Towards the composition of Ad Hoc B2B Applications : Semantics, Properties and Complexity Management

Rosa Gil(*)(**), Jaime Delgado(**)

(**)Universitat Pompeu Fabra (UPF)
Departament de Tecnologia
Pg. Circumval.lacio 8, E 08003 Barcelona, Spain
rosa.gil@upf.edu, jaime.delgado@upf.edu

Zièd Choukair(*)

(*)Ecole Nationale Supérieure des Télécommunications de Bretagne (ENSTB)
Laboratoire d'Informatique des TéléCommunications (LIT)
Technopole Brest- Iroise- BP 832-29285 - Brest Cedex- France
zied.choukair@enst-bretagne.fr

Abstract

The goal is to build a general framework for ad hoc applications over large networks using semantics and complex systems analysis.

The first part deals with semantics because applications need to be described. Ontologies are used to achieve it. They systematize application knowledge in a graph. Sometimes, the amount of information can be huge, in other words, the graph could be a large network. This is the reason why in the second part called complex systems analysis, statistical analysis is used to extract properties from the graph. It allows to discover new knowledge because some structures appear in the graph as connections degree, clustering and "small world". For instance, this information will show which application components are highly and poorly connected and it is translated to a vulnerability to attacks. Finally, a telecommunication project is used as benchmark to validate the previous work.

1. Introduction

Ad hoc applications need an open framework to become real applications for service providers. It is easier when the problem is constrained and the environment is closed.

This previous feature implies that sometimes, there is no information a priori about new ad hoc components application going into our environment. However, if new application components are described semantically and related to an ontology, this problem can be solved . Even if the new component uses an unknown ontology, there is an opportunity to find information as we are going to see.

Well-defined ontologies are always related to 'upper-ontologies', ontologies of high abstraction level. They can be viewed as something common between the ontology that describes our application and the unknown ontology. For example, a concept named "pressureEvent" doesn't say too much. However if it is related to "Event" concept of an upper ontology, it is described in a high level and can be connected to another concept depending on "Event" such "temperatureEvent"(http://suo.ieee.org/)

Problematic is detailed in section 2, after, section 3 describes how it is useful to use semantics and which representations and languages are more suitable to describe component applications. Following section 4, where the statistical analysis is explained how it is going to be used. Finally, a model and an example: weather forecast is used to validate the it, where semantics and complex systems analysis are the pillars of telecommunication applications to build services : platform CoCTelS.[14]

2. Problematic

Applications in open environments where there is no a priori knowledge are developed based on existing components. These components define the application. When the components are usually renewed because they are required or the execution environment evolves, it is necessary to find a dynamical way to get valuable aspects as autonomy and mobility. Thus, we are describing just the situation that will be held on the next telecommunication generation.

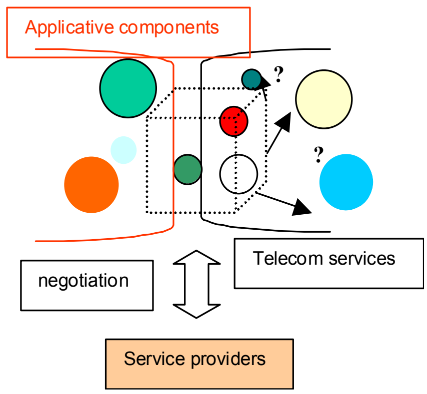

Figure 1. An open ad hoc application based on new components

First of all, a review of the current situation is needed. There is an application based on known components as in Figure 2 where it is necessary to know not only the components but their operators and functions also, to be able to connect to them and to keep our application going on, which are the basis of EAI and BDI (see section 2).

This approach does not fit with next telecommunication requirements, so a new model must be developed to accomplish them, we can see an overview at

Figure 1, where the application is built in a dynamical and self-organizing way, moreover there is also a negotiation with service providers to get the most appropriate component every time is needed.

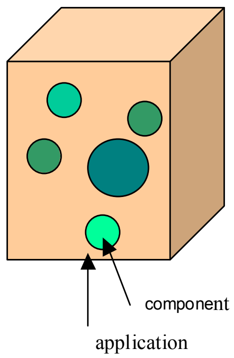

Figure 2. Classical application model design constrained by existing components.

There are some initiatives proposed as EAI (Enterprise application integration)[1]. BPI (Business Process Improvement, http://www.ebizq.net/topics/bpm) is at the top of the EAI abstract level. It is thought to integrate components as we saw in Figure 2. In order words, it creates a limited environment where collaboration is made without hierarchy and components know each other. Therefore, these models are limited by a priori knowledge and thus they are not really dynamic.

EAI market originated with the installation of enterprise resource planning (ERP) systems on a wide scale in the early 1990s. Companies once used client/server technology to build departmental applications, but later realized the gains in linking multiple business processes. Enterprises built distributed computing environments; only to find a competitive advantage in expanding those applications to include external business partners but nowadays with Web projects a new strategy has to be develop. The size and the continuous change in the web topology are the most important issues to be solved.

EAI Market Segmentation could be divided into five hierarchical categories of integration.

Abstract level

Business Process Integration

Process of workflow integration

Application integration

Data translation and transformation; rules base routing; pre-programmed adapters; monitoring and management tools

Component Integration

Application Servers

Data Integration

Tools for extracting, transforming and loading data

Platform Integration

Messaging

Nowadays, the solution taken in EAI is to design global packages and to customize depending of the user necessities. They use as background technologies great applications servers which usually are multi-component, cross-platform application servers and also are designed for building applications and web services with J2EE, PowerBuilder, XML or CORBA.

Agents platforms have been integrated in some applications servers as BlueJADE project in HP (BlueJADE project. http://sourceforge.net/projects/bluejade/ , where JADE-LEAP (JADE-LEAP platform, http://leap.crm-paris.com/) agent platform has been added to application server (Jboss, http://www.jboss.org/index.html, developed in Java and also HP-AS).

Other interesting purpose is XMI (OMG management group web site. http://www.omg.org/technology/xml/) , which enables easy interchange of metadata between modelling tools (based on the OMG UML) and between tools and metadata repositories (OMG MOF based) in distributed heterogeneous environments. XMI integrates three key industry standards: 1) XML - eXtensible Markup Language, a W3C standard; 2) UML - Unified Modelling Language, an OMG modelling standard; 3) MOF - Meta Object Facility and OMG modelling and metadata repository standard.

Thus, there is a tendency to :

1) integrate services and applications over networks (as the Web is)
2) use semantic representation for metadata, for instance XML is widespread over applications servers.
3) integrate different technologies as web services and applications using agents technologies.

These concepts have to face to the fact that every day the amount of components grows not only in number if not in quality. Thus, these existing technologies provide a great basis to build a new ways to provide a more specific analysis as we are going to see, first of all adding semantics and complex systems analysis as a second phase.

In the next section, it is going to be pointed why it is better to take RDF or OWL languages instead XML.

3. Semantics: Efficient information system integration

Enterprises are using XML technology as a basis but it is not the only one and it is not the most effective, many propositions are being made in the last times.

Web is an important content and service provider. Telecommunication service providers can not forgot it [2]. A little state of the art in web languages will help us to understand what the new technologies are going to propose.

3.1. Representing applications and services

Several markup languages are designed to represent concepts. For instance, web pages are based on HTML and their evolutions as DHTML. This was a revolution that takes its greatest maximum in the web, however something relevant was missing yet. We refer to semantics, because the amount of information that can be found in the web claimed to have a new way for representing data, a good metadata representation was necessary. XML (eXtensive Markup Language) was born to accomplish it with schemes concept aid. Some initiatives have arisen in enterprise world as ebXML (Electronic Business using eXtensible Markup Language - http://www.ebxml.org/)). Nowadays it is the most widespread metadata representation in new technologies as we have seen in the previous section.

However, RDF (Resource Description Language) represents an evolution that uses XML syntax but it is not constrained by that, so RDF is a foundation for processing metadata and it provides interoperability between applications that exchange machine-understandable information on the Web. RDF emphasizes facilities to enable automated processing of Web resources. In this way the most up technology is OWL (Ontology Web language - http://owl.mindswap.org/) or the vertebral column of what it is called Semantic web that has its starting point at DAML-OIL (Darpa Agent Markup Language-Ontology Inference Layer) where ontologies are knowledge representation in almost its high degree.

As an example we take an application from telecommunications domain which is described by XML files using PHP (PHP: Hypertext Preprocessor) and JSP (Java Server Pages) as a technology background for devices I-mode project [3] ,where there are different services as: weather or traffic among others. Afterwards we are going further in this project.

Illustrating the differences between files, in our case an user profile, described by XML and after by RDF, we must understand how the file is interpreted when it is processed. Both formats use API's (Application Programming Interfaces) ,ex: Xerces (http://xml.apache.org/) for XML and Jena for RDF.

They are defined by schemas: XMLSchema and RDFSchema which define vocabulary and relationships constraints over the XML tree and the RDF graph as we are going to see.

RDF is an abstraction layer over XML, so it integrates XML and the serialization files are very similar, however its interpretation differs because XML file is interpreted as a tree while RDF file is as a directed graph. This makes a great difference because elements order is important in an XML file and it can be or not in a RDF file, for instance, in the description of place, it doesn't matter the order but in a XML file we have to put it correctly, if not, the application will not success interpreting the file. RDF allows to specify if order is important or not by list or sets for example. Thus, RDF interpretation can be partial, but it is not the case of XML file, where all the document must be understood. It means that the structure it is well known, however it must be known 'a priori', and then we can consult, but it is not the case for RDF file where it is only enough to know the element and its relationships.

DAML-OIL and OWL allow to provide more detailed constrains over the RDF graph. For instance, we can specify elements cardinality, as how many days we need. Classify elements by their properties, i.e. define implicit classes. In our example every week or every day can be inside every time things because they are have a characteristic period and it is done without 'a priori' information. It is performed automatically as they were rules.

3.2. Publishing applications and services

DAML-S (Darpa Agent Markup Language – Services, Ontology http://www.daml.org/services/) seems to be in a near future a veritable standard in services world. DAML-S supplies Web service providers with a core set of markup language constructs for describing the properties and capabilities of their Web services in unambiguous, computer-interpretable form. DAML-S markup of Web services will facilitate the automation of Web service tasks including automated Web service discovery, execution, interoperation, composition and execution monitoring. Following the layered approach to markup language development, the current version of DAML-S builds on top of DAML+OIL.

Also initiatives as UDDI (Universal Description, Discovery Integration, http://www.uddi.org/specification.html) which claims to be a point of reference from industry with WDSL (Web Services Description Language, http://www.w3.org/TR/wsdl.html) as a web service definition language. Thus, WSDL is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information.

Both initiatives, DAML-S and UDDI will coexist at the same time surely and they will complement each other because the first is focused in Knowledge Management since

UDDI is understood as a widespread services discovery all over the world.

From OWL will emerge a new way for describing services more effective. It will be easier to map concepts from one ontology to other, because every concept will have not only a meaning but also a well-defined context and relationships with the other concepts and this will make the difference with nowadays technologies.

3.3. Applications components discovery and negotiation

Agents as a technological concept seem to be the natural approach to realize components discovery and negotiation. It is a crucial point to achieve dynamical behavior because we need something that can take decisions, act autonomously and of course making a deal, negotiating at last in the most effective way.

Some work will be showed in the model description, where agents negotiate multimedia content with service providers using expert systems and semantics to achieve this in a dynamical and effective way. However, a good mechanism to perform discovery was missing. It is the reason why we look at complex systems, where a way to get information in a quickly way it is possible in huge networks also, and moreover, properties can be extracted to be analyzed and make decisions to maintain a feedback in the system.

3.4. Services and applications integration technology

Many enterprises are working to provide services, some are proprietary, others provide an open environment , for instance, JWSDP (Java Web Services Developer Pack, http://java.sun.com/webservices/) , using XML.

APIs are being developed to integrate services, not only web services but EJB (Enterprise Java Beans) with XML, it provides interaction interfaces between clients, containers, components and concentrators.

4. Working with large networks: world wide web

Real systems doesn't behavior as perfect systems (periodically) nor chaotic. Their behavior is just in the middle and not predictable.

Recent researching over large networks among scientists have focused on a number of distinctive statistical properties that large networks seem to share as: Multi-Agent Collaboration networks [4], technological networks such as Peer to Peer systems [5], the World Wide Web [6], power grids [7], biological networks such as neural networks [8] among others disciplines.

Which are these features? The next lines want to make light about it.

Scale free networks [9] have appeared to be accurate descriptions of real networks as Internet or the Web. Using graphs theory we can make a representation of concepts and their relationships among them, the graph is enormous when we analyze the web, for instance, or it can be only a partial analysis but the great it is that it doesn't matter, it's scale free!, it means that it has a power-law distribution, so you can find the same properties a different scales: as a fractal.

Following these lines, small world phenomenon is described [10], showing that "there are many connections between near neighbors and few with far nodes "

4.1. Small World

Small world systems exhibit properties as Average path length, clustering coefficient, degree distribution and spectral properties [11]. Defining properties, the small world definition will appear.

- **Path length**: For random graphs, we have very short distances if we analyse *path length (d)*, defined as the average minimum distance between any pair of nodes. It can be shown that in random graphs, $d_{random} \approx log(n) / log(z)$. Graphs where $d \approx d_{random}$ are said to be 'small-world' networks that propagate information very efficiently.
- **Clustering coefficient**: It measures the probability that two neighbours of a given node (z) are also neighbours of one another. For a random graph, $C_R \approx z/n$ and is thus a very small quantity. Also, it is noticed that in real networks, $C >> C_{random}$. High clustering favours small-worldness.

Therefore, for a given network, if it is observed a small path length, i.e. $d \approx d_{random}$, but a big clustering coefficient, i.e. C >> Crandom, it can be said that very likely it is a small world.

5. A new model: beyond integration and interoperability

The previous sections have showed the difference between XML and RDF and why using graphs can be a great advantage. However, this is only the beginning. Figure 3 is a model schema which is going to be described.

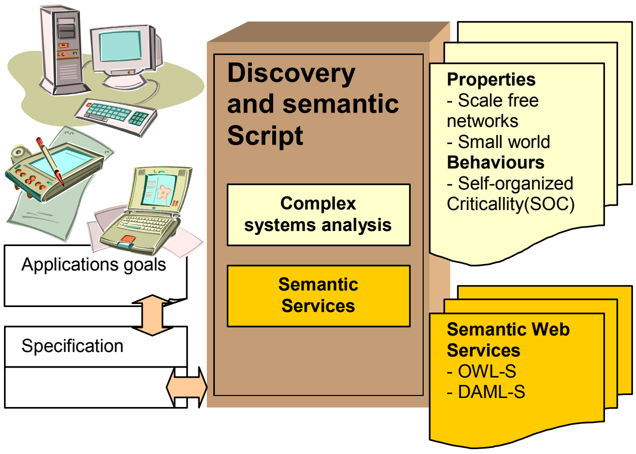

Figure 3. New application model based on complex systems analysis and semantics

It is supposed to be some components all over a large network, and the objective is to discover resources that will build applications over the network.

The premises are:

1. The network is huge for our proposes, however we have seen that there is an interesting behavior that can be characterized. (section 4)
2. We know components applications semantics

The goal is to build an application that makes a weather forecast. At first, as we have explained in the premises we don't know where we can find the components to build this application, perhaps because a resource has disappeared or we have lost the information we had in the past.

Software engineering provides interesting solutions to assemble software components according to proven methodologies and technologies [12] [13]. We are going to work over a composition platform for telecommunication services which are composed of applications. These are made of components. The platform CoCTelS [14] offers to each user an environment which allows him to select services he is interested in.

Bouquets are composed of elementary components and the connector-factory generates the connector which is an abstraction for component interactions, see Figure 4. Each user has his own connector being able to call shared services and aspects. Hence, the connector-factory should create connectors as small as possible to ease their storage, their activation induce a minimal resource consumption when executed. The synchronization of the accesses to the elementary services have to be managed as many connectors may invoke the same service at the same time. They are still managed as before.

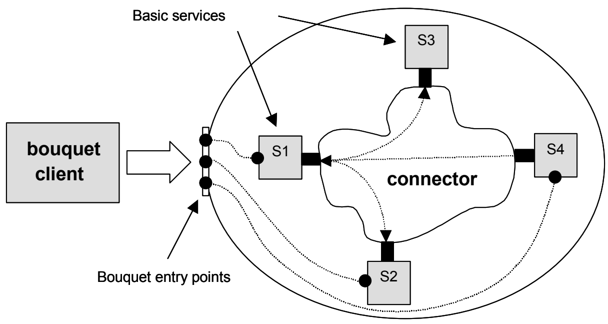

Figure 4. The bouquet and its entry points

The connector-factory will base his generation according to the description of the elementary services and aspects which are selected by the client. Two points of views have to be considered : the static point of view and the dynamic point of view. The static point of view deals with the connection links between the elementary components of the bouquet of services. The dynamic point of view deals with the management of the dynamic aspects of the collaboration between the components participating in the virtual bouquet. For example, when a client asks his bouquet to load a video mail, the connector has to call the mail service, get the reference of the video and its description and then, according to network resources availability propose, to the caller the video or to select an alternative among rendering degradation, or getting notified later when the network will offer enough resources to assure enough quality.

Adding values to the services is a factor of satisfaction to the users and of income to the operators. When helping the client to configure his virtual bouquet of services, the bunch manager has to propose, when appropriate, the aspects which may potentially be composed with the services already retained by the user. For this purpose, the bunch manager has to roam the entry points of the description of the service and whenever available, propose potential aspects to the user. If the user retains an aspect, the composition of the aspect is done the same way as between primary level services and secondary or system level services.

The bunch manager now has all the elements to generate the description file of the connector: a path in the ontology. It has at its disposal the services and the aspects which will compose the virtual bouquet, as well as the invocation sequence for each potential invocation. The connector description file generated by the bunch manager specifies trees whose roots are the high level services. The first leaves are the high level methods. Each high level method is associated to one or many aspects and has an execution tree to help the connector factory generate the right sequence of calls. They are the paths that we are going to build in the ontologies.

The first stage of connector-factory generation consists in getting the information concerning the high level services, the high level methods, the aspects and the processed calls in the execution tree. Then the factory has to get the associated technical information (localization of the service, the invocation protocol, the invocation parameters, pre and post conditions) and then generate the source code of the connector directly based on the execution tree. Afterwards, the connector-factory will look up the rules which have in their left side one of the methods inside the connector description file and take them into account. These rules which apply to the virtual bouquet of service will be considered

If we use semantics, we can fix an entry point: a weather ontology – for example we can take this one: http://www.mindswap.org/~mhgrove/weather-ont.daml, described in RDF, DAML or OWL, in other words: a graph. In this case it is useful to go to an ontology library (http://www.daml.org/ontologies/)

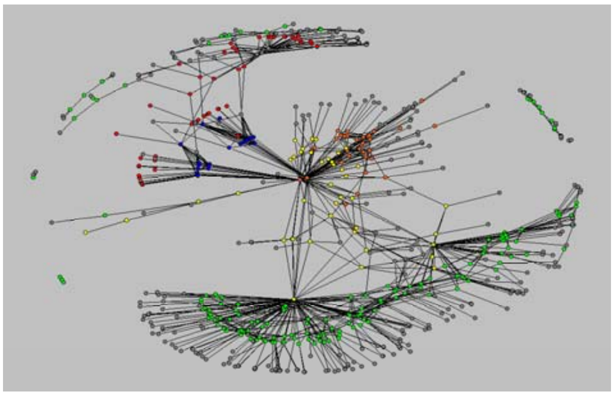

Figure 5. Network analysis: Green: weather Ontology , Yellow: daml , Red: purl, Orange: w3 consortium, Grey: anonymous, literals

The key point is to make ontologies related to others or to an upper ontology, and this is relevant when we represent it as we can see in Figure 5.

Pajek (http://vlado.fmf.uni-lj.si/pub/networks/pajek), a large network analysis tool, was selected. The RDF N-Triples serialisation was translated to a '.net' Pajek network file. The triples subjects and objects became network nodes connected by directed edges from subject to object. Nodes are identified by their original URIs to allow network construction and the edges are unnamed so duplicated edges are ignored.

An analysis is performed over the graph to locate concepts that can describe applications components, for instance, time. There are some ontologies related to time, we have taken one (http://www.isi.edu/~pan/damltime/time-entry.owl) and a GIS ontology (http://www.daml.org/2001/06/map/map-ont). In fact, ontologies are chosen dynamically depending on user profile (it is a graph and it describes customer preferences). The graph is showed in Figure 6.

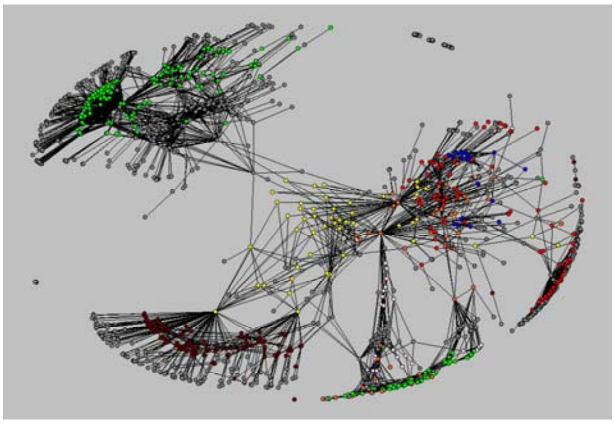

Figure 6. Network drawing. Green isi, Red dublincore , Blue purl, Yellow daml+oil , Orange w3 consortium, Brown mindswap, Pink ww.daml.org/2001/06/map/map-ont

In this way, we can discover from an ontology, which is describing a service, connections to other components in other ontologies and discover a path to find new information. Data extracted is summarized in Results section.

Moreover, it is possible to discover new information from the topology. The network analysis allow us to locate 'hubs', in other words nodes over the network that have many relations. They are good candidates to be important resources. Also it is possible to see clustering belonging to small world phenomenon as it was explained in section 4.

For each bouquet of service being composed, new paths are generated according to the elementary services which already compose the bouquet and its entry points. In order to simplify proof obligations, we assume that we should only verify the correctness all along execution paths and then consider states to be correct only if they belong to a valid execution trace. A trace ends when it reaches a stable state or an illegal state (component's input precondition is not satisfied when it is activated).

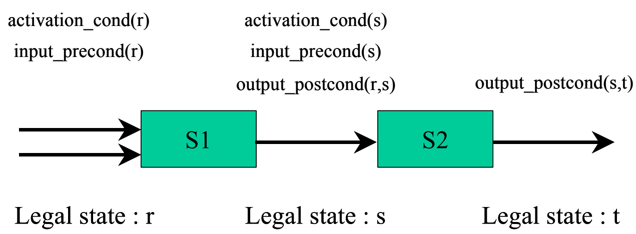

Figure 7. Activation obligation example

We can add semantics with DAML-S or OWL, to represent telecom services and design our connector because as we can see in Figure 7 and Figure 8, we have to describe services and we think that the structure we can find in DAML-S will be helpful to describe them. In order to check properties in the calling paths, constraints in DAML-S allows us to infer new properties.

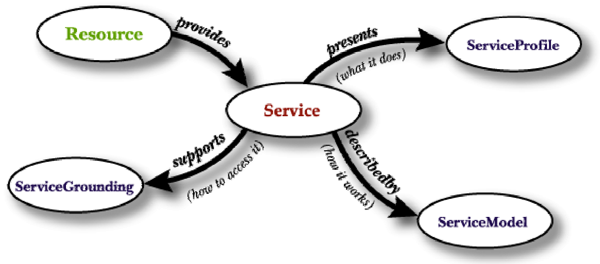

Figure 8. DAML-S Upper Ontology (image from Terry Paine)

We can go beyond, in our example, for instance, we can first describe our service, in DAML-S, it can be seen in Figure 8, an upper ontology for describing services. A service presents what it does in the Service Profile where are defined: input types, output types, pre-conditions and post-conditions as we have seen to be necessary for telecom platform in Figure 7. For example see table 2:

Table 2. Weather Project Service Profile

```
<profile:input>
    <profile:ParameterDescription rdf:ID="Date">
            <profile:parameterName>Date</profile:parameterName>
    <profile:restrictedTo                                    rdf:resource="
http://www.w3.org/2000/10/XMLSchema#Date ">
</profile:ParameterDescription >
</profile:input>
<profile:input>
    <profile:ParameterDescription rdf:ID="Location">
            <profile:parameterName>Location</profile:parameterName>
    <profile:restrictedTo                                    rdf:resource="
http://site.uottawa.ca/Context#Location">
            </profile:ParameterDescription >
</profile:input>
<profile:input>
    <profile:ParameterDescription rdf:ID="Device">
            <profile:parameterName>Device</profile:parameterName>
    <profile:restrictedTo
rdf:resource=""http://www.w3.org/2002/11/08-ccpp-
client#deviceIdentifier">
</profile:ParameterDescription >
</profile:input>
```

Resource is the URI who provides the service, Service Grounding explains how it is possible to access: communication protocol as (RPC, HTTP, …), port number, marshalling/serialization and finally the Service Model describes how it works: process flow, composition hierarchy and process definitions.

5.1. Applicability and use advantages

Once, we find what we are looking for, we negotiate the component. This is done in a transparent way for the user. We employ the same methodology and technologies that we have employed in previous projects about multimedia content negotiation [15, 16].

Thus, we are going to make deals based on offers (see table 3), and counteroffers, then we have been developing an architecture for negotiation with mobile agents which at last requires to make at the end a negotiation with the content provider or services provider, where an agent representing user searches for multimedia content, it can be done in a automatic way using techniques that comes from artificial intelligence as expert systems as Jess (Java Expert Shell System, http://herzberg.ca.sandia.gov/jess/) developed as rules connected to semantic tools also for describing content and components. Customization for different devices where device profiles are performed using semantic tools also.

Table 3. Example of offer serialized as RDF/XML

```
<rdf:RDF xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:ipr  = "http://dmag.upf.es/ontologies/ipronto#"
    xmlns:xsd  = "http://www.w3.org/2000/10/XMLSchema#">
<ipr:Offer rdf:about="http://dmag.upf.es/mars/offer20020211183424">
    <ipr:patient>
<ipr:PurchaseLicense>
    <ipr:licenser rdf:resource="http://www.howlinwolf.com"/>
        <ipr:licensee rdf:resource="http://chiTouristGuide.org"/>
<ipr:permision> <ipr:Access>
    <ipr:place
rdf:resource="http://chiTouristGuide.com/issues/march02"/>
    <ipr:patient
rdf:resource="http://www.howlinwolf.com/imgs/0973.jpg"/>
    <ipr:user rdf:resource="http://chiTouristGuide.com/members"/>
<ipr:timeFrom>  <xsd:date rdf:value="2002-03-01"/> </ipr:timeFrom>
<ipr:timeTo> <xsd:date rdf:value="2003-03-01"/> </ipr:timeTo>
        </ipr:Access> </ipr:permission>
<ipr:obligation> <ipr:Compensation>
        <ipr:payer rdf:resource="http://chiTouristGuide.com "/>
        <ipr:payee rdf:resource="http://www.howlinwolf.com"/>
                <ipr:input><ipr:DollarQuantity
rdf:value="100"/></ipr:input>
        </ipr:Compensation> </ipr:obligation>
<ipr:time>    <xsd:date rdf:value="2002-02-11"/> </ipr:time>
    </ipr:PurchaseLicense>
</ipr:patient>
</ipr:Offer>
</rdf:RDF>
```

6. Results

Analyzing the networks in Figure 5 and Figure 6, we extract the following results.

1) It has been noticed some **errors**, nodes that are not connected, it means that they are not well defined and they can not be well accessed. Ex: *weather-ont.daml#Heavy.Heavy Intensity, weather-ont.damlWeatherDescriptor, or weather-ont.daml#Partial*

2) Ontologies not always are related to other ontologies as upper ontologies. If it is not done, no other connections will be allowed and no new information can be discovered. This must be a parameter design.

3) Nodes are related semantically and they build clusters related each other with paths and belonging to 'small worlds'. Ex: we have discovered a kind of weather report used in the aviation domain, METAR reports.

METAR is the international standard code format for hourly surface weather observations which is analogous to the SA coding currently used in the US. The acronym roughly translates from French as Aviation Routine Weather Report. Therefore, we have the MetarReport ontology class that describes METAR formatted reports.

4) Nodes degree can be normalized and represented, so the hierarchical semantics is showed graphically. This information shows the relevance in connections of every component. For security reasons is important to see where these components are situated in the graph and how important they are. It is the information the degree gives. The size of the node is a measure of that phenomenon.

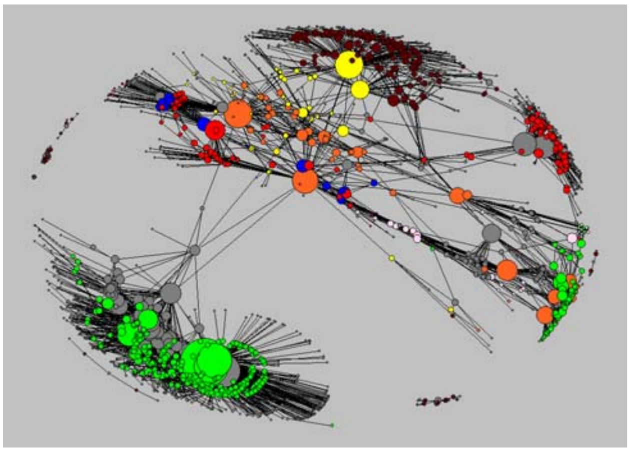

Figure 9. Normalized degree network. Green isi, Red dublincore , Blue purl, Yellow daml+oil , Orange w3C, Brown mindswap Pink ww.daml.org/2001/06/map/map-ont

7. Conclusions

We can summarize them in the following lines:

1) Application components that are services or part of services have to be semantically described to achieve ad hoc applications with no information about where they are (a path directly related to them)

2) Semantically related, it means that application components are members of an ontology. Well-defined ontologies are related to 'upper ontologies', that are the *common point* to establish new relations and discover new knowledge.

3) Semantic relations build a graph, if there are many, it becomes a large graph or large network. Statistical analysis can be applied. Many properties arise such "small world", degree or clustering. This information shows changes in topology, community phenomenon, clustering and even security analysis can be performed. For example, when a component is highly connected, it becomes a vulnerable component, so it is necessary to control how many there are and where they are situated in the graph.

8. References

[1] Linthicum, D. S. Enterprise Integration: An Architecture for Enterprise Application and Systems Integration. John Wiley & Sons; 1st edition (February 1, 2002)

[2] Choukair, Z. and Sfar, S. Run Time Adaptation of UMTS Services to Available ResourcesZ.; Aina March 2003; China.

[3] Choukair, Z. & Sfar, S.:"Design an development of Multimedia Telecomunication Semantics" Technical Report, ENST, June 2003 http://projets.enst-bretagne.fr/pap/pap5k/

[4] Aguirre, C., Martínez, J., Huerta, R., Corbacho, F.: A New Topology for Multi-Agent Collaboration. 1st First Technical Workshop of the Computer Engineering Department, Universidad Autónoma de Madrid (2000)

[5] Zhang, H., Goel, A., Govindan, R.: Using the Small-World Model to Improve Freenet Performance. Proc. IEEE Infocom, 14 (2002)

[6] Adamic, L.A.: The Small World Web. Proceedings of ECDL'99, LNCS 1696, Springer-Verlag (1999) 443-452

[7] Watts, D.J., Strogatz, S. H.: Collective dynamics of small-world networks. Nature 393 (1998) 440-442

[8] Albert, R., Barabási, A.L.: Statistical mechanics of complex networks. Reviews of Modern Physics, 74 (2002) 47-97

[9] Barabási, A.L and Bonabeau, E.. Scale-Free Networks. Scientific American, May 2003

[10] Milgram, S.: The small world problem. Psychology Today 2 (1967) 60-67

[11] Solé, R.V., Ferrer, R., Montoya, J.M., Valverde, S.: Selection, tinkering and emergence in Complex Systems. Complexity, 8(1) (2002) 20-33

[12] Batory, D., Geraci, B.J; Validating component compositions in software system generators, Proceedings of Fourth International Conference on Software Reuse, 23-26 April 1996 Page(s): 72 –81

[13] Succi, G., Wong,.R , Liu,E., Smith, M., Supporting dynamic composition of components, Proceedings of the 2000 International Conference on Software Engineering, 4-11 June 2000 Page(s): 787

[14] Choukair, Z. "COCTELS, A Model to Compose Private Adaptive Bouquets of Services in Telecom Networks"., (JOIN) Special Issue on Advanced Information Networking: Architectures and Algorithms, sept 2003

[15] Delgado, J.; Gallego, I.; García, R. and Gil, R. An Architecture for Negotiation with Mobile Agents Mobile Agents for Telecommunication Applications, MATA 2002. Lecture Notes in Computer Science, Vol. 2521, pp 21-31. Springer-Verlarg 2002

[16] Gil R., García R., Delgado J. Delivery context negotiated by mobile agents using CC/PP. Mobile Agents for Telecommunication Applications, Lecture Notes in Computer Science, Vol. 2881, pp 99 - 110. Springer-Verlarg, 2003