows a distinct collection of P2P protocols. Chapter six examines the four major P2P protocols for routing and finding peers. Finally, chapter seven shows different existing implementations of P2P systems that are interesting for SWAP because they use semantic routing or provide a basis to easily implement protocols for semantic routing. Systematically, for each system the topics of the previous chapters are taken and discussed.

### 3 “P2P” models

It is important to be aware not to give a too tight description of what P2P means. It shouldn't distract us from the success of systems that initialised the phenomenon. Therefore we describe P2P only as a class of applications that takes advantage of resources –storage, cycles, content, human presence- available at the edges of the Internet, nothing more. If we look to the existing systems on the Web, we can roughly divide them in three different models: the broker mediated model, the direct P2P model and the resource-sharing model.

#### 3.1 Broker mediated model

In a **mediated** P2P network (figure 1), central servers contain an index of all the content and where it can be found. When the server receives a request, peers hosting the content are identified from the index. The server acts as a broker and “**mediates**” a direct communications link between the requesting peer and the closest, most efficient host peer. The server instructs the requesting peer where it can obtain the file, but is not otherwise involved in the transfer. This greatly reduces bandwidth and storage costs over a central server architecture.

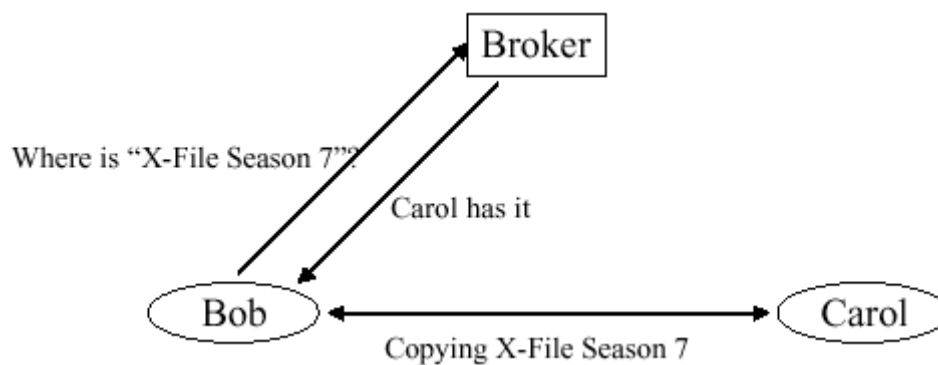


Figure 1: broker mediated model

Napster is a good example of this model. When a user starts a Napster client, it registers itself to the Napster server (the broker) and the filenames from the users' hard disk are stored at the server. When another user queries Napster, the server responds with a list of systems that matches on the query.

### 3.2 *direct P2P model*

This model let users register information with network neighbors (figure 2). Searching across the network to find information is done by sending queries to neighbors, and if the neighbors don't know the answer they send the query to their (or a selection of their) neighbors. Techniques like a history profile of the query could prevent cyclic behavior and restricts the chain length.

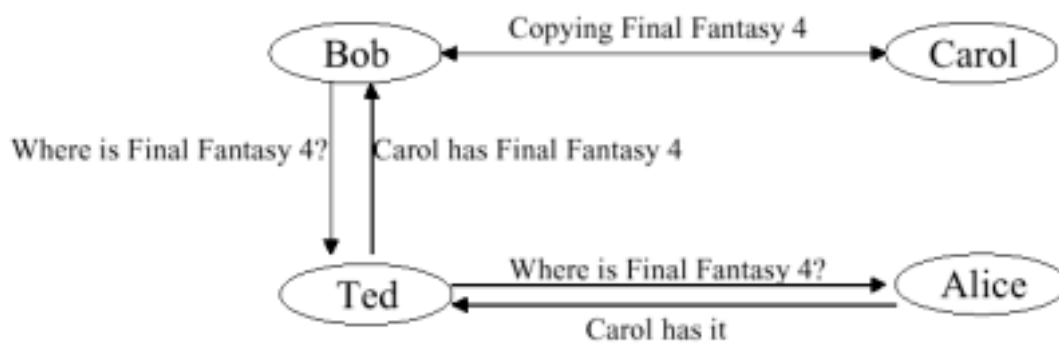


Figure2: direct P2P model

The network can be organized in different ways, for example:

- Random Graph (Gnutella Approach): Queries may have a TTL scope (Time To Live scope that is a maximum number of peer hops), where cycles are prevented by maintaining a history of messages at each peer and dropping repeated messages.
- Power Law Networks (Adamic, 2001): In this approach there are only a few nodes with very high connectivity and many nodes with low connectivity. This idea is implemented by KaZaA [KaZaA].
- Publish/Subscribe Networks
  - Information consumers subscribe their needs with a publish/subscribe server located in a network of P/S servers.
  - P/S servers propagate and aggregate subscriptions to other P/S servers.
  - Information producers publish their advertisements to the P/S servers
  - Event notifications flow from information producers to subscribers through the servers

The big advantage of this approach is the independence of a centralized server that could be a bottleneck in CPU or storage capacity and it also prevents the possibility of censorship.

A disadvantage is the difficulty in finding the peers you need, but as already said, more efficient methods than just broadcasting are developed and discussed in this paper.

### 3.3 Resource sharing model

In this model, a “master” uses “slaves” for any kind of purpose (figure 3). The master could, for example, use the unused CPU cycles from the slave for calculating extraterrestrial data (SETI@home, Popular Power), using disk space (Mojo Nation) or using the data stored on the slave (like Google: it visits sites to get information and include it in its database).

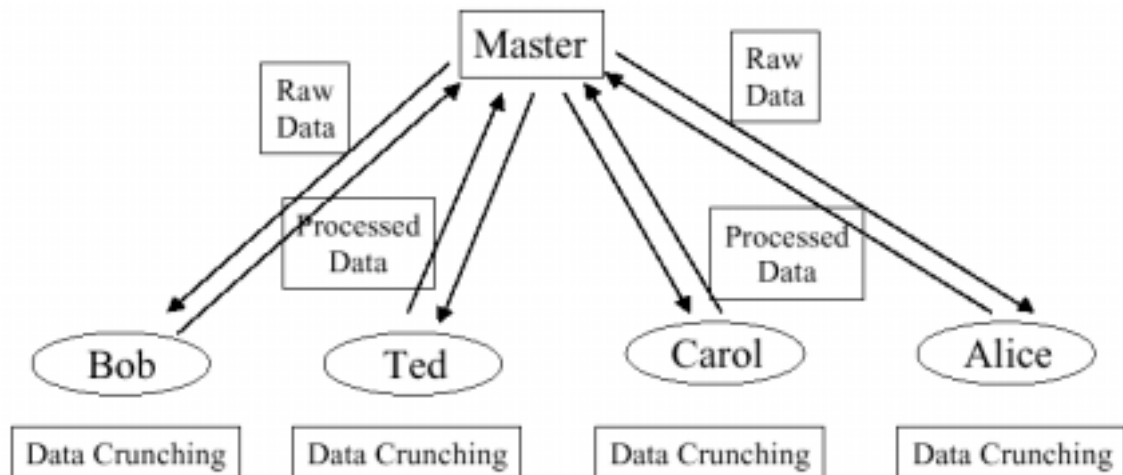


Figure 3: resource sharing model

One could argue that this method isn't peer to peer, because the peers are not connected with each other, but are only connected with one server. We share this opinion therefore we only mention it but don't examine systems in this paper that is implemented in this way. The reason for mentioning it after all is because well-known systems like SETI@home are called P2P but follow this master/client protocol.

## 4 Different peer metaphors

One can roughly have three different viewpoints a P2P system, namely a system based on *remote procedure calls*, as a *web service* or as an *agent*. The viewpoint strongly depends on the community of the viewer and the application of the system itself.

## **4.1 Remote procedure calls**

In this situation, a peer invokes methods at another peer. This can be done directly by a public interface on the peer or through a server that provides the interface to the peer. One can distinct roughly two types of RPC's namely those that are readable by humans and those who are not. Examples of the first are XML-RPC [XML-RPC] and SOAP [SOAP] which implement RPC's in XML<sup>1</sup> and examples of the latter are COM+ /DCOM, CORBA /IIOP and EJB / RMI/IIOP which represents message data in a binary format. Besides the readability also some other problems can be identified with the older protocols like DCOM, IIOP and RMI/IIOP [Sanat]. The chief among them is that these protocols are incompatible. A DCOM based system cannot talk to an EJB based system, for example. If an enterprise has diverse applications on different platforms, these applications cannot be integrated using the older protocols. Another problem is that these protocols are not firewall friendly. Most firewalls are configured to allow access traffic only through specific ports, the most popular being the HTTP port 80. The older protocols use different ports, which are blocked by most corporate firewalls. This means that applications residing in different physical locations cannot talk to each other even if they have been built on the same platform.

## **4.2 Peers as web services**

P2P and Web Services are two streams that still arbitrarily being held apart despite increasingly convergent themes, protocols, technologies, and applications. P2P not only delivers resources at the edge of the Internet but, as a nice side-effect, presence and identity -- two vital ingredients in Web Services. Web Services extend P2P beyond file- and CPU-sharing, instant messaging, and the like to deliver services and resources beyond simple files and computational drones. The above story about remote procedure calls can be applied on how web services can invoke methods on other web services. On top of that, there are already initiatives that add a put a layer above these RPCs. For example, the Web Services Description Language (WSDL) is a specification to describe networked XML-based services. It provides a simple way for service providers to describe the basic format of requests to their systems regardless of the underlying protocol (such as SOAP). WSDL is a key part of the effort of the Universal Description, Discovery and Integration (UDDI) initiative to provide directories and descriptions of such on-line services for electronic business. IBM and Microsoft have managed to work together to establish the basic Web-services standards of SOAP, WSDL and UDDI. But the gloves come off when it comes to selling products based on those standards. The first field of combat is that of development tools, in which Microsoft has recently launched its comprehensive Visual Studio .NET<sup>2</sup> suite. IBM is countering with its WebSphere Studio suite<sup>3</sup> and

---

<sup>1</sup> For a comparison between XML-RPC and SOAP please look at [Rhodes]

<sup>2</sup> <http://msdn.microsoft.com/vstudio/>

SUN with their Sun ONE Studio<sup>4</sup>, which offers one major advantage over Microsoft's tools: they are firmly ensconced in the Java world. When we look at the P2P aspect of web services, IBM, Microsoft and SUN shouldn't become too aggressive in hosting its own subscription-based Web services. Customers should be free in hosting web services on their own server, which results in a real P2P network of ad-hoc web service communication. The catch is that SOAP, WSDL and UDDI, on their own, only begin to scratch the surface of the requirements necessary to deliver robust interoperability among applications. IBM and Microsoft have been particularly aggressive in developing and promoting new Web-services specifications to address security, routing, workflow and other functional requirements. Among those specifications are:

- WS-Inspection, developed jointly by IBM and Microsoft [WSIL]. It specifies where to look for Web services on Websites, to allow the discovery of Web services that aren't registered in UDDI directories
- Web Services Modelling Framework (WSMF) [FB02], provides a conceptual model for developing and describing web services and their composition (complex web services).
- DAML-S [ABC<sup>+</sup>01] that employs semantic web technology for service description.
- Web Services Experience Language (WSXL), designed to help developers create interactive services accessed by people, rather than by other software programs. IBM submitted WSXL to the Organization for the Advancement of Structured Information Standards (OASIS), which has a Web Services for Interactive Applications technical committee that IBM chairs, for consideration;
- Web Services Standard for Remote Portals (WSRP), also underway within OASIS and chaired by IBM, which aims to provide a mechanism by which content and applications can be represented in portals without any custom coding required; and
- Web Services Flow Language (WSFL), a workflow language intended to help link business processes into aggregate workflows, and also to allow individual business processes to "advertise" themselves as Web services. In this case, IBM is at odds with Microsoft, which is proposing its own workflow language, called XLANG.

### **4.3 Peers as agents**

Besides web services, also another important area in the IT world exist: the agent community. When the peers in the network behave like agents we could, for example, use Agent Communication Languages (ACLs) [LFP99] for communicating messages. Agent architectures fit into a general effort to support interactions among various software entities. Numerous approaches and technologies exist to support inter-

---

<sup>3</sup> <http://www.ibm.com/websphere/>

<sup>4</sup> <http://www.sun.com/software/sunone/>

process, inter-application, and inter-object communication over computer networks (e.g. DCE, TCP/IP, OMG/CORBA, OLE, ODBC, OpenDoc, ToolTalk). Agent interactions (communications in an ACL) can be layered on top of many of these protocols. Some initiatives in ACL's are KQML [FLM97], KIF [KIF] and FIPA-ACL [Fip01a]. An ACL provides language primitives that implement the agent communication model. ACLs are commonly thought of as wrapper languages in that they implement a knowledge-level communication protocol that is unaware of the choice of content language and ontology specification mechanism. Nearly all the ACLs around derive their language primitives from the linguistic theory of speech acts [Vas98]. Speech act theory categorizes human (or machine) "utterances" into different categories depending on the intent of the speaker (human or agent), the effect on the listener (human or agent), and any other physical manifestations of the act of uttering the utterance. Since speech acts are a human knowledge-level communication protocol, it is felt that they would be effective as an agent communication protocol, esp. since agents might operate on behalf of humans.

## 5 Overview of main characteristics in the P2P area

This chapter gives an overview of issues in the P2P domain. Due to the fact that [MKL<sup>+</sup>02] already wrote a perfect survey on the P2P area, we provide only a brief overview, extend it on some points and focus on the most important ones as that came out of the SWAP meeting in Warsaw, September 2002.

### **5.1 Interoperability: protocols vs. APIs.**

Interoperability means consistency across platforms, applications, and programming languages. A protocol determines how an application will be accessed, owned and created by a large group and defined outside software. An API is defined by a small group, owned by a small group, and closely tied to the software. APIs break over time in ways protocols don't. Classes of API breakages: (1) Unintended consequences, (2) underestimating the value of backward compatibility and (3) overestimating the coolness factor of new features.

A question that a developer should ask him/herself is how much control can and should be seeded from the application designers to the pool of users? What we've learned from the Network is that the more this is shared, the more it scales.

### **5.2 Scalability and network efficiency**

With scalability we mean the ability of the network and the peer itself to continue to function well as it is changed in size or volume. The networks' scalability depends largely on the routing protocols used to do advertisements, queries and answers. The peers' scalability itself depends on the balance between the resources (e.g. CPU cycles, memory and storage space) from and needs to the peer. The question about network scalability plays more on the Internet than on the intranet, because on the Internet the number of users that going to use the applications is mostly larger and difficult to predict.

In centrally controlled systems like Napster and SETI@HOME, a mediating server does the coordination task. In this case the network usage of querying and retrieving is very efficient but also vulnerable to the problems facing a centralized server.

Early P2P systems such Gnutella [Gnu01] and Freenet [Fre01] are ad-hoc in nature. A peer has to "blindly" send its requests to many other peers, causing the rest of the peers to search for the requested document. When the peer system is decentralized coordinated, like Gnutella and Freenet, each peer handles the communication individually. This can cause the time to retrieve a document to be unbounded. In addition, searching may fail even when an object exists, making the behaviour of the system nondeterministic. Therefore, in the case of decentralized coordination, increased network efficiency above the Gnutella approach is needed.

Increasing network efficiency means:

- Reducing redundant traffic
- Reducing load per node

Gnutella style networks optimizations are:

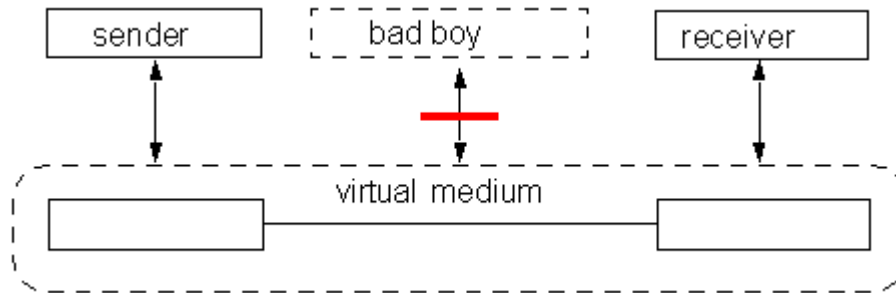
- Content caching and migration in the direction of demand (replication of content in the direction of high demand)
- Power law networks with accompanied routing algorithms (peers with a higher bandwidth have to route more queries than peers with a lower bandwidth).
- If queries are much more frequent than advertisements, it makes more sense to propagate advertisements to multiple peers.
- Propagation of advertisements could also be made in the direction of demand.

Recent P2P systems, represented by CAN [RFH<sup>+</sup>01], Chord [SMK<sup>+</sup>01], Oceanstore [KBC<sup>+</sup>00], and Pastry [RD01], dictate a consistent mapping between an object key and hosting node. Therefore, an object can always be retrieved as long as the hosting nodes can be reached. These systems are designed to scale to billions of users, millions of servers and over  $10^{14}$  files. Other systems like NeuroGrid [Jos02], Poblano [CY01] and Edamok [Eda02] try to use semantic knowledge about the data to forward queries to peers that have a high probability to answer the query. These last systems will be discussed in more detail in this paper.

### 5.3 Authenticity, confidentiality and security of information

In short, the P2P approach provides a new level of autonomy to marketplace participants.

When information is shared between different systems, which guarantee does the receiver have that it is the authentic document and isn't altered by a 'bad boy'? Or how can we prevent third persons of reading confidential information sent over the P2P network?



Security mechanisms are becoming more and more important in open network communications. To guarantee confidentiality, a huge variety of encryption algorithms and protocols are available. Digital signatures are widely used to ensure the authenticity of digital data. That means on the one hand, a signing party can not deny having signed a particular message (non-repudiation), and on the other hand, an entity verifying a signature can be sure that the message was not manipulated (data integrity). Signature schemes like MD5withRSA, SHA1withRSA, DSA or ECDSA can be utilized for these purposes.

To compute a digital signature from a message, the entire message must be available. That is, the message must be stored somewhere as a whole. This also applies to the verification process. A digital signature can only be verified if the signed message and the signature are entirely accessible. In the context of digital streams, these conditions cannot be met offhand due to the fact that they can be very long (or potentially infinite). Streams may include for instant online news, stock or data feeds, radar or medical data transmissions or digitized video and audio. Efforts like the Java Cryptography Extension [JCE02] provide also support for encrypting streams.

### 5.4 Privacy/anonymity for users

Anonymous publication and storage services allow individuals to speak freely without fear of persecution. Another goal of anonymity is to guarantee that censorship of digital content is not possible. The authors of the Free Haven [Dingledine 2000] have identified the following forms of anonymity:

- Author: A document's author or creator cannot be identified

- **Publisher:** The person who published the document to the system cannot be identified
- **Reader:** People who read or otherwise consume data cannot be identified
- **Server:** Servers containing a document cannot be identified based on the document
- **Document:** Servers do not know what documents they are storing
- **Query:** A server cannot tell what document it is using to respond to a user's query

There are five techniques to provide anonymity for peers [MKL<sup>+</sup>02]:

- **Multicasting.** Here the peer that wants to be anonymous sends the document to a multicast group. The net result of this approach is that every peer that sends and request is *beyond suspicion*.
- **Identity Spoofing.** Besides changing the originator's address, anonymity can also be ensured by changing the identity of a communicating party. For example, in Freenet [CSW<sup>+</sup>2001], a peer passing a file to a requestor, either out of its own cache or from an upstream peer, can claim to be the owner of the content. The responder is possibly innocent, from an attacker's point view, because there is a nontrivial probability that the real responder is someone else.
- **Covert paths.** Instead of communicating directly, two parties communicate through some middle nodes. Most existing techniques ensure only sender anonymity. A party that wishes to hide its identity prepares a covert path with the other party as the end of the path. Examples include Mix [Cha81], Onion [SGR97] and Herdes [SL00]. The covert paths can use store/forward or persistent connection. Varying the length of the covert paths and changing the selected paths with different frequency can achieve different degrees of anonymity.
- **Intractable aliases.** LPWA [GGK<sup>+</sup>97] is a proxy server that generates consistent untraceable aliases for clients from the servers. The client can open an account and be recognized upon returning to the opened account, while hiding the true identity of the client from the server. Techniques of this kind ensure sender anonymity and rely on a trusted proxy server. The degree of anonymity that can be achieved falls in between absolute privacy and beyond suspicion.
- **Non-voluntary placement.** An interesting new approach is anonymity via non-voluntary placement of a document on a hosting node. Here, a publisher forces a document onto a hosting node using. Because the placement is non-voluntary, the host cannot be held accountable for owning the document.

It must be stated that anonymity does not mean the absence of identity. Indeed, sometimes a certain degree of identification is unavoidable. For example, a cell phone number or a SIM card identification number cannot be kept anonymous, because it is

needed by the phone company to authorize and set up calls. As another example, the IP number of a PC cannot be hidden from its nearest gateway or router if the PC wants to send and receive network traffic.

Moreover, anonymity can be built on top of identity, but not vice versa. And often there are multiple ways to ensure anonymity like, for example, hashing techniques to make a unique identifier of the user's identity. In the examples above, it is much more difficult to link a pre-paid SIM card sold over the retail counter for cash to the actual cell phone user. Likewise, a co-operative gateway or router can help hide the PC's true IP address from the outside world by using message relays or NAT.

## **5.5 Digital Rights management**

When providers want to prevent or control unauthorized distribution of digital content we speak about Digital Rights Management (DRM). While consumers want immediate delivery and user-friendliness when using online distribution channels, content providers demand a legal and highly secure system that will accurately track sales and usage data to compensate copyright holders. Integrating support for DRM into a mediated P2P platform provides the secure, cost-effective solution the entertainment industry is searching for. DRM protects the digital goods, and can do so at a number of levels. This flexibility supports an e-commerce model by allowing different rights to be assigned to content at different price points. DRM allows business rules to be assigned for each piece of content.

Using music as an example, the DRM can be set to allow promotional content to be played three times, then become "locked" and unusable. Or, for content available in a paid subscription service, the rights can be set to allow unlimited plays but the file cannot be copied to another device or CD.

Effective DRM technology has not yet been created. The tendency of bits to reproduce, to flow and to be copied may well be a fundamental part of their nature. However DRM may not have to be perfect to work: if circumvention is inconvenient enough, then average users may prefer to pay.

Despite the fact of the mentioned problems there are different initiatives in the area of DRM. For example, the eXtensible rights Markup Language XrML provides a universal method for specifying and managing rights and conditions associated with all kinds of resources including digital content as well as services [XrML]. Another initiative is the Open Digital Rights Language. ODRL provides the semantics for a DRM expression language and data dictionary pertaining to all forms of content. The ODRL is a vocabulary for the expression of terms and conditions over content including permissions, constraints, obligations, conditions, offers and agreements with rights holders [ODRL].

## 5.6 popularity (follow the users, what is already out there)

Last but not least, we shouldn't have the "if we build it, they will come" mentality, where interesting technological challenges of decentralizing applications are assumed to be the only criterion that a peer-to-peer system needs to address in order to succeed. Clay Shirky provides the reason:

*It seems obvious but bears repeating: Definitions are useful only as tools for sharpening one's perception of reality and improving one's ability to predict the future. Whatever one thinks of Napster's probable longevity, Napster is the killer app for this revolution.*

*If the Internet has taught technology watchers anything, it's that predictions of the future success of a particular software method or paradigm are of tenuous accuracy at best. Consider the history of "multimedia." If you had read almost any computer trade magazine or followed any technology analyst's predictions for the rise of multimedia in the early '90s, the future they predicted was one of top-down design, and this multimedia future was to be made up of professionally produced CD-ROMs and "walled garden" online services such as CompuServe and Delphi. And then the Web came along and let absolute amateurs build pages in HTML, a language that was laughably simple compared to the tools being developed for other multimedia services.*

[Ora01]

At the time of writing this document a close cooperation is announced with the people of the EDAMOK project at Trento in Italy<sup>5</sup>. A big advantage of this cooperation is that we have a bigger chance of influencing the peer community (e.g. the JXTA group of SUN).

## 6 Specific P2P+SW issues

Peer-to-Peer computing combined with Semantic Web technology will be an interesting path to switch from the more centralized KM solutions that are currently implied by ontology-based solutions to a decentralized approach. P2P scenarios open up the way to derive consensual conceptualizations among employees within an enterprise in a bottom-up manner. Another feature of this scenario is that it makes knowledge management solutions be nearly free of administration in order that they may be used by everyone including private persons and small cooperating companies being e.g part of a virtual company. Unfortunately, the current P2P solutions have some important limitations that should be solved when we combine it with Semantic Web technology:

---

<sup>5</sup> <http://sra.itc.it/projects/edamok/>

- In many projects, P2P is discussed as a solution at the protocol level (no client-server dichotomy) and as a means for distributing disk space. However, this is of minor importance for improved service in knowledge management. Here it is the actual sharing of information and knowledge, which needs to be supported, and not the organization of disk storage or network traffic.
- Existing solutions such as Napster or Gnutella provide limited support in information and knowledge sharing. Napster supports only keyword-based search of music titles and author names, Gnutella does not provide any pre-defined search support. Each Gnutella client is completely free in how it interprets a query.
- Leading industry efforts such as JXTA by Sun Microsystems are limiting P2P services to string matching. No support for shared ontologies is provided. Queries are specified in arbitrary XML formats. No use is being made of the opportunities to use RDF/RDF Schema for expressing shared vocabularies. Finally, JXTA limits query answering to using resources in a single location, while in fact many queries will require the combination of information retrieved from different sources at different locations.
- Peer selection is currently not really based on content. This needs to be improved to route a query to the next knowledgeable peer rather than some arbitrary one.

These flaws however make current P2P systems unsuitable for knowledge sharing purposes. Key to the success of combining Peer-to-Peer solutions with Semantic Web technologies is the use of Emergent Semantics. Emergent semantics builds on lightweight and/or heavyweight ontologies that different individuals, departments, or organizations have created. It considers the overlap between ontology definitions and the use of concepts and relations with actual data in order to extract shared ontologies for sets of individuals or groups of people. Intelligent tools will use such definitions to ensure that knowledge will be appropriately structured, so that it can be easily found. Knowledge Management can occur in a distributed fashion without overhead through central administration.

The next sections describe the specific issues when combining P2P systems with Semantic Web technology:

### **6.1 Peer selection service**

In order to receive the right answers without flooding the peer network with queries one must ask the “right” peers. Ontology-based peer selection mechanisms need to exploit similarity of ontologies for this purpose.

### **6.2 Variation of ontologies and lack of ontological precision**

Different peers will use different, though overlapping ontologies. Alignment, mapping and visualization tools will have to cope with different ontologies, even though no alignments are explicitly specified. Some of the alignments and the mappings may be

found by analysis of peer knowledge using methods of the just emerging field of Emergent Semantics (e.g. same file categorized to different concepts indicates alignment).

Ontologies will be produced from various user interactions, like classifications into folders or usage of meta-data. Ontology definitions will be imprecise and “sloppy” ontologies will be the norm rather than the exception. An inference engine for these ontologies must be able to ask and answer queries to peers in a robust, scalable, often locally contained, manner.

### **6.3 *Ontological drift***

In a P2P environment, one cannot expect any maintenance to happen on the ontologies (in fact, users will often not know what is in the ontologies on their machine). As a result, we must design mechanisms that allow the ontologies to update themselves, in order to cope with ontological drift. Based on the queries and answers elsewhere in the P2P network, ontologies will have to adjust their own definitions accordingly.

## **7 Four different P2P protocols**

This section provides a brief overview of four distinct P2P protocols, namely Gnutella, JXTA, FIPA and Pastry.

Gnutella [Gnu01] is a file sharing protocol [GP01]. Applications that implement the Gnutella protocol allow users to search for and download files from other users connected to the Internet. Although the Gnutella protocol supports a traditional client/centralized server search paradigm, Gnutella’s distinction is its peer-to-peer, decentralized model. In this model, every client is a server, and vice versa. These so-called Gnutella servents perform tasks normally associated with both clients and servers. They provide client-side interfaces through which users can issue queries and view search results, while at the same time they also accept queries from other servents, check for matches against their local data set, and respond with applicable results. Due to its distributed nature, a network of servents that implements the Gnutella protocol is highly fault-tolerant, as operation of the network will not be interrupted if a subset of servents goes offline.

**Connecting and discovering.** To connect to the network one has to know the IP address and port of any servent that is already connected. The first thing the servent does when it connects is announce its presence. The servent it is connected to passes this message on to all of the other servents it is already connected to, and so on until the message propagates throughout the entire network. Each of these servents then responds to this message with a bit of information about itself: how many files it is

sharing, how many KBs of space they take up, etc. So, in connecting, you immediately know how much is available on the network to search through.

**Search & retrieval.** Searching works similarly to connecting: the peer sends out a search request, it is propagated through the network, and each servent that has matching terms passes back its result set. Each servent handles the search query in its own way. The simple query ".mp3" could be handled in different ways by different servents: one servent might take it literally, matching anything with ".mp3" in it, while another might interpret it as a regular expression and match any character followed by "mp3". To save on bandwidth, a servent does not have to respond to a query if it has no matching items. The servent also has the option of returning only a limited result set, so that it doesn't have to return 5000 matches when someone searches for "mp3".

Since all of the searches are to the local servent's database, the servent sees what everyone else is searching for. Using this, most clients have a Search Monitor that allows the user to see, in real time, the searches that their servent is responding to.

The retrieval of files works as follows: when someone finds a search result that he wants to download, he just connects to the servent in the same way your web browser would connect to a web server. Of course, the servent has this built-in, so the normal web browser never has to enter the picture.

Servents are also smart enough to compensate for firewalls. If someone is behind a firewall that can only connect to the outside world on certain ports (80, for instance) he will just need to find a servent running on port 80. Since the servents can serve on any port, one is likely going to find one that is serving on a firewall-friendly port. Also, if one is trying to download a file from a servent that is behind a firewall, he can ask the firewalled servent to push the file to him since he will not be able to connect to it directly. The only thing the protocol cannot compensate for is file transfers between two servents behind two different firewalls. In such a case, there really isn't anything that can be done.

**Applications.** BearShare<sup>6</sup>, Gnucleus<sup>7</sup>, LimeWire<sup>8</sup>, Morpheus<sup>9</sup>, Phex<sup>10</sup>, Shareaza<sup>11</sup>, Swapper<sup>12</sup>, Xolox<sup>13</sup>, Gtk-Gnutella<sup>14</sup>, Mutella<sup>15</sup>, Qtella<sup>16</sup>

---

<sup>6</sup> <http://www.bearshare.com>

<sup>7</sup> <http://www.gnucleus.com>

<sup>8</sup> <http://www.limewire.com>

<sup>9</sup> <http://www.morpheus-os.com>

<sup>10</sup> <http://phex.sourceforge.net>

<sup>11</sup> <http://www.shareaza.com>

<sup>12</sup> <http://mywebpages.comcast.net/jthomas497/swapper/swapper.html>

<sup>13</sup> <http://www.xolox.nl>

<sup>14</sup> <http://gtk-gnutella.sourceforge.net>

<sup>15</sup> <http://mutella.sourceforge.net>

<sup>16</sup> <http://www.qtella.net>

## 7.1 JXTA Protocol suite

The JXTA protocols are a set of six protocols that have been specifically designed for ad hoc, pervasive, and multi-hop peer-to-peer (P2P) network computing. Using the JXTA protocols, peers can cooperate to form self-organized and self-configured peer groups independent of their positions in the network (edges, firewalls, network address translators, public vs. private address spaces), and without the need of a centralized management infrastructure.

The design of the JXTA protocols seeks to create a set of protocols that have very low overhead, make few assumptions about the underlying network transport and impose few requirements on the peer environment, and yet are able to be used to deploy a wide variety of P2P applications and services in a highly unreliable and changing network environment.

**Connecting and discovering:** The Endpoint Routing Protocol defines a set of request/query messages that are processed by a routing service to help a peer route messages to their destination.

When a peer is asked to send a message to a given peer endpoint address, it looks in its local cache to see if it has a route to this peer. If it does not find a route, it sends a route resolver query message to its available peer routers asking for routing information. A peer can have as many peer routers as it can find or they can be pre-configured. Pre-configured routers are optional.

When a peer router receives a route query, if it knows a route to the destination, it answers the query by returning the route information as an enumeration of hops. Once a route has been discovered, a message can be sent to the first router and that router will use the route information to route the message to the destination peer. The route is ordered from the next hop to the final destination peer. At any point the routing information may become obsolete requiring the current router to discover a new route in order to complete the message delivery.

**Search & retrieval.** The goal of the JXTA group is to make a P2P system for the Web and also to make search within a distributed network very efficient. And so those are the two sides of JXTA Search: both deep and wide. Deep in the sense of finding content at the edges, deep into the database on the edge of the Web network; and wide in the sense of helping peers shuttle queries around more efficiently because of a varied distributed network.

Although one can implement its own search protocol, the Query Routing Protocol is a basic protocol to handle queries [Wat01]. The QRP defines mechanisms for sending and responding to queries in the JXTA Search network, as well as mechanisms for

defining meta-data for nodes in the network. The JXTA Search network consists of the following participants:

- JXTA Search Information Providers, anything that responds to requests formatted in the JXTA Search QRP language. Information providers may be JXTA peers or Web servers, such as cnn.com.
- JXTA Search Consumers, anything that makes requests in the JXTA Search QRP language. Consumers may be JXTA peers or Web sites with HTTP client interfaces to the JXTA Search network.
- JXTA Search Hub, a mechanism that facilitates efficient query routing over the JXTA Search network by handling message routing between consumers and providers. Providers register a description of themselves with the hub and wait for matching requests. Consumers query the network through the hub and await responses. Hubs can also be providers or consumers they can be chained together in a network.

Consumer applications send requests to the JXTA Search network via the nearest JXTA Search hub. The hub determines which of the known providers should receive the query based on provider meta-data. The hub sends the requests to providers, receives responses, and sends responses back to consumers.

**Applications.** Although JXTA is a platform independent protocol, SUN has implemented it<sup>17</sup> in Java. The next table shows applications<sup>18</sup> that make use of this JXTA implementation.

<b>name</b>	<b>description</b>
<u><a href="#">aisland</a></u>	agent framework
<u><a href="#">allhands</a></u>	Event Notification application
<u><a href="#">angelopeerrendezvous</a></u>	A p2p based interactive software for intra enterprise communication
<u><a href="#">gnougat</a></u>	Gnougat: Fully decentralised file caching
<u><a href="#">gnovella</a></u>	Some experiments with JXTA and document storage in an enterprise
<u><a href="#">go</a></u>	A Go Tournament based on the JXTA Protocols
<u><a href="#">halu</a></u>	JXTA media distribution application
<u><a href="#">instantp2p</a></u>	JXTA Demonstration GUI
<u><a href="#">jnushare</a></u>	Information sharing application based on GISP
<u><a href="#">juxtaprose</a></u>	a web / discussion content sharing application
<u><a href="#">jxta-httpd</a></u>	Provides a Set of service & tools providing web publishing
<u><a href="#">myjxta</a></u>	myJXTA - JXTA Demonstration Application (InstantP2P)
<u><a href="#">myjxta2</a></u>	myJXTA2 - Enterprise Version of myJXTA

---

<sup>17</sup> <http://www.jxta.org>

<sup>18</sup> All these projects can be found on <http://apps.jxta.org/servlets/ProjectHome>

<a href="#"><u>parlor</u></a>	App framework for creating collaborative P2P spaces
<a href="#"><u>project2p</u></a>	A peer to peer solution to share project document
<a href="#"><u>radiojxta</u></a>	delivering audio content over JXTA networks
<a href="#"><u>rosettachat</u></a>	Localized JXTA Peer Text Messaging
<a href="#"><u>shareddiary</u></a>	Shared Diary for a Small workgroup
<a href="#"><u>shell</u></a>	JXTA Command Line Shell for interactive access
<a href="#"><u>vop2p</u></a>	voice over P2P network
<a href="#"><u>www</u></a>	A project for HTML documents and information

## 7.2 Fipa

The core mission of the FIPA standards consortium is to facilitate the interworking of agents and agent systems across multiple vendors' platforms. This is expressed more formally in FIPA's official mission statement:

*The promotion of technologies and interoperability specifications that facilitate the end-to-end interworking of intelligent agent systems in modern commercial and industrial settings.*

The core message of FIPA is that through a combination of speech acts, predicate logic and public ontologies, we can offer standard ways of interpreting communication between agents in a way that respects the intended meaning of the communication. This is much more ambitious than, for example, XML, which only aims to standardize the syntactic structure of documents.

To support this, FIPA has adopted and is working on specifications that range from architectures to support agents' communicating with each other, communications languages and content languages for expressing those messages and interaction protocols which expand the scope from single messages to complete transactions. In the future, there are plans to extend this even further to cope with longer term relationships between agents.

**Connecting and discovering:** Agent management provides the normative framework within which FIPA agents exist and operate [FMS01]. It establishes the logical reference model for the creation, registration, location, communication, migration and retirement of agents.

The entities contained in the reference model (see *Figure 1*) are logical capability sets (that is, services) and do not imply any physical configuration. Additionally, the implementation details of individual APs and agents are the design choices of the individual agent system developers.

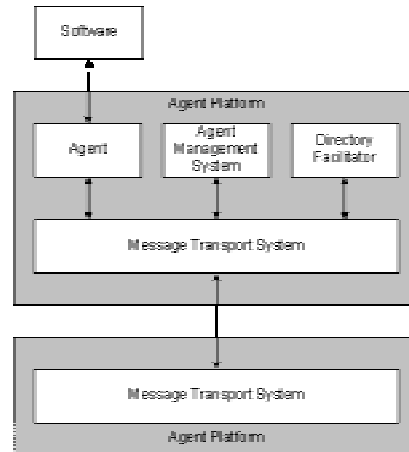


Figure1

The agent management reference model consists of the following logical components, each representing a capability set (these can be combined in physical implementations of APs):

- An **Agent** is the fundamental actor on an AP that combines one or more service capabilities into a unified and integrated execution model that may include access to external software, human users and communications facilities.
- A **Directory Facilitator (DF)** is a mandatory component of the AP. The DF provides yellow pages services to other agents. Agents may register their services with the DF or query the DF to find out what services are offered by other agents. Multiple DFs may exist within an AP and may be federated.
- An **Agent Management System (AMS)** is a mandatory component of the AP. The AMS exerts supervisory control over access to and use of the AP. Only one AMS will exist in a single AP. The AMS maintains a directory of AIDs which contain transport addresses (amongst other things) for agents registered with the AP. The AMS offers white pages services to other agents. Each agent must register with an AMS in order to get a valid AID.
- An **Message Transport Service (MTS)** is the default communication method between agents on different APs [FAS01].
- An **Agent Platform (AP)** provides the physical infrastructure in which agents can be deployed. The AP consists of the machine(s), operating system, agent support software, FIPA agent management components (DF, AMS and MTS) and agents.
- **Software** describes all non-agent, executable collections of instructions accessible through an agent. Agents may access software, for example, to add

new services, acquire new communications protocols, acquire new security protocols/algorithms, acquire new negotiation protocols, access tools which support migration, etc.

### Search & retrieval.

When an agent has a question but doesn't know whom to ask, it can search in the DF. A DF is a mandatory component of an AP that provides a yellow pages directory service to agents. It is the trusted, benign custodian of the agent directory. It is trusted in the sense that it must strive to maintain an accurate, complete and timely list of agents. It is benign in the sense that it must provide the most current information about agents in its directory on a non-discriminatory basis to all authorised agents. At least one DF must be resident on each AP (the default DF). However, an AP may support any number of DFs and DFs may register with each other to form federations.

Every agent that wishes to publicise its services to other agents, should find an appropriate DF and request the **registration** of its agent description. There is no intended future commitment or obligation on the part of the registering agent implied in the act of registering. For example, an agent can refuse a request for a service which is advertised through a DF. Additionally, the DF cannot guarantee the validity or accuracy of the information that has been registered with it, neither can it control the life cycle of any agent. An object description must be supplied containing values for all of the mandatory parameters of the description. It may also supply optional and private parameters, containing non-FIPA standardised information that an agent developer might want included in the directory. The **deregistration** function has the consequence that there is no longer a commitment on behalf of the DF to broker information relating to that agent. At any time, and for any reason, the agent may request the DF to **modify** its agent description.

An agent may **search** in order to request information from a DF. The DF does not guarantee the validity of the information provided in response to a search request, since the DF does not place any restrictions on the information that can be registered with it. However, the DF may restrict access to information in its directory and will verify all access permissions for agents that attempt to inform it of agent state changes.

**Applications.** The following table provides an overview of systems that make use of the FIPA standards<sup>19</sup>

<b>name</b>	<b>description</b>
Agent Development Kit <sup>20</sup>	The ADK features dynamic tasking, JXTA-based P2P architecture with XML message-based communication that supports FIPA and SOAP, JNDI directory services, using a lightweight runtime environment based on Java.

<sup>19</sup> A detailed overview can be found on <http://www.fipa.org/resources/livesystems.html>

<sup>20</sup> <http://www.tryllian.com/>

April Agent Platform <sup>21</sup>	AAP is written using the April programming language and the InterAgent Communication System (IMC),
Comtec Agent Platform <sup>22</sup>	Unique to the Comtec Platform is the implementation of FIPA Ontology Service and Agent/Software Integration, which require SL2 as the content language.
FIPA-OS <sup>23</sup>	FIPA-OS was the first Open Source implementation of the FIPA standard and supports most of the FIPA experimental specifications currently under development.
Grasshopper <sup>24</sup>	Grasshopper is compliant to both available international agent standards, namely the OMG MASIF and FIPA specifications.
JACK Intelligent Agents <sup>25</sup>	JACK Intelligent Agents, is an environment for building, running and integrating commercial-grade multi-agent systems using a component-based approach
JADE <sup>26</sup>	Will be discussed in detail in the next chapter
JAS (Java Agent Services API) <sup>27</sup>	The Java Agent Services (JAS) project defines an industry standard specification and API for the deployment of agent platform-service infrastructures.
LEAP <sup>28</sup>	LEAP (Lightweight Extensible Agent Platform (IST-1999-10211)) is a development and run-time environment for Intelligent Agents, is the precursor of the second generation of FIPA compliant platforms.
ZEUS <sup>29</sup>	ZEUS is an Open Source agent system entirely implemented in Java, developed by BT Labs and can be considered a toolkit for constructing collaborative multi-agent applications.
JFIPA <sup>30</sup>	JFIPA [Tve01] is a set of java-based tools that supports parsing and routing FIPA Agent Communication Language (ACL) messages represented in XML.

<sup>21</sup> <http://sf.us.agentcities.net/aap/index.html>

<sup>22</sup> <http://ias.comtec.co.jp/ap/>

<sup>23</sup> <http://fipa-os.sourceforge.net/>

<sup>24</sup> <http://www.grasshopper.de/>

<sup>25</sup> <http://www.agent-software.com/>

<sup>26</sup> <http://jade.csel.it/>

<sup>27</sup> <http://www.java-agent.org/>

<sup>28</sup> <http://leap.crm-paris.com/>

<sup>29</sup> <http://193.113.209.147/projects/agents/zeus/>

<sup>30</sup> [www.jfipa.org/](http://www.jfipa.org/)

### 7.3 Pastry

Pastry is a scalable, distributed object location and routing substrate for wide-area peer-to-peer applications. Pastry performs application-level routing and object location in a potentially very large overlay network of nodes connected via the Internet. It can be used to support a variety of peer-to-peer applications, including global data storage, data sharing, group communication and naming [RD01]. Pastry is completely decentralized, scalable, and self-organizing; it automatically adapts to the arrival, departure and failure of nodes. Experimental results obtained with a prototype implementation on an emulated network of up to 100,000 nodes confirm Pastry's scalability and efficiency, its ability to self-organize and adapt to node failures, and its good network locality properties.

**Connecting and discovering:** Each node in the Pastry peer-to-peer overlay network is assigned a 128-bit node identifier (nodeId). The nodeId is used to indicate a node's position in a circular nodeId space, which ranges from 0 to  $2^{128}-1$ . The nodeId is assigned randomly when a node joins the system. It is assumed that nodeIds are generated such that the resulting set of nodeIds is uniformly distributed in the 128-bit nodeId space. The nodeIds could be generated by computing a cryptographic hash of the node's public key or its IP address. As a result of this random assignment of nodeIds, with high probability, nodes with adjacent nodeIds are diverse in geography, ownership, jurisdiction, network attachment, etc.

The operation for connection of a node in the network:

```
nodeId = pastryInit(Credentials, Application)
```

It causes the local node to join an existing Pastry network (or start a new one), initialize all relevant state, and return the local node's nodeId. The application-specific credentials contain information needed to authenticate the local node. The application argument is a handle to the application object that provides the Pastry node with the procedures to invoke when certain events happen, e.g., a message arrival.

#### Search & retrieval:

The Pastry routing procedure is shown in pseudo code form in Table 1. The procedure is executed whenever a message with key  $D$  arrives at a node with nodeId  $A$ . We begin by defining some notation.

$R_l^i$ : the entry in the routing table  $R$  at column  $i$ ,  $0 \leq i < 2^b$  and row  $l$ ,  $0 \leq l < \lceil 128/b \rceil$ .

$L_i$ : the  $i$ -th closest nodeId in the leaf set  $L$ ,  $-\lceil |L|/2 \rceil \leq i \leq \lceil |L|/2 \rceil$ , where negative/positive indices indicate nodeIds smaller/larger than the present nodeId, respectively.

$D_l$ : the value of the  $l$ 's digit in the key  $D$ .

$shl(A,B)$  the length of the prefix shared among  $A$  and  $B$ , in digits.

```

(1) if ( $L_{\lfloor L/2 \rfloor} \leq D \leq L_{\lceil L/2 \rceil}$ ) {
(2)     //  $D$  is within range of our leaf set
(3)     forward to  $L_i$ , such that  $|D - L_i|$  is minimal;
(4) } else {
(5)     // use the routing table
(6)     Let  $l = shl(D, A)$ ;
(7)     if ( $R_i^{D_l} \neq null$ ) {
(8)         forward to  $R_i^{D_l}$ ;
(9)     }
(10)    else {
(11)        // rare case
(12)        forward to  $T \in L \cup R \cup M$ , such that
(13)             $shl(T, D) \geq l$ ,
(14)             $|T - D| < |A - D|$ 
(15)    }
(16) }

```

**Table 1.** Pseudo code for Pastry core routing algorithm.

Given a message, the node first checks to see if the key falls within the range of nodeIds covered by its leaf set (line 1). If so, the message is forwarded directly to the destination node, namely the node in the leaf set whose nodeId is closest to the key (possibly the present node) (line 3). If the leaf set does not cover the key, then the routing table is used and the message is forwarded to a node that shares a common prefix with the key by at least one more digit (lines 6–8). In certain cases, it is possible that the appropriate entry in the routing table is empty or the associated node is not reachable (line 11–14), in which case the message is forwarded to a node that shares a prefix with the key at least as long as the local node, and is numerically closer to the key than the present node’s id. Such a node must be in the leaf set unless the message has already arrived at the node with numerically closest nodeId. And, unless  $\lfloor L/2 \rfloor$  adjacent nodes in the leaf set have failed simultaneously, at least one of those nodes must be live.

This simple routing procedure always converges, because each step takes the message to a node that either (1) shares a longer prefix with the key than the local node, or (2) shares as long a prefix with, but is numerically closer to the key than the local node.

## Applications.

The next table shows applications that make use of Pastry.

name	Description
------	-------------

SCRIBE <sup>31</sup>	group communication/event notification system.
PAST <sup>32</sup>	archival storage systems
SQUIRREL <sup>33</sup>	a co-operative web cache.
PASTA <sup>34</sup>	Same as PAST but will provide mutability and a decentralized hierarchical namespace
Herald <sup>35</sup>	a publish/subscribe event notification service

## 8 Existing systems

This chapter shows a number of systems that are interesting for SWAP. We discuss them in a systematic way by walking through the previous chapters and see how the mentioned system relates it. This means that for each system we look:

- to which model the system belongs (broker mediated, direct P2P, resource sharing).
- to which metaphor this system fits best (RPC, web service, peer as an agent)
- how the system relates to the different main characteristics (interoperability, scalability, authenticity/confidentiality/security, privacy/anonymity, digital rights and popularity)
- how the system relates to the specific P2P+SW issues (peer selection, ontology variation, ontological precision and ontological drift)
- which kind of protocol it uses (in the case that the system uses another protocol we point out the most similar protocol)

### 8.1 Neurogrid

NeuroGrid [Jos02] is an approach to decentralized search involving adaptation to ongoing network activity, each successive search changing the knowledge that each network node possesses about the contents of other nodes. NeuroGrid aims to use this adaptation to support fast, reliable and efficient decentralized search. NeuroGrid consists of two components that complement one another: a semantic routing technique and a learning algorithm. In NeuroGrid, user responses to search results are stored and used to update the meta-data describing the content of remote servers. If a remote server is queried with a word such as “automobile” and returns adverts for

<sup>31</sup> <http://www.research.microsoft.com/~antr/SCRIBE>

<sup>32</sup> <http://www.research.microsoft.com/~antr/PAST>

<sup>33</sup> <http://www.research.microsoft.com/~antr/SQUIRREL>

<sup>34</sup> <http://www.kcsu.org.uk/tm25/pasta/index.htm>

<sup>35</sup> <http://research.microsoft.com/sn/Herald/>

weight-loss products, the reliability of this server with respect to this query should be reduced. The system knows that the response was inappropriate due to the absence of positive feedback such as storing links to the discovered data, or explicit negative feedback. Each NeuroGrid node maintains a knowledge base that stores associations between keywords and other NeuroGrid nodes, facilitating search of the network by forwarding queries to a subset of nodes that it believes may possess matches to the search query. NeuroGrid operates under the assumption that files (or file pointers) are referenced by a number of 'keywords'. The knowledge base of keyword-node associations maintained by each node represents local 'belief' concerning the contents of remote nodes.

### 8.1.1 Model

NeuroGrid in its current server side incarnation is not pure P2P, but is based around a large number of small servers being linked to one another in a P2P fashion, with each server supporting a small community of users. Ideally the system would be completely distributed, which could be implemented with a suitable address space that gave each node a fixed identity. Being able to rely on nodes fixed identities is important for the way in which NeuroGrid adapts.

### 8.1.2 Methaphor

The writers of NeuroGrid are not clear of the course they want to follow. In [Neu02] they argue that it would be much better to implement the protocol in XML-SOAP or by bit encoding like Gnutella. However, their initial objective with the NG (NeuroGrid) protocol is to provide a method of communication that will piggy-back on top of HTTP, not interfere with existing browsers, and to allow simple redirects when users select recommended URLs or search nodes. If the implementers of NeuroGrid follow the HTTP approach where XML-SOAP will be the message format, each peer can be seen as a web service.

### 8.1.3 P2P characteristics

#### Interoperability

The implementers have chosen to keep the first implementation very simple. They use an own language on top of the HTTP GET protocol. Although it is very simple it isn't interoperable with any standard. In the 'to do' list they want to make the implementation on XML-SOAP.

#### Scalability and network efficiency

NeuroGrid nodes maintain a list about which queries other nodes have been good at answering in the past. This allows a NeuroGrid node, when queried about something, to pass back a list of other nodes that could be queried, along with potential answers to the query. The direct advantage over systems like Gnutella is that rather than automatically flooding the network with queries and then trying to mop up with nodes

checking to see if they already sent a message or giving queries TTL (time to live), the emphasis is given back to the user so that they can decide if the local results are adequate or if they want to delve deeper. The disadvantage of this approach is that the user has to do the effort.

[Neu02b] provides an overview of a simulation is done to compare the effect of query forwarding in a network that was connected at random, and a network with a similar number of connections, but where those connections were formed by the presence of similar content on nodes at either end of the connection.

#### Authenticity, confidentiality and security of information

When a user tries to get information it has to click on the link shown in the browser. This link is provided after the user did the query. The link can be the direct route to the information source (e.g. <http://www.google.com>) but also a pointer to the providing node with an argument that points to the source (e.g. <http://www.peer1.com/peerservlet?http://www.google.com>). This feedback can be very useful for revising recommendations however in this way a malicious peer can alter information. The user has to check the provided URL, and check if the provided information really is from the source.

#### Privacy/anonymity for users

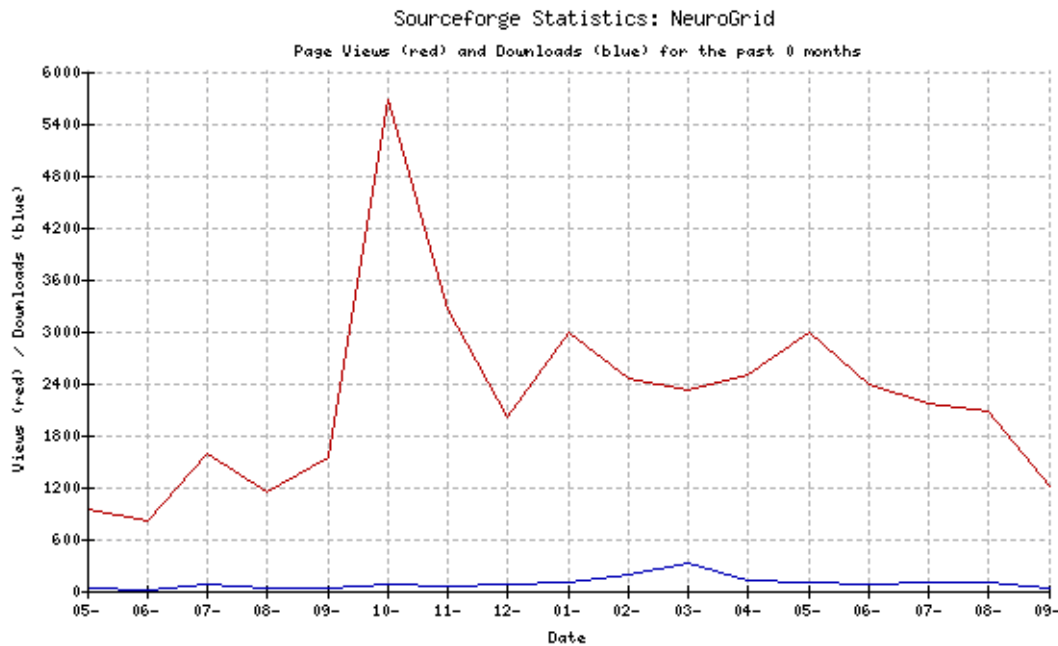
The queries and results are URL's via the HTTP GET protocol. Therefore the receiver can see the senders IP of the query or source. Therefore the sender isn't anonymous.

#### Digital Rights Management

The writers claim that the implementation will be extended to the XML-SOAP protocol. Then it is possible to extend it the XML message with XrML and ODRL.

#### Popularity

The next graph shows the views and downloads on the Sourceforge site from May 2001 till September 2002. The number of downloads is a bit disappointing.



#### 8.1.4 P2P+SW characteristics

##### Peer selection service

NeuroGrid nodes maintain a list about which queries other nodes have been good at answering in the past. This allows a NeuroGrid node, when queried about something, to pass back a list of other nodes that could be queried, along with potential answers to the query.

Initially upon querying a NeuroGrid node a user should receive a set of recommendations. URLs are clicked through; the NG node monitors the click through and then redirects the user's browser to the appropriate page. e.g.

```
<a href="/servlet/com.neurogrid.prime.AdvancedSearchServlet?
sub_type=feedback&keyword_id=339&user=other&url_id=73&
url=http://www.webdav.org/dasl">http://www.webdav.org/dasl</a>
```

The NG protocol really comes into play when a node recommends another NeuroGrid node, such as with the following link:

```
<a href = "/servlet/com.neurogrid.prime.AdvancedSearchServlet?
sub_type=feedback&keyword_id=13+1&user=other&node_id=11">
http://www.neurogrid.net</a>
```

On this way also a feedback loop is created where the receiving node can adjust the popularity of the content.

#### Variation of ontologies and lack of ontological precision

Each NeuroGrid node stores an index of user's bookmarks relationships with keywords, and also an index that contains the relation between keywords and other nodes, be they existing search engines, or other NeuroGrid nodes. Initially other searchable systems are entered by users and associated with a subset of keywords, but once they are used to provide data for users on subsequent searches they become associated with the keywords used in those searches. Due to the fact that the system only uses keywords and not relations (e.g. isa relations) between them, the problem of variation doesn't exist. Only in the case of homonyms (one word has more than one meaning), the system can propose wrong links to the user. However, the user will after clicking on the link, directly see that the information was wrong and can provide negative feedback or doesn't bookmark the link. Therefore the system will provide lower rankings for these 'wrong' links and will hesitate more to show them in the future. In [Jos02] they say the following:

*“Nodes come and go rapidly in some P2P systems. Since NeuroGrid nodes rely on there being some degree of regularity for them to learn, any environment in which there is little or no consistency will not support this kind of learning framework. However it is important to consider that a node's ability to supply content is bound up not just with the files it can serve or provide pointers to, but also how frequently it is connected to the network. The statistical learning process presented above will prefer those nodes that are regularly connected to the network. To the extent that a disconnected node provides particularly relevant content it will still be queried.”*

#### Ontological drift

NeuroGrid doesn't support any way to deal with ontological drift or with data that is not valid anymore. However in [Jos02] they want to use a timestamp for determine how recent the data:

*Clearly nodes will be limited in the amount of data they can maintain on other nodes, and a further development of the statistical framework being implemented in the latest NeuroGrid system maintains not just information about the extent to which valid data was returned by a node, but also how long ago that was. An adjustable discounting scheme will be used to follow the stability of different nodes.*

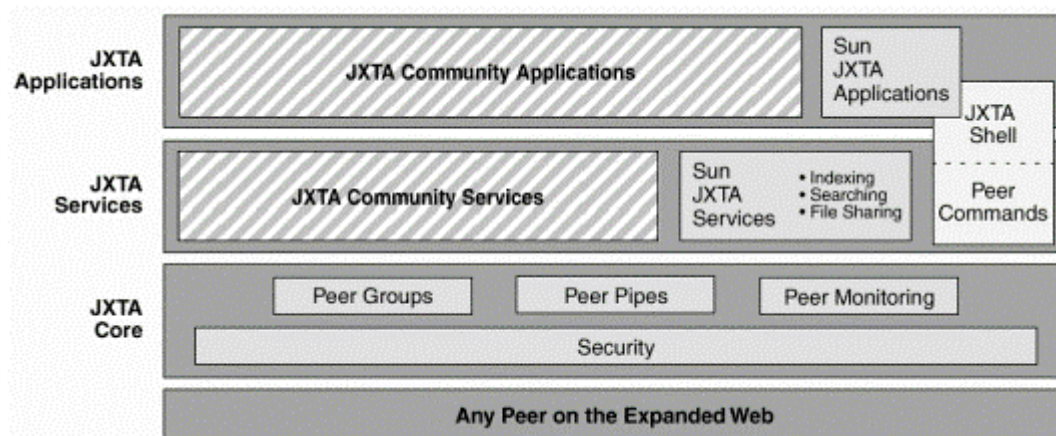
### 8.1.5 Protocol

NeuroGrid doesn't build on an existing platform or extends a protocol, however it use similar techniques to Gnutella to prevent loops and indefinite propagation: Time To Live (TTL) counters, which limit the range of queries, and Globally Unique Identifiers (GUIDs) which prevent a node forwarding a query twice.

The initial objective with the NG protocol is to provide a method of communication that will piggy-back on top of HTTP, not interfere with existing browsers, and to the extent possible allow simple redirects when users select recommended URLs or search nodes. The important thing to remember with the NG protocol is that efficiency, or speed of processing is not the first priority. Each NG node only moves from one transmission to the next at the speed of individual (human) user actions. Gnutella nodes forward queries as fast as they can, i.e. much faster than any human; but once an NG node provides a recommendation of a host, it waits for the human user to select where queries are forwarded to, if they are forwarded at all.

### 8.2 JXTA implementation by SUN

As of April 2001, the first prototype implementation is available on <http://www.jxta.org>. It is implemented on JDK™ release 1.1.4, by which they decided is the most common Java platform available on machines running Microsoft Windows and the UNIX operating system. The code should run on Windows95, 98, 2000, ME, and NT out of the box. It also runs on the Solaris™ Operating Environment and Linux with the appropriate level of Java runtime environment support. Version 1.0 is a starting point for the developer community than a finished product. Without any effort to optimize the code size, the core classes packed into a jar file have about 250 KB. The implementers expect that much smaller implementations will soon emerge.



### 8.2.1 Model

JXTA provides technologies to replace two of the advantages of centralized systems - discovery and firewall/NAT traversal. In fact, the current mechanisms are largely centralized. But the promise of JXTA is to abstract these problems away from application designers, allowing them to easily build decentralized applications.

JXTA provides discovery capabilities. When a peer joins a JXTA group, it can query that group for anything it may want: a service, a file, a peer with a particular name. The application does not need to worry about the details of how this query operates, because the JXTA platform handles it. It may query a central server (like Napster), or it may query the group in a decentralized fashion (like Gnutella). Either way, the application author gets discovery for free from JXTA. Concluding, we can say that JXTA provides the freedom to choose which model you want to implement (broker, direct or resource sharing).

### 8.2.2 Metaphor

JXTA give one the freedom of building a Web Service, an agent or do RPC's with it, so therefore it depends on the application that makes use of the platform which metaphor can be applied.

### 8.2.3 P2P characteristics

#### Interoperability

JXTA technology is designed to enable interconnected peers to easily locate each other, communicate with each other, participate in community-based activities, and offer services to each other seamlessly across different P2P systems and different communities. Project JXTA aims to bring to the P2P world what the browser brought to the Internet [Gon01].

JXTA technology is designed to be implementable on every device with a digital heartbeat, including sensors, consumer electronics, PDAs, appliances, network routers, desktop computers, data-center servers, and storage systems.

Many P2P systems, especially those being offered by upstart companies, tend to choose (perhaps unsurprisingly) Microsoft Windows as their target deployment platform. The cited reason for this choice is to target the largest installed base and the fastest path to profit. The inevitable result is that many dependencies on Wintel-specific features are designed into (or just creep in) the system. This is often not the consequence of technical desire but of engineering reality with its tight schedules and

limited resources. This approach is clearly shortsighted, as P2P does not stand for PC-To-PC. Even though the earliest demonstration of P2P capabilities are on Wintel machines — the middle of the computing hardware spectrum, it is very likely that the greatest proliferation of P2P technology will occur at the two ends of the spectrum — large systems in the enterprise and consumer-oriented small systems. In fact, betting on any particular segment of the hardware or software system is not future proof. Project JXTA envisions a world where each peer, independent of software and hardware platform, can benefit and profit from being connected to millions of other peers. For now, JXTA runs in JAVA on PC's. However Sun Microsystems Inc. announced in November 2001, that it would expand its JXTA P2P protocol to include handheld devices such as PDAs and wireless phones. The update adds support for J2ME (Java 2 Micro Edition), which will enable users of handheld devices to access files or applications and share data with other handhelds, PCs or servers in a Jxta-based P2P network, much in the same way that Napster users once swapped files between computers [JXME].

#### Scalability and network efficiency

JXTA does not mandate how messages are propagated. For example, when a peer sends out a peer discovery message, the Peer Discover Protocol does not dictate if the message should be confined to the local area network only, or if it must be propagated to every corner of the world.

The current implementation of JXTA uses the concept of a peer group as an implicit scope of all messages originated from within the group. In theory, any scope can be realized with the formation of a corresponding peer group. For example, a peer in San Francisco looking to buy a used car is normally not interested in cars available outside of the Bay Area. In this case, the peer would like to multicast a message to a subset of the current worldwide peer group, and a subgroup can be formed especially for this purpose. On the other hand, it seems more convenient and efficient if such a multicast can be done without the formation of a new peer group.

The JXTA group envisions a number of approaches to solving this problem. For example, all messages can carry a special scope field, which indicates the scope for which a message is intended. Any peer who receives this message can propagate the message based on the scope indicator. Using this approach, it is desirable that a sending peer is bootstrapped with some well-defined scopes and also can discover additional scopes. Implementations on top like Poblano or the efforts of the SWAP project are examples of external communities from JXTA to handle the routing problem.

JXTA does also not mandate exactly how discovery is done. It can be completely decentralized, completely centralized, or a hybrid of the two. In JXTA Version 1.0, the following discovery mechanisms are supported:

- LAN-based discovery. This is done via a local broadcast over the subnet.
- Discovery through invitation. If a peer receives an invitation (either in-band or out-of-band), the peer information contained in the invitation can be used to discover a (perhaps remote) peer.

- Cascaded discovery. If a peer discovers a second peer, the first peer can, with the permission of the second peer, view the horizon of the second peer, discovering new peers, groups, and services.
- Discovery via rendezvous points. A rendezvous point is a special peer that keeps information about the peers it knows about. A peer that can communicate via a rendezvous peer, perhaps via a pipe, can learn of the existence of other peers.

Rendezvous points are especially helpful to an isolated peer by quickly seeding it with lots of information. It is conceivable that some web sites or its equivalent will be devoted to providing information of well-known rendezvous points.

### Authenticity, confidentiality and security of information

At many places JXTA is independent of specific security approaches. Nonetheless, Version 1.0 is a first step towards providing a comprehensive set of security primitives to support the security solutions used by JXTA services and applications. To be more specific, JXTA Version 1.0 attempts to provide the following security primitives:

- A simple crypto library supporting hash functions (e.g., MD5), symmetric encryption algorithms (e.g., RC4), and asymmetric crypto algorithms (e.g., Diffie-Hellman and RSA).
- An authentication framework that is modeled after PAM (Pluggable Authentication Module, first defined for the UNIX platform and later adopted by the Java security architecture).
- A simple password-based login scheme that, like other authentication modules, can be plugged into the PAM framework.
- A simple access control mechanism based on peer groups, where a member of a group is automatically granted access to all data offered by another member for sharing, whereas non-members cannot access such data.
- A transport security mechanism that is modeled after SSL/TLS, with the exception that it is impossible to perform a handshake, a crypto strength negotiation, or a two-way authentication on a single pipe, as a pipe is unidirectional.
- The demonstration services called InstantP2P and CMS (content management service) also make use of additional security features provided by the underlying Java platform.

In other words, JXTA has security API's and a library that has implements the different security aspects, mentioned above. A description of the existing API's and implementation can be found at [JSP].

### Privacy/anonymity for users

For a JXTA peer to authenticate itself in a peer group a peer identity is required. Such an identity must be unique across the universe of peers. To provide privacy for the users, hashing techniques can be used to make a unique string of a user's identity. Of course, the source of the software that generates the key should not be open. Given

this unique identity, a peer can use it as part of a peergroup's authentication policy which also may require a password to be created to grant, for example, group, account privileges, and a renewal period. This is done over a private connection to protect the password. Finally, a group credential can be returned to the peergroup member, which acknowledges and embodies the authorized privileges. This same credential is then required whenever any of the associated peergroup services are used.

Admittedly, a great deal is missing in the above scenario, and the implication is that such a scheme may require JXTA authorization services. Peergroup members will need to be aware of which systems do authorization, and a source for list of URI's for these systems will need to be published using the JXTA advertisement mechanism. Finally, while authorization services must support the JXTA protocols, their internals may not be open source for either proprietary or security reasons.

### Digital Rights Management

To keep Version 1.0 small, they do not use a XML parser that can parse any XML document. Instead, they use a lightweight parser that supports a subset of XML. They are working towards normalizing this subset according to an existing effort called MicroXML [MicroXML]. It is unclear if this effort supports parsing of the two digital rights management XML languages described in section 4.5.

### Popularity

In April 2002, SUN provided a press release<sup>36</sup>, where it made the following statements about the popularity of the JXTA project:

*As an open source effort, Project JXTA has benefited greatly from the significant and numerous contributions of the active JXTA.org community. Nearly 10,000 registered members of the JXTA.org community actively contribute to the technology and thousands more are investigating and working with JXTA. The Web-based collaborative environment of JXTA.org provided by CollabNet has been instrumental in bringing the JXTA community together to work on approximately 80 known JXTA-based projects. These projects include: ipeers - artificial intelligence for P2P networks, Voice-Over P2P and Edutella - a P2P network of 20 universities. The JXTA technology and documentation have been downloaded for investigation and use close to a half million times by a variety of companies, developers and organizations looking to build P2P services and applications using JXTA.*

The project site of JXTA<sup>37</sup>, the weekly stats are show to give an indication of the popularity of the platform:

---

<sup>36</sup> <http://www.sun.com/smi/Press/sunflash/2002-04/sunflash.20020423.1.html>

<sup>37</sup> <http://www.jxta.org>

Weekly Stats	
October 5, 2002	
Members	11,179
Posts	96
CVS Commits	1039

#### 8.2.4 P2P + SW characteristics

##### Peer selection service

As already mentioned, JXTA doesn't mandate how messages are propagated and how peers are discovered. However, JXTA provides several discovery mechanisms (see section 7.2.3. about scalability) although they don't use any semantic knowledge about the data of the peer, to do more efficient routing. Efforts to make routing and peer selection possible, based on knowledge about the data on top of JXTA are, for example, Poblano [CY01], Edutella [NWQ<sup>+</sup>], Edamok [Eda02] and probably the SWAP project.

##### Variation of ontologies and lack of ontological precision

The current JXTA implementation doesn't work with semantic knowledge about the data, therefore it doesn't deal with variation of ontologies. How the systems built on top of JXTA (Poblano, Edutella and Edamok) deal with it, can be read in this paper.

##### Ontological drift

As already stated above, the lack of using semantic knowledge about the data, implies that it also doesn't handle ontological drift. Please look at the systems built on top, described in this paper (Poblano, Edutella and Edamok) how they deal with it.

#### 8.2.5 Protocol

Initially Project JXTA has defined the following six protocols. The developer community may develop more protocols.

- *Peer Discovery Protocol*  
This protocol enables a peer to find advertisements on other peers, and can be used to find any of the peer, peer group, or advertisements. This protocol is the default discovery protocol for all peer groups, including the World Peer Group. It is conceivable that someone may want to develop a premium discovery mechanism that may or may not choose to leverage this default protocol, but the inclusion of this default protocol means that all JXTA peers can understand each other at the very basic level. Peer discovery can be done with or without specifying a name for either the peer to be located or the group to which peers belong. When no name is specified, all advertisements are returned.
- Peer Resolver Protocol

This protocol enables a peer to send and receive generic queries to find or search for peers, peer groups, pipes, and other information. Typically, this protocol is implemented only by those peers that have access to data repositories and offer advanced search capabilities.

- *Peer Information Protocol*  
This protocol allows a peer to learn about other peers' capabilities and status. For example, one can send a *ping* message to see if a peer is alive. One can also query a peer's properties where each property has a name and a value string.
- *Peer Membership Protocol*  
This protocol allows a peer to obtain group membership requirements (such as an understanding of the necessary credential for a successful application to join the group), to apply for membership and receive a membership credential along with a full group advertisement, to update an existing membership or application credential, and finally, to cancel a membership or an application credential. Authenticators and security credentials are used to provide the desired level of protection.
- *Pipe Binding Protocol*  
This protocol allows a peer to bind a pipe advertisement to a pipe endpoint, thus indicating where messages actually go over the pipe. In some sense, a pipe can be viewed as an abstract, named message queue that supports a number of abstract operations such as create, open, close, delete, send, and receive. Bind occurs during the open operation, whereas unbind occurs during the close operation.
- *Peer Endpoint Protocol*  
This protocol allows a peer to ask a peer router for available routes for sending a message to a destination peer. Often, two communicating peers may not be directly connected to each other. Example of this might include two peers that are not using the same network transport protocol, or peers separated by firewalls or NAT. Peer routers respond to queries with available route information, which is a list of gateways along the route. Any peer can decide to become a peer router by implementing the Peer Endpoint Protocol.

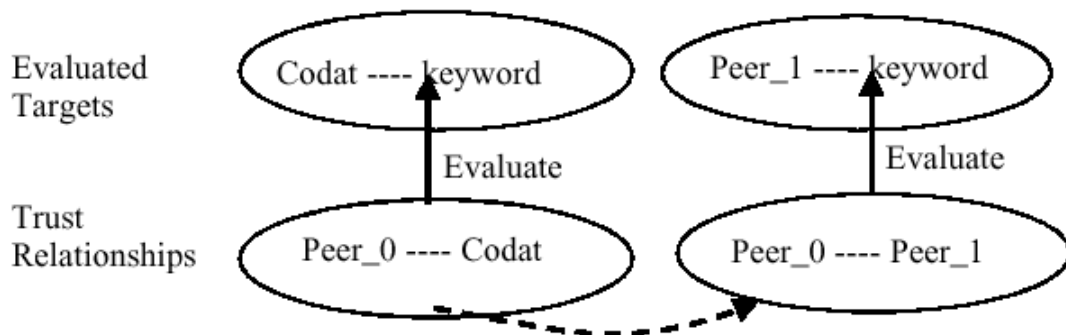
### 8.3 Poblano

The goal of the Poblano project is to establish a decentralized model of trust on the JXTA platform. This model presents trust relationships between peers and also relationships between peers and their content. One of the initial applications of this model is to perform reputation guided searching. A second application of Poblano is to build a recommendation system for security purposes.

Other reputations models, such as those used in Free Haven [Sni00] and Publius projects [WRC00] the degree of trust is calculated with parameters such as performance, honesty, reliability etc, of a given peer. If a peer cheats at playing cards, for example, the peer might be deprived of his ability to authenticate and join another card game.

But for the group of people interested in cooking, the above measurement is too biased towards personal risk, and not on content, and will be of little use. Hence, for the cooking group, the trust should be biased towards data relevance, or the quality of recipes. For us trust has multiple components or factors, and we are looking at a factor of trust, which is based on the group's interests, or group content relevance.

In order to collect the information on a group's "interests," Poblano introduces the codat trust component, as shown in the following figure:



At first, a user (Peer\_0) needs to evaluate his trust on a codat to build a trust relationship with that codat. In order to evaluate trust with respect to the peer's interests, the latter are represented as keywords. Then, the results of the "interests" evaluation on the set of codat that Peer\_0 received from Peer\_1 will be used to evaluate the Peer\_0's trust of Peer\_1 as a source for the given collection of keywords. In this way, the evaluation of the trust for the peer is based on the user's interests or keywords. Again, the codat trust component is different than the traditional trust concept, which is identified as the risk trust component in Poblano. The risk trust component's value will be at least determined by codat integrity, e. g., the codat contained a virus as noted by a virus pre-processor, peer accessibility (is the peer up most of the time), and peer performance (long delays in retrieving data).

### 8.3.1 Model

Each peer in the Poblano network bases its experience on other peers only on communication with these peers. Therefore it functions in a completely distributed fashion and can we categorize it in the "direct P2P" model.

### 8.3.2 Metaphor

The messages that are sent over the JXTA platform are XML messages. That's it, JXTA doesn't say what the content should be of the message, only some tags are reserved to specify JXTA specific commands. Poblano uses this feature to specify their way to make up the XML message. Due to the fact that they have developed their own language, it doesn't fit into the Web Service metaphor. Also the agent metaphor cannot be applied to Poblano, because, for example, they don't use speech acts, messaging protocols more than just query/answer and the peers are not proactive and autonomous.

### 8.3.3 P2P characteristics

#### Interoperability

As said, Poblano is implemented on top of the JXTA platform, thus it is independent of the platform and on the programming language. The messages sent between the peers are in XML, thus easy to parse in standard XML parsers. Also, the messages are easy to understand by human readers.

#### Scalability and network efficiency

Each peer contains a "CodatConfidence" table and a "PeerConfidence table" where confidence of respectively the codat and the peers are stored. It is not clear so far, how the system performs with a high number of peers in respect of memory and CPU usage. Due to the fact that Poblano asks only those peers wherein it has a high confidence to forward the query to, the network efficiency is higher then when the queries are broadcasted. There is, however, no public information available to see the results of some experiments about this topic. In section 7.3.4 and 7.3.5, we will discuss the algorithm of forwarding queries in more detail.

#### Authenticity, confidentiality and security of information

Each user will have security parameters that must be keep absolutely private. Here one might find data such as the user's private key, root certificates, local and remotely stored codat, and peer group credential. In order to protect this user's personal security environment a pass phrase mechanism will be implemented [CY01??].

For private, peer-to-peer communication Poblano implements the JXTA Transport Layer Security on top of the JXTA core protocols. They will use the TLS\_RSA\_WITH\_RC\_128\_SHA cipher suite from the JXTA security toolbox.

The model implies that at the least secure end-point of this spectrum, self signed certificates *may* be sent in the TLS handshake (instead of sending this over the network one can send it by post or via fax, or using a Certificate Authority (CA)). And, thus the 'imposter-in-the-middle' attack will be possible as it is for any PGP, Web-of-Trust where self-signing cannot prevent forged certificates, however cosigned certificates are more difficult to forge and provide very good privacy [CY01??].

Poblano plans on implementing more than two points in the trust spectrum, i.e., self-signed, cosigned, and CA signed certificates. And with cosigning, and satellite CA's they offer good privacy TLS for zero-dollar cost. The way this works is that Poblano provides algorithms for calculating codat trust based on a peer's reputation in a given peergroup. Since a certificate is one form of codat, it follows that the Poblano algorithms can be applied to a peer's peergroup-keyring, i.e., a peer group member's collection of signed certificates for a given peergroup.

### Privacy/anonymity for users

For a JXTA peer to authenticate itself in a peerGroup a peer identity is required. Such an identity must be unique across the universe of peers. Also, certificates issued to a peer need a unique user identification (UUID). For X.509 [HVP<sup>+</sup>99] certificates this must be a X.500 distinguished name that is unique across the Internet. An example is

CN=Yu Chang, OU=Fun Water Slides, O=Summer Play, Inc., C=FR

PGP certificates require user information but are less stringent about the details. This can be "identity" information about the user such as her or his name, user ID, photograph, and so on [PGP]. In either case, we can generate a unique UUID. For example:

CN=UserName, OU=<Twenty digit pseudo-random id>, O=www.jxta.org, C=Country

A suitable concatenation of the above DN name identifiers is also suitable for a PGP certificate. Thus, given that each peer has its own certificate, self-signed, cosigned, or CA signed, we can create a peer identity by hashing the concatenation of the UUID and the public-key fields, signing this hash with the private-key, and using the this digital signature as the identity. This hash provides the user privacy about its identity. Since such a signature can be large, for large keys it is the key length, the first twenty bytes can be used as a digital fingerprint. Another possible fingerprint is the MD5 or SHA-1 hash of the private key. Both are reproducible only by the owner of the private key, and verifiable, and can be used as a challenge. The identity can be used as the peer's credential in JXTA messages.

### Digital Rights Management

The literature on Poblano did not mention this topic therefore please look at the description of the JXTA implementation (section 7.2.4).

### Popularity

The literature on Poblano didn't mention anything on this topic.

### 8.3.4 P2P + SW characteristics

#### Peer selection service

When a peer receives a query it first looks up in the keyword in its own CodatConfidence table. If there is a local codat associated with the keyword, i. e., the answer is returned. If not, the peer looks up the keyword in its PeerConfidence table. If there are Peers highly associated with the keyword the request is forwarded to those peers. Each of those peers will perform the same steps as the requesting peer did. If such a peer has a valuable codat related with the keyword, the peer would inform the requesting peer, and terminate the search.

#### Variation of ontologies and lack of ontological precision

When a peer gets a query from a user, it does a lookup in its own CodatConfidence table. If there is a local codat associated with the keyword, i. e., the local flag is true, then the answer is returned. It is not clear how the system responds on a keyword that refers to more than one codat. Perhaps this is not possible. Clear is however that it only matches on strings, thus a response is only provided if the keyword in the query exactly matches with the keyword in the table. This means that the system doesn't make use of relations between keywords (the same as Neurogrid does), and therefore limits the system in responding to queries.

#### Ontological drift

In the case of Poblano it is more useful to talk about the drift of expertise of the peers than about the drift of the validity of an ontology. In this way a peer can devalue the confidence on the knowledge of the providing peer. The way that Poblano does this is as follows. The requester (user) accesses the codat. Through clicking on the codat, the provider's popularity value for the codat will be increased by 1. By the means of the requester's rating of the codat, the old confidence object for the provider with respect to the keyword will be updated. Now the requester has an entry added in his own CodatConfidence table. The collection of such codat ratings for the given peer will be used to generate a PeerConfidence value for the provider with respect to the given keyword, codat pair. This is from the requester's eyes. The same codat rating will be also fed back to the provider. The provider may update his CodatConfidence table based as a function of the previous value, feedback on codat, and his confidence in the requester.

### 8.3.5 Protocol

Each peer will have PeerConfidence tables and CodatConfidence tables. Since codat is associated with peer groups, there is one CodatConfidence table for each group. If the peer belongs to more than one peer group, it will have more than one CodatConfidence table. The first PeerConfidence table is for those peers in a given peer group for which the peer has keyword, codat information. The second

PeerConfidence table is across all the peer groups to which the peer belongs. The latter table permits to calculate a peergroup independent PeerConfidence value.

When a peer does a search on a keyword, it will do the following:

Step 1) Lookup the keyword in its own CodatConfidence table. If there is a local codat associated with the keyword, i. e., the local flag is true, then succeed<sup>38</sup>; else

Step 2) Lookup the keyword in its PeerConfidence table. If there are Peers highly associated with the keyword, forward the request to those peers. Each of those peers will perform step 1 and 2 using their own tables again. If such a peer has a valuable codat related with the keyword, the peer will inform the requesting peer, and terminate the search.

Step 3) If the lookup fails, then the peer can resort to the peergroups' keyword tables discussed in [CY01] for other possible sources.

## **8.4 Edutella**

The goal of the Edutella project is a multi-staged effort to scope, specify, architect and implement an RDF-based metadata infrastructure for JXTA to enable interoperability between heterogeneous JXTA applications. The initial services that will be developed are:

- Query Service: Standardized query and retrieval of RDF metadata.
- Replication Service: Provide data persistence / availability and workload balancing while maintaining data integrity and consistency.
- Mapping Service: Translate between different metadata vocabularies to enable interoperability between different peers.
- Annotation Service: Annotate materials stored anywhere in the Edutella Network.

The first application that the project will focus on is a P2P network for the exchange of educational resources between German universities (including Hannover, Braunschweig and Karlsruhe), Swedish universities (including Stockholm and Uppsala), Stanford University and others.

---

<sup>38</sup> There will be instances when a peer will wish to search remotely even if the local codat is good.

#### 8.4.1 Model

The possibility of JXTA to completely distribute services is used by Edutella through the provision of Edutella Services (described in web service languages like DAML-S or WSDL, etc.) that complement the Jxta Service Layer, building upon the JXTA Core Layer, and Edutella Peers live on the Application Layer, using the functionality provided by these Edutella services as well as possibly other JXTA services.

#### 8.4.2 Metaphor

The Edutella approach uses web service languages like DAML-S or WSDL to describe the Edutella services that the peers provide. Therefore we can categorize Edutella in the Web Service metaphor.

#### 8.4.3 P2P characteristics

##### Interoperability

The main goal of the project is to enable interoperability between heterogeneous JXTA applications. The *Edutella Mapping Service* will be able to manage mappings between different schemata and use these mappings to translate queries over one schema X to queries over another schema Y. Mapping services will also provide interoperation between RDF- and XML-based repositories. Mediation services actively mediate access between different services, clustering services use semantic information to set up semantic routing and semantic clusters. The *Edutella Query Service* is intended to be a standardized query exchange mechanism for RDF metadata stored in distributed RDF repositories and is meant to serve as both query interface for individual RDF repositories located at single Edutella peers as well as query interface for distributed queries spanning multiple RDF repositories. One of the main purposes of the query service is to abstract from various possible RDF storage layer query languages (e.g., SQL) and from different user level query languages (e.g., RQL, TRIPLE): The Edutella Query Exchange Language and the Edutella common data model provide the syntax and semantics for an overall standard query interface across heterogeneous peer repositories for any kind of RDF metadata.

##### Scalability and network efficiency

For now, no hard statements can be made about the scalability of the system. The prototype environment will be up and running at the end of this year, further work will concentrate on refining the existing architecture and scalability of the Edutella network. The Edutella network offers a service called 'Edutella Replication'. These type of peers are complementing local storage by replicating data in additional peers to achieve data persistence / availability and workload balancing while maintaining data integrity and consistency. Since Edutella is mainly concerned with metadata, replication of metadata is our initial focus. Replication of data might be an additional possibility (though this complicates synchronization of updates).

#### Authenticity, confidentiality and security of information

Edutella did not mention this topic in the literature; therefore please look at the description of the JXTA implementation (section 7.2.4).

#### Privacy/anonymity for users

Edutella also did not mention this topic in the literature therefore please look at the description of the JXTA implementation (section 7.2.4).

#### Digital Rights Management

Edutella is able to handle different schemata, where the mapping services will provide interoperation between RDF- and XML-based repositories. In this way it is easy to include the different DRM protocols in XML.

#### Popularity

The prototype will be ready end of this year, so therefore we have to wait for these results.

### 8.4.4 P2P + SW characteristics

#### Peer selection service

Peers register the queries they may be asked through the query service (i.e., by specifying supported metadata schemas (e.g., “this peer provides metadata according to the LOM 6.1 or DCMI standards”) or by specifying individual properties or even values for these properties (e.g., “this peer provides metadata of the form `dc title(X,Y)`” or “this peer provides metadata of the form `dc title(X,'Artificial Intelligence')`”). Queries are sent through the Edutella network to the subset of peers who have registered with the service to be interested in this kind of query. The resulting RDF statements are sent back to the requesting peer.

The above text shows that the selection is done not at the selecting peer but at the receiving peer. Thus this means that, for now, there is a broadcast of the query and then a matchmaking process follows. This scales well in a small peer group, however isn't scalable in a larger context. Due to the semantic knowledge that each peer provides about its data, it should be easy to implement an efficient peer selection mechanism.

#### Variation of ontologies and lack of ontological precision

While groups of peers will usually agree on using a common schema (e.g., SCORM or IMS/LOM for educational resources), extensions or variations might be needed in some locations. The Edutella Mapping service will be able to manage mappings between different schemata and use these mappings to translate queries over one schema X to queries over another schema Y. Mapping services will also provide interoperation between RDF- and XML-based repositories. Mediation services actively mediate access between different services, clustering services use semantic information to set up semantic routing and semantic clusters.

Evaluating the following query (plain English) “Return all resources that are a book having the title ‘Artificial Intelligence’ or that are an AI book.” we get the query results shown in Figure 3, depicted as RDF-graph. <http://www.lit.edu/types#BookArtificialIntelligence> dc:title rdf:type

The strength of Edutella lies in its capability to specify in metadata exactly which queries a peer can receive. This metadata allows queries in the form `title(X,'Artificial Intelligence')`, that means that everything that math

### Ontological drift

The literature on Edutella, doesn't mention anything about this topic.

### 8.4.5 Protocol

The purpose of Edutella is not to implement an efficient routing mechanism, and therefore doesn't specify the way how messages are routed and how peers are found. They use the standard services of JXTA to find peers and to route messages. The strength of Edutella lies in the possibility to connect highly heterogeneous peers (heterogeneous in their uptime, performance, storage size, functionality, number of users etc.).

Each Edutella peer can make its metadata information available as a set of RDF statements. The goal is to make the distributed nature of the individual RDF peers connected to the Edutella network completely transparent by specifying and implementing a set of Edutella services. Each peer will be characterized by the set of services it offers:

**Query Service.** The Edutella query service is the most basic service within the Edutella network and is described in more detail in the second part of this paper. Peers register the queries they may be asked through the query service (i.e., by specifying supported metadata schemas (e.g., “this peer provides metadata according to the LOM 6.1 or DCMI standards”) or by specifying individual properties or even values for these properties (e.g., “this peer provides metadata of the form `dc_title(X,Y)`” or “this peer provides metadata of the form `dc_title(X,'Artificial Intelligence')`”). Queries are sent through the Edutella network to the subset of peers who have registered with the service to be interested in this kind of query. The resulting RDF statements are sent back to the requesting peer.

**Edutella Replication.** This service is complementing local storage by replicating data in additional peers to achieve data persistence / availability and workload balancing while maintaining data integrity and consistency. Since Edutella is mainly concerned

with metadata, replication of metadata is our initial focus. Replication of data might be an additional possibility (though this complicates synchronization of updates).

**Edutella Mapping, Mediation, Clustering.** While groups of peers will usually agree on using a common schema (e.g., SCORM or IMS/LOM for educational resources), extensions or variations might be needed in some locations. The Edutella Mapping service will be able to manage mappings between different schemata and use these mappings to translate queries over one schema X to queries over another schema Y. Mapping services will also provide interoperation between RDF- and XML-based repositories. Mediation services actively mediate access between different services, clustering services use semantic information to set up semantic routing and semantic clusters.

## **8.5 InfoQuilt/PSW**

The InfoQuilt System developed at the LSDIS Lab. provides a framework for formulating complex information requests, which can capture the semantics of user's request involving multiple ontologies, and support a form of knowledge discovery. Their current work involves integrating InfoQuilt's semantic capabilities with technologies more appropriate for developing a distributed and collaborative Semantic Web platform. As a step towards this objective, they investigate the use of Peer-to-Peer (P2P) computing as a possible infrastructure. They call the new concept a Peer-to-Peer Semantic Web (PSW). In realizing PSW, they use DAML+OIL that provides a specification framework for independently creating, maintaining, and interoperating ontologies while preserving their semantics, and use P2P that is used to provide a distributed architecture which can support sharing of independently created and maintained ontologies. The writers claim that their PSW facilitates:

- Distributed and autonomous creation and maintenance of local ontologies,
- Advertisement (i.e., registry) of local ontologies,
- Introducing inter-ontological relationships between relevant ontologies as-needed basis once they are located,
- Controlled sharing of knowledge base components among users in the network,
- Ontology-driven semantic search of concepts and services,
- Knowledge discovery and exploration of inter-ontological relationships.

The two important problems the PSW approach intends to address are 1) what knowledge to share, in other words semantic search of relevant ontologies and 2) how to share, meaning a platform to share independently created and maintained ontologies.

### **8.5.1 Model**

The InfoQuilt System has users connected in a P2P network, accessing a shared directory of information and services. The P2P network in InfoQuilt is user centered,

where a client registers itself to a directory or directories, and searches the directory for peers providing *semantically relevant* information and services. The directory also provides the necessary contact information to connect to a specific peer, so that a direct peer-to-peer connection can be established (see Figure 2). Currently they are using a proprietary directory service but are also investigating UDDI [UDD00] for a possible use for advertising and registering different ontologies. This Napster alike approach points out that this system fits in the broker model.

### 8.5.2 Metaphor

InfoQuilt is an agent-based system that allows users to semantically request information, semantically correlate data from different sources and of heterogeneous type or representation, and analyze data available from diverse autonomous and heterogeneous sources [STP01]. Furthermore, it also supports knowledge discovery through interactive what-if analysis or information hypotheses including conjectures and relationships within and across domains. InfoQuilt system includes: (a) language and tools to specify IScapes (*i.e.*, semantic information requests), and (b) tools and algorithms to perform what-if analyses to search the information space of semantically related data. IScapes allow parameterized specification of information requests and correlation that utilizes the domain ontologies, inter-ontological relationships and user defined functions to accurately describe a user's information need.

Each user in the P2P network runs a multi-agent Information Brokering System whose architecture is built upon the work done in [LSA+01]. The information and services that a user is interested in are semantically identified using the Knowledge Space Navigation algorithm, which provides means for locating relevant ontologies, hence relevant content.

Although, the literature on InfoQuilt talks about agents, it doesn't use standard ACL's (e.g. FIPA or KQML) for communication between agents, but use an own language. Also, the 'agents' are not pro-active and autonomous. Therefore, this system doesn't really fit in the 'agent metaphor', but can be better categorized in the 'RPC metaphor'.

### 8.5.3 P2P characteristics

#### Interoperability

The system is written in JAVA, and is therefore platform independent. Unfortunately they are using their own proprietary directory service for advertising and registering different ontologies. Thus, they developed their own API for accessing this service. In [STP01] the writers mention that they are investigating UDDI for this purpose. The current work changes the representation of the knowledge to DAML+OIL and DAML-S (for service discovery), which provides more features than our earlier XML and RDF based specification. This change strengthens the expressability of the knowledge, however limits the interoperability, because the DAML+OIL standard is still in development. It is even so, that DAML+OIL is already outdated, the new OWL language is the successor.

### Scalability and network efficiency

The P2P network in InfoQuilt, the client registers itself to a directory or directories, and searches the directory for peers providing *semantically relevant* information and services. The directory also provides the necessary contact information to connect to a specific peer, so that a direct peer-to-peer connection can be established. The scalability of this approach is not clear yet. The network efficiency is high, because the centralized (Napster alike) approach, prevents the broadcast problem that arises in a Gnutella type of protocol. However, the directory service could form a bottleneck in processing power and/or storage.

When a peer decides to share an ontology in the global knowledge space, s/he has to upload the ontology into the PSW. This process is called registration of ontologies. New concepts (KObjects) and relationships (Links) in the uploaded ontology are created appropriately. Once ontology is registered, other peers can refer to the definitions in this ontology for their use. This approach enables an environment for controlled sharing of ontologies. The disadvantage clearly lies in the scalability.

### Authenticity, confidentiality and security of information

The purpose of Infoquilt is to share ontologies through registering these in a directory. Once ontology is registered, other peers can refer to the definitions in this ontology for their use. This approach enables an environment for controlled sharing of ontologies. For example a peer can protect his definition by not sharing, but use assertions that refer to shared definitions. In case a peer decides to remove or deregister his ontology, all the definitions and the assertions that refer to these definitions become invalid in the knowledge space. In this way, the control of the ontology remains by the source. Although the literature on Infoquilt doesn't mention the confidentiality and security aspect, it is thinkable to send an authentication string with a request, that makes it possible that users still can search the relevant ontologies, but if you want to access them the user should have the right permissions.

### Privacy/anonymity for users

The directory service provides the necessary contact information to connect to a specific peer, so that a direct peer-to-peer connection can be established. This information is an URL, this means that the user isn't anonymous.

### Digital Rights Management

The InfoQuilt system, focuses on the sharing of knowledge. The DRM languages are not suited for this, because they provide descriptions to protect copyrighted data like music, books etc. The knowledge in the InfoQuilt system is meant to be shared, used and referenced. The knowledge that a user wants to keep private, are not registered in the directory service.

### Popularity

The literature doesn't mention anything about the popularity of InfoQuilt.

## 8.5.4 P2P + SW characteristics

### Peer selection service

One of the key advantages of constructing a knowledge space is semantic search. In the IScape Builder, the user specifies the keywords (usually common nouns) used in the information request. The IScape Builder is a stand-alone Java application that provides a graphical interface to create and execute IScapes. In the figure 3, a screen shot of this builder is provided and it enables users to provide the keywords for searching ontologies. The next step in the construction of the IScape is selection of ontologies that help the system understand the user's query. In the previous version of InfoQuilt, on which our PSW approach is built, the users need to create either their own ontologies from scratch or manually select some of the previously created local ontologies. However, the PSW approach enables InfoQuilt System to search the knowledge space for the relevant ontologies. When the peer sends a list of keywords to the directory service, a set of ontologies, accompanied with their sources is sent back.

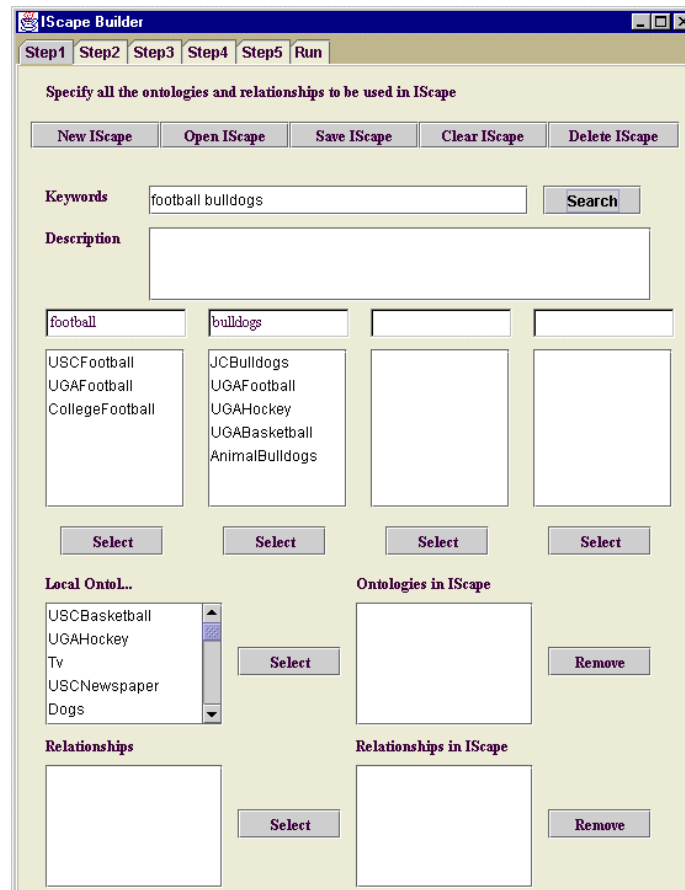


Figure 3: screenshot of IScape

Now the user can make relations between ontologies and make a selection.

## Variation of ontologies and lack of ontological precision

InfoQuilt depends on the user to select ontologies. The IScape user interface provide means to make this easier. On this way, the system can handle different ontologies, and the user can decide (in case of contradictive ontologies) which ontologies to use.

[??] depends on natural selection:

*“It does not necessarily mean that we have to agree to all the definitions existing in the knowledge space, but DAML+OIL helps to create a democratic and ecological knowledge space where different concepts and definitions can co-exist. One that survives is the one that is most referenced, other definitions go by unnoticed. In using DAML+OIL, the only concepts everybody agrees upon are the basic classes like Thing [Thi]. These are the predefined most or least general classes of all the classes that are defined in DAML+OIL. With the use of DAML+OIL the knowledge space is constructed from roots. When a new ontology is created, it is already hooked up in the knowledge space because of the use of imports and namespaces.”*

## Ontological drift

As said, InfoQuilt depends on natural selection, where the ‘best’ ontologies will survive. The future will show if this approach is sufficient to get the full potential out of the semantic web. In my opinion, you need more stimulation for competition and a measure mechanism to measure the quality of an ontology.

## 8.5.5 Protocol

The process of searching involves the following steps. The input is a set of keywords and the output is a list of ontologies .

1. Take each keyword and run a basic keyword match on the subject, object and the predicate (in that particular order) in the entire knowledge space,
2. Retrieve the name of the ontologies that satisfies the above match along with the ownership details,
3. If the keywords result in a number of ontologies, compare the ontologies for common parents and eliminate the ontologies without any common links,
4. If there is more than one ontology describing the same keyword, perform search with more keywords or compare the resulting ontologies to help user select the ontology.

Even though this search is invoked using keywords, they call it semantic, because the keyword matches are done with the KObjects qualified in the knowledge space, which by very construction preserves the semantics of the knowledge it describes. Unlike ordinary keyword searches, this kind of search has specific direction, because the writers of InfoQuilt believe that the keywords used in a valid information request are semantically woven in the holistic meaning they represent. One other utility they have developed using the knowledge space is the ability to compare two ontologies. This involves the following steps.

1. Identify the KObjects (concepts) used in each ontology in knowledge space,
2. Find a common parent KObject that links two KObjects that are defined in each of the compared ontologies; in other words find a connecting link (relationship) between the two ontologies and trace it for the user.

With DAML+OIL used in defining the ontologies, any two concepts will be linked together by the fact that both of them are *Things*. The scope of defining how close a common KObject can be in terms of the number of Links to be defined as a certain relationship is an interesting research problem and is beyond the scope of this paper. This utility can thus help the user explore inter-ontological relationships, and hence knowledge discovery.

## **8.6 Jade**

JADE (Java Agent DEvelopment Framework) is a software framework fully implemented in Java language. It simplifies the implementation of multi-agent systems through a middle-ware that claims to comply with the FIPA specifications and through a set of tools that supports the debugging and deployment phase. The full FIPA communication model has been implemented and its components have been clearly distinct and fully integrated: interaction protocols, envelope, ACL, content languages, encoding schemes, ontologies and, finally, transport protocols. The agent platform can be distributed across machines (which not even need to share the same OS) and the configuration can be controlled via a remote GUI. The configuration can be even changed at run-time by moving agents from one machine to another one, as and when required. JADE is completely implemented in Java language.

### **8.6.1 Model**

Every user can start an agent on an existing agent platform, or can start an agent platform him/her self. JADE agents can communicate among separate platforms; in other words, JADE provides mechanisms for inter platform communication. These mechanisms are normally CORBA based and use the IIOP protocol rather than RMI. Such an arrangement allows JADE agents to talk to FIPA compliant agents on any platform supporting CORBA and FIPA standards. This is the whole idea of FIPA: to make diverse agent systems interoperable. Also, there exist other protocols for inter platform communication. One popular method uses the HTTP protocol.

The JADE platform itself is made up of agents. The two key ones are the AMS (Agent Management System agent) and the DF (directory facilitator), which are mandatory and thus always there on the platform. Users can register their own agents on the platform.

As said, the DF is a mandatory component of the JADE platform that provides a yellow pages directory service to agents. It is the trusted, benign custodian of the agent directory. It is trusted in the sense that it must strive to maintain an accurate,

complete and timely list of agents. It is benign in the sense that it must provide the most current information about agents in its directory on a non-discriminatory basis to all authorised agents. At least one DF must be resident on each AP (the default DF). However, an AP may support any number of DFs and DFs may register with each other to form federations.

An AMS is a mandatory component of the AP and only one AMS will exist in a single AP. The AMS is responsible for managing the operation of an AP, such as the creation of agents, the deletion of agents, deciding whether an agent can dynamically register with the AP and overseeing the migration of agents to and from the AP (if agent mobility is supported by the AP). Since different APs have different capabilities, the AMS can be queried to obtain a description of its AP.

The AMS represents the managing authority of an AP and if the AP spans multiple machines, then the AMS represents the authority across all machines. An AMS can request that an agent performs a specific management function, such as quit (that is, terminate all execution on its AP) and has the authority to forcibly enforce the function if such a request is ignored.

JADE offers the freedom to use the DF to find agents, however it is not mandatory. One can also implement its own DF agent to do, for example, semantic search. When each user runs a JADE platform, thus where different platforms are connected with each other, one can say the model is real P2P.

## 8.6.2 Metaphor

Clearly, JADE fits in the agent metaphor. JADE simplifies the implementation of multi-agent systems through a middle-ware that claims to comply with the FIPA specifications and through a set of tools that supports the debugging and deployment phase.

## 8.6.3 P2P characteristics

### Interoperability

JADE is a software framework fully implemented in Java language. The agent platform can be distributed across machines (which not even need to share the same OS) and the configuration can be controlled via a remote GUI. The configuration can be even changed at run-time by moving agents from one machine to another one, as and when required. The only system requirement is the Java Run Time version 1.2.

JADE is FIPA compliant where FIPA tries to support both agent-level and platform-level interoperability through a comprehensive set of specification. At the agent level, FIPA mainly deals with ACL, interaction protocols, message content and message ontology issues.

On the platform level, the agents can communicate among separate platforms where JADE provides mechanisms for inter-platform communication. As said, these mechanisms are CORBA based and use the IIOP protocol rather than RMI. Such an arrangement allows JADE agents to talk to FIPA compliant agents on any platform supporting CORBA and FIPA standards. This is the whole idea of FIPA: to make diverse agent systems interoperable. Also there exist other protocols for inter platform communication. On popular method uses the HTTP protocol that can solve the firewall problem.

JADE can run on PC's and there is an initiative called JADE-LEAP [BP01] that allows JADE compatible agents to run on PDAs and phones.

### Scalability and network efficiency

About the scalability of the JADE Message Transport System [CQV02] says the following:

*“The obtained results show that JADE well performs in scalability in several scenarios (intra- and inter-platform). Messaging performance for intra-platform configuration highlight that JADE does not introduce relevant overhead compared to its underlining technology Java RMI, and that it well exploits different configurations of the agent platforms.”*

Unfortunately, the writers didn't check how the system performs on other MTPs (Message Transfer Protocols) then the IIOP MTP, like the HTTP MTP.

On the scalability of the JADE agents on CPU and storage aspects, we can say that JADE let developers completely free on how complex the agent will be. The LEAP project shows that the LEAP agents can be very small (run on mobile devices like PDAs) and still can connect on a JADE platform

### Authenticity, confidentiality and security of information

No form of security has been built into the JADE agent platform so far [PRT01], and the system itself is a single-user system, where all the agents belong to a single authority and have equal rights and permissions. This means that it is not possible to use JADE in several real world application, such as electronic commerce.

To deal with this limitation, a multi-user support has been designed and implemented within an experimental version of JADE called JADE-S [Vit02], leveraging the Java Security APIs to cast a security model on the agent platform. JADE-S makes the JADE platform a controlled multi-user environment, where all the components are owned by authenticated users, whom in turn are authorized by the platform administrator to perform only certain privileged actions.

JADE-S is based on the Java security model [Java-sec] and extends it for multi-agent systems. It also takes advantage of JAAS (Java™ Authentication and Authorization Service), JCE (The Java™ Cryptography Extension) and JSSE (The Java™ Secure

Socket Extension) technologies in order to provide a rich set of security features to agent-based application developers.

#### Privacy/anonymity for users

The JADE platform knows where its agents are running, due to the RMI connections between the agent and the platform. However, the names of the agents can be arbitrary, only the platform name is fixed to the URL where the platform is running (e.g. fooAgent@barHost.com:1099/JADE). This means that the URL of the agent itself is anonymous but the URL of its platform is known.

#### Digital Rights Management

JADE is not really meant for sharing files, however the content of a message can be an arbitrary string, thus also a file. One could think of a message that contains the file accompanied with another message that contains the DRM content.

#### Popularity

An indication of the popularity of JADE can be induced on the number of posts on their mailing list (about 4-10 posts per day) and the list of academic and industrial users. For an overview please look at [JP]. Unfortunately the number of downloads can't be found on their site.

### 8.6.4 P2P + SW characteristics

#### Peer selection service

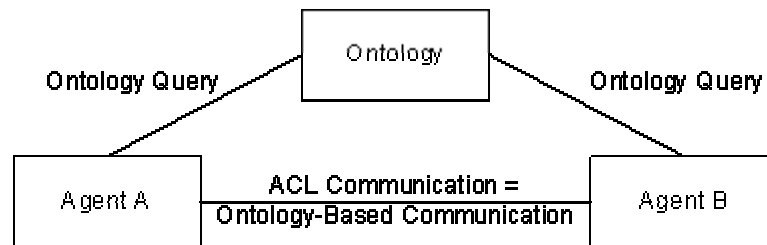
As said, in FIPA, the DF (Directory Facilitator) is a mandatory component of an AP that provides a yellow pages directory service to agents. Every agent that wishes to publicize its services to other agents, should find an appropriate DF and request the registration of its agent description. There is no intended future commitment or obligation on the part of the registering agent implied in the act of registering. For example, an agent can refuse a request for a service that is advertised through a DF. Additionally, the DF cannot guarantee the validity or accuracy of the information that has been registered with it, neither can it control the life cycle of any agent. An object description must be supplied containing values for all of the mandatory parameters of the description. It may also supply optional and private parameters, containing non-FIPA standardized information that an agent developer might want included in the directory. Due to the fact, that one can implement their own DF agent, JADE offers the freedom to make peer selection as complex as one wishes. Thus like in JXTA, there is a standard DF delivered, however one can choose to use another.

#### Variation of ontologies and lack of ontological precision

On the usage of varying ontologies, [Fip01c] says the following:

*“The FIPA communication model defined in [Fip01b] is based on the assumption that communicating agents share an ontology of communication defining speech acts and protocols (see Figure 1). In order to have fruitful*

communication, agents must also share an ontology of their domain of application. In an open environment, agents are designed around various ontologies (either implicit or explicit). For allowing their communication, explicit ontologies are however necessary, together with a standard mechanism to access and refer to them (such as an access protocol or a naming space).



**Figure 1: Ontology-Based Communication Model**

Without explicit ontologies, agents need to share intrinsically the same ontology to be able to communicate and this is a strong constraint in an open environment where agents, designed by different programmers or organizations, may enter into communication.

An explicit ontology is considered to be declaratively represented as opposed to implicitly, procedurally encoded. It can be then considered as “a referring knowledge” and, as a consequence, could be outside the communicating agents; managed by a dedicated ontology agent.”

JADE has the ability to deal with ontologies that are made beforehand in JAVA (and compiled). These ontologies have a unique identifier (e.g. animals.java), and agents can register the ontologies they ‘understand’ at the DF. Now agents can search other agents that use the same ontologies. The problem with the JADE ontology approach is that ontologies are static, thus can’t be changed on the fly (one has to adapt the java ontology source code, recompile it and restart the agent). Another problem is the rigid and limited query possibility. Fortunately, one doesn’t has to use the JADE ontology approach, it is very easy to send, for example, pointers to a DAML+OIL ontology that an agent uses, or send the ontology itself via a message.

### Ontological drift

JADE does not provide default mechanisms to judge the validity of ontologies. They also don’t mention anything on this subject in their papers. Therefore, the agents themselves have to deal with it.

### 8.6.5 Protocol

At the interplatform level, JADE offers the flexibility to ‘plug-in’ MTPs (Message Transport Protocols) like HTTP, WAP, IIOP and Orbacbus (these can be found on the

website of JADE [JADE]. This means that one also can build its own MTP by using the Java interfaces, defined in the jade.mtp package.

At the agent level, FIPA mainly deals with ACL, interaction protocols, message content and message ontology issues. The agents use the FIPA ACL (agent communication language) [Fip01a], where the syntax of the ACL is very close to the widely used communication language KQML [FLM97]. However, despite syntactic similarity, there are fundamental differences between KQML and ACL, the most evident being the existence of a formal semantics for ACL that should eliminate any ambiguity and confusion from the usage of the language.

## 9 Summary

This report provided an overview of P2P systems that make use of semantic knowledge about the data, or provide easy means to achieve this. All systems use the pure P2P model to find peers to answer a query. They don't depend on a centralized server, where all the addresses and/or the descriptions of the data are stored. Only JADE has a hybrid approach: agents have to be registered on an agent platform, that means a centralized approach. However, platforms can communicate with each other on a pure P2P basis, where they don't have to register themselves to a central registry. The main difference between Poblano / Neurogrid and InfoQuilt and Edutella is that the former do their own matchmaking when a query comes in, the latter depend on specialized peers that do the job. We didn't show any systems that use the 'resource sharing' model, because this isn't really P2P.

We described three metaphors, namely:

- the agent metaphor (JADE and InfoQuilt)-, where the peers use Agent Communication Languages (ACLs) for communication
- the web service metaphor (Edutella and in the future NeuroGrid), where the peers can be seen as a web service and make use of the webservice standards (e.g. WSDL, UDDI, SOAP).
- the Remote Procedure Call metaphor (JXTA, NeuroGrid, Poblano) , where peers don't make use of standard message protocols. The peers have to know exactly what it can ask to other peers, e.g. their API, to invoke its methods.

“Older” P2P systems, like Napster and [SETI@home](#) that made the P2P concept famous, all have their own specific API, but the newer systems trend to use standardized protocols and/or implement their system on top of a basic P2P platform like JXTA.

For each system we discussed some general characteristics that apply in the P2P world, namely interoperability, scalability and network efficiency, security and digital

rights management. Also the popularity of each system is discussed to see how the world responds on the system. Some remarks on each topic:

- Interoperability: This depends mainly on the protocol that is used. Systems like Poblano, Edamok and Edutella make use of the JXTA protocols. Their messages are made up in the XML standard and are therefore easy to understand. The JXTA implementation in JAVA is platform independent, and can be installed on any PC with JAVA, but also a small version is available for PDAs and mobile phones. InfoQuilt and JADE use ACLs (Agent Communication Languages) like FIPA-ACL and KQML. InfoQuilt and JADE are both implemented in JAVA, and are therefore platform independent and there is an initiative called JADE-LEAP [BP01] that allows JADE compatible agents to run on PDAs and phones. NeuroGrid is the only system that doesn't use a standard protocol or system, but in the literature can be found that in the future, it will use SOAP for implementing NeuroGrid peers as webservice.
- Scalability: Looking at the network efficiency, only Poblano and NeuroGrid peers use algorithms to calculate the best peer to ask a query. In contrast InfoQuilt and Edutella peers broadcast requests over the network to specialized peers that do the query matching. It is reasonable to say that the former two systems are more scalable in this sense. Both, JXTA and JADE provide standard 'directory' peers that can be seen as a kind of yellow pages. In JADE each platform has at least one directory agent and in JXTA each peer group has a service that knows the address of each peer. The nice thing of JADE and JXTA is that one can also implement their own yellow page peers. Especially for the SWAP this is an important feature of a peer platform to build peers that do query matching based on semantic knowledge.
- Security: JXTA is independent of specific security approaches; nonetheless Version 1.0 provides a set of security primitives like a crypto library and a password-based login scheme. For systems built on top of JXTA (Poblano and Edutella) it is easy to integrate these security protocols. JADE provides standard security protocols with an experimental version called JADE-S. JADE-S makes the JADE platform a controlled multi-user environment, where all the components are owned by authenticated users, whom in turn are authorized by the platform administrator to perform only certain privileged actions. InfoQuilt doesn't have any security means.
- Privacy: In P2P systems, messages are directly sent between peers over the internet, therefore the address of the receiver should be known by the sender by having its IP address. To provide privacy, one could introduce router peers that maintain a list of pseudonyms accompanied with the real address. This is possible in JADE and JXTA, so therefore it is also possible for the systems built on top of it. In InfoQuilt and NeuroGrid this principle is more difficult to implement, because they have already one fixed solution.
- Digital Rights Management: The message structure in JXTA is XML, so therefore all the systems on top of JXTA have the possibility to use the DRM XML messages (ODRL and XrML). The content of JADE messages can be

anything, thus also XML. Both, NeuroGrid and InfoQuilt don't use XML as the message structure.

- **Popularity:** Only JXTA and JADE are of real importance when we look at the number of publications, references and number of downloads. JXTA is popular in the P2P community, where JADE it is in the agent community.

The specific Semantic Web related topics like peer selection, variation of ontologies and ontological drift are discussed for each system. This report shows that Edutella and InfoQuilt use ontologies for describing and structuring the data that the peers share. JADE offers the possibility for agents to register their ontologies at the 'Directory Facilitator' (a kind of yellow page). Poblano and NeuroGrid use just keywords for describing the data. JXTA has no standard methods for using ontologies.

## 10 Literature

- [ABC<sup>+</sup>01] A. Ankolenkar, M. Burstein, T. Cao Son, J. Hobbs, O. Lassila, D. Martin, D. McDermott, S. McIlraith, S. Narayanan, M. Paolucci, T. Payne, K. Sycara, and H. Zeng: DAML-S: Semantic Markup For Web Services, <http://www.daml.org/services/damls/2001/10/daml-s.html>.
- [BP01] F. Bergenti and A. Poggi *Leap: A pa platform for handheld and mobile devices*. In presented at ATAL, 2001.
- [Cha81] D. Chaum. *Untraceable Electronic Mail Return Addresses and Digital Pseudonyms*, Communication of the ACM 24, 2, Feb. 1981, pp. 84-88.
- [CQV02] E. Cortese, F. Quarta, G. Vitaglione *Scalability and Performance of JADE Message Transport System*. 1st International Workshop on "Challenges in Open Agent Systems" held at AAMAS'02 in Bologna, Italy in July 2002. [http://www.agentcities.org/Challenge02/Proc/Papers/ch02\\_19\\_cortese.pdf](http://www.agentcities.org/Challenge02/Proc/Papers/ch02_19_cortese.pdf)
- [CSW<sup>+</sup>01] I. Clarke, O Sandberg, B Wiley, T.W. Hong *Freenet: A Distributed Anonymous Information Storage and Retrieval System*. Designing Privacy Enhancing Technologies: International Workshop on Design Issues in Anonymity and Unobservability, LNCS 2009, ed. by H. Federrath. Springer: New York, 2001
- [CY01] R. Chen and W. Yeager, *Poblano, a Distributed Trust Model for Peer-to-Peer Networks*. June, 2001, <http://www.jxta.org/project/www/docs/trust.pdf>
- [Eda02] EDAMOK, a joint project of the Institute for Scientific and Technological Research (IRST, Trento) and of the University of Trento. <http://sra.itc.it/projects/edamok/>
- [FAS01] FIPA Agent Software Integration Specification  
<http://www.fipa.org/specs/fipa00079/>

- [FB02] D. Fensel and C. Bussler *The Web Service Modeling Framework WSMF*, submitted to Electronic Commerce Research and Applications.
- [Fip01a] The FIPA Agent Communication Language Specification: <http://www.fipa.org/specs/fipa00061/>
- [Fip01b] The FIPA Agent Management Specification: <http://www.fipa.org/specs/fipa00023/>
- [Fip01c] The FIPA Ontology Service Specification: <http://www.fipa.org/specs/fipa00086/>
- [FLM97] T. Finn, Y. Labrou and J. Mayfield *KQML as an agent communication language*. In: Software Agents. Bradshaw, J. (Ed.) AAAI Press/MIT Press. ISBN 0-262-52234-9. 1997
- [FMS01] FIPA Agent Management Specification <http://www.fipa.org/specs/fipa00023/XC00023H.html>
- [Fre01] FREENET. 2001. The FreeNet home page, <http://www.freenetproject.org>.
- [GGK<sup>+</sup>97] E. Gabber, P Gibbons, D Kristol, Y Matias, A Mayer *Consistent, yet anonymous, Web access with LPWA*. Communications of the ACM. 42, 2. February 1999. pp. 42-47.
- [Gnu01] The gnutella home page, <http://gnutella.wego.com>
- [Gon01] L. Gong, JXTA: A Network Programming Environment . IEEE Internet Computing, (5)3:88--95, May/June 2001.
- [GP01] The gnutella protocol <http://capnbry.net/gnutella/protocol.php>
- [HFP<sup>+</sup>99] R. Housley, W. Ford, W. Polk, D. Solo. *Internet X.509 Public Key Infrastructure Certificate and CRL Profile*. rfc2459, January, 1999.
- [JABBER] <http://www.jabber.org/ietf/draft-miller-jabber-00.html>
- [JADE] The JADE project website: <http://sharon.cselt.it/projects/jade/>
- [Jav-sec] Java Security: <http://java.sun.com/security/>
- [JCE02] The Java Cryptography Extention (JCE) page: <http://java.sun.com/products/jce/>
- [Jos02] S. Joseph. *NeuroGrid: Semantically Routing Queries in Peer-to-Peer Networks*. International Workshop on Peer-to-Peer Computing in Pisa, Italy, May 2002, [http://www.neurogrid.net/NeuroGridSimulations\\_mod\\_a.pdf](http://www.neurogrid.net/NeuroGridSimulations_mod_a.pdf)
- [JP] Who is using JADE? <http://sharon.cselt.it/projects/jade/currently.htm>

- [JSP] The SUN JXTA Security Project. <http://security.jxta.org/Security-project.html>
- [JXME] The SUN JXTA for J2ME (JXME) Project. <http://jxme.jxta.org/>
- [JXTA] The JXTA project homepages <http://www.jxta.org/>
- [Kal02] V. Kalogeraki *Managing Peer-to-Peer Applications* OMG's Third Workshop on Real-time and Embedded Distributed Object Computing, San Fransisco, CA (January 2002)
- [KaZaA] KaZaA Media Desktop, Jan. 2002. <http://www.kazaa.com/en/index.htm>
- [KBC<sup>+</sup>00] J. Kubiawicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao. *Oceanstore: An Architecture for Global-State Persistent Store*. Proceedings of ASPLOS 2000, Nov. 2000.
- [KIF] The KIF specifications: <http://logic.stanford.edu/kif/>
- [FLM97] T. Finn, Y. Labrou, and J. Mayfield *KQML as an Agent Communication Language*, in J. M. Bradshaw Ed. *Software Agents*, MIT Press, pp. 291-316, 1997.
- [LFP99] Y. Labrou, T. Finin and Y. Peng *Agent Communication Languages: The Current Landscape*, Intelligent Systems, Vol. 14, No. 2, March/April 1999, IEEE Computer Society. <http://www.cs.umbc.edu/~jklabrou/publications/ieeeIntelligentSystems1999.pdf>
- [LSA<sup>+</sup>01] T. Lima, A. Sheth, N. Ashish, M. Guntamadugu, S. Lakshminarayan, N. Palsena, and D. Singh, *Digital Library Services Supporting Information Integration over the Web*, WIIW 2001
- [Rhodes] K. Rhodes. *Difference between SOAP and XML-RPC*: [http://weblog.masukomi.org/writings/xml-rpc\\_vs\\_soap.htm](http://weblog.masukomi.org/writings/xml-rpc_vs_soap.htm)
- [MicroXML] Micro XML parser. [www.udanex.com/tek/os/MicroXML.html](http://www.udanex.com/tek/os/MicroXML.html)
- [MKL<sup>+</sup>02] D. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, and Z. Xu. *Peer-to-Peer Computing* HP Labs Technical Report HPL-2002-57, March, 2002.
- [Neu02a] The NeuroGrid protocol <http://www.neurogrid.net/php/protocol.php>

- [Neu02b] NeuroGrid results <http://www.neurogrid.net/php/results.php>
- [NWQ<sup>+</sup>] W. Nejdl, B. Wolf, C. Qu, S. Decker, M. Sintek, A. Naeve, M. Nilsson, M. Palmer, and T. Risch. *Edutella: A P2P networking infrastructure based on RDF*. WWW2002, Honolulu, Hawaii, USA. ACM 1-58113-449-5/02/0005. May, 2002 <http://www2002.org/CDROM/refereed/597/>
- [ODRL] The ODRL homepage: <http://odrl.net/>
- [Ora01] Oram, A., ed., *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*, Sebastopol, CA: O'Reilly & Associates, March 2001
- [PGP] The International PGP Homepage, [www.pgpi.org](http://www.pgpi.org)
- [PRT01] Poggi, A., Rimassa, G. and Tomaiuolo, M. Multi-user and security support for multi-agent systems, in Proc. of WOA 2001. <http://sharon.cselt.it/projects/jade/papers/woa2001.pdf>
- [RD01] A. Rowstran and P. Druschel. *Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems*. In Proc. 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001), (Heidelberg, Germany), November 2001. <http://www.research.microsoft.com/~antr/PAST/pastry.pdf>
- [RFH<sup>+</sup>01] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. *A Scalable Content-Addressable Network*. In Proc. of ACM SIGCOMM, Aug. 2001.
- [Sanat] G. Sanat *SOAP Vs DCOM & RMI/IIOP* <http://www.stylusinc.net/technology/soap/soap2.shtml>
- [SGR97] P.F. Syverson, D.M. Goldschlag, M.G. Reed. *Anonymous Connections and Onion Routing*, IEEE Symposium on Security and Privacy. pp. 44-53. 1997
- [SL00] C. Shields, B.N. Levine. *A protocol for Anonymous Communication Over the Internet*. 7th ACM Conference on Computer and Communication Security (ACM CCS 2000), November 2000.
- [SMK<sup>+</sup>01] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. *Chord: A scalable peer-to-peer lookup service for internet applications*. In Proceedings of the ACM SIGCOMM '01 Conference, San Diego, California, August 2001.
- [SOAP] The SOAP specification: <http://www.w3.org/TR/SOAP/>
- [Sni00] B.T. Sniffen, *Trust Economies in the Free Haven Project* May 2000, <http://theory.lcs.mit.edu/~cis/cis-theses.html>.

- [STP01] A. Sheth, S. Thacker, and S. Patel, *Complex relationships and Knowledge discovery support in the InfoQuilt System*, Technical Report, LSDIS Lab, Univ. of Georgia, September 2001 (submitted for publication).
- [Tve01] A. Tveit. jfipa: XML-based FIPA ACL Message support for Software Agents, (Living) Technical Report, published on the Internet, Trondheim, Norway, September 2001
- [Vas98] Venu Vasudevan *Comparing Agent Communication Languages* OBJS Technical Note Object Services and Consulting, Inc. July 1998
- [Vit02] G. Vitaglione. *JADE Tutorial, Security Administrator guide*. TILAB S.p.A. September 2002.  
<http://sharon.cselt.it/projects/jade/doc/tutorials/SecurityAdminGuide.pdf>
- [Wat01] Steve Waterhouse. *JXTA Search: Distributed Search for Distributed Networks*. White Paper from Sun Microsystems, Inc. May, 2001.  
<http://search.jxta.org/JXTAsearch.pdf>
- [WRC00] M. Waldman, A.D. Rubin and L.F. Cranor, *Publius: A Robust, Tamper-Evident, Censorship-Resistant Web Publishing System*. In Proc. 9th USENIX Security Symposium, pp 59-72, August, 2000
- [WSIL] The Web Service Inspection Layer Specification: <http://www-106.ibm.com/developerworks/webservices/library/ws-wsilspec.html>
- [XML-RPC] The XML-RPC homepage: <http://www.xmlrpc.com/>
- [XRML] The XRML homepage: <http://www.xrml.org/>