Semantic Security for Devices and Services

Version 08

Project Identifier	WoO
Project Title	Web of Objects
Document Version	V07
Planned Delivery Date	June, 2013
Actual Delivery Date	December, 2013
Document Title	WoO Semantic Security for Devices and Services
Work Package	WP5
Task	T5.1
Document Type	Word
Abstract	Template for the definition of the security language of WoO

Version history

Version nº	**Date**	**Revised by**	**Description**	**File name**
00	1/09/2012	Juanvi Vera	Table of contents and work assignments	
01	14/09/2012	S. Deleu D. Excoffier	First draft	WoO-D5.1 Semantic Security-v01.docx

Version nº	Date	Revised by	Description	File name
02	19/11/2012	Juanvi Vera	Privacy and context management	WoO-D5.1 Semantic Security-v02.docx
03	1/2/2013	Juanvi Vera	Privacy Management (original content by Visual Tools)	WoO-D5.1 Semantic Security-v03.docx
04	7/9/2013	Juanvi Vera	Privacy Management, final version (VT). Security parameters and policy (UPC)	WoO-D5.1 Semantic Security-v04.docx
05	12/11/2013	Juanvi Vera		WoO-D5.1 Semantic Security-v05.docx
06	1/12/2013	Juanvi Vera	Security parameters, security solutions	WoO-D5.1 Semantic Security-v06.docx
07	16/12/2013	Juanvi Vera	General review of the document	WoO-D5.1 Semantic Security-v07.docx
08	20/12/2013	Juanvi Vera	Privacy in smart meter networks by Aurensis	WoO-D5.1 Semantic Security-v08.docx

1. Contents

2. Acronyms

IoT	Internet of Things
DAML	DARPA Agent Markup Language
KIF	Knowledge Interchange format
OIL	Ontology Inference Language
OWL	Web Ontology Language
QoS	Quality of Service
RDF	Resource Description Framework
RIF	Rule Interchange Format
SA-WSDL	Semantic Annotations for WSDL
SA-REST	Semantic Annotation for REST
Sensor ML	Sensor Model Language
SOA	Service Oriented Architecture
SWS	Semantic Web Services
TTP	Trusted third party
URI	Uniform Resource Identifier
W3C	World Wide Web Consortium
WSDL	Web Services Description Language
WSN	Wireless sensor network
WoT	Web of Thing (or Internet of Things)

WSML Web Semantic Modeling Language

3. Introduction

This task will create definitions, properties and relations for objects based on the models that were defined in document D3.3. These points will lead together to the creation of a meaningful representation of the functionalities that objects offer to the network, the interfaces to connect to these functionalities and services, and the constraints that the object imposes to the external clients due to security configuration and privacy settings. The output of this task will enhance the security properties of the Web of Object scenarios by securing features such as service discovery, context negotiation and validation of interface consistency.

The goal of this task is to create or extend a common language that captures the semantic representation of the security requirements in the Web of Objects. Standard binary representation of ontologies, such as RDF(S), may be sufficient to describe static concepts, but in the case of the Web of Objects dynamic aspects also need to be considered. Therefore, extending the representation model with n-ary ontologies may be required to model dynamic concepts such as self-* mechanisms (reconfiguration and adaptation).

This document is structured as follows. Chapter 4 studies the security parameters to be provided in the Web of Objects, including a taxonomy of the attackers, a description of the network architecture and the security services than objects must provide. Chapter 5 explores different security solutions developed inside the web of object projects, namely 1) a system to manage large groups of objects, allowing the creation of clusters of users and fast distribution of security keys 2) an study of the privacy management provided in the WoO project 3) the definition of a secure recommendation system to offer a search service for the objects and 4) negotiation methods for context switching. Chapter 6 explores a semantic language to define the security provided by the Web of Object project. Finally, chapter 7 explores the best security practices to be implemented by all members of the Web of Object project. The deliverable finishes with the conclusions and references.

4. Security parameters and policy

The introduction of REST architecture in the web of objects paradigm has several advantages: in presents a common way to access services and information in remote objects, and then there is no need of using proxy services at the application level; allows the creation of complex network using simple technologies that present an unified way to access local services and data, and thus there is no need of using proxy services at the application level. In addition, it allows autonomous configuration of the objects, which now are smart enough to take their own decisions.

[Garcia06] identifies the lifecycle of an object in the internet of things paradigm. An object starts its life by being manufactured. In this moment, the designers decide the processing power of the object, the protocols it is able to run and the common information it has with other objects by the same maker. Regarding security, this includes common security keys, identity namespaces and possibly light authentication protocols. In this moment, the builders of an object may severely limit the security available for an object. Some of these security mechanisms cannot be changed later.

This chapter analyzes the security parameters of the project Web of Objects, and gives some insight on the actual implementation of the security mechanisms in the demonstrators of the project.

Type of Attackers

Above assets of the web of objects (objects, services, private data, network infrastructure) are potentially the targets of attackers which generally differ in:

- knowledge;
- activity; and
- degree of intrusion.

The ability to carry out a specific type of attacking activity is heavily linked to the cost an attacker is required to spend in terms of equipment to carry out an attack successfully. This can range from extremely cheap, where only a hammer is required, to prohibitively high, where top-of-the-line semiconductor test equipment is needed. A key characteristic of the skills is also if traces are left behind by an attack; if after the attack the node is left in the same state as before the attack, including unaltered memory contents, then this is harder to notice than an attack which causes physical destruction of the node. We will now briefly summarize the properties pertaining to these characteristics, and the thus resulting and typically used class scheme.

Attacker Knowledge

The prior knowledge or knowledge acquired during the stage of the attack can be differentiated as follows:

- **Outsider**: An outsider attacker can only inflict external denial of service or leak of service but no falsification of service. A good anti-jamming combined with a cryptographic system that uses integrity mechanisms, however, is generally able to defend outsider attacks.

- **Insider**: Insider attackers are very dangerous to the functioning of a network, mainly because it is very difficult to identify an insider and/or insider attack. For example, an insider attacker can easily fabricate a false event report to mislead the decision makers, or keep injecting bogus data to cause network outage, etc. Unfortunately, internal attacks cannot be solved by the classic cryptographic techniques solely [19]-[20]. Conventional methods such as encryption, authentication, etc., have the ability to verify the correctness and integrity of an operation, but cannot eliminate insider attacks. An insider attacker typically knows key materials and can therefore easily modify and forward with access to these valid cryptographic keys. An insider attacker detection scheme, possibly through trust management protocols, must be designed to ensure many of the mission-critical applications.

A prime design goal is hence to defend key materials which prevents or minimizes the chances of an outsider attacker to become and insider attacker.

Attacker Activity

Attacking activities can roughly be classified as passive versus active:

- **Passive Attacks.** They extract information from the device merely by observing physical properties of the devices and the communication between them. They typically do not leave traces behind and are generally, but not always, of comparably low cost.
- **Active Attacks.** They involve the manipulation of the devices and/or the information exchanged between them. This kind of attacks requires physical tampering or entering the cryptographic protection.

Degree of Intrusion

Attacks are also classified as non-invasive versus semi-invasive:

- **Non-Invasive Attacks.** They typically do not manipulate the device or the infrastructure at large.
- **Invasive Attacks.** They have practically full access to the device and/or the infrastructure.

Examples of passive attacks are traffic analysis and camouflaging. Most viable attacks, however, are active attacks and are going to be discussed below.

Resulting Attacker Classes

To grasp both ability and activity, we capitalize on a taxonomy on the class of attackers:

- **Class I (funded organizations).** They are able to assemble teams of specialists with related and complementary skills backed by great funding resources. They are capable of in-depth analysis of the system, designing sophisticated attacks, and using the most advanced analysis tools. Some example assets targeted by these attackers are commercial secrets and private data. They may use Class II adversaries as part of the attack team.
- **Class II (knowledgeable insiders).** They have substantial specialized technical education and experience. They have varying degrees of understanding of parts of the system but potential access to most of it. They often have highly sophisticated tools and instruments for analysis.
- **Class III (clever outsiders).** They are often very intelligent but may have insufficient knowledge of the system. They may have access to only moderately sophisticated equipment. They often try to take advantage of an existing weakness in the system, rather than try to create one.
- **Class IV (weekend hacker).** They are often very skilled but only look for some challenge to expose security weaknesses of a system. They typically constitute a threat when grouped and connected to coordinate any form of attack.

Class II and III attackers typically weigh the tradeoff of cost of staging an attack and the potential benefit this would yield. Any security design ought to ensure that the cost of staging any type of attack is as large as possible.

Network architecture

This section explores the network architecture of the Web of Objects.

Centralized Architecture

A centralized architecture implies the availability of a single (or a few) entities which have control over the entire network. A typical embodiment of a centralized approach is shown in Figure 1. Note that a centralized approach typically means one-hop connectivity to all network members, but in the context of short-range embedded systems is typically realized via a multi-hop network.

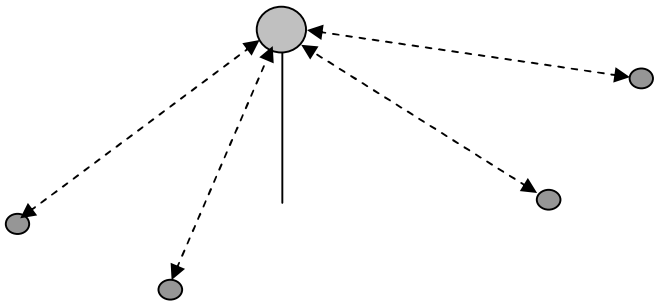

Figure 1: Embodiment of a centralized embedded architecture.

From a performance point of view, centralized approaches are known to be superior to other approaches, given that a) optimum algorithms are used and; b) the central entity has complete knowledge of the entire system. The latter, usually comes along with a large overhead which needs to be gauged against the performance gains.

From a security point of view, this single centralized entity needs to monitor the safety of the entire network. The utmost important task here is, if needed, to generate and to distribute security keys to all the members via a pair-wise secure channel established with each member. This requirement might generally be too stringent for embedded multi-hop systems since a central key server must be continuously available and present in every possible subset of a group in order to support continued operation in the event of arbitrary network partitions. Continuous availability can be addressed by using fault-tolerance and replication techniques; unfortunately, the omni-presence issue is difficult to solve in a scalable and efficient manner.

Hierarchical Architecture

A hierarchical architecture implies the availability of clusters, which is controlled by a cluster head which communicates to its associated node members. A simple node is typically but not necessarily associated to a single cluster head. Communication between node and cluster head is typically but necessarily done in a single hop. Cluster heads typically communicate with each other by means of a flat architecture, or via another hierarchical tier, or via a central entity. Hierarchical networks can be heterogeneous, i.e. cluster heads (super nodes) are more powerful than simple nodes, or homogenous, i.e. cluster heads and associated nodes have a complexity of approximately the same order of magnitude. A typical embodiment of a hierarchical approach is shown in Figure 2. Note that a hierarchical approach is essentially a hybrid between a centralized and flat architecture.

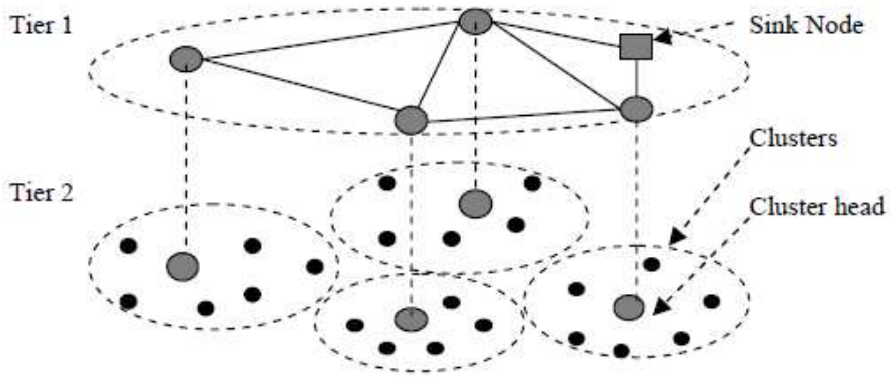

Figure 2: Embodiment of a hierarchical embedded architecture.

From a performance point of view, hierarchical approaches offer the advantage of trading the pros and cons of the centralized and flat architectural approaches. Notably, the overhead related to the knowledge of the cluster state is diminished, at the expense of a sub-optimum performance when compared to centralized approaches.

From a security point of view, the load of key management is now distributed among cluster heads. Typically, each cluster head would now generate and distribute keys only to its associated members. The obvious advantage is that no single point of failure is present and the problem of omni-presence as well as scalability hence diminished. The disadvantage is that more points of attack and failure are created since cluster heads are easier compromised than some centralized security entity.

Flat Architecture

A flat architecture implies that all nodes of the network are equal from a networking point of view and also typically but not necessarily have the same processing capabilities. Short-range embedded systems typically require the presence of multiple hops over such a flat architecture until the sink node or gateway is reached. A typical embodiment of a flat approach is shown in Figure 3.

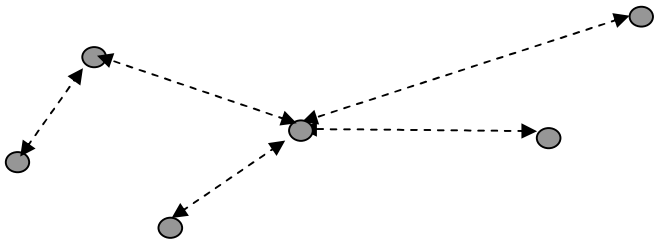

Figure 3: Embodiment of a flat embedded architecture.

From a performance point of view, flat architectures are known to be fairly poor. Notably, the entire MANET community has been designing protocols for these types of architectures without any viable commercial solution being on the table today. However, when the network is not very large, then flat approaches were shown to perform sufficiently well.

From a security point of view, the flat peer-to-peer architecture requires that every member contributes an equal share to the common group secret, computed as a function of all members' contributions. This is particularly appropriate for dynamic networks since it avoids the problems with the single point(s) of trust and failure.

Standard security protocols

Security protocols are heavily supported by the following security algorithms:

- **Symmetric Cryptography.** It implies that both the origin and destination share the same security credential (i.e. secret key), which is utilized for both encryption and decryption. There are two types, i.e. stream ciphers which are simpler and faster; and block ciphers which are more flexible and powerful. Typical symmetric block cipher algorithms are Skipjack, RC5, AES and Twofish; a typical symmetric stream cipher algorithms is RC4.
- **Asymmetric Cryptography.** It is a form of cryptography that uses two keys: a key called secret key, which has to be kept private, and another key named public key, which is publicly known. Any operation done with the private key can only be reversed with the public key, and vice versa. Typical asymmetric block cipher algorithms are Elliptic Curve Cryptography (ECC), Rabin, NTRU and MQ-Schemes. There are no known (secure) asymmetric stream cipher algorithms.
- **Homomorphic Cryptography.** It is a form of encryption that allows arithmetic operations on the plaintext by performing a specific algebraic operation on the ciphertext. These schemes are especially useful whenever some party not having the decryption key(s) needs to perform arithmetic operations on a set of ciphertexts, such as in the case of the data aggregation in the objects of the WoO scenerio.
- **Hash Functions.** Cryptographic hash functions or hash primitives are utilized in order to compress a set of data of variable length into a set of bits of fixed length. The result is a “digital fingerprint" of the data, guaranteeing integrity, identified as a hash value. A typical algorithm is SHA-1.
- **Digital Signatures.** It is a public-key method to verify the authenticity of a received data from the peer by using a pair of keys, a ‘sign private-key’ and a ‘sign public-key’. Only the device knows its sign private-key whereas the sign public-key is distributed to all the communicating devices. Typical algorithms used for facilitating digital signatures are RSA, DSA and ECDSA.

Security Services

Above-described threats need to be counteracted by suitable security services. Invoking the often-used CIA security model, the following high-level services can be identified:

- Confidentiality
- Integrity
- Availability
- Authentication and authorization
- Group communication and key management

- Privacy

We will now detail these services and briefly summarize their properties.

Confidentiality

Confidentiality essentially means keeping information secret from unauthorized parties. A sensor network should not leak sensor readings to neighboring networks or adversaries. The confidentiality objective is required in sensors' environment to protect information within the nodes as well as traveling between the sensor nodes of the network or between the sensors and the base station from disclosure, since an adversary having the appropriate equipment may eavesdrop on the communication. By eavesdropping, the adversary could overhear critical information such as sensing data and routing information. Providing confidentiality at large typically implies the following security services:

- **Data Secrecy.** The data content should generally be kept secret. This prevents unauthorized nodes to obtain access to the information. It typically involves encryption algorithms.
- **Forward and Backward Secrecy.** As per [10], as new sensor nodes can be deployed whenever other sensor nodes fails, there are two properties that need to be considered: forward secrecy, where a sensor should not be able to read any future messages after it leaves the network, and backward secrecy, where a joining sensor should not be able to read any previously transmitted message. These properties may not be important in certain scenarios, where there is no need to hide the contents of the network from old nodes and new nodes authorized to perform the same tasks as their partners. However there are other scenarios where these properties must be taken into account, such as in networks with nodes that must be authorized to perform certain tasks.
- **Code Obfuscation.** It is a mechanism that allows the protection of a valuable piece of information (e.g. the security credentials) contained inside the node. By obfuscating the code and data, the amount of time needed by the attacker to analyze the compromised nodes will increase, thus it will be more difficult to deduce the secrets from the extracted contents of program hash, the EEPROM (memory transistor) or the SRAM (memory transistor). The obfuscation methods must not be equal for all the nodes. This is to prevent the attacker from using the same method to retrieve the secrets once he/she is successful in compromising one node.
- **Resilience.** It refers to the ability of the network to minimize (or eliminate) any information available to an attacker once the node and/or network is compromised. High resilience means that the accessed information has either been purged or is of little use to the attacker.
- **Key Management.** To facilitate above security services for maintaining confidentiality, some key management schemes are typically needed. It is used in symmetric encryption schemes, etc. It is important here to protect the keying material through suitable key protection mechanisms, such as code obfuscation.

Integrity

Integrity means that the data produced and consumed by the sensor network must not be maliciously altered. Unlike confidentiality, integrity is, in most cases, a mandatory property. The wireless channel can be accessed by anyone, thus any peer (outsiders and insiders) can manipulate the contents of the messages that traverse the network. Even more, data loss or damage may occur due to the harsh communication environment, and in the worst case the network will accept corrupted data. As the main objective of a sensor network is to provide services to its users, the sensor network will fail in its purpose if the reliability of those services cannot be assured due to inconsistencies in the information. Providing integrity at large typically implies the following security services:

- **Authentication.** An adversary can easily inject messages, so the receiver needs to make sure that the data used in any decision-making process originates from the correct source. As in conventional systems, authentication techniques verify the identity of the participants in a communication, distinguishing in this way legitimate users from intruders. In the case of sensor networks, it is essential for each sensor node and base station to have the ability to verify that the data received was really sent by a trusted sender and not by an adversary that tricked legitimate nodes into accepting false data. If such a case happens and false data are supplied into the network, then its behavior could not be predicted, and most of times the mission of WSN will not be accomplished as expected. However, authentication for broadcast messages requires stronger trust assumptions on the network nodes.
- **Authorization.** It implies that only authorized entities (sensor nodes and base station) can be able to perform certain operations in the network (e.g. information providing, controlling the system). Since a sensor network can be considered as one single entity, where all nodes perform the same tasks and acknowledge the role of the base station as manager and supervisor, it could be supposed that any authenticated device is inherently authorized to perform its tasks. Nevertheless, there might be situations (e.g. when nodes actuate over physical systems) where some members of the network need to have a proper authorization in order to perform certain tasks. Is in these situations where authorization must be taken into account.
- **Freshness.** Some services require the data to be recent. This is an important security requirement to ensure that no message has been replayed meaning that the messages are in an ordering and they cannot be reused. This prevents the adversaries from confusing the network by replaying the captured messages exchanged between sensor nodes. To achieve freshness, security protocols must be designed in such a way that they can identify duplicate packets and discard them preventing replay attack.
- **Code Attestation.** Software based attestation enables a third party to verify the code running on the system to detect any maliciously altered code. Usually code attestation is done through the use of special hardware mechanisms proposed by the Trusted Computing Group (TCG) and Next Generation Secure Computing Based (NGSCB). However, these hardware mechanisms are costly and are not implemented in typically available nodes. Thus this kind of software attestation is designed to provide the detection of malicious code alteration and verify that the nodes are using the correct codes.

- **Key Management**: Above integrity services typically require the availability of keys, which can but not necessarily have to be different from those used for confidentiality. Therefore, a proper key management service needs to be in place to guarantee a proper functioning of the security services. Again, key material needs to be sufficiently protected.
- **Trust**. The basic tasks of a trust management system are determining initial trust, observing the trustee's actual behavior and updating trust accordingly [7]. Usually, trust management systems can be classified into two categories: credential-based trust management systems and behavior-based trust management systems. This classification is based upon the approach used in order to establish trust among the peers of a system.

Availability

Availability implies that the users of a web of objects must be capable of accessing its services when they need them. For example, when a WoO is used for monitoring purpose in manufacturing system, unavailability of nodes may fail to detect possible accidents. Availability ensures that sensor nodes are active in the network to fulfill the functionality of the network. It should be ensured that security mechanisms imposed for data confidentiality and authentication are allowing the authorized nodes to participate in the processing of data or communication when their services are needed. As sensor nodes have limited battery power, unnecessary computations may exhaust them before their normal lifetime and make them unavailable. Sometimes, deployed security protocols or mechanisms in the WoO are exploited by the adversaries to exhaust the sensor nodes by its resources and makes them unavailable for the network. Providing availability at large typically implies the following security services:

- **Security Policies**. Such policies should be in place so that sensor nodes do not need to spend extra computation or do not try to allocate extra resources for security purposes.

- **Self-Organization**. As per [8], there is no fixed network infrastructure as WSNs are typically forming in an ad hoc fashion. Therefore, nodes must have the self-organizing and self-healing capability to support multi hop routing, frequency hopping MACs, etc. The self-organizing security protocols should e.g. support efficient key management schemes so that sensor nodes organize themselves according to the key distribution and can build trust relations with the neighbor nodes and secure virtual infrastructure as well.

- **Non-Repudiation**. It involves providing some ability to allow traceability or network management review of participants of the routing process including the ability to determine the events and actions leading to a particular routing state. Non-repudiation relies on the logging or other capture of on-going routing exchanges. Given the limited resources of a node and potentially the communication channel, and considering the operating mode associated with embedded systems, transaction logging or auditing process communication overhead is not be practical; as such, non-repudiation is not further considered in this document.

Authentication

Authentication involves knowing the identity of the entities that participate in the communication. In this context, having a definition of identity shared by every party of the communication and some map between identities and participants, we say that an entity is authenticated by others if the authenticated party can prove that she is the only entity in the network with some specific identity. Typically, one of the parties decides before the communication the identity she wants to contact to. During a communication exchange, none of the parties could be authenticated, only one or both of them.

It is not only necessary to identify and object, but provide the mechanisms necessary for other objects to test and verify this identity. For example, it is not enough to provide an ID card if the issuer of the ID card cannot be identified.

Several mechanisms exist to authenticate objects in a distributed environment.

- **Certificates**. A party trusted by all participants of the communication (trusted third party, TTP), issues signed certificates, small pieces of data, to the parties that wish to be authenticated. These certificates rely on public/private keys. The certificate includes the public key of the entity A and some information about her identity, as its internet address, and it is signed by the TTP. Any entity B receiving this certificate can use the public key of the entity A, encrypt some data and she knows only A can decrypt these data. If A proves she can decrypt the data, B authenticates A.

- **Username/password**. The entities A and B agree on some username/password pair. This pair is kept secret. If A is able to present the pair to B, B authenticates A.

- **Tickets**. Similarly to username/password, a ticket is a small piece of data that an object and a TTP agree. The ticket includes the signature of the TTP and some mechanism to limit their timespan. Any entity in the network presented the ticket authenticates the object A.

In a typical web of objects, entities are planted in the scenery and configure automatically to discover other objects and share data with them. One of the main challenges this autonomous configuration and deployment must face is how to limit the access to the sensitive information only to those entities, users and objects that have the suitable rights. These rights may have changed after the object in planted in the field, for example, because there is a new resident at the house.

Object identification and authentication

The first step to authenticate an object involves the creation of identities that can be assigned to the objects. In this sense, any other object in the network can decide whether or not a specific

entity is allowed to access a service or download some data. This section covers some special cases for identities:

- Group identities
- Recommendations

Group identities

Sometimes, identifying a single object and separate it from other similar objects is not necessary or desirable. For example, several objects in the same area and owned by the same user, like the objects in a home network, may prefer some way to show some common identity to the external world. In this case, objects external to the house cannot put apart a door sensor from a camera, being all "Alice's devices". In a similar way, objects may show different entities in different contexts. For example, a mobile phone could share “Alice’s identity” while at the mall, but present a more specific “mobile phone identity” when at home. The management of these entities and how objects identify other objects according to the context is something the semantic description must capture. For example, they may present different profiles depending on the identity of the object that requested them.

These common identities must be, in turn, verified. Indeed, if a camera at Alice's home is not using its "camera identity" but a more generic "Alice's object", an external object must be able to verify if this identity is true. Hence, identity verifiers must cope with hierarchical identities and complex descriptions.

Recommendation systems

When users of the web of objects join a network, they make lots of decisions about the objects in the environment: which personal objects must be available to the rest of the network, which objects already in the network must be accessed. That is, the user of a web of objects identifies his objects and uses the available objects according to his needs.

An automatic decision of the most suitable object available in the network needs some semantic description about the user needs and the kind of service that objects offer. This way, the user may identify objects and select the most suitable. In order to make these decisions, the system must provide some mechanism that captures the semantic description of the users’ needs and the capabilities of the objects in the network. In addition, the system must provide a smart system that takes these semantic descriptions as inputs and output the objects the user must access.

Gathering all the information to make a well-grounded decision is a very time consuming process. Recommender systems appeared to assist the user in quickly making the right decision and

saving time and maybe money. The additional intelligence that semantic descriptions set on the web of objects networks makes possible to approach service discovery systems as recommender systems.

The process of receiving a useful recommendation begins with the creation of a view of the user that contains their interests. In the context of a recommendation system, the users' profiles are their identity. Thus, user's profiles include sensitive information which captures the personal description of a particular user. Protecting the users' privacy is not only a necessity for users, since it can improve the result of the recommendation process. Indeed, if users are not afraid of declaring their likes and dislikes, the recommendations that they get from the system will be more accurate. Protecting the user's privacy is not the only security service to provide in recommender systems. In a distributed environment, other actors of the system may need additional protection. The providers of a recommendation, for example, expose their own opinion of the resource that they are recommending. Thus, recommenders should be protected in the same way than users. Furthermore, recommenders and other participants that assist in the recommendation process may be, even unknowingly, committing a copyright infringement. The risk of being prosecuted may affect the quality of the output of the system, for example, preventing the recommendation of a certain movie even if the recommender thinks that it is the most suitable for the user. This is a new kind of legal attack, and providing protection for this attack may improve the quality of the recommendations.

We formalize next the steps that the web of objects takes for providing a recommendation about the most suitable object. We aim for a general description of a recommender system, trying to fit many different types within the same structure. Thus, actual systems may reduce some of these steps to a simple process while others perform complex tasks inside them.

1. **Document collection and profiling**. During this step, the web of objects collects and identifies the objects that it is going to offer to the users. For example, in a centralized system, the object owners insert the semantic description of their objects in the local database. If the recommendation is based on the description of the objects and not only the description of the user accessing the object, this step includes the creation of a profile that captures the defining characteristics of the object. This is the most common case in the web of objects scenario.

2. **User profiling**. During this step, the object owned by a user enters the web of objects. In this moment, a user profile is created and assigned to the user. This profile could be controlled by the user, for example, if it is the output on the answers of a test or a self-configuration of her needs. The profile can also be created by an external observer by means of the analysis of the user behavior. This is the case of profiles that involve the

study of the buying habits of the users. In any case, the information that is included in the user profile is highly sensitive and the system must provide mechanisms to protect and secure this profile.

3. **Recommender selection**. The web of objects may have many entities that register and are able to select the most suitable object, service or item after a recommendation request. For example, some recommenders may be specialized only on some object categories, or being specific for some context. During this step, the web of objects identifies and selects those recommenders that are more suitable to answer the query of the user. For example, the number of users and objects in a shopping mall may be in the order of thousands. In order to manage this amount of information, an initial classification of objects according to some criteria (proximity, affinity…) takes place. In a social network, participants often select an initial set of "friends" or "similar people" that can be used to make recommendations. We can map this behavior to the web of objects.

4. **Query the system**. During this step, objects send a query to the system that includes a semantic description of the object she is interested in. The complexity of the process of querying the system varies with the different recommender types. For example, this is a very simple process in a centralized repository shop, since it is reduced to sending a message to some database. In distributed systems, on the other hand, this step involves routing the query to the selected recommenders and it may be a complex task. As in the case of users' profiles, the query of a user to the web of objects includes sensitive information that must be protected.

5. **Recommendation process**. The selected recommenders search their internal databases to select those objects that are more suitable to answer the query of a user. Then, the recommenders return a set of objects they believe that are interesting to the user. At this point, we find useful to imagine a recommender system as a system where users evaluate objects. In this case, we can model the knowledge of the system as a matrix, where rows are users and columns resources. Figure 4 shows this model. An element of the matrix rij is the rate that a user i gave to a resource j. This matrix is scarcely populated and most of the elements are empty, since it is usually impossible for users to evaluate a significant subset of the available resources. The goal of the recommender system is making a good guess of the rate that a user would give to a resource that is not yet evaluated. Given these guesses, the recommender decides whether a resource interests the user or not with an algorithm that is often as simple as "the document is interesting if its calculated rate is higher than a threshold λ". The mechanisms that are used to populate the elements of the matrix, the input that the recommender needs for guessing rates and the actual location of

the matrix in the system are the main differences between the different implementations of real recommender systems.

	Doc 1	Doc 2	Doc 3	Doc 4
User A	0.9	-	-	-
User B	-	0.5	0.8	0.5
User C	0.2	-	0.8	-
User D	0.1	0.8	-	-

Figure 4: A recommender as a matrix

6. **Accessing the recommended objects**. During the final phase of the system, users access the recommended objects. The final output of the process may be useful to enhance future recommendations, and hence some feedback mechanism can be included. This is the case of user profiles that are based on buying habits. From the security point of view, an access to a resource implies that a recommendation was correct, and since it tells something about the user profile, this is a security leakage. Even if the other steps of the process are conveniently protected, an attacker may learn something about a user's profile by means of inspecting only the objects the user accesses. A system that aims to protect the user's privacy must consider the final access to the objects as part of process to process.

Authorization

A offers the security service of authorization if only authorized entities can be able to perform certain operations. Usually, authorization implies authentication: the object has a list of identities authorized to access a service or data. If any entity is authenticated using one of these identities, then it is authorized to access the asset. Some objects may be authorized just because they show some shared identity. This is the case of a home scenery: all objects inside the house could be authorized to access all resources inside the house. Authorization may include some mechanism to allow entities to act on behalf of another, authorized entity.

OAuth

OAuth is a protocol built on HTTP to authenticate a proxy of a user as an authorized party to act on behalf of the user. For example, users (in this case, the owners of a mobile device) may authorize an external object to access their private data in a specific server. The simplest approach is providing the username and password to the object, but this is undesirable in many contexts. For example, if the object is not controlled directly by the user but by an external party. OAuth is used in this context.

One of the most extended uses of OAuth is accessing personal data in the social networks by third party applications. For example, a photo application installed on our mobile phones may send our photos to Facebook. If the user does want to trust the developers of the application whit his Facebook username and password (and he will be a very sensible user, if he doesn't), the application must implement OAuth. In this case, the application requests the user and Facebook permission to publish new data on behalf of the user. After the protocol takes place, the application is allowed to publish photos in the user's timeline without knowing the username and password of the user, only some security tokens. The user can at any moment revoke this permission. Google, Facebook, Twitter and many other social networks use OAuth to protect the users' private data.

OAuth defines the following roles:

- The data **owner** is the ultimate owner of the data. For example, the user of the web of objects.
- The data **server** is the database where the owner stores his data. For example, it may be a database, a server in the internet, or a trusted object in the network.
- The **client** is the object accesses these data, to insert, remove or modify items. For example, an application running on the owner's mobile phone, or an object in the network that gives some kind of service to the owners of the data.

In addition, these assumptions are taken:

- The client acts on behalf of the user in the server.
- The owner does not trust enough on the client to provide the authentication tokens that the uses in the server.
- The server knows the identity of the client prior to the operation of the protocol. Developers of new applications register their clients in the server prior to the distribution of the clients.

- The owner must authorize the client at least the first time the client access his data.
- The owner will be able to revoke the authorization of the client at any moment. For this moment on, the client won't be able to access the owner's data in the remote server. The reader will notice that the client is still able to access the data if it stored a local copy of these data.

The authentication process makes a server authenticates owners and clients. When first contacted by a client, the server authenticates the client and asks the owner to authorize the operation. If the owner is authenticated and authorizes the operation, the client receives a ticket. This ticket will be valid as long as the owner desires. For subsequent operations, the client just presents this ticket and the operation takes place without contacting the owner. This process is shown in detail in Figure 5.

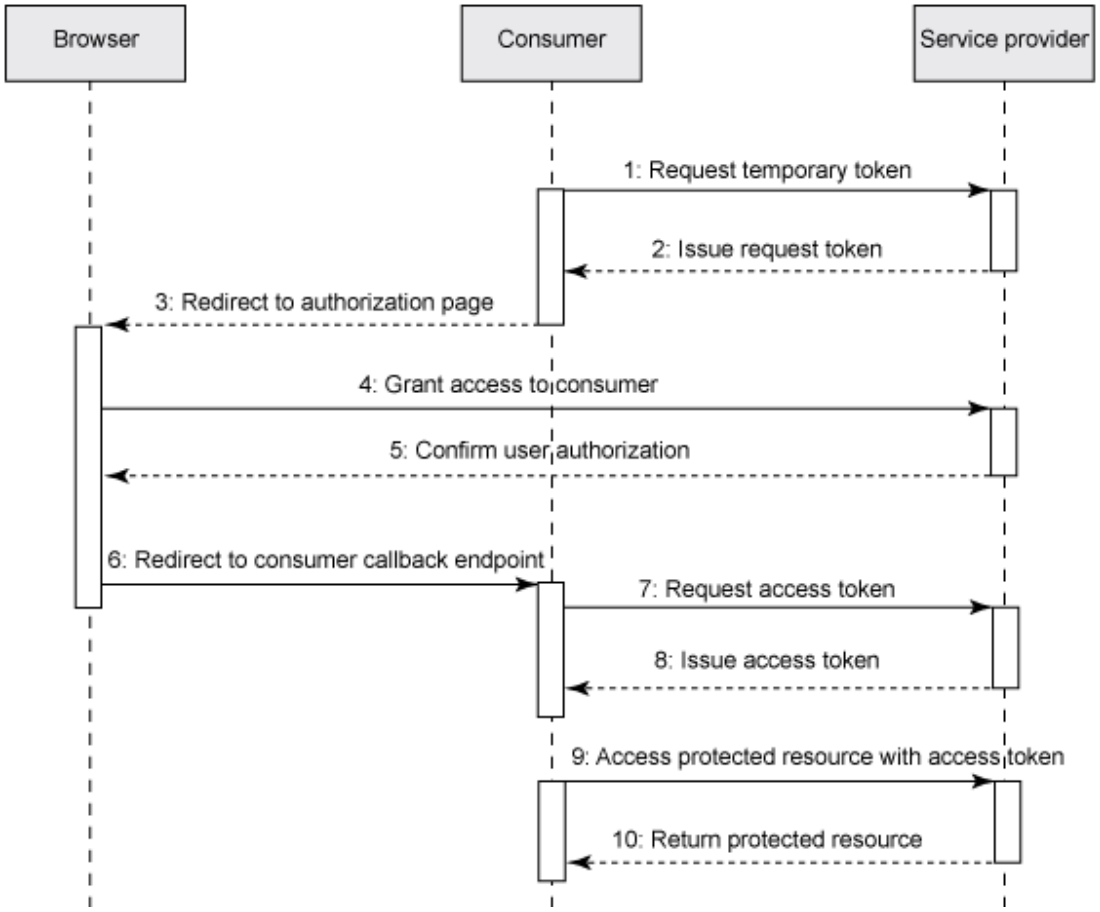

Figure 5: The OAuth authentication process

The complete OAuth work-flow requires from the owner many interactions before being able to access to her data the first time. For example, the user must introduce her username/password, grant access to the client, write down the PIN number and enter this large number into the client. To provide some usability to the process and avoid all these interactions, many servers accept a simplification of the protocol that they call xAuth. In xAuth, the owner gives her username/password to the client, which performs the authorization process by its own without any further action from the owner. After agreeing with the client's token, the specification of xAuth requires that the client forgets the owner's username/password but there it not any mean to enforce this. As the reader will notice, a malicious client may store the owner's username and password. This pair can be sent to a third party in the background without the owner never noticing.

We believe that, in spite of the apparent simplicity of the xAuth process, it should never be used neither by the client developers nor by the data owners. For security reason, a client never should ask for a username/password and we mustn't introduce them. Unfortunately, the usability improvement seems to be reason enough for many server and developers to use xAuth. It is our opinion that xAuth must never be used in the Web of Objects paradigm.

Group communications and key management

After the object selection, it is necessary to establish a secure communication channel between users and objects where no attacker can eaves-drop, modify, replay, or inject messages on. This is what is called group security, which is targeted to provide group privacy, since data should be protected just from outsiders, and group authentication, since the only sources of communication should be the members of the group. In order to achieve this goal, every member knows a set of keys that are usually classified as: keys shared by two nodes or pairwise keys, and keys shared between group nodes or groupwise keys. Pairwise keys allow secure routing by hop-by-hop encryption and authentication, and provide easy isolation of a kidnapped member since compromised keys are just not used anymore. On the other hand, groupwise keys allow secure routing without the need of costly hop-by-hop encryption but providing hop-by-hop authentication and integrity (checking message authentication codes at every link). However groupwise keys are not resilient against node kidnapping and thus must be updated whenever a member is compromised.

Nevertheless both types of keys are part of a vicious circle if asymmetric cryptography is not used: in order to securely agree pairwise keys, nodes need a pre-shared secret that it is often the groupwise key; and in order to securely update a groupwise key, secure communications must be provided often by means of non-compromised pairwise keys.

Once the group of objects is created, they may use a shared security key to provide the next security services:

- **Confidentiality**. If all members of a group share de same key, they can use this key to encrypt communications and they won't be accessible by any member in the group.
- **Backward and forward secrecy**. If the security key is updated each time objects joins and leaves the network, new objects won't have access to past communications and old objects won't be able to access to the new data in the network.
- **Authentication**. An object can authenticate another object as member of the same group if they can prove that they know the group key, for example, using a challenge-response mechanism.
- **Basic authorization**. Depending of the security policy, it may be adequate if objects of the same group are able to access resources inside the group. In this sense, authenticate an

object is equivalent to authorize the object. This is the case of most deployments of objects at a home.

Service Timeline

Above services are not always offered but rather occur at very specific periods during the network lifetime:

- **Pre-Deployment Phase**. Prior to deployment, the services of code obfuscation and some pre-programmed key management can be provided. This, however, will not be further considered here as it depends on the manufacturing process. This phase includes any pre-shared security parameter, as static security keys.
- **Start-Up Phase**. During this time, all security services are invoked which guarantee a safe ramp up of all networking activities. Typically, the following services are invoked: initial key management; authentication; authorization; establishment of security policies; suitable self-organization; etc.
- **Run-Time Phase**. This is the ideal state the network ought to be in as long as possible where only the following security services need to be provided: forward, instantaneous and backward secrecy; freshness; authentication; code attestation; and trust management.
- **Adaptation Phase**. This phase is entered if changes in the network are detected, such as detection of new nodes, compromised nodes, malfunctioning, regularly triggered overhaul, etc. This phase involves a re-initialization of a subset of the services provided at start-up, such as local re-keying; authorization to new nodes; self-organization in the areas where problems have been detected; etc.
- **Disposal phase**. When an object leaves the WoO, the rest of the objects must adapt to the disappearance. For example, other objects in the network may provide services and data the leaving object provided. Furthermore, some security items may change. This is the case of the security keys. Finally, a protocol must exist to ensure that the sensitive material the disposed object may contain is saved and removed.

In the next chapter, we explore how the Web of Objects provide some of these services. The chapter is going to be especially focused on the new services under study in the Web of Object project.

5. Security solutions in the Web of Objects

In this chapter, we explore the security solutions proposed during the development of the Web of Objects project. Specifically, we focused our efforts on these specific issues, since they have been identified as novel in the scenery under study:

- Group management
- Privacy management
- Recommendation systems
- Context negotiation and dynamic scenarios

The partners of Web of Object are aware these are not the only security services that must be provided by the objects in WoO. Chapter 4 explored some traditional services and existing solutions for its provision.

Group management

In this section, we explore how object can create groups depending on their interests, and how these groups can be secured.

Groups of interests

The web of objects aims to the creation of a network of autonomous entities that join and leave without the intervention of a human operation. These networks may include thousands of services, resources, users and objects

Social networks are a kind of P2P network where objects create links according to their similarities. Indeed, the fact that a link exists in a social network means that two nodes conclude that they share some common features. In these networks, users share their profiles, including their likes and dislikes, and ask the community for tips on other objects that may be of interest to them. Two objects that share a link in a social network probably also have common interests and similar likes. In a social network where objects link with similar objects, the problem of searching for resources and services boils down to asking for recommendations from a small neighborhood.

One desirable characteristic of the mechanisms to create a social network is that they must be fast and efficient. That is to say, the social network must help users to locate other users that are similar to them, not only when the user joins for the first time, but in every moment of the process.

Clusters of similar users also have an impact from the point of view of the privacy of the user. If a communication arrives from a cluster of similar users and the recipient cannot put apart the source of the communication inside this cluster, then the anonymity of the source (using the k-anonymity concept as a definition of anonymity) is preserved.

In this section, we aim to improve the performance of the web of objects based on the creation of a social overlay on top of an unstructured P2P network. Objects are clustered according to their affinity. For this approach to be successful, it requires fast identification and location of clusters or other users that are similar, and an efficient construction of these clusters. On this social network, other complex services based on similarities are easily deployable.

The objective of this section is that given a query by a user and an affinity function (based on threshold), it is possible to efficiently find objects in the network that are affine to the query. In this context, efficiency means to retrieve as fast as possible the set of affine objects.

Joining the network

When a new object joins the network, it contacts to a random object that was previously inside the network and requests to “search users that are similar to me”.

In general, we assume that an object enters the network with the ability of finding random nodes that are already members of the network. Actually, real implementations of social networks rarely start with random links. Often, new objects link initially to those who are their already linked in real life, as other objects owned by the same person. Creating links to objects that are initially affine, as real social networks do, improve the performance of the joining mechanism during the initial phase.

The initially contacted objects check first whether they are affine with the joining object or not. Next, they check if a maximum number of hops has been reached. If not, they forward the query to a selected subset of objects in the neighborhood, and the process goes on until the maximum number of hops is reached.

In order to select the subsets of neighbors that will receive the message, objects use an epidemic algorithm. Nodes are chosen based on three parameters: a number of more similar nodes, a number of less-clustered nodes and a number of nodes taken at random. These are parameters M1, M2 and M3. All of these parameters are important for the epidemics of the route to aid in the spreading of the message from the first node. First, the similarity criterion through parameter M1 enables finding objects that are affine in the social space. Meanwhile, nodes that are alike in the social space are likely linked to each other, and thus they have a high relative clustering coefficient. Therefore, through parameter M2, the node sends the query to nodes that are not in the same cluster as the user, disregarding their similarity. This rule has been less explored in the

literature. The intuition that supports the use of M2 is that it enables to arrive to nodes of the social space that share some interests but do not yet belong to the same cluster as the user. Nodes with a low clustering coefficient are, therefore, shortcuts to other clusters in the network. Finally, the criterion of random nodes through parameter M3 is used in many random-walk and epidemic protocols, and it was be evaluated in our research as well. It is the main rule during the first steps of a new object in the network, since they don't know anybody affine and will try random links at first.

If the joining object gets an answer from node affine to itself, it creates a link to this object. During the development of this project, we discovered that objects can limit to the number of linked elements: if a new affine node is found and there is no room in the neighborhood, the less-similar node is removed to make room for the new one. Limiting the number of links in the network has the desirable effect of enhancing the performance of the network and set a bound to the number of messages. As a consequence, a low number of links while still maintaining the usefulness of the network is a desirable property.

After a round of these algorithms, the new node will have in her neighborhood either nodes that are alike to them, or the same initial of random objects.

Browsing the network

After the joining phase, objects should run a new algorithm periodically in order to find new nodes that are affine to them. Networks are dynamic and links are created and destroyed all the time. If an object is not able to find another user that is affine at a given moment, it may be possible later on. In this regard, there is a “slow joining period” before the maximum performance in the network is reached. During this period, users are learning about the structure of the social network. This algorithm is similar to the one run during the initial phase, but now objects are not searching for similar objects but for the desired data and services. This searching mechanism, run by a service location protocol, allows the creation of new interesting links.

When an object receives a service query from another object and they are able to offer the service or data, they test whether or they are linked. The algorithm to search for services in the ad-hoc service discovery protocol uses the same parameters M1, M2 and M3 described in the previous section.

All these mechanisms are explored in depth in reference [Vera2012-2]. In this document, we analyzed how epidemic algorithms can be used to locate interesting objects in a social network where links depend on the user's preferences. The substrate of the network structure is created based on the affinity between the users' profiles, with enough random links to induce a small-world

behavior. From this point forward, we can create new links and discover new user communities by taking advantage of the search results and the small-world behavior of social networks.

In [Vera2012-2], we showed that the criteria of similarity, clustering and randomness do not improve enough the results of the epidemic algorithm. Hence, we proposed a SoftDHT, a structure of sample user profiles that aids in the location of islands of similar users that were not identified at first. In addition, we discussed that there is little gain if a node links to another node that already belongs to a highly clustered affinity group. It may be more useful to limit the number of neighbors and include links to other less clustered, external nodes. In this case, external and unknown groups of users that share the same interests but have not being discovered yet can be found.

We performed throughout simulations, and we found that the proposed enhancements improve the performance of basic epidemics algorithms in dynamic scenarios and shortens the convergence time, while having a comparable performance in the long run. In addition, we tested the network structure of the basic searching algorithm and the improved version, and found that the improvements aid in the creation of a network that shows the desired small world behavior.

Simulations used networks of up to tens of thousands nodes. In the next section, we explore how networks of this size can change their security parameters fast and efficiently.

Group key management

In Chapter 4, we analyzed how group key management can be used to provide standard security services in groups of objects. In this section, we explore how the group key management can be implemented on a large dynamic network, where objects are constantly joining and leaving. The work in this task have been done in collaboration with the ITEA2 project DiCoMa [DiCoMa]

Group key management (GKM) studies the generation and updating of afore-mentioned keying material during the whole group life thus warranting that only the current group members can authenticate and understand or decrypt messages within the group. We are certain that the application of known GKM techniques can secure the exchanged critical data while incurring a low impact on the network performance. However, many wireless sensor networks (WSN), such as the unattended WSNs (UWSNs), preclude the fixed presence of a centralized data collection point, which usually manages the group security. Within this unattended nature, a secure distributed cooperation framework for sharing data, resources and/or services between the UWSN members becomes mandatory and thus guarantying GKM for UWSNs arises as a very challenging task.

In a typical GKM, every member knows a set of keys that are usually classified as:

- **Individual Key**. Every node has a unique key that it shares with the base station. This key is used for secure communication between the node and the base station. For example, a node can use its individual key to compute message authentication codes for its sensed readings if the readings are to be verified by the base station. A node may also send an alert to the base station if it observes any abnormal or unexpected behavior of a neighboring node. Similarly, the base station can use this key to encrypt any sensitive information, e.g. keying material or special instruction that it sends to an individual node.

- **Group Key**. It is a globally shared key that is used by the base station for encrypting messages that are broadcast to the whole group. For example, the base station issues missions, sends queries and interests. Note that from the confidentiality point of view there is no advantage to separately encrypting a broadcast message using the individual key of each node. However, since the group key is shared among all the nodes in the network, an efficient rekeying mechanism is necessary for updating this key after a compromised node is revoked. Periodic rekeying could also be used in intervals smaller than the average time needed to break the security mechanisms via e.g. brute-force approaches; this increases security at the expense of communication overhead.

- **Cluster Key**. It is a key shared by a node and all its neighbors, and it is mainly used for securing locally broadcast messages, e.g., routing control information, or securing sensor messages which can benefit from passive participation. Researchers have shown that in-network processing techniques, including data aggregation and passive participation are very important for saving energy consumption in sensor networks. For example, a node that overhears a neighboring sensor node transmitting the same reading as its own current reading can elect to not transmit the same. In responding to aggregation operations such as MAX, a node can also suppress its own reading if its reading is not larger than an overheard one. Clearly, for passive participation to be feasible, sensor nodes should be able to decrypt or verify some classes of messages, e.g., sensor readings, transmitted by their neighbors. This requires that such messages be encrypted or authenticated by a locally shared key. As such, LEAP provides each node a unique cluster key shared with all its neighbors for securing its messages. Its neighbors use the same key for decrypting or verifying its messages.

- **Pairwise Shared Key**. Every node shares a pairwise key with each of its immediate neighbors. In LEAP, pairwise keys are used for securing communications that require privacy or source authentication. For example, a node can use its pairwise keys to secure the distribution of its cluster key to its neighbors, or to secure the transmission of its sensor readings to an aggregation node. Note that the use of pairwise keys precludes passive participation.

Pairwise keys allow secure routing by hop-by-hop encryption and authentication, and provide easy isolation of a kidnapped member since compromised keys are just not used anymore. On the other

hand, groupwise keys allow secure routing without the need of costly hop-by-hop encryption but providing hop-by-hop authentication and integrity (checking message authentication codes at every link). However groupwise keys are not resilient against node kidnapping and thus must be updated whenever a member is compromised. Nevertheless both types of keys are part of a vicious circle if asymmetric cryptography is not used: in order to securely agree pairwise keys, nodes need a pre-shared secret that it is often the groupwise key; and in order to securely update a groupwise key, secure communications must be provided often by means of non-compromised pairwise keys.

Most of current distributed GKM proposals on the state-of-the-art for unattended sensor networks focus on generating the necessary group keys from a set of pre-distributed keying material. In any case, either the process of secure agreeing a common groupwise key is a heavy process or relies on a powerful central base station or it is not provided at all. Regarding to WoO, the former is the only option and thus the lack of an efficient re-keying scheme from the non-compromised keys will result, at the very best, in a costly repetition of the initial creation process every time the group membership changes. This fact is especially heavy for common sensor nodes which have limited computational resources and battery, and not to mention how worse the situation becomes when the group is very large. Therefore distributed GKM techniques focusing on re-keying efficiency with an autonomous ability to regenerate the group (unattended creation/updating of the group keying material) must be provided.

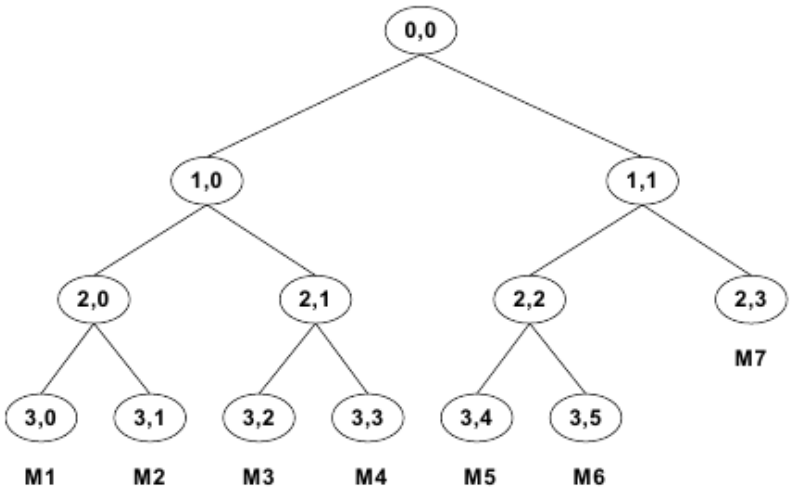

Figure 6: Logical tree of KEKs

Considering that the more energy consumption activity in WoO is the transmission of a message over a wireless channel (reaching up to 3 orders of magnitude), any GKM proposal for WSNs must be

designed to minimize the number of messages that has to be transmitted for re-keying. The most successful proposals for reducing the re-keying problem order are even nowadays based on the old well-studied logical key encryption keys (KEK) tree hierarchies. The simplest method for providing group security is merely based on the use of a groupwise key shared by all the group members that it is often called the network wise key, or group key or session encryption key (SEK). This key allows every group member to: 1) send encrypted data; 2) decrypt received data, and 3) authenticate itself as a group member since the knowledge of the session key guarantees that it belongs to the group. However, in order to securely update the SEK when the group membership changes, some other keys are necessary and these keys are the ones commonly called KEKs. Many GKM re-keying algorithms use these KEK trees since they substantially improve the efficiency in terms of bandwidth and latency. In this kind of methods, a key server builds a KEK tree only known to him and assigns a set of these KEKs to each of the members of the group. This set of keys corresponds to the path from the tree's leafs –where the members are- to its root. When a member leaves or joins a group only the KEKs belonging to that member need to be changed. Then new keys are delivered to the remaining members and the tree is reconstructed using the unchanged keys. In short, the cost of re-keying is reduced from O(N) to O(L), with N the number of members and L the depth of the tree.

Next, the steps of the protocol are described from a broad point of view. A more detailed description of our proposal and an evaluation, which was developed inside the Web of Objects and DiCoMa projects, can be found in [Serrano2013].

Initial creation of the group

During its initial execution all group nodes must be preloaded with an initial symmet-rical key K^k_g. This key is the k-th hash of an initial group key K_g randomly generated by the base station prior to the group deployment. Index k represents the upper limit of new members during the life of the group without direct intervention of the base station because, as explained in section 4, consecutive previous hashes of Kg are revealed whenever the base station inserts a new sensor node on the network (a node joins the group).

The shared secret K^k_g is the assumed to be the current SEK that initially allows group members to protect the group data from outsiders and to authenticate them-selves as members of the group (group authentication). Group authentication is sufficient if we assume that all members are trusted and hence attacks from insiders are not possible. Once the group security is autonomously established, a new SEK is generated, and K^k_g is no longer necessary but to validate new member joins.

As aforementioned, a secure UWSN can run with just the use of a SEK for group-secure routing. Nevertheless, when a node is kidnapped the SEK is compromised and in order to isolate such kidnapped node and guarantying secure rekeying, pairwise keys are also needed for hop-by-hop encryption. As a result, besides providing a group key management scheme, every node must agree a

pair-wise key with all the nodes with which it has direct visibility at the link. From now on we will call these nodes as neighbors.

Once the leaving process is started, the RM deletes every spare node and consequently moves itself or its subtree. Then, the RM updates its key and regenerates all the keys in its path to the root. Next, the RM securely communicates with every leader (GL) of the subtrees hanging from the sibling nodes of the leaving member by means of hop-by-hop encryption with the pre-established pairwise keys. After that, the RM sends to the GLs the needed new blinded keys to reconstruct the tree. After receiving the blinded keys, each involved GL sends them to the rest of members in its subtree encrypted with the subtree root key. Now every member in the tree regenerates the tree keys in its path to the root and a new SEK K0,0 is assumed. At this moment all compromised keys have been updated and the leaving process is finished with a total of just 1 + 2(L − 1) with L = log 2 N for the case of a balanced binary tree with N members or leafs.

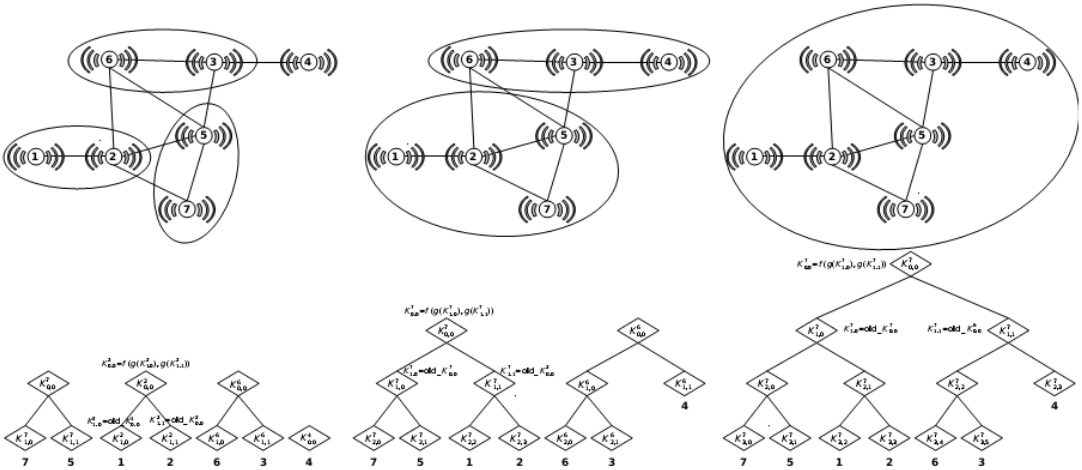

Figure 7: Example of the creation of the group key with three interactions

Managing members leavings/losses/expulsions

When a sensor node leaves the group, in order to guarantee the backward secrecy, the SEK (and consequently the compromised tree KEKs) must be updated. When the member voluntarily leaves (due to e.g. battery exhaustion), it notifies its leave to its sibling (the node hanging from the same node of the tree) or, if it does not exist, the leader (GL) of the sub-tree hanging from the first sibling node of the leaving member in a leaf-root way. The node in charge of initiating the rekeying process is called rekeying master (RM). If the member just crashes or it is compromised, any member detecting it broadcasts its leave in order to notify it to its RM. As a result, intrusion detection systems (IDS) are completely necessary in order to find malicious activities within the network, but they are by now out of the scope of this work. IDS schemes are often based on analyses of node behavior. As a result the

use of tamper-evidence (cheaper than tamper-proof ones) devices may enhance the IDS performance. With these devices, kidnapping a node may produce evidences that could be detected in its transmitted EM waves or messages.

Managing new member joins

When the base station wants to insert a new node in the network, it preloads it with the previous unused hash of the group key. That is to say, if the current SEK is K^k_g, the base station preloads the node with K^{k-1}_g where K^k_g is the hash of K^{k-1}_g. When the new node is deployed it authenticates itself by means of broadcasting its preloaded key and the rest of nodes check the new join by computing the hash of the preloaded key. After the node is authenticated, the rekeying process is similar as it is for a member leaving. First, the node sends its blinded key to the chosen RM, Then, the RM adds a new leaf to the tree (for the new member) moving itself if necessary. Then, the RM updates its key and regenerates all the KEKs in its path to the root. Next, the RM securely sends then necessary blinded KEKs to the rest of member encrypted with the previous shared tree KEKs. After receiving the blinded keys, every member in the tree regenerates the tree keys in its path to the root and a new SEK $K'_{0,0}$ is assumed. Once again we illustrate such behavior with the example of Fig. 4 where member 9 joins the group and the RM is member 5.

A more detailed description of our proposal and an evaluation, which was developed inside the Web of Objects project, can be found in [Serrano2013]

The house scenery

The aim of this section is to securing a scenario where objects create a P2P network with fairly simple gateways and nodes of low complexity, or with complexity of the gateways and nodes is in the same order of magnitude. In this scenario we assume that some nodes (intermediate) route traffic from/to the gateways. This is the typical scenario of a home network, where objects like security cameras, automatic locks, light management sensors… are of constrained resources.

The wireless channels in this scenario must be protected from outsiders. The use of a static, pre-shared security key is a first approach to solve the problem. But his approach shows several drawbacks:

New objects entering the network have access to the security key that old objects were using. Hence, this approach does not provide backward secrecy.

Old objects leaving the network know the secret key. If there is not a disposal policy, these objects may end in the hands of an attacker, and as a result, the security key. Hence, this approach is not able to provide forward secrecy.

A second approach for this scenery is the use of par-wise keys between all nodes. This way, nodes know a secret key to communicate to any of their neighbors. However, now the use of pair-wise keys between the nodes and the gateways would not allow intermediate nodes to authenticate the packets they route.

As a result, two solutions arise:

- Use of pair-wise keys between each node and all of its neighbors. Packets follow the path from the node to the gateway with hop-by-hop encryption; that is to say that every intermediate node must decrypt and check every forwarding packet and then re-encrypt (and calculating the MAC) with the following pair-wise key.
- Use of a network-wise keys shared by all the nodes of each network. An intermediate node must only check group authentication in order to forward de packet, but it does not need to decrypt/encrypt again.

The first solution is significantly more costly but provides greater resilience against node kidnapping. The second one is less costly but an attacker kidnapping a node could access to all network communications (poor resilience).

Pair-wise keys solutions rely on the negotiation of the necessary pairs of keys just after the network deployment. A naïve solution is to pre-load the nodes with a temporal common shared key, the network-wise key, which initially allows them to securely negotiate pair-wise keys with their neighbors. In this case, when a node is compromised the data is not exposed to the attacker (apart from the data passing through or getting out the node) but the network-wise key must be updated in order to allow for new nodes (pair-wise negotiations). We will discuss how to update this key below. However, the use of a trusted third party that securely manages the negotiation is more widely accepted. Another traditional approach would be the use of any known key agreement protocol. However, key agreement protocols are based on asymmetric cryptographic and hence, as previously stated, its use in embedded systems ought to be avoided or minimized due to inherent resource constraints.

In order to achieve a better understanding of pair-wise approaches, we detail next the basic behavior of the SPINS proposal, which has been widely accepted in the literature. Within this proposal, Perrig *et al.* presented a node-to-node key agreement constructed from symmetric-key algorithms and hence with an overall low cost per node. The proposed symmetric protocol is based on a hierarchical centralized infrastructure and, therefore, uses the base station (the gateway in our scenario) as a trusted agent for key setup. The authors avoid the use of any pre-deployment scheme beyond a shared secret between any sensor node and the base station. The protocol is detailed below.

Suppose that the node *A* needs to establish a shared secret session key SK_{AB} with node *B*. Initially, *A* and *B* do not share any secret and thus they use a trusted third party *S*, which is the base station.

Both member *A* and *B* share a secret key with *S*: K_{AS} and K_{BS} respectively. The sequence of messages of the protocol is the following:

$A \rightarrow B : N_A, A$

$B \rightarrow S : N_A, N_B, A, B, MAC(K_{BS}, N_A|N_B|A|B)$

$S \rightarrow A : \{SK_{AB}\}_{KAS}, MAC(K'_{AS}, N_A|B|\{SK_{AB}\}_{KAS})$

$S \rightarrow B : \{SK_{AB}\}_{KBS}, MAC(K'_{BS}, N_A|A|\{SK_{AB}\}_{KBS})$

The nonces N_A and N_B ensure strong key freshness to both *A* and *B*. Confidentiality is ensured through encryption with the keys K_{AS} and K_{BS} of the established session key SK_{AB}, and message authentication through the MAC using keys K'_{AS} and K'_{BS}.

As one can easily extract, this approach (which is the basis of the most accepted ones) relies on the presence of a trusted third party in order to secure the establishment of the pair-wise keys and thus cannot be applied in isolated WSN scenarios without a fixed presence of a powerful base station (or any kind of power node).

Solutions based on the use of network-wise keys are, unlike the pair-wise ones, low-cost solutions more targeted to less dynamic scenarios. These network-keys can provide: confidentiality (symmetric encryption/decryption), group authentication (since the use of the key clearly identifies the node as a member of the network) and message authentication and integrity (by means of a message authentication code – MAC).

However, if an attacker is able to compromise a node, it will get access to the keying material and consequently become an “insider”. When this fact happens, the network-wise key of the network the node belongs to is exposed and thus not usable any longer since the attacker could publish it. The overall previous solution is therefore no more valid as is. Consequently a new challenge appears: 1) the network key must be updated but kept secret for the compromised node.

Consequently, each network needs an entity in charge of securely managing network-wise key updates in order for that key to be only known by the current non-compromised. In this scenario, because of its abilities, this entity should be the gateway. As a result, potentially secure dedicated channels between every node and its gateway must be provided. Through these channels, the gateway can securely send a new network-wise key to all of its associated nodes apart from the compromised one/s. That is easily achieved by means of different dedicated pair-wise keys between every node and its gateway.

It is important to notice that the rekeying messages for updating the network-wise key must use end-to-end encryption and thus rely on the application layer. The packets are relayed to their destinations (the nodes) over a non-secured topology (only the application content is secured). As a result, the

attacker could become aware of the rekeying process but it can neither stop it nor disrupt it, apart from possibly isolating a part of the network from the gateway.

In summary, securing this scenario involves that every node stores at least:

- Either a pair-wise key with all of its neighbors, or a network-wise key for protecting MAC layer communications.
- A (or a set of) pair-wise key with its gateway in order to allow secure negotiation/update of the previous keys.
- A pair-wise key with the base station for authentication at application layer.

In terms of overhead needed, there is the necessary frame overhead for security and either the necessary messages in order to negotiate the pair-wise keys or an order of the number of network nodes messages when rekeying becomes necessary. Although the last is not very costly since the transmitter is the “powerful” gateway and the nodes are only involved when forwarding is necessary. The use of a group key management, as the one proposed in the last section, solves these issues.

Privacy Management

In this section, we explore the privacy issues of the web of Objects and how they must be managed inside the project.

Privacy in technological environments

Privacy can be defined as the area of personal life that every individual has the right to keep confidential, deciding who has access to his personal information and protecting it from interferences **Erreur ! Source du renvoi introuvable.**.

The emerging communications and information technologies make easier to have access to private information of individuals and organizations, reducing the control that they have over their own data. Besides, the cross-border data flows and the storage of information in distributed networked environments becomes a new challenge in the area of privacy. In this context it is necessary to strengthen the mechanisms and regulations that protect the right to privacy of individuals.

Privacy concerns become particularly delicate in the specific case of complex environments such as the Web of Objects, where a huge number of sensors and smart objects with a high degree of autonomy interoperate to provide sophisticated services for home automation, security, entertainment and many other domains. The ability of these objects to automatically collect, process and transfer a large amount of data, to adapt themselves to different contexts and situations, and to take decisions without the need of user interaction lead to new potential risks related to privacy and data protection, that should be carefully taken into consideration.

According to Daniel J. Solove, there are four main groups of activities related to privacy which constitute a potential risk to the public **Erreur ! Source du renvoi introuvable.**:

- The *collection of information*, either from the individuals or the environment, without the user participation. This group includes tasks such as pervasive sensing and monitoring, that allow to gather a big amount of information from different resources.
- The *data processing*, which covers the use, storage and management of the information collected. The data, when analysed together, can reveal new personal information that was not expected to be exposed, and that could be used for behaviour profiling. During the processing of information a number of threats to user privacy may appear due to an inadequate management of information *(insecurity)*, to changes in the purpose of the collection of data, which can be done without the explicit consent of the user *(secondary use)*, or a lack of information from service providers or organizations to users about how their personal data is collected and for which purpose *(exclusion)*.
- The *dissemination and sharing of information*, which in a context such as the Web of Objects can be performed without any human interaction. There are many potential risks derived from the dissemination activities: unwanted disclosure of private information of users, increased access to information by persons other than the owner, the misappropriation of the identity or the reputation of others, the distortion of information or even blackmail. All this creates a sense of mistrust in users as they lose control over their personal information.
- *Invasion of privacy*: in this case the activities are more related to the individual than with the management of his private information. Highlighted among them are the intrusive activities which invade the personal lives of individuals, even in virtual spaces (e. g. spam).

The European Commission, as a result of the public consultation on the Internet of Things that took place between April and July 2012, published a document reviewing the main challenges an objectives of the Internet of Things in terms of privacy and security, that can be applied to the Web of Objects ecosystem **Erreur ! Source du renvoi introuvable.**. In this document, privacy, data protection and information security are considered complementary requirements that should be present in any IoT solution or service, where information security refers to preserve the confidentiality, integrity and availability of the information.

In addition to the risks identified by Solove, the European Commission defines in the mentioned document other relevant privacy issues and challenges for the IoT:

- Privacy should be considered since the design stage of any service or system to improve the effectiveness of the security measures and reduce the implementation costs (*Privacy by design*).

- *Ensuring the continuity and availability of the services offered.* This is an essential requirement that involves taking the necessary measures to provide access to the services when needed and to recover any personal information even in case of system failure.
- *The risks of the services and systems are context dependant*, as the same risk should be treated differently depending on the country regulations and the domain where the service is provided, which makes them more difficult to be evaluated and mitigated.
- *The changes in the purpose for which the data were collected without the users' knowledge*, which not only threaten the right to privacy of individuals but also have a negative impact on the trust users have in the service provider.
- The realisation of *malicious attacks against systems and devices* which may expose personal information if adequate controls and safeguards are not implemented.
- *Lack of flexibility between service providers*. In some cases, the service providers difficult the portability of a user to another company, which reduces the control users have over their data and their right to choose service provider.
- The diversity of laws and regulations on privacy and data protection, which are difficult to apply when the services are provided in different countries.

Clearly, privacy is not only about data protection but also about defining responsibilities, increasing the control of users over their personal data and imposing strict limits to the treatment of personal information. On the one hand it is necessary to create an adequate legislation to limit the collection and management of personal information and to define the responsibilities of the treatment and protection of that information. Moreover it will be mandatory to include in all the systems that manage personal information mechanisms to protect privacy and to detect and control system vulnerabilities.

As users are becoming more aware of the importance of ensuring that their private information remains private, companies and organizations need to protect adequately all the information managed or stored and inform users about how their personal data is treated to build trust in the services they offer.

Finally, it is desirable to increase the data portability and homogenize the legislation on data protection to facilitate coherent law enforcement worldwide.

Legal framework and good practices on privacy protection

Apart from the particular characteristics of the Web of Objects environment, the general recommendations and regulations on data and privacy protection shall be taken into consideration in the development of services and applications. The most relevant are summarized below.

International context

The right to privacy of individuals is recorded in the *Universal Declaration of Human Rights* written in 1948 by the General Assembly of the United Nations. This document establishes that the right to privacy is a human right as it is specified in the following articles **Erreur ! Source du renvoi introuvable.**:

- *Article 12:* "No one shall be subjected to arbitrary interference with his privacy, family, home or correspondence, nor to attacks upon his honour and reputation. Everyone has the right to the protection of the law against such interference or attacks."
- *Article 19:* "Everyone has the right to freedom of opinion and expression; this right includes freedom to hold opinions without interference and to seek, receive and impart information and ideas through any media and regardless of frontiers."

The protection of privacy is a constant concern in today's society. At a global level a series of laws and policies has been defined to defend the right to privacy of individuals and to regulate data protection. These laws vary significantly from one country to another as can be seen in , which displays a map showing information about the data protection laws currently being used or considered worldwide.

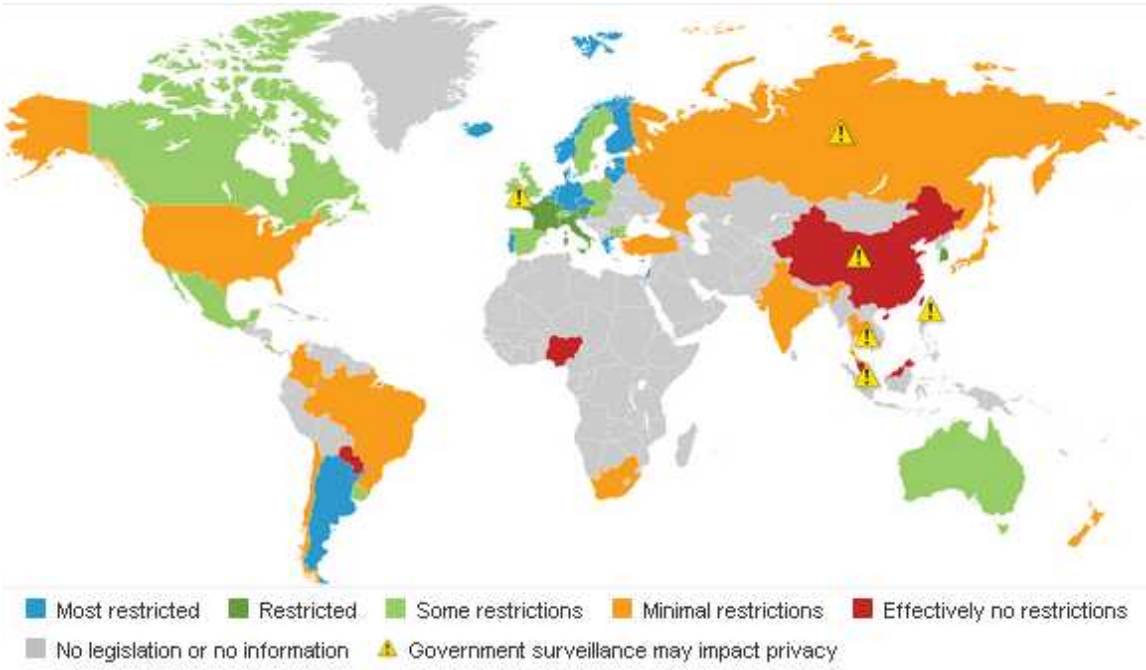

Figure – Privacy and Data Protection Heat Map **Erreur ! Source du renvoi introuvable.**

Internationally there are two main documents whose recommendations on the treatment of personal information should be taken into account in the development of services for the Web of Objects:

- The **Fair Information Practice Principles** (FIPs), initially proposed by the US Secretary's Advisory Committee on Automated Personal Data Systems in a 1973 in response to the increasing development of automated data systems containing personal information about users. The Privacy Protection Study Commission also contributed with their report *Protecting Privacy in an Information Society* in 1977 to the creation of this code of fair information practices for the treatment of personal data in automated systems. These principles can be summarized as follows **Erreur ! Source du renvoi introuvable.**:

  - *Notice/Awareness:* Users should be informed first about the information practices of an entity before any personal information is collected, which includes providing information about who wants to collect their data, for which purpose and how it is going to be managed and protected.
  - *Choice/Consent:* Users should be able to control how their data is used and prevent the data to be used without their consent, especially in the cases of secondary uses of the information not related with the completion of the main service offered to the user.
  - *Access/Participation:* Individuals must be allowed to find out what information has been collected from them, to verify its accuracy and to edit or delete them in an inexpensive and easy way.

    From the user perspective, the privacy policies defining the capacity of organizations to obtain user information can be classified into three types **Erreur ! Source du renvoi introuvable.**:

    - *Opt-in*: those policies which require the prior explicit consent of the user to disclose his personal information to third-parties.
    - *Opt-out*: in this environment the user has the option to refuse to share his personal information with third parties, but by default the data holder is allowed to share it.
    - *Anonymity*: in this case the access to the personal information collected and stored is not allowed.
  - *Integrity/Security:* The entities collecting the information are responsible for ensuring that the data required from users is accurate and secure.

These principles where reviewed later by the Federal Trade Commission (FTC) which added another one in 1998 [FTC-1]:

- *Enforcement/Redress:* There must be appropriated enforcement measures to penalize the breach of these recommendations as a critical component of any government program to protect privacy online.

And also these other three recommendations in 2012 considered as the three basic pillars of the privacy infrastructures of the future **Erreur ! Source du renvoi introuvable.**:

- *Privacy by Design (PbD):* this is a systematic approach stating that privacy and security should be taken into account since the design stage of any product, including design decisions and the development of reasonable security mechanisms to protect user information and verify its accuracy, and limiting the collection and retention of the data.

  This concept was initially developed by Ontario's Information and Privacy Commissioner in the 90's to deal with the challenges derived from the application of the new information and communication technologies. Both in Europe and in the United States regulatory agencies support the privacy by design because it provides better results than other approaches that worry about privacy once the system is designed **Erreur ! Source du renvoi introuvable.**.

  Privacy by Design covers the following elements in practice **Erreur ! Source du renvoi introuvable.**:

  1. Recognition of the need to address proactively the privacy issues and the benefits this brings, such as the increase of the customer trust and satisfaction, reputation enhancement, increase of the competitive advantage, etc.

  2. Application of the different recommendations on appropriate handling of information for privacy protection of the different national and international organizations (OECD, FTC, etc).

  3. Identification of the privacy aspects when developing systems and information technologies.

  4. Need for qualified and dedicated leadership on privacy issues and for contributions of professionals in the field.

  5. Adoption and integration of privacy enhancing technologies (PETs), which minimize the

use of personal data, increase the safety of such data and favour the users control over their personal information.

6. Embedding privacy in a way that improves both the privacy and the functionality of the system.

7. Respect for user's privacy.

This philosophy has already been implemented by many companies, such as Google, Twitter and Mozilla, which now offer SSL encryption by default in some of their online services, or Apple that implements in the Safari browser the blocking of third-party tracking cookies by default **Erreur ! Source du renvoi introuvable.**. In these examples the privacy by design reduces the responsibility of users over the protection of their privacy, but it also can be used to provide more mechanisms to let users express their privacy preferences and control the access of others to their personal data, which makes this approach very interesting to be applied in the Web of Objects.

- *Simplified Choice for Businesses and Consumers:* the data holders should let the users control what information is shared and with whom, including the tracking of their online activities, giving them clear and simple options at a relevant time instead of lengthy policies or terms of service.
- *Greater Transparency:* the entities should publish their practices on the collection and treatment of personal information and let users have access to their data. The FTC proposes to do this through simplified and standardized privacy policies that users can understand easily and compare across companies.

The **Guidelines on the Protection of Privacy and Transborder Flows of Personal Data** of the Organisation for Economic Cooperation and Development (OECD) developed in 1980. This document defines a set of directives that suppose an international unanimity on the general guidelines for the collection and management of personal information. There are eight principles established by these guidelines [PRIOPT] J. Bouckaert and H. Degryse, "Opt In Versus Opt Out: A Free-Entry Analysis of Privacy Policies", 2005

[PRIVHM] Interactive Data Protection Heat Map, Forrester Research, Inc, 2011. Available online in: http://heatmap.forrestertools.com/

- :

- *Collection Limitation Principle:*
  "There should be limits to the collection of personal data and any such data should be obtained by lawful and fair means and, where appropriate, with the knowledge or consent of the data subject."
- *Data Quality Principle:*
  "Personal data should be relevant to the purposes for which they are to be used and, to the extent necessary for those purposes, should be accurate, complete and kept up-to-date."
- *Purpose Specification Principle:*
  "The purposes for which personal data are collected should be specified not later than at the time of data collection and the subsequent use limited to the fulfilment of those purposes or such others as are not incompatible with those purposes and as are specified on each occasion of change of purpose."
- *Use Limitation Principle:*
  "Personal data should not be disclosed, made available or otherwise used for purposes other than those specified in accordance with Paragraph 9 except:

  a) with the consent of the data subject; or

  b) by the authority of law."
- *Security Safeguards Principle:*
  "Personal data should be protected by reasonable security safeguards against such risks as loss or unauthorised access, destruction, use, modification or disclosure of data."
- *Openness Principle:*
  "There should be a general policy of openness about developments, practices and policies with respect to personal data. Means should be readily available of establishing the existence and nature of personal data, and the main purposes of their use, as well as the identity and usual residence of the data controller."
- *Individual Participation Principle:*
  "An individual should have the right:

  a) to obtain from a data controller, or otherwise, confirmation of whether or not the data controller has data relating to him;

  b) to have communicated to him, data relating to him within a reasonable time; at a charge, if any, that is not excessive; in a reasonable manner; and in a form that is readily intelligible

to him;

c) to be given reasons if a request made under subparagraphs(a) and (b) is denied, and to be able to challenge such denial; and

d) to challenge data relating to him and, if the challenge is successful to have the data erased, rectified, completed or amended."

- *Accountability Principle:*
  "A data controller should be accountable for complying with measures which give effect to the principles stated above."

The technological changes produced in the last decades make necessary to adapt all these guidelines to the current situation where privacy is becoming increasingly important for a society that moves towards the globalization of information. Thus, the OECD member countries, the FTC and other organizations are actively working on adapting the fundamental principles of privacy to the twenty-first century and its implementation, in cooperation with business and industry, civil society and other international organizations, to develop appropriate policies to current economic and technological trends. This work is reflected in other more recent publications, such as the Privacy Online: Guidance on Policy and Practical (2003) or Recommendation of the Council on Cross-border Co-operation in the Enforcement of Laws Protecting Privacy (2007) developed by the OECD [PRIOPT] J. Bouckaert and H. Degryse, "Opt In Versus Opt Out: A Free-Entry Analysis of Privacy Policies", 2005

[PRIVHM] Interactive Data Protection Heat Map, Forrester Research, Inc, 2011. Available online in: http://heatmap.forrestertools.com/

. Anyway, both the FIPs and the OECD guidelines have become a referent in the creation of privacy regulations worldwide, being adapted in each case to the laws and situation of each country.

European legal and regulatory framework

This section summarises the main European Directives on the treatment and free movement of personal information that could be used as guidance in the development of services for the Web of Objects.

The European legislation on personal data protection is implemented by means of directives, which are legislative acts of the European Union that require to be transposed to the legislation of each country of the member states [EUDIR].

These are the most relevant directives that can be applied to protect personal data in the Web of Objects:

- **EU Directive 95/46/EC**, which is a key directive on the protection of personal information

that was published in 1995 and adopted by the member states over the past few years [EC9546].

This document was based on the 1980 OECD recommendations described in the previous chapter and sets the limits for the collection and use of personal data protecting the privacy of individuals, all in the context of free movement of personal data within the European Union. It also calls for the creation of an independent national body in each EU Member State responsible for the protection of such data.

This directive is of relevance to the WoO services which require the collection or management of information that may identify users, such as services processing sensitive data, capturing images in which people can be identified or involving other data that can reveal personal information of users (e.g. location). Moreover, it should be taken into account in the development of services which require the movement of personal data between different countries.

At high level, these are the six basic elements included in this directive:

1. Notice: users shall be informed about the collection of their personal data and the purpose of that process.

2. Choice: users shall be able to choose to either have or not have their data collected. Regarding this issue, this directive emphasizes the use of the opt-in approach to collect user information.

3. Use: users shall be able to be informed about how their personal information will be used and to restrict this use.

4. Security: data holders must implement the necessary mechanisms to protect personal data and inform the users about those security measures.

5. Correction: data holders must ensure that the information stored is updated and shall let users access to their information to verify its accuracy.

6. Enforcement: the same as the Enforcement/Redress principle (FIPs).

- **EU Directive 97/66/EC**, concerning the processing of personal data and the protection of privacy in the telecommunications sector [EC9766].

  This directive establishes that member states shall guarantee the confidentiality of the communications made over public communications networks and establishing appropriate

regulations for each country.

- **EU Directive 2000/31/EC** *(Electronic Commerce Directive)*, which creates a basic legal framework for the services of the information society, in particular electronic commerce in the Internal Market [EC0031].

  This directive aims to facilitate the cooperation between member states removing obstacles to the establishment of providers of information society services and to cross-border online services in the European Union, providing legal certainty to both companies and citizens.

- **EU Directive 2002/58/EC**, also known as *E-Privacy Directive*, which replaces directive 97/66/EC dealing with the protection of privacy in the electronic communications sector [EC0258].

  It contains a number of essential rules to ensure user confidentiality in services and in electronic communications technologies. These standards focus on the prohibition of unsolicited email (spam), ensuring the user's prior consent (opt-in) and the use of cookies.

- **EU Directive 2006/24/EC**, which amends Directive 2002/58/EC with regard to the retention of data generated or processed in connection with the provision of electronic communications services or of public communications networks [EC0624].

  It establishes that service providers of publicly available electronic communications or public communication networks must keep the data that allows the identification of such communications: source and destination of the communication, the date and time, call duration, type of communication, equipment used and its location. This shall be applied to traffic and location data on both legal entities and natural persons and to the related data necessary to identify the subscriber or registered user. This will not be applied to the content of electronic communications, including the information consulted using an electronic communications network. This directive is aimed to ensure the availability of data for research, detection and prosecution of serious crime, as defined in the national legislation of each Member State.

- **Directive 2009/136/EC** [EC09136] concerning networks and electronic communications services.

  In 2009 the new EU telecoms rules enters into force, which provides among others the creation of the Body of European Regulators for Electronic Communications (BEREC), whose main

objective is to strengthen the cooperation between the national regulatory authorities (NRA) and the internal market for electronic communications networks.

Regarding Directive 2009/136, also known as "EU Cookie Directive", it requires the explicit consent of the end user to store or access cookies *(opt-in)* unless the cookie is absolutely necessary for the provision of the service requested by the user. It supposes the enforcement of Directive 2002/58/EC in this regard.

Apart from promoting the privacy by design approach, the European Commission recommends the use and development of technologies that guarantee privacy, such as the Privacy-Enhancing Technologies (PETs), particularly to improve the protection of privacy in those cases where personal information is processed through ICT networks. In their report Communication from the Commission to the European Parliament and the Council on promoting data protection by privacy-enhancing technologies of 2007 a PET is defined as "system of ICT measures that protects privacy by eliminating or reducing personal data or by preventing unnecessary and/or undesired processing of personal data, all without losing the functionality of the information system" [EC-IOT] European Commission. "IoT Privacy, Data Protection, Information Security". European Commission Website, Digital Agenda for Europe, Newsroom: Internet of Things Factsheet Privacy and Security. News Section,28/02/2013.

[ECPET-1]. This system consists of a set of tools, applications or mechanisms that allow users to obtain maximum control over their data by protecting the privacy of their Personally Identifiable Information (PII). Examples of PETs are access security and role based authorisation tools, encryption tools, policy tools, filtering tools, etc. The suitability of the technologies will depend mainly on the characteristics of the information system in which they are applied, the level of protection desired and the sensitivity of the data used [ECPET-2].

Smart meters privacy

One of the scenarios under consideration of the Web of Objects is an electric network where the company monitors the energy consumption of a house through smart sensors deployed at the user's home. In this scenario, electricity meters collect information about a customer's electricity use. Unlike the old mechanical meters, Smart Meters are digital, two-way communication devices that can display and transmit more accurate and close to real-time usage information. In this scenario, regulations and company policies apply and as a consequence there exist strict guidelines in place for the storage and use of the collected data, which apply to both electricity retailers and distributors.

There are some privacy concerns that smart meters will reveal the activities of people inside of a home by measuring their electricity usage frequently over time.

Detailed Information on Household Activities: Smart meters offer a significantly more detailed illustration of a consumer's energy usage than regular meters. Traditional meters display data on a consumer's total electricity usage and are typically read manually once per month. In contrast, smart meters can provide near real-time usage data by measuring usage electronically at a much greater frequency. According to the Department of Energy of the US, this may be able to reveal occupants' "daily schedules (including times when they are at or away from home or asleep), whether their homes are equipped with alarm systems, whether they own expensive electronic equipment such as plasma TVs, and whether they use certain types of medical equipment.

Increased Potential for Theft or Breach of Data: Smart grid technology relies heavily on two-way communication to increase energy efficiency and reliability, including communication between smart meters and the utility (or other entity) that stores data for the grid. Many different technologies will transmit data to the grid, including "traditional twisted-copper phone lines, cable lines, fiber optic cable, cellular, satellite, microwave, WiMAX, power line carrier, and broadband over power line." Of these communications platforms, wireless technologies are likely to play a "prominent role" because they present fewer safety concerns and cost less to implement than wireline technologies. According to the Department of Energy, a typical utility network has four "tiers" that collect and transmit data from the consumer to the utility. These include "(1) the core backbone—the primary path to the utility data center; (2) backhaul distribution—the aggregation point for neighborhood data; (3) the access point—typically the smart meter; and, (4) the HAN—the home network." Energy usage data moves from the smart meter, and then to an "aggregation point" outside of the residence such as "a substation, a utility pole-mounted device, or a communications tower." The aggregation points gather data from multiple meters and "backhaul" it to the utility using fiber, T1, microwave, or wireless technology. Utilities typically rely on their own private networks to communicate with smart meters because they have found these networks to be more reliable and less expensive than commercial networks. So, consumer data moving through a smart grid becomes stored in many locations both within the grid and within the physical world. Thus, because it is widely dispersed, it becomes more vulnerable to interception by unauthorized parties and to accidental breach. The movement of data also increases the potential for it to be stolen by unauthorized third parties while it is in transit, particularly when it travels over a wireless network—or through communications components that may be incompatible with one another or possess outdated security protections.

Additional information about this issue can be found in []

Addressing the privacy challenges in the Web of Objects

Considering both the technological and the regulatory aspects described in the previous sections, the design of privacy-aware applications and services in the Web of Objects should cover the following aspects:

- Use of adequate mechanisms for the protection of the data and the communications between the different objects or with users.
- Definition of privacy policies, in which the service providers state their practices concerning data treatment and protection.

Regarding the security mechanisms for data protection, the suitability of the technologies mainly depends on the characteristics of the information system, the level of protection desired and the sensitivity of the data used.

The privacy policies are information management rules that establish the internal practices on privacy for each organization. An adequate privacy policy should be based on the existing recommendations and regulations on privacy and data protection, thus, these are the elements that every privacy policy should include extracted from the recommendations reviewed:

- Contact information of the business, organization or person responsible for the service.
- Detailed information about the treatment of personal data, indicating clearly what information is collected, for which purpose, how it will be used, where and how long it will be stored and what happens to the data once the user unsubscribes from the service.
- Description of the security measures, technical or administrative, adopted for the protection of the personal information.
- Information about the disclosure of personal information with third parties, providing users the opportunity to refuse sharing their personal information.
- Information about how can users access and correct their personal information.
- Dispute resolution procedures.

Privacy policies do not guarantee the security of the data but provide users a sense of transparency while building trust between providers and consumers.

Privacy Policy Languages

The privacy policy languages can help in many of the steps required for the management of privacy policies (review, assessment, policy enforcement...). They were designed to express the privacy controls of both consumers and service providers, and generally respond to a specific need. Most of the initiatives that have come out for the design of this kind of languages have emerged in the last ten years.

In 1997, the W3C began to develop the Platform for Privacy Preferences (P3P) to express the privacy policies of web sites using a single standardized format. That same year, the P3P Exchange Language (A P3P Preference Exchange Language, APPEL) was created to describe preferences for P3P policies between user agents. This language, now obsolete, gave way to the P3P protocol 1.0 and later to 1.1 P3P [W3C-1].

CPExchange was developed in 2000 to facilitate business-to-business communication concerning privacy policies. During that year ODRL (Open Digital Rights Language) was proposed as a language for rights management in digital media and in the context of electronic commerce, which has been accepted by the OMA (Open Mobile Alliance) as standard for rights expression for all mobile contents [FIDIS].

Later in the industry came out the need to express internal privacy policies on the part of organizations. With this aim, first E-P3P (Platform for Enterprise Privacy Practices) was developed, and from this IBM designed EPAL (Enterprise Privacy Authorization Language). They only care about the privacy control within a company or between companies, and do not consider the information from customers or consumers. In order to ensure the consistency between the public policies and the internal policies, Karjoth et al. developed a method to translate EP3P policies into P3P policies. EPAL can also be translated to DPAL (Declarative Privacy Authorization Language) which includes both the customer perspective and the company perspective [FIP-1] Robert Gellman, “Fair Information Practices: A Basic History”, 2010

[FTC-1] Federal Trade Commission, “Privacy Online: A Report to Congress”, June 1998

[FTC-2] Federal Trade Commission, “Protecting Consumer Privacy in an Era of Rapid Change: Recommendations For Businesses and Policymaker”, March 2012

[Garcia06] O. Garcia-Morchon et al. “Security Considerations in the IP-based Internet of Things”. Internet draft. http://tools.ietf.org/html/draft-garcia-core-security-06

[GeoXACML] GeoXACML WebSite: http://www.geoxacml.org/

[GML-1] OGCI – Open Geospatial Consortium Inc., “OGIS Geography Markup Language (GML) Encoding Standard”, 2007

[GML-2] OGCI – Open Geospatial Consortium Inc., “OGC Geography Markup Language (GML) – Extended schemas and encoding rules”, January 2012

[GML-3] PennState, “Introduction to Geographic Markup Language (GML)”, N/A

[IBM] [BARTH].

By the same time, in 2003, the consortium OASIS developed the XACML language (eXtensible Access Control Markup Language) in collaboration with SUN as a new standard for data protection, based on XML. From this language WSPL (Web Services Policy Language) was created, which chooses a subset of XACML standards that are modified to make decisions in the Web services environment. WSPL allows to unify two different policies into one that meets the requirements of both [W3C-1] W3C Website. All Standards and Drafts: http://www.w3.org/TR/#tr_P3P

[WSPL].

WS-XACML (Web Services Profile of XACML), the new version of WSPL, is intended to complement the policies of the Web services adding the potential of XACML concerning access and privacy controls. In this context of Web services, OASIS also developed SAML (Security Assertion Markup Language) and GeoXACML (Geospatial XACML) [FIP-1] Robert Gellman, "Fair Information Practices: A Basic History", 2010

[FTC-1] Federal Trade Commission, "Privacy Online: A Report to Congress", June 1998

[FTC-2] Federal Trade Commission, "Protecting Consumer Privacy in an Era of Rapid Change: Recommendations For Businesses and Policymaker", March 2012

[Garcia06] O. Garcia-Morchon et al. "Security Considerations in the IP-based Internet of Things". Internet draft. http://tools.ietf.org/html/draft-garcia-core-security-06

[GeoXACML]. The first is a communication platform for the authentication and authorization processes between an identity provider and a service provider, and the second can be used to enforce the specific access restrictions of geographic services [PRIOPT] J. Bouckaert and H. Degryse, "Opt In Versus Opt Out: A Free-Entry Analysis of Privacy Policies", 2005

[PRIVHM] Interactive Data Protection Heat Map, Forrester Research, Inc, 2011. Available online in: http://heatmap.forrestertools.com/

.

Privacy policy languages are normally designed as lightweight XML languages as the policies should be as simple and clear as possible. To determine which language policy or protocol is most appropriate it is necessary to study the application context, the legal and privacy needs of the service and the target customers.

User profiles

One of the mechanisms to protect the user profiles in the Web of Objects is introducing some kind of distortion to the description of the user. This distortion is managed by a module inside the

user's objects (the privacy manager), which modifies the user's profile according to the context and identity of the object that is trying to access the data.

In [Vera2012] and as a result of the WoO project, we proposed a method to protect the data that a user profiles captures by means of a lineal projection of the vector profiles into a different social space with fewer dimensions. This way, since the projected vectors have fewer components than the original profiles, we have the intuition that they will content less private information about the user, and therefore the exposition of the user to privacy attacks is reduced.

Despite this projection, the recommender still has to be able to calculate similarities and affinities with other users and documents, using some specially crafted similarity functions.

To analyze the projection of the user profiles, we will take advantage of two lemmas: the Johnson-Lindestrauss lemma and the undecomposability of random matrices. These lemmas are applied as follows. Given a set of profiles that are modeled as vectors of a social space, Johnson-Lindestrauss proved that is possible to calculate a projection into a metric space of fewer dimensions that keeps the distances between profiles bounded, and hence their similarities and affinities. In addition, it is possible to configure the error of the final distances to an error suitable for our need. According to this lemma, if we have a social space of n categories, we can define a projection into a space of $m<n$ categories that keeps the distances between profiles bounded. If $m<n$ and the attacker access only to the projected profile and the projection matrix, he cannot reconstruct the original profile since the linear system is undetermined. The second lemma allows us to take a step forward. According to the undecomposability of random matrices lemma, if $n > 2m -1$, the attacker won't be able to calculate any single component of the original profile.

According to these two lemmas, if we use a matrix M to project the user's profiles in a n-space into another m-space, with high probability it is not possible to recover the original components of the user's profile and the distances in the projected space are related to the distances in the original space.

In our scenario these projections may be used to preserve privacy of data. The reader will notice that these projections are a kind of data distortion, a technique that has been often used to preserve privacy.

Johnson-Lindestrauss lemma ensures that there is a projection with these characteristics, but it gives no hint about the actual matrix. During the development of this project, we tested three different matrices to create the projections:

- A matrix with random components
- A matrix following the structure proposed in [Achlioptas]. This matrix holds the first lemma.

- A hybrid matrix, linear combination of the other two.

We simulated profiles where users can be included in 200 different categories and project into a space of 20 dimensions. Given the projected profile and the projection matrix, a malicious user that tries to reconstruct the original profile has to solve a lineal system of 200 variables with 20 equations. There are 180 freedom degrees, and then we can safely establish that the original profile cannot be reconstructed. Under these circumstances, the privacy of the user is preserved.

The fact that this mechanism generates false positives is not a drawback for the social groups created by the Web of Objects. Even in those networks that make links according to the social distance of the links, some amount of randomness should be introduced in order to minimize the network diameter. Additionally, random links do enhance the recall ratio of the random walk searching protocol, as studied in the group management section. As there is only a limited number of false positives in the system, the analysis of this section allows us to determine a minimum dimension for the projected profiles, in order to ensure the privacy of the users while keeping the amount of false positives low.

Lineal projections of profiles into spaces of less dimensions let users add a configurable degree of protection to their profiles. Indeed, users can modify the amount of anonymity lost and uncertainty of the projected profile by means of selecting a different projection matrix (for example, modifying parameter p of the hybrid matrix) or increasing the dimension m of the projected space. Especially crafted projection matrices let not only to protect the user's privacy, but still make the calculation of the profiles similarity possible, with a bounded uncertainty.

We explored a kind of attack against this mechanism and concluded that the process of attacking a specific user with forged profiles and triangulation of the original user's profile need an initial good guess of the user position. If the initial guess of the attacker is further away in the social space, it is increasingly difficult to estimate the position of the original user profile.

More information about this distortion mechanism can be found in [Vera2012]

Recommendation systems

The WoO project aims to the provision of personalized services that are created as an automatic aggregation of other existing services that the objects in the network offer. The provision of personalized services need is based on the creation, management and transmission of the personal data of the user, in the form of model or view of the user. This is the user's profile, and it contains personal information about the user that the Web of Objects must protect.

This section of the document studies the protection of the privacy of a user that seeks for personalized content inside in the Web of Objects environment. The search for personalized services and content will be based on recommendations. In our definition, a recommendation may be a distributed selection of interesting items, such as books in a library; personalized advertising targeted to users that show some characteristics chosen by the advertiser or; personalized services that are automatically created according to the preferences of the user, such as personalization of tourism tours inside a smart city.

A personal recommender requires some amount of data from the customer. Providing more information may improve the accuracy of a recommendation, as well as increase the exposure of the private data of the user. In [Lam06], authors explore the attacks to the user's privacy using different points of view. Lam identifies data that is useful for a recommendation, such as user's interests, and data that is highly private but (possibly) useless for the system, such as her ZIP code. He also proposes the definition of several privacy metrics:

- The value of the information that the user inserts into the system (from the point of view of accurate recommendations) For example, the knowledge that a user likes a popular movie may not be as meaningful as the fact that she likes an obscure, fan-made tribute movie by the HP Lovecraft Historical Society. In addition, the importance of these data decreases with the volume of data, and a unit of additional information provides a marginal increase of the system knowledge. For example, in a movie rating service, the overall profile of the user is constructed with the first hundred ratings. When the user has rated 1,000 movies, a new rate can hardly modify the user's profile.
- A metric of the risk of exposure of the user (from the point of view of data privacy), and likelihood of publishing sensitive data in the system. Lam also discussed about the nature and amount of information that a user needs to insert into the system in order to get a useful recommendation. Finally, Lam foresaw the usual trade-off between both metrics in a real environment. There are some usual techniques that can be applied to enhance the protection of the user's private data and make identification of the user harder. For example, users could add noise to the ratings of their documents [Domingo04], insert forged, random queries between legitimate accesses [Rebollo10] or join a self-organized coalition of different users to present a joint query to the recommender system [Domingo09].

During the development of the project WoO, we explored a recommender system where users create a social network of smart objects that share their profiles and seek for recommendations in a distributed way. Objects link each other in this social network according to their preferences and then if two objects share a link, they will show similar profiles with high probability.

In this context, an attacker is an insider of the WoO. If an attacker is able to learn somehow the interest of a certain node, or even push a node with a crafted profile into the system, he may be able to make at least some educated guesses about the interests of his neighbors. This kind of attack is

referred as the neighborhood attack [Zhou08]. Zhou et al were interested in de-anonymizing a social network by means of analyzing their links, but their analysis and ideas where extended in later works.

For example, [rastogi2009] analyzed the number of nodes that an attacker must subvert to get a significant knowledge about the network. They conclude that the attack was feasible even the current social networks, in the size of millions.

Thus, there is a need to set a bound to the personal information that users push into the network. This limit will affect accuracy that the user expects for her recommendations, and any system that is concerned with the privacy of its users will find necessary a privacy and utility metrics, and resolve the trade-off between these two metrics. Several techniques exist to protect the user's privacy in the literature, but there is lack of research on how these techniques affect the recommendation output. Finally, protection of the private information of a single user is not enough if her likes and dislikes can be guessed from her neighbors' likes and dislikes.

Attacks against intermediate nodes

Recently, a new kind of attack against intermediate nodes in a distributed system like the one studied by WoO have appeared. These are not traditional attacks. The attackers in this case are legions of lawyers and policemen that put content distributors down using copyright infringement laws.

A new international treaty regarding copyright protection called ACTA is being negotiated at the moment of writing this document. After some secrecy, the consolidated text is now public [ACTA]. Article 2.15 copes with liability of legal persons, and states that "the provisions of this section shall apply *to inciting, aiding and abetting* the offenses referred in article 2.14". The penalties that this article proposes "include imprisonment as well as monetary fines". Thus, not only downloading or the provision of a protected document is punished under the ACTA, but also the abetting to the downloading. According to a EU Parliament member, "the lack of transparency of the negotiations has made it very difficult for both civil society and the European Parliament to monitor the drafting process". Despite of this, many European states endorsed ACTA on January 26th in Japan [Latif2012], but the treaty was not ratified by the European Parliament on July, 2012 [Meyer2012]. Many political groups, both inside and outside the EU Parliament, have expressed their concerns that the ACTA treaty could enact new barriers for individual rights, even with massive protests as in the case of Poland.

The reader should notice that the entity that made documents available, the one that recommend documents, and the one that distribute them may be not the same. This way, a participant recommending or distributing a document may be oblivious of its legal status. In this case, we believe that *recommending* is dangerously close to *inciting*, which is a punished behavior according to ACTA.

This situation is even worse for the recommender system if it includes mechanisms to upload and/or download the protected document.

The ACTA treaty was the beginning of a trend in legislators throughout the world to make people that help or even incite to download copyrighted documents liable of copyright infringement. All over the world, bills with a similar nature and spirit to ACTA are passed to the national Parliaments. This is the case, for example, of the Stop Online Piracy Act (SOPA) and the Protect Intellectual Property Act (PIPA) in the USA, or the so called *Sinde-Wert law* in Spain. These bills have raised lively discussions about the balance between the protection of the rights of the copyright holders, and the open character of Internet where most data is exchanged freely. Most of these bills are currently under heavy modifications, but they have something in common with ACTA: they make companies liable of user's actions if the companies do not react to a copyright infringement notification. Under most legislation these notifications were previously issued only by judicial authorities, but supporters of supplementary controls criticize the slow pace of justice courts. In order to match the fast timings of current economy, they propose that administrative authorities or even IP holders could issue copyright infringement notifications. In some cases, system administrators have a short time to react to these administrative notifications, as short as 5 days in the case of SOPA.

These laws allegedly aim to “the worst of the worst” of the document providers, but according to some opinions [Higgins2011], the “broad and vague definitions” that these laws include are dangerous and may be applied on nearly every site. For example, large action sites and huge social networks will find extremely difficult to monitor every transaction and activity of their users. eBay or Etsy, with hundreds of thousands of fast trades between particular users, cannot pro-actively control the copyright status of the items that users exchange.

Are these threats too exaggerated?

The Pirate Bay is a popular web that indexes files in the BitTorrent network. The Pirate Bay does not provide access to the actual document but only lists a set of addresses that allow potential downloaders to locate the document in the BitTorrent network, outside The Pirate Bay's servers. In April, 2009, the administrators of The Pirate Bay were found guilty of complicity to provide unauthorized access to copyrighted content and sentenced to one year of jail and nearly 3 million euros in damages by a Swedish court [PirateBay]. Short after The Pirate Bay's trial, Rapidshare, a popular intermediate node to download documents, handed over persona information about content uploaders from Germany to the courts in order to prevent legal action against the company [RAPIDSHARE].

Recently, the administrators of the direct downloading site MegaUpload have been put under arrest in New Zealand on behalf of the north-American FBI [MEGAUPLOADFBI]. According to the FBI, this action “directly targets the misuse of a public content storage and distribution site to commit *and facilitate* intellectual property crime”. The FBI accuses the managers of MegaUpload of: 1.- massive

copyright infringement and money laundering; 2.- ``willfully reproduce and distribute many millions of infringing copies of copyrighted works''; 3.- Creation a business model completely centred on *incentive* the copyright infringement, by means of directly paying to the users that upload the most successful files and enforcing rules that prevent the distribution of long term, private data (such as removing data if it is not access after a short time) 4.- Finally, *not removing copyrighted material* even after they are informed of its existence.

In June 2008, Warner Music, Universal Music, Emi, and Sony pressed charges against Pablo Soto, author of several P2P applications. The companies asked for 13 million euros for unfair competition, since the software developed by Mr. Soto could be used to download documents under the copyright of the reporting companies.

Mr. Soto did not upload any protected material to the P2P networks, and he pleaded that he was not able to control the activity of the network users. Pablo Soto was acquitted about the charges of copyright violation, but he had to wait for three years for a final ruling [MercantilNo4deMadrid2011].

The high expenses of a lawsuit, the criminal charges that the defendants face and the long time to get a ruling persuaded other site administrators to react to these legal actions. For example, only two days after that the FBI took actions against MegaUpload administrators, dozens of similar sites (Filesonic, Fileserve, Uploaded.to, VideoBB, FileJungle, UploadStation, FilePost, UploadBox, x7.to, 4shared, etc.) either changed their policies or announced a voluntary shutdown [enigmax2012].

Other direct download sites, especially European companies out of USA soil such as Putlocker or NovaMov, took advantage of the new situation [Labovitz12].

Finally, other actors acknowledge these dangers and are moving the technologies of their services to safer grounds, at least from their point of view. For example, The PirateBay is going to introduce an additional level of indirection by means of moving their service from torrent distribution and tracking to the indexing of magnet URLs. In a few months, The PirateBay will store only URLs that link to a external, uncontrolled and privately run node in a distributed hash table that stores the equivalent to the metadata that torrent files contained in the past [Ernesto2012].

Currently, PirateBay no longer stores links to a P2P network where the desired file can be found, but links to second level links to identify nodes that participate in the desired BitTorrent network. It is not clear if this indirection could prevent legal prosecution against The PirateBay in the future.

Our recommender system was developed in T3.5 of the project Web of Objects, and the architecture of the recommendation system is described in deliverable D3.3. In the next section, we explore the mechanisms that allows the recommender system to protect the privacy of the users.

Oblivious databases

One of the mechanism used in the Web of Object project to protect the user's privacy is oblivious databases. We say that a database is oblivious of the query if a user if (1) the user receives an answer to the query he issued to the database (2) the database does not know the query issued by the user, nor the answer the user got. Private information retrieval is a stronger form of oblivious databases if another feature is met (3) the user does not learn anything apart from the answer she was seeking.

Most oblivious databases schemes use a form of homomorphic cryptography. These schemes are especially useful whenever some party not having the decryption key(s) needs to perform arithmetic operations on a set of ciphertexts, such as in the case of the data aggregation in the objects of the WoO scenario.

Oblivious databases are used in two demonstrators of the Web of Objects:

1. Checking whether or not a vehicle is authorized to park on a specific place. In this case, a surveillance camera asks the centralized database about the permissions of a vehicle without leaking the identification (plate) of the vehicle. The reader is referred to the demonstrators package for additional details about these mechanism.

2. A recommender that is able to issue recommendations related to the interest of a user without learning the specific interest of the user. The high level description of this mechanism is included next, and additional details can be found at [Vera2013].

We call our recommender system DocCloud. DocCloud involves a social network of similar users, a cloud system of recommenders and a distributed secure filesystem. These networks are built on the Web of Objects structure.

Assortative mixing is the property that a network shows when nodes link to other nodes that are similar to them, under some quantitative definition for similarity. Most social networks show an assortative behavior. We work on a social network where there are some clusters of nodes that gather users that share similar interests and hence are affine. When objects join the network, they identify their most suitable cluster according to their interests, as described in the section about group management. In addition, we will define a cloud system that includes all indexers. We will use these objects as indexers of documents of the recommender system. Finally, our recommender system will use another network to store and distribute the real documents. We will assume that, given an URL, users are able to download a file from this filesystem in a private way. Figure 8 shows this scenario and the technologies involved in each one of the steps.

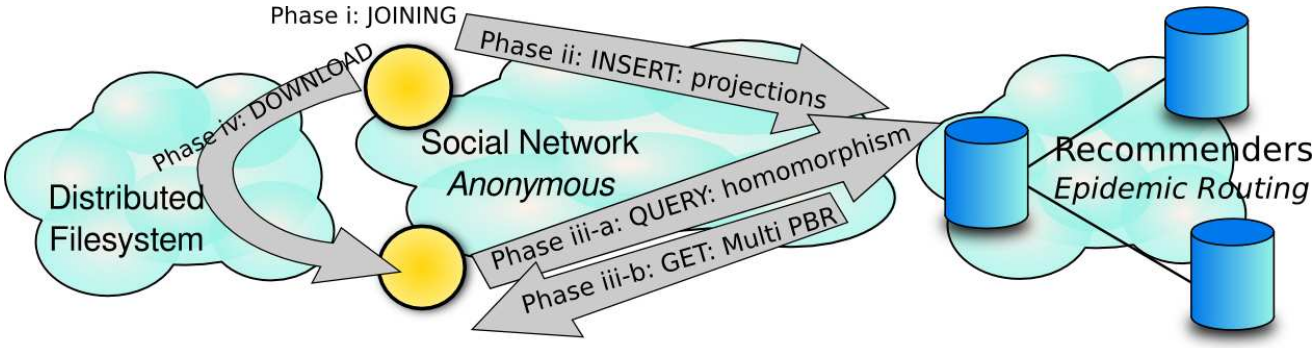

Figure 8: the process of a secure recommendation

Context negotiation and dynamic scenarios

The WoO project builds a dynamic and moving scenario where the users, objects and devices that are available at any moment change over time. From the point of view of the operator or the users of a scenario, the set of available services and resources that objects may access is not constant.

Given the auto-configuration and context-aware capabilities of the objects of WoO, new participants in a scenario may include very different views and definitions of the resources that they share, or a set of constrains that cannot be applied in a different scenario. For example, a mobile phone that includes the detailed profile of its user is allowed to share all the attributes of these profiles with other objects inside the home of the user. On the other hand, the view of these profiles that is shared with the environment must be completely different when the mobile phone enters a public space such a mall. In a similar sense, objects may expose different views of the services that they offer to other objects depending on the current context of the object.

Serious challenges in how objects dealing with distributed data and distributed services should be designed managed and deployed. While some important areas (such as security, transactions and federation) are already being addressed, these approaches tend to cover purely technology issues of how to, for example, secure a protocol or connect federate directories, without wider consideration of the change in paradigm that occurs when large numbers of services are deployed and managed over time. In particular:

- In any non-trivial environment it must be assumed that not all services are owned by the same group of actors.
- Many of the configurations and settings needed by a service to operate must be aligned with and fit to its operational environment, to its context.
- Issues of trust, rights, obligations and permissions immediately arise - and may significantly affect access to data and service executions.

- Workflows must be agreed by all parties before they can be executed since it can no longer be assumed that all parties are either benevolent or will deliver results unless explicit obligations are recorded and held.
- Critical applications may simply cease to function of services provisioned from third parties disappear or malfunction.

All these factors point to the need for a "social level" which properly allows users to define richer contexts for interactions regarding data handling and service execution. The following subsections develop the motivations for this need and the mechanisms that can be used to tackle it.

Generality, locality, and context as first-class object

Generality has always been an issue in computing and more specifically in Artificial Intelligence [McCarthy87], evident in the fact that there has yet been no real "general" theory of the world by which variations of a specific problem can be automatically reasoned about successfully. Such defect is inherent of any theory of the world, as such theories will always be limited from a given perspective. Generality is usually achieved by formulating an appropriate local theory for each possible problem at hand. As a natural consequence, the revision of a problem will often require for a proper revision of the corresponding theory. In the sense that such a local theory is meant to embody the subjective perspective that an individual, e.g. a system designer or a user, has about the world, such theories are called contexts [Bouquet96].

While not being the common case, contexts can be treated as first-class objects in formalisms. This can be achieved by choosing subsets of the global knowledge base, never considering all we know in the reasoning but only a small subset of it [Giunchiglia93]. In consequence, the notion of context is used as a mean of formalizing this idea of localization of the reasoning: a set of local facts along with a set of inference mechanisms.

Even when contexts are not considered, it can always be said that any representation of knowledge is context dependent [Benerecetti00]. In other words, the content of a representation cannot be established by simply composing the content of its parts; in addition, one has to consider extra information that is left implicit in the representation itself. Context dependence can be made explicit by using the box metaphor [Giunchiglia97].

Reasoning can be done both locally and globally, by using one or multiple contexts at the same time respectively; additionally, contexts can be subsumed by other contexts and such subsumptions can be automatically inferred by using context arithmetics [Bouquet96]. Many of the context arithmetics existing also allow for expressing formal properties of contexts, such as partiality, approximation or perspective of one context with respect to another context [Benerecetti00].

Context-awareness in distributed systems

In contrast with the theoretical view of the notion of context, context-aware applications in software engineering are defined as applications that appropriately react to information sensed from the environment, as opposed to applications that just elaborate information explicitly provided by users [Dey01]. Context in this case is rather seen as a collection of features of the physical or virtual environment which can affect the behavior of the application. Such contextual features may include location, time, sender, receiver, and other participants or objects. The practical advantage of context-awareness is that it allows designers to create applications that can use contextual features to automatically adapt their behavior to a dynamic environment [Benerecetti01]. [Vladoiu10] introduces a classification of such features:

- personal context: user's interests and intentions, knowledge-ability, social customs and cultural habits, motivation, social abilities, cognitive abilities, learning style, objectives and goals, and so on;

- task context: operations, goals, operating mode – static or dynamic, etc;
- device context: mobile phone, gipix, PDA, laptop, desktop etc.;
- social context: friends, family, colleagues, acquaintances etc.;
- spatio-temporal context: date, time, user's location, orientation and movement, space – e.g. public, private, limitations – e. g. time interval, location area, etc;
- environmental context (things, persons, services, weather, indoor/outdoor, illumination, noise, crowded etc. from user's surroundings);
- user interface: textual, graphical, 3D, web-based, resolution, dimensions, versatility, etc.;
- infrastructure: network related (availability, bandwidth, stability, price, and so on), or other resources related (coverage, battery, charger etc.);
- strategic context: something important for a planned effect;
- historical context: for keeping trace of the past experience.

Context-awareness has no serious implications in standalone applications where context definition and change can be constrained and controlled, but in distributed systems, with different people interacting with each other to achieve goals not given in advance and using multiple autonomous applications with different definitions of the contexts, the situation is radically different. This arises several interesting issues [Benerecetti01]:

1. Each autonomous application may operate in a different environment, thus making contexts not necessarily shared among applications.
2. If two applications operate in a different context, meaningful interactions between them may require the ability to communicate to each other information about their current context, and to establish relationships between their context and the context of the other applications.

3. Even if the environment is the same, applications may not share the representation of such environment.

Such issues make the definition of [Dey01] insufficient for fully designing distributed context-aware systems. In distributed systems, context-aware applications not only must be aware of their own context, but also need to take into account the fact that other applications operate in different contexts, and that this has many important consequences in their interaction. [Benerecetti01] studies the implications of these requirements with some interesting conclusions:

There is not such a thing as a shared representation of the environment. If two actors of a system are autonomous, it cannot be assumed that an actor will have the power of expressing a concept X coming from other actor in its own local language.

Consequently, there is no need of assuming that there is a list of objectively relevant contextual features.

The semantics of what is represented by two actors is local. In other words, the local relationships between concepts may not be shared between a pair of actors.

Therefore, objectivity cannot be achieved when dealing with contexts in distributed systems. Each actor can, and has to, be considered to have their subjective interpretation; the only workaround to this issue is to eliminate as much as possible the ambiguity of the represented contexts by making the actors to agree on some common grounds, e.g., context-aware SLAs or contracts.

Such issues have usually been too high-level or resource consuming at the practical level and most context-aware solutions are hardcoded rather than theoretically-grounded. Work on context-awareness in Service-Oriented Architectures has historically been limited, yielding negative effects on the advancement of dynamic composition mechanisms [Papazoglou06]. Most of the actual implementations have focused on context as location plus other minor features primarily for mobile applications [Kovács09][Nguyen10], in the form of ad hoc platforms that cannot be easily integrated within the Wed of Objects scope.

Contexts and electronic organizations

A social context is usually defined by public or private agreements or contracts between parties, including shared knowledge and rules of interaction. Without it, it is impossible to enforce a high enough level of predictable behavior and safe interactions necessary to ensure security and privacy of shared information [Horne07].

Groups of parties may share domain and action ontologies, and enforce similar rules of actuation. These shared artifacts are referred to as a social context. Context provides full meaning to the terms, actions, and processes described in a contract between interested parties [Aldewereld10-2]..

When an actor reasons and acts about a regulated frame of interaction agreed with other party, it makes use of the context between both. Actors within a common context share a common vocabulary. This implies that each context has an associated domain ontology defining the meaning of the terms used in the interactions [Panagiotidi08]. Therefore, the ontology bound to a context must contain at least all the predicates, roles, role hierarchy, actions and processes that are part of its domain. A world model may contain sets of rules that use predicates and actions from the ontology, placing constraints on the evolution of the domain (for example, the world model may state that an object may not be placed on top of itself).

Advanced Negotiation Methods

The online, digital presence of businesses that offer online information and web services require new models to negotiate access to resources, e.g. the exchange of information and composition of services. These resources include low-level resources, such as computational resources, as well as higher-level services, such as the level of access to a desired level of functionality. For example, in a large business, a stakeholder may have access by default to only a part of the system, but depending on the context of interaction, it may try to negotiate access to a different part to meet its current goals.

In order to do so, the parties involved need to negotiate and agree on various levels of interaction. Content and services are often distributed across multiple domains, each with their different management policies concerning resources. Moreover, negotiations in such settings will occur not only between individual parties, but also between groups of agents, representing different parties. For example, a negotiation over the level of access to a desired level of functionality in a large system may affect equally all agents of the same type. As such there are inter-dependencies not only between the issues negotiated over, but also between the different contexts in which this negotiation can take places, and over which parties or coalitions participate in a given agreement.

In order to address such challenges, new negotiation protocols will need to be developed, that go beyond the current state of the art in modeling complex multi-issue and multi-party negotiations. In existing literature, several techniques [ItoEtAl05, RobuEtAl05, HindriksEtAl08] consider the issue of handling interdependencies between issues being negotiated, but they do not model the multi-context and multi-party aspect of real negotiations. Other works [NguyenJennings05, AnEtAl09, Endriss06] study concurrent, multi-party negotiations, but they do not consider the complex interdependencies that arise between different issues in a negotiation, or the fact that several agents can prefer to negotiate together as a group.

6. Semantic Language

Ontologies get an important role to play in the realization of the Web of objects. These ontologies provide the vocabulary and the structure of metadata: it is the main representation for the integration of heterogeneous data sources.

Semantic Ontologies can be applied to every business domain: military, environment, health, home and all other possible areas. It needs security, extendable properties describing from space exploration to chemical processing via disaster relief.

This section zooms into a specific technology for the interaction with the environment by semantic language. We propose a study of common languages used for service model representation that answers to static and dynamic concept and securing features.

Semantic languages for devices

RDF & RDFS – Resource Description Framework

RDF describes the semantic networks of information on Web pages and RDF-Schema describes how to use RDF to describe RDF vocabularies. [RDF-2 & RDFS-1]

Security

The document “Security standards for the semantic” [RDF-2] associates RDF with Fine Grained Security that can be supplied by:

- **OLS** – Oracle Label Security for RDF data allows sensitivity labels to be associated with individual triples stored in model. For each query, access to specific triples is granted by comparing their labels with the user’s session labels.
- **VPD** – Virtual Private Database for RDF data allows security administrators to define policies that conditionally restrict a user’s access. The data stored in the models is classified using its metadata and each user query is rewritten to include context-dependent data access constraints that enforce access restrictions.

Pros & Cons

– An RDF graph is inconsistent if it contains a contradiction and thus is interpreted as false.
– RDF Language is not deductive and is limited to binary relations between objects: no reasoning.

Components

The basic RDF model is based on three notions: resources, properties and statements [RDF-1]

- Resources: objects in a system described by expressions
- Properties: attribute used to describe a resource
- Statements: couple of resources with a named property and value of the property

DAML + OIL

"DAML – DARPA Agent Markup Language + OIL Ontology Inference Language" is a semantic markup language builds on RDF and RDF Schema, that provides modeling primitives commonly found in frame-based languages.

DAML+OIL is revised in OWL

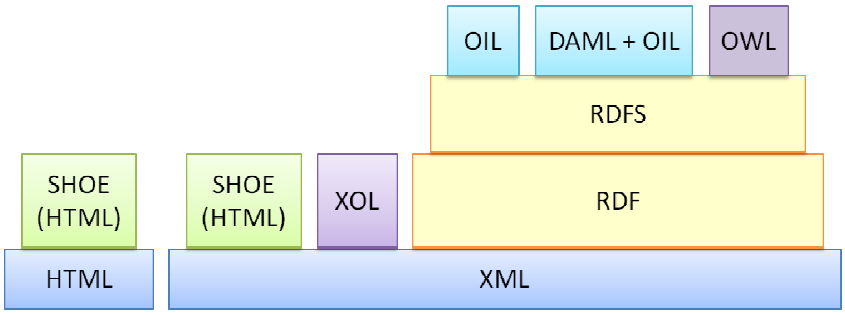

Figure : Evolution of the ontologies markup languages

Security

SRI International paper [DAML+OIL-1] proposes a security ontology for DAML+OIL for handling access control handling, data integrity measures of web resources.

SAML for Security Assertions Markup Language is an XML-based security standard for exchanging authentication and authorization information led by OASIS (Organization for the Advancement of Structure Information Standards) [SAML-1]

Pros & Cons

N/A

Components

The DAML+OIL ontology consists of instances, class elements, property elements as RDF (see paragraph)

OWL –Web Ontology Language

OWL supplies specification, publication, discovery, integration and access. In OWL, ontologies can be modularized and explicit its dependencies. [OWL-1]

OWL can be distributed to any systems, is W3C recommendation and is extensible. Three languages are descendant of it: OWL-Lite, OWL-DL and OWL-Full.

- OWL-Lite: easy to program and reasoning; it is subject to simple classes with low constraints (OWL-limited).
- OWL-DL: with a high expressivity, it is preferred for automated reasoning. The most used.
- OWL-Full: it matches with all RDF data model and is subject to a panel of decisions. It is not suitable for automated reasoning because of complex and slow reasoning.

OWL Full includes OWL DL, which includes OWL Lite. « *Most of the systems use OWL Lite or OWL DL implementation*. » [OWL-2]

Security

Security solutions are not detailed by the language: care should be taken when using OWL with any kind of personal data that might be linked with other data sources or ontologies. [OWL-1]

Pros & Cons

+ OWL-DL is based on description logics (DLs) and permits to calculate the hierarchical class of the structure and to verify the conflicts (errors and inconsistencies in the ontology & errors and contradictions in the data sets).

Components

The OWL ontology work with 3 main components: individuals, classes and properties.

- Individuals : objects in the system (DL-individual)
- Classes: group of individuals or concepts directed by formals descriptions (conditions) (DL-concept)

- Properties: the relation between two individuals. The properties present logical capabilities such as being transitive, symmetric, inverse and functional (DL-role)

Extension

OWL 2 – Web Ontology Language is an ontology language that can be used along with information written in RDF. The OWL 2 has a very similar structure do OWL and adds new features as: Keys, Property chains, Richer data types, data ranges, Qualified cardinality restrictions, Asymmetric, reflexive and disjoint properties, Enhanced annotation capabilities.

The OWL 2 provides individuals, classes, properties and data values [OWL 2-1] see OWL.

IN BRIEF - SHOE Language – Simple HTML Ontology Extension

SHOE is a small extension to HTML which allows annotating the web documents with machine-readable knowledge. [SHOE-1]

The way is to:

- Define an ontology describing classifications of web objects and relationships between them.
- Annotate HTML pages to describe themselves, other pages or subsections.

SHOE is using a standard syntax that allows information to be analyzed and processed. It allows query systems to use the information without having to reparse the text.

The terms used are:

- Category
- Data
- Element
- Instance
- Instance Key
- Name
- Ontology
- Ontology Identifier
- Prefix
- Relation
- Rule
- Version (Number)

RIF – Rule Interchange Format

RIF defines a standard for exchanging rules among rule systems among Web rule engines. It focuses on exchange rather than defining a single one-fits-all rule language such as RDF, OWL and SPARQL.

The approach is to design a family of languages (dialects) with rigorously specified syntax and semantics that is intended to be uniform and extensible (possible to define a new RIF dialect as a syntactic extension to an existing RIF dialect, with new elements corresponding to desired additional functionality that might eventually become standards). [RIF-1]

RIF rules are able to interface with RDF and OWL ontologies: the RIF Working Group has defined in [RIF-2] the necessary concepts to ensure compatibility of RIF with RDF and OWL.

SKOS – Simple Knowledge Organization System

SKOS is a common data model for sharing and linking knowledge organization systems via the Web. It may be used on its own, or in combination with formal knowledge representation languages such as the Web Ontology Language (OWL).

SKOS is not a formal knowledge representation language because it doesn't assert any axioms or facts. Rather, as a thesaurus or classification scheme, it identifies and describes a set of distinct ideas or meanings referred as "concepts". SKOS data are then expressed as RDF triples. [SKOS-1]

Security

In the SKOS data model, most statements of definition are not integrity conditions, but are statements of logical dependency between different elements of the data model.

Pros & Cons

N/A

Components

The element of the SKOS data model are classes and properties, and the structure and integrity of the data model is defined by the logical characteristics of, and interdependencies between, those classes and properties.

SKOS semantic relations are links between SKOS concepts, where the link is inherent in the meaning of the linked concepts. The System distinguishes between two basic categories of semantic relation: hierarchical and associative.

- Hierarchical: direct link between two concepts indicating that one is in some way more general than the other, is used only to make assertions. It provides applications with a convenient and reliable way to access the direct and indirect links for any concepts.
- Associative: direct or indirect link between two concepts indicating that the two are inherently "related", is used to improve search recall through query expansion.

Interaction map of XML based languages

In order to understand the interactions between the XML based languages described earlier, two maps have been built.

The (zoom available on appendix) represents the interactions between XML based ontology languages: RDF. It is described in XML and presents some extensions such as RDFS and OWL.

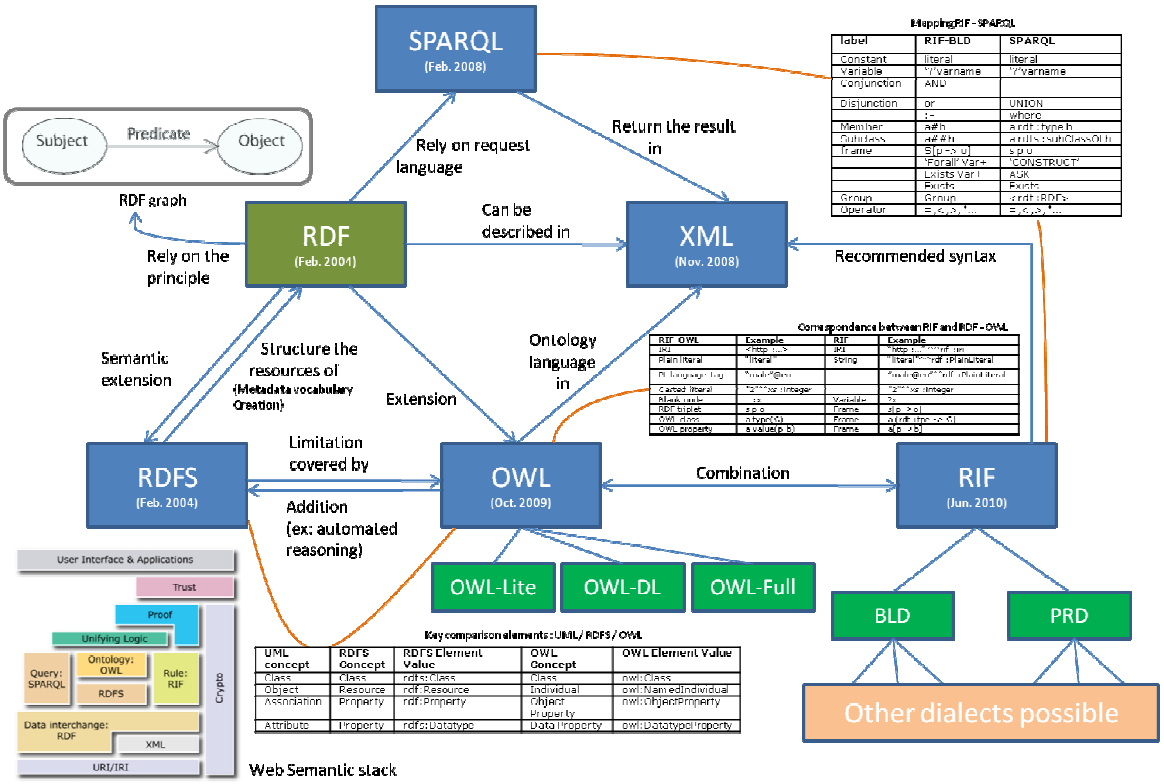

Figure 9: Interaction map of XML based ontology languages

Erreur ! Source du renvoi introuvable. (Zoom available on appendix) represents the interaction between the ontology languages based on XML. SOAP is using directly XML Languages, and some bridges are available to make RIF, RDF, SPARQL and OWL interacting with XML Language.

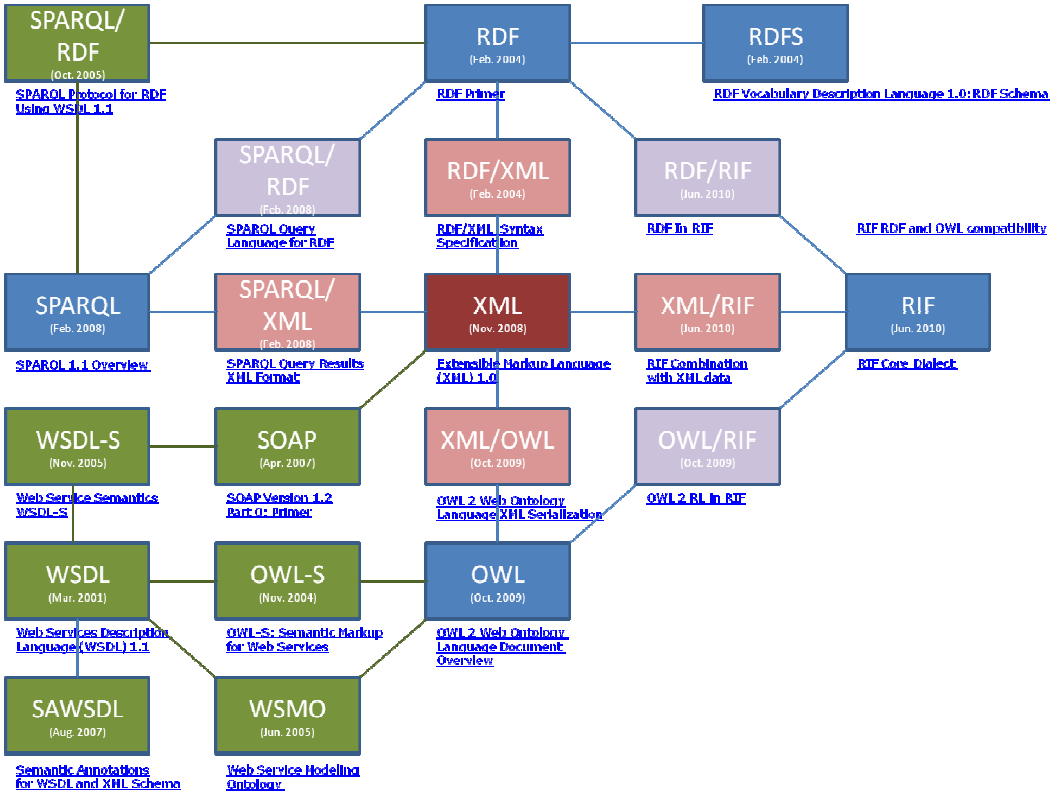

Figure 10: XML interaction map

SWRL – Semantic Web Rule Language

SWRL is a Rule Markup Language (RuleML) that builds a hierarchy of rule sublanguages upon XML, RDF and OWL. [SWRL-1]

The proposed rules are of the form of an implication between an antecedent (body) and consequent (head). The intended meaning can be read as: « *whenever the conditions specified in the antecedent hold, then the conditions specified in the consequent must also hold*. ».

Security

N/A

Pros & Cons

+ For an extensive use of rules, the form or expressiveness of the rules can be restricted in order to increase interoperability, reusability, extensibility, computational scalability or ease of implementation.

Components

All rules are expressed in terms of OWL concepts: classes, properties and individuals (see RDF or OWL)

KIF – Knowledge Interchange Format (KIF)

KIF is a language designed for use in the interchange of knowledge among disparate computer systems. KIF is

- Not intended as primary language for interaction with human users (appropriate applications such as Prolog, conceptual graphs, natural language, etc.)
- Not intended as an internal representation for knowledge within computer systems (a computer converts the data of a knowledge base in KIF into its own internal form.

The basis of KIF is a conceptualization of the world in terms of objects and relations among those objects where a universe of discourse is the set of all objects hypothesized to exist in the world.

KIF does not require every user to share the same universe of discourse, but requires every universe to include certain basic objects. [KIF-1]

Security

N/A

Pros & Cons

– KIF is a highly expressive language that:

- Can complicate the job of building fully conforming systems.
- Can tend the result systems to be “heavyweight”.

Components

Basic objects include:

- Numbers (real and complex)

- ASCII characters
- Finite string of ASCII characters
- Words = objects along with the things they represent
- Finite lists of objects

Relationships among objects take the form of relations: finite lists of objects that jointly satisfy the relation. A function is a special kind of relation by associating a unique object (value) for every finite sequence of objects (arguments)

Extension

In order to deal with the problems identify on the paragraph , the KIF committee voted to augment the basic language specification with a set of “conformance dimensions” in order to accommodate varying capabilities and/or computational constraints while providing a migration path from more restrictive to more expressive.

SUO-KIF – Standard Upper Ontology-Knowledge representation Interchange Format

SUO-KIF is the knowledge representation language used in the SUMO – Suggested Upper Merged Ontology. [SUMO-1]

SUO-KIF was derived from KIF to support the definition of the Suggested Upper Merged Ontology. It is a language designed for use in authoring and interchange of knowledge among disparate computer systems. It is intended primarily a first-order language: a compromise between computational reasoning and richness of representation. [SUO-KIF-1]

MILO is an ontology that provides deeper conceptual support to the various domain ontologies that have been developed under SUMO. [MILO-1]

Security

N/A

Pros & Cons

– The SUMO consists of approximately 4 000 assertions (including over 800 rules) and 1 000 concepts.

+ The SUMO is designed to be relatively small so that these assertions and concepts will be easy to understand and apply.

– KIF is a highly expressive language. Some disadvantages are expressed in:

- Building fully conforming systems
- Having the resulting systems that tend to be “heavyweight”

+ In order to deal with these problems, the basic language specification is augmented with a set of “conformance dimensions”. Although this conformance profile scheme is more complex than one based on conformance levels, it accommodates varying capabilities and/or computational constraints while providing a migration path from more restrictive to more expressive. [SUO-KIF-1]

Components

SUMO is a modular ontology that is divided into self-contained sub-ontologies. Each sub-ontology is indicated by a section header and the dependencies between them.

The syntax of SUO-KIF is described in three layers: characters, lexemes and expressions where terms are individual (class), function (relation) and sentence (facts). [SUO-KIF-2]

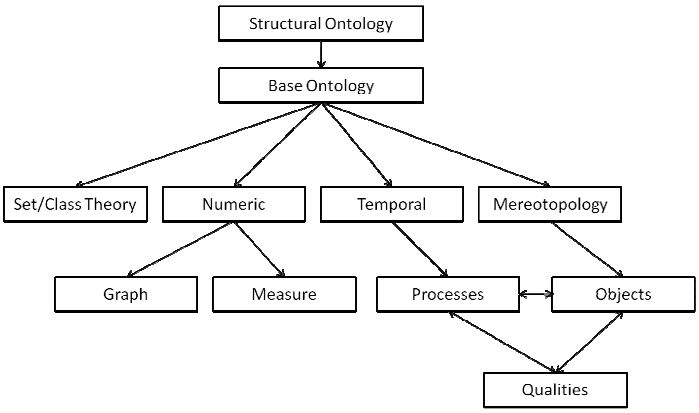

Figure : Sumo structure [SUMO-2]

Ontology extension

The MSO – Multi Source Ontology is part of the public KB of WebKB-2 that permits Web users to browse, search, filter and display, update knowledge bases (KBs). The SUO Working Group has voted the MSO as one of the materials to work on.

The ontology of the shared knowledge base is currently an integration of various top-level ontologies. It permits:

- The categories and statements from the various ontologies complement, illustrate or precise the various ontologies.
- Lexical conflicts are avoided
- Semantics conflicts or redundancies can be detected when new categories or statements are added.

- The categories and statements may be filtered on their creators.
- The contribution of each knowledge provider is acknowledged.

CYCL ontology – Cycorp ontology

CycL is a formal language whose syntax derives from “first-order predicate calculus” and from Lisp. [CYC-1] It is essentially an augmentation of first-order logic with extensions to handle equality, default reasoning and some second-order features.

CycL is the Cyc representation language used to describe Cyc ontology which is one of the general-purpose or ‘universal’ ontologies that are publicly available to perform human-like reasoning. It offers a graph to illustrate the generalization relation (sub/superset) among entities, which can show individual action. [CYC-2]

Security

+ Cyc establish mission critical problem prevention within CycSecure. It is a security risk analysis tool that capitalizes on the power and richness of the Cyc Knowledge Base and reasoning system. It provides network security professional with analyses of an organization’s network vulnerabilities at several levels.

Pros & cons

The potential of the software to assist in the provision of consistency among definitions and relations has not been tested.

The Cyc ontology claims to be “formalized common sense”, and the problem is that a “common sense” vocabulary is not coherent and consistent.

Components

The vocabulary of CycL consists in a set of terms in a natural language (constants that represents a collection or an individual), object, non-atomic terms, variables, etc.) combined into expressions (quantifier and relation) used to make assertions in the Cyc knowledge base. [CYC-1]

Semantic language for services

OWL-S – Web Ontology Language for Web Services (or DAML-S)

DAML-S for DAML based Web Services is the precursor of OWL-S

OWL-S – Web Ontology Language for Web Services, is an ontology that describes properties and web services capacity: it permits discovery, invoking, composing and monitoring Web resources with a high degree of automation. [OWL-S-1]

Security

Different types of information are considered like:

- The category of the service
- The quality rating to showcase the quality of service it provides
- Any kind of information: estimation of the max response time, to the geographic availability of a service.

Pros & Cons

This language gives a formal representation of the services capacities and processes models that are used for automatic compositions (reasoning):

Components

OWL-S is based on 3 notions: profile, grounding and process model.

- Service Profile: what the service does (service requester, provider or infrastructure components).
- Service Grounding: how to access the service (concrete realization).
- Service Model: descriptive information on the functionality of a service (conceived as a process) and its composition out of other services (how the service works).

SWSL – Semantic Web Services Language

The SWSL Committee of the Semantic Web Services Initiative (SWSI) prepared the submission of SWS Framework. SWS for Semantic Web for Services or Semantic for Web Services [SWS-2] is an ontology-base markup that enables automatic service discovery, service composition and service invocating and monitoring.

SWSL includes two distinct knowledge representation languages: SWSL-Rules and SWSL-FOL.

- SWSL-Rules: declarative rule-based language based on the logic programming/deductive database paradigm
- SWSL-FOL: first order classical logic based language (logic description of services composition)

Security

SWSL-Rules are suited to represent available knowledge and desired patterns of reasoning for several tasks [SWSL-1]:

- Authorization policies (for security, access control, confidentiality, privacy and other kinds of trust)
- Monitoring of processes to recognize and handle exceptions or other dynamic conditions
- Advertising, discovery and matchmaking
- Semantic mediation: translation mappings that mediate between different ontologies or contexts and between knowledge expressed in those ontologies (translate from the output of one service to the input expected by another service)
- Object-oriented ontologies that use default inheritance with priorities and/or cancellation

Pros & Cons

– The solution is complete but the reasoning is based on complex description logics (DLs).

Components

SWSL basic syntactic components are head, body and query

Extension

The SWSL-FOL is a first-order sublanguage of SWSL.

WSML – Web Service Modeling Language

WSML is a semantic web language targeted specifically at SWS – Semantic Web Service. It is based on three logical formalisms for the modeling of services [WSML-1]: Description Logics (DLs), first-Order Logic and Logic Programming

WSML consists of five bases: WSML-Core, WSML-DL, WSML-Flight, WSML-Rule and WSML-Full.

WSML is used in WSMO (for Web Service Modeling Ontology) that is defined in a Meta Model Language based on the MOF (Meta Object Facility). WSMO elements (generic information item descriptions and service-specific properties related to the quality of service) are described by [WSMO-1]:

- non-functional properties (be associated with every main component model)
- goal-related information (describe the requested Web service capability)
- Web service functional capabilities (describe what the service does in terms of inputs, output, pre-conditions and post-conditions)
- Choregraphy (describe how to communicate with a Web service)

- Grounding (describe how the semantic declarations are associated with a syntactic specification such as WSDL)
- Orchestration (specify the capabilities by functionalities
- Mediators (specify the elements that are connected and the mismatches that can be resolved between them)

Security

Non-functional properties are an extensible list of attribute values which consists of the attribute identifier:

- **Quality of Service**: Accuracy, Network-related QoS, Performance, Reliability, Robustness, Scalability, Security, Transactional, Trust.
- **Dublin Core Metadata**: Description, Format, Identifier, Language, Publisher, Subject, etc.
- **Other**: Financial, Owner, Type of Match, etc.

Security provides authentication, authorization, confidentiality, traceability/auditability, data encryption and non-repudiation.

Pros & Cons

N/A

Components

WSML syntax consists of two major parts: the conceptual syntax and the logical expression syntax.

The conceptual syntax is used for the modeling on ontologies, goals, web services and mediators (these are the elements of the WSMO conceptual model

- Ontologies: provide the terminology.
  - *Concepts*: the basic elements of the agreed terminology for the system
  - *Relations*: model interdependencies between several concepts
  - *Functions*: special relation with a unary range and a n-ary domain where the range value is functionally dependent on the domain values (parameters inherited from relation)
  - *Instances*: objects defined explicitly or by a link to an instance store
  - *Axioms*: axiomatic expressions (logical statement)

- Goal repository: defines the problems that should be resolved by web services.

- Web services: describes aspects of a web service.
  - *Capability*: functional description (preconditions, assumptions, post conditions and effects)
  - *Service interfaces*: how the service behaves in order to achieve its functionality
    - Choreography: describes the interface for the client-service interaction required for service consumption
    - Orchestration: describes how the functionality of a service is achieved.

- Mediators: bypasses interoperability problems.
  WSMO needs ontology mediators to be used for resolving mismatches: aligning, merging and transforming imported ontologies.

The general logical expression syntax: it occurs within axioms and the capabilities which are specified in the descriptions of goals and Semantic Web services. [WSML-1]

SA-WSDL– Semantic Annotation WSDL

SAWSDL is a W3C Recommendation that defines how to add semantic annotations on schema types that can be used during web service dynamic discovery, composition, and for specifying the data mapping of xml schema types to and from ontology useful for invocating the services.

The semantic annotation mechanism does not rely on any particular semantic modeling language, but requires that the semantic concepts defined in it be identifiable via URI references. (An URI should refer to concepts in a semantic model) [SAWSDL-1]

The document [SAWSDL-2] describes a representation of that model in the Resource Description Language (RDF) and in the Web Ontology Language (OWL), and a mapping procedure for transforming particular WSDL descriptions into their RDF form.

SAWSDL is based on member submission WSDL-S: SAWSDL is a restricted and homogenized version of WSDL-S including a few changes [SWS-1]:

- Do not introduce “precondition” and “effect” since there was no agreement on how to model the semantic web.
- Replace the “category” annotation by a more general extension attribute.
- Decompose “schemaMapping” annotation into two different extension attributes to specifically identify the type of transformation performed.

Security

The security is brought by the Web Services (WS-*) specifications. The broad acceptance and adoption can be slow, because of the complexity and the confusion resulting from WS-* specifications. [SAWSDL-3]

Pros & Cons

« Mapping SAWSDL into RDF»: WSDL extensions are mapped into an RDF form compatible with the WSDL 2.0 RDF Mapping. [SAWSDL-1]

Components

Conceptually, WSDL has the following components to represent service descriptions: element declaration, type definition, interface, binding and service.

- The first three deal with the abstract definition of a service
- The latter two deal with service implementation.

Extensions

SAWSDL4J extends WSDL4J.

WSDL-Semantics

WSDL – Semantics defines a mechanism to associate semantic annotations with Web Services that are described using WSDL. [WSDL-S-1] This approach offers multiple advantages over OWL-S:

- Describe, in a upwardly compatible way, both the semantics and operation level details in WSDL
- Take an agnostic approach to ontology representation languages by externalizing the semantic domain models.
- Preconditions and effects are used by WSDL-S to describe the conditions that must be met before an operation can be invoked, the result that the invocation of the operation will have.

The initial WSDL-S proposal was done by the METEOR-S group from the University of Georgia.

Security

The support of Quality of Service assertions for Web services are investigating by using ontologies and rules by extending the WS-Policy framework. WSDL-S should stay close to the WSDL specification.

Pros & Cons

WSDL-S allows Web Service developers to annotate their Web services with their choice of ontology language unlike in OWL-S.

It is relatively easy to update the existing tooling around the WSDL specification to accommodate an incremental approach.

Components

WSDL has the following constructs to represent service descriptions: interface, operation, message, binding, service and endpoint.

- The first three deal with the abstract definition of a service
- The remaining three deal with service implementation.

Extensions

WSDL-S supports UML class models for annotation that is an alternative modeling language (because of the extensive use of UML to create domain and enterprise models). Alternatively, a UML class model can be converted to OWL using the mapping rules defined by the Ontology Definition Metamodel (ODM) developed by the Object Management Group. [WSDL-S-1]

METEOR-S

The METEOR project is focused on workflow management techniques for semantic Web services. It extends Web Services standards by applying Semantics Web technologies in Annotation, Quality of Service, Discovery, Composition and execution to achieve greater dynamism and scalability.

SA REST – Semantic Annotation for REST

SA-REST is a data format that adds additional meta-data to REST API descriptions in HTML or XHTML.

SA-REST annotations facilitate better exploration and composition capabilities in SOSNs (concept of SOA – Service Oriented Architecture, into the sensor network domain). [SA-REST-1]

SA-REST is developed from many of the ideas that were first presented in WSDL-S and then adapted in SAWSDL. SA-REST does not enforce the choice of a language for representing ontology or a conceptual model. SAWSDL service can be translated into a SA-REST service and back. [SA-REST-2]

Security

N/A

Pros & Cons

Adding semantic annotations to a RESTful web service yields many benefits and alleviates many of problems associated with RESTful web services:

+ Facilitate data mediation: it adds a way to RDF to specify the format of the inputs or outputs of the service

+ Determine automatically how services are invoked (via http post, http get) by annotating the type of request that should be used. That allows a tool to invoke a service knowing which type of request should be chose. [SAREST-3]

Components

SA-REST defines three basic properties to non-intrusively annotate HTML/XHTML documents:

- Domain-rel: allows a domain information description for a resource
- Sem-rel: captures the semantics of a link and adds externalized annotations to third party documents
- Sem-calss: used to markup a single entity within a resource.

Sensor ML - Sensor Model Language

SensorML is a key component for enabling autonomous and intelligent sensor webs. It provides the information needed for discovery of sensors, including the sensor's capabilities, location and taskability. It also provides the means by which real-time observations can be geolocated and processed "on the fly" by SensorML-aware software.

The openGIS SensorML Encoding Standard specifies models and XML encoding that provide a framework within which the geometric, dynamic and observational characteristics of sensors and sensor systems can be defined. [SensorML-1]

Within SensorML, sensors and transducer components are all modeled as processes that can be connected and participate equally within a process chain or system, and which utilize the same process model frame as any other process. All processes and component are encoded as application schema of the Feature model in the Geographic Markup Language (GML).

Security

A SensorML resource description can be constrained by three properties: national and international securityConstraints, validTime and legalConstraints.

- securityConstraints: is based on the Security Banner Marking model of the Intelligence Community Information Security Marking (IC ISM) standard.
- validTime: indicates the time instance or range over which this process description is valid. This property is important for processes in which parameter values or operation modes may change with time.
- legalConstraints: is based on ISO 19115 and specifies whether Privacy Act, Intellectual Property Rights or copyrights apply to the content of the process description or its use.

The role of the SensorML is to provide characteristics required for processing, georegistering, and assessing the quality of measurements from sensor systems. Most of the simple scalar types have a quality property that can provide some measure of the quality of a scalar value. For numerical scalars such as Count, Quantity and Time, the quality may be expressed as values of precision, accuracy, tolerance and confidence level.

Pros & Cons

N/A

Components

Sensor ML proposes four components: base, process, method and system.

- **Base**: provides the basic definitions and abstract elements used by several SensorML schema and defines the metadata groups and components
- **Process**: provides definitions for base process types, ProcessChain and ProcessModel
- **Method**: describes the process methodology including algorithms, documentation, fine-grained validation and links to software implementations
- **System**: defines a System and Component derived from Process with additional positional and interface information

SWE – Sensor Web Enablement

The goal of the SWE initiative of the OWS – OGC Web Services, is the definition of web service interfaces and data encodings to make sensors discoverable, taskable and accessible on the World Wide Web by defining a framework of data models and encodings describing sensors and their observations.

The SWE specifications enable a standardized communication and interaction with arbitrary types of sensors and sensor systems.

The main adopted OGC standards in the SWE framework includes:

- **O&M** – Observation & Measurements: standard that specifies an XML implementation for observations and for features involved in sampling when making observations. The XML schemas provide document models for the exchange of information describing observation acts and their results, both within and between different scientific and technical communities. [O&M-1].
- **SensorML** – Sensor Model Language: standard models and XML schema for describing the processes within sensor and observation processing systems.
- **PUCK**: defines a protocol to retrieve a SensorML description, sensor "driver" code, and other information from the device itself, thus enabling automatic sensor installation, configuration and operation.
- **SOS** – Sensor Observation Service: Open interface for a web service to obtain observations and sensor and platform descriptions from one or more sensors.
- **SPS** – Sensor Planning Service: open interface for a web service by which a client can
    - Determine the feasibility of collecting data from one or more sensors or models
    - Submit collection requests
- **TML** – TransducerML: model and encoding for streaming multiplexed data from a sensor system, and for describing the system and data encoding. It defines a set of models describing the response characteristics of a transducer and an efficient method for transporting sensor data. The TML response models are formalized XML descriptions. The specification has been RETIRED [TML-1]
- **SAS** – Sensor Alert Service: service for advertising, subscribing to and publishing alerts to alert listener clients.
- **WNS** – Web Notification Service: a service by which a client may conduct asynchronous dialogues with one or more other services. This service is useful when many collaborating services are required to satisfy a client request, and/or when significant delays are involved in satisfying the request.

OCML – Operational Conceptual Modeling Language

OCML is a language used for representing service models (metamodelling), rules and logical expressions. It combines a frame system with a tightly integrated forward and backward chaining rule system and includes constructs for defining classes, instances, relations, functions, procedures and rules.

OCML contains import/export facilities to RDF(S) and import facilities for OWL. It is used for internal representation of IRS-III (Internet Reasoning Service part 3) which API contains a module for translating between OCML based SWS descriptions and WSML based SWS descriptions. [OCML-1]

IRS that is currently being applied within the WSMO Working Group, is a service ontology forming the epistemological basis and providing semantic links between the knowledge level components describing SWS and the conditions related to its use.

IRS-II follows the UPML framework developed within the IBROW (An Intelligent Brokering service for knowledge-component Reuse On the World-wide-web) project. The UPML framework partitions knowledge into ontologies, domain models, task models, and problem solving methods (PSMs) which are connected via bridges.

Security

The IRS-III service ontology contains the same items than WSMO in particular Non-functional properties that describe component model, information about the provider and the service (category, cost and quality of service, scalability, security and robustness).

Pros & Cons

N/A

Components

The IRS-III service ontology contains the same items than WSMO and uses OCML for modeling ontologically the semantic descriptions of goals, Web services and Mediators, and for implementing components that perform selection, Choregraphy, Orchestration and Mediation

GML – Geography Markup Language

The GML is an XML grammar for expressing geographical features and serves as a modeling language for geographic systems [GML-1]. It defines syntax, mechanisms and conventions [GML-2]:

- Provide an open, vendor-neutral framework for the description of geospatial application schemas for the transport and storage of geographic information in XML;
- Allow profiles that support proper subsets of GML framework descriptive capabilities;
- Support the description of geospatial application schemas for specialized domains and information communities;
- Enable the creation and maintenance of linked geographic application schemas and datasets;
- Support the storage and transport of application schemas and datasets;
- Increase the ability of organizations to share geographic application schemas and the information they.

A digital representation of the real world may be thought of as a set of features.
The state of a feature is defined by a set of properties, where each property may be thought of a set of {name, type, value] triple. [GML-1]

Security

N/A

Pros & Cons

Despite the fact that GML has the advantage of being reasonably easy to read (even if you no longer have the original program that made the archive), making use of GML for data storage is still quite rare, because GML is a text based language that produces large files. [GML-3]

Components

There are two parts to the grammar:

- The schema that describes the document
- The instance document that contains the actual data

Examples of Working Projects and Initiatives

WSAN – Wireless Sensor and Actor Networks

WSAN is referred to a group of sensors and actors linked by wireless medium to perform sensing (event detection, event identification, location sensing), communication (acting tasks via local control) and computing.

Further research problems should be investigated in WSANs [WSAN-1]:

- Algorithms providing ordering, synchronization and elimination of actions redundancies between sensors and actors.
- A unified framework that can be exploited by different applications to select the best networking paradigm and to provide efficient actor-actor communication
- Open WSAN layering architecture
- Effective algorithms for a single layer as well as cross layering requirements.
- Real-time communication protocols for both sensor-actor and actor-actor coordination in WSANs
- Low cost and reliable architecture for networked sensors

The question of reliability is resolved by [WSAN-2] by a design methodology independent of the computation and communication platforms upon which the WSAN is built, and the environment in which the WSAN is deployed. It also does not rely on the system models and controller design of

the target control applications. It can be applied in a wide range of WSAN-based control applications.

SWAN – Semantic Web Applications in Neuromedecine

The SWAN project aims to develop a practical, common, semantically-structured framework for scientific discourse in bio-medicine in general and neuro-medecine in particular.

The project is the result of collaboration between the Alzheimer Research Forum and computer scientists at Massachusetts General Hospital and Harvard University.

The ontologies integrated in the SWAN ontology are natively (or new versions defined) expressed in OWL-DL [SWAN-1]

The SWAN ontology is organized in three types of modules:

- Basic modules: ontology building blocks.
  - Collections: set and bags (unordered) and lists (ordered)
  - PAV – Provenance, authoring and versioning: relationships for defining provenance, authoring and versioning (which are not part of Dublin Core vocabularies)
  - Discourse relationships: relationships have been defined without constraining domains and ranges
  - FOAF – Friend Of A Friend: persons, activities and their relations are described as projects, organizations, groups.
  - Agents: extension of FOAF to cover all the requirements
  - SKOS, Qualifiers, Scientific discourse: classification/annotation through scientific vocabularies or terminologies related to biomedicine
- Extensions modules: fields of science.
  - Life science entities: relationships between entities related to biomedicine
  - Citations: requirements of the SWAN applications
  - Qualifiers extension modules: perform annotation
- Distributions: includes basic modules and extensions for serving a specific domain.

Conclusion

In this study we described common languages used for service model representation. Some of these languages are answering to static and dynamic concept and securing features, and some are not but there are bridges to make them answering to the wanted features.


WoO
web of objects

7. Best security practices

In this chapter and as a result of the security mechanisms explored in the last chapters, we describe the best practices to be implemented by all partners of the Web of Objects project to secure all the entities that participate in the network.

Identity management

Manage groups of objects with a common, shared identity. Identify, create and manage groups of similar objects in the same environment. For example, objects own by the same person inside a house, objects in the area under the control of a large mall, a stadium, the group of objects inside a car… The management of the security parameters using groups of objects is easier, and security administrators can device and enforce group security instead of checking individual policies and authorization. The creation and management of large groups of objects according to some interests is described in chapter 5 of this deliverable.

Authenticate all the objects. Ideally, authenticate the object. As a second level, authenticate at least the group of the objects. Some authentication methods that are useful in the scenery of the Web of Objects were explored in chapter 4.

Always use authorization schemes to give access to private data. Some light authorization methods that are useful in the scenery of the Web of Objects were explored in chapter 4.

Privacy protection

Identify and the private data to be protected in the network, and label the data accordingly. This includes any form of direct authentication, like names or IDs, any information that must be kept private, as medical sensors or bank data, and any other piece of data that can be used to gain some knowledge of a user. Which data is private depends on the context and sometimes on the relation of the data and objects with their owners. If an object is collecting and sending its location to the network, the location is a private piece of data if it can be linked to a specific identity. For example, a car constantly sending its location to another object server can be linked to a user identity just monitoring the location at night, since this location will be with high probability the owner's home.

Limit the amount of personal data sent to other objects. Consider using a protocol that supports homomorphic encryptions, oblivious databases and secure multiparty computations. Also, consider adding distortion to the user's profiles before sending them to not trusted objects. The demonstrators of the Web of Object protocol show that, even if not all participants in the communication can be trusted, most services can be offered by means of the necessary

security protocols. These techniques were discussed in chapter 4 and 5 of this deliverable, for the specific cases of profile protection and recommendation systems.

Show different profiles depending on the context of the object. For example, an object may have different semantic descriptions depending on if it is at home (a trusted environment) or at the mall (an untrusted environment). In the former case, more private data can be shared. In the latter, not only less data must be share, but some of these data could be modified somehow. These techniques were described in chapter 5 of this deliverable.

Network and service protection

Create a separate network segment for those objects that cannot protect themselves. Not all of the objects in WoO are powerful enough to protect their data. For example, small sensors or security cameras may lack the resources to provide a secure environment. It the data they provide must be accessed from the outside, provide a proxy with enhanced security. Besides, these objects must be hidden in some secure environments, protected by other objects in the network. For example, by means of using virtual private networks, firewalls or demilitarized zones. All of these examples use traditional, well known security mechanisms.

Use dynamic keys inside the objects that change often. The use of built-in, pre-shared security parameters is not recommended in the Web of Objects, since some objects can be tampered, careless disposed or compromised over time. The objects must use some mechanism to update their security mechanism, especially their security keys, in a dynamic way. In chapter 5, we explored a key management mechanism suitable for large networks of small objects, suitable for the development of the project.

Deploy the objects taking into account it will need future updates. The objects in the Web of Objects are autonomous, but the operator must take into account they need to be updated when new vulnerabilities are discovered. A sensor deployed in the field for years will be unsecure for sure!

Consider the management of denial of service attacks, and at least consider the consequences of a denial of service attack. For example, in the case of an object controlling a lock, under a DoS attack, can the user access to her house?

Object protection

Include in the description of the object and its services some information about the expected security. If this information is included in the description of the object, the users of the object can modify its behaviour when interacting in the web of objects, or even decide not to participate in a communication with an unsafe object. The owner of the objects must decide about some policy when the security of the communication is not defined. In this case, not

trusting the remote object seems like the safest option, and objects can use their security profiles for the most unsafe contexts. For example, not participating in the communication, adding noise to the user's profile or not exposing all their services.

Keep your software up to date. Some of the devices used in the Web of objects are constrained devices (mobile phones, sensors, actuators), and not all of them are updated after new attack vectors are releases to the public. For example, the RESTful services offered by the Web of Objects rely on a web server installed on the different devices and some kind of additional web development framework at the server side (PHP, Python…) Web servers and web development frameworks such as Django are common targets for external attackers. They are complex pieces of software running on constrained devices, and sometimes they are not updated as often as necessary.

Do not consider the objects owned by a user secure because of this ownership. As a result of the last point, an object may be compromised even if it is owned by a trusted owner. The web of objects must take this fact into account, and objects shouldn't trust each other just based on this ownership. Hence, always use high levels of security regardless the current context of the user or the identity of the parties involved in the communication.

Consider also the human factor of the security. Finally, some of the objects are controlled by a human. This is the case of mobile phones or vehicles. In this case, the makers must take into account the user of the object may be not educated enough to understand the risks involved in the communication or provided the best configuration. The default configuration for an object must take into account the safest parameters, even if it endangers the provision of some services. If the maker decides to use an unsafe configuration as default to make this kind of services easier, the user must be informed each time the object try to use them.


WoO
web of objects

8. References

[BARTH] A. Barth, J. C. Mitchell, J. Rosenstein (2004). Conflict and Combination in Privacy Policy Languages.

[CYC-1] Cycorp, “The Syntax of CycL”, March 2002

[CYC-2] Aalborg University, “Integrating Ontologies: Assessing the use of the Cyc Ontology for Cadastral Applications”, N/A

[DAML+OIL-1]SRI International, “Access Control and Data Integrity for DAML+OIL and DAML-S”, N/A

[DiCoMa] Disaster Control Management (DiCoMa). ITEA2 Project. http://www.dicoma.eu/

[EUDIR] Wikipedia, “European Union Directive”, link: http://en.wikipedia.org/wiki/European_Union_Directive

[EC0031] Directive 2000/31/EC of the European Parliament and of the Council of 8 June 2000 on certain legal aspects of information society services, in particular electronic commerce, in the Internal Market ('Directive on electronic commerce'). Official Journal L 178, 17/07/2000 P. 0001 – 0016

[EC0258] Directive 2002/58/EC of the European Parliament and of the Council of 12 July 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector (Directive on privacy and electronic communications). Official Journal L 201, 31/07/2002 P. 0037 – 0047

[EC0624] Directive 2006/24/EC of the European Parliament and of the Council of 15 March 2006 on the retention of data generated or processed in connection with the provision of publicly available electronic communications services or of public communications networks and amending Directive 2002/58/EC. Official Journal L 105, 13/04/2006 P. 0054 – 0063

[EC09136] Directive 2009/136/EC of the European Parliament and of the Council of 25 November 2009 amending Directive 2002/22/EC on universal service and users’ rights relating to electronic communications networks and services, Directive 2002/58/EC concerning the processing of personal data and the protection of privacy in the electronic communications sector and Regulation (EC) No 2006/2004 on cooperation between national authorities responsible for the enforcement of consumer protection laws Text with EEA relevance, Official Journal L 337, 18/12/2009 P. 0011 – 0036

[EC9546] Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data. Official Journal L 281, 23/11/1995 P. 0031 – 0050

[EC9766] Directive 97/66/EC of the European Parliament and of the Council of 15 December 1997 concerning the processing of personal data and the protection of privacy in the telecommunications sector. Official Journal L 024, 30/01/1998 P. 0001 – 0008

[EC-IOT] European Commission. “IoT Privacy, Data Protection, Information Security“. European Commission Website, Digital Agenda for Europe, Newsroom: Internet of Things Factsheet Privacy and Security. News Section,28/02/2013.

[ECPET-1] European Commission, “Communication from the Commission to the European Parliament and the Council on promoting data protection by privacy-enhancing technologies” [COM (2007) 28], 2007

[ECPET-2] London Economics, “Study on the economic benefits of Privacy enhancing technologies (PETs)”, Final Report for the European Commission, 2010

[FIDIS] FIDIS Consortium (Future of Identity in the Information Society) (2005). D3.8: Study on protocols with respect to identity and identification – an insight on network protocols and privacy-aware communication.

[FIP-1] Robert Gellman, “Fair Information Practices: A Basic History”, 2010

[FTC-1] Federal Trade Commission, “Privacy Online: A Report to Congress”, June 1998

[FTC-2] Federal Trade Commission, “Protecting Consumer Privacy in an Era of Rapid Change: Recommendations For Businesses and Policymaker”, March 2012

[Garcia06] O. Garcia-Morchon et al. “Security Considerations in the IP-based Internet of Things”. Internet draft. http://tools.ietf.org/html/draft-garcia-core-security-06

[GeoXACML] GeoXACML WebSite: http://www.geoxacml.org/

[GML-1] OGCI – Open Geospatial Consortium Inc., “OGIS Geography Markup Language (GML) Encoding Standard”, 2007

[GML-2] OGCI – Open Geospatial Consortium Inc., “OGC Geography Markup Language (GML) – Extended schemas and encoding rules”, January 2012

[GML-3] PennState, “Introduction to Geographic Markup Language (GML)”, N/A

[IBM] IBM, Günter Karjoth, Matthias Schunter, and Michael Waidner (2002). Platform for Enterprise Privacy Practices: Privacy-enabled Management of Customer Data

[KIF-1] draft proposed American National Standard (dpANS), “Knowledge Interchange Format”, N/A

[MILO-1] Tecknowledge, “MILO (Mid-Level Ontology)”, April 2003

[Murrill2012] “Smart Meter Data: Privacy and Cybersecurity”. Congressional Research Service, 2012. https://www.fas.org/sgp/crs/misc/R42338.pdf

[OASIS] OASIS Website, Security Services: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security

[OECD] OECD, “OECD Guidelines on the Protection of Privacy and Transborder Flows of Personal Data”, September 1980

[OCML-1] The Open University, “IRS-III: A broker-based approach to semantic Web services”, Journal of Web Semantics, January 2008

[O&M-1] OGCI – Open Geospatial Consortium Inc., “Observations and Measurements - XML Implementation 2.0”, March 2011

[OWL-1] W3C Recommendation, “OWL Web Ontology Language Reference”, February 2004

[OWL-2] University of Aix-Marseille, “Introduction au language « Ontology Web Language » (OWL)”, march 2009

[OWL-S-1] W3C Member Submission, “OWL-S: Semantic Markup for Web Services”, November 2004

[OWL 2-1] W3C Recommendation, “OWL 2 Web Ontology Language Document Overview”, October 2009

[PBD-1] I. Rubinstein, “Regulating Privacy by Design”, Information Law Institute, NYU School of Law 2011

[PBD-2] A. Cavoukian Ph.D., “Privacy By Design”, 2009

[PBD-3] E. Ramirez, “Privacy By Design and the New Privacy Framework of the U.S. Federal Trade Commission”, Privacy by Design Conference, Hong Kong June 13th, 2012

[PRIOPT] J. Bouckaert and H. Degryse, “Opt In Versus Opt Out: A Free-Entry Analysis of Privacy Policies”, 2005

[PRIVHM] Interactive Data Protection Heat Map, Forrester Research, Inc, 2011. Available online in: http://heatmap.forrestertools.com/

[RAE-1] Royal Spanish Academy, “Dictionary of the Spanish Language of the Royal Spanish Academy”, 22nd Edition, 2001

[RDF-1] W3C Recommendation, “RDF Primer”, February 2004

[RDF-2] Bhavani Thuraisingham, “Security standards for the semantic web”, Computer Standards & Interfaces 27, Elsevier, August 2004

[RDFS-1] W3C Recommendation, “RDF Vocabulary Description Language 1.0: RDF Schema”, February 2004

[RIF-1] W3C Working Group Note, “RIF Overview”, June 2010

[RIF-2] W3C Recommendation, “RIF RDF and OWL Compatibility”, June 2010

[SAMEL] OASIS Standard, “Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) v1.1”, September 2003

[SA-REST-1] W3C Member Submission, “SA-REST: Semantic Annotation of Web Resources”, April 2010

[SA-REST-2] University of Georgia & Wright State University, Dayton, “SA-REST and (S)mashups: Adding Semantics to RESTful Services”, N/A

[SA-REST-3] University of Georgia, “SA-REST: Bringing the power of semantics to REST-Based Web services”, 2005

[SAWSDL-1] W3C Recommendation, “Semantic Annotations for WSDL and XML Schema”, August 2007

[SAWSDL-2] W3C Working Draft, “Web Services Description Language (WSDL) Version 2.0: RDF Mapping”, May 2007

[SAWSDL-3] Wright State University, “Beyond SAWSDL – a game plan for broader adoption of semantic Web Services”, IEEE Intelligent Systems Trends & Controversies, Nov-Dec 2007

[SensorML-1] OGCI – Open Geospatial Consortium Inc., “OGIS Sensor Model Language (SensorML) Implementation Specification”, July 2007

[Serrano13] Juan Hernández-Serrano, Juan Vera del Campo, Josep Pegueroles, Carlos Gañán: Low-cost group rekeying for unattended wireless sensor networks. Wireless Networks 19(1): 47-67 (2013)

[SHOE-1] SHOE Project proposed specification, “SHOE 1.01”, April 2000

[SKOS-1] W3C Recommendation, “SKOS Simple Knowledge Organization System Reference”, August 2009

[SOLOVE] Daniel J. Solove, “A taxonomy of privacy”, University of Pennsylvania, Law Review, vol. 154, nº3, 2006

[SOS-1] OGCI – Open Geospatial Consortium Inc., “OGC Sensor Observation Service Interface Standard”, April 2012

[SUMO-1] IEEE SUO Working Group, “SUMO (Suggested Upper Merged Ontology) 1.73”, 2004

[SUMO-2] Adam Pease, “The suggested Upper Merged Ontology (SUMO) at Age 7: Progress and Promis”, Articulate Software, September 2007

[SUO-KIF-1] IEEE SUO Working Group, “Standard Upper Ontology Knowledge Interchange Format”, April 2003

[SUO-KIF-2] Adam Pease, “Standard Upper Ontology Knowledge Interchange Format”, February 2004

[SWAN-1] W3C Interset Group, “Semantic Web Applications in Neuromedicine (SWAN) Ontology”, October 2009

[SWRL-1] W3C Member Submission, “SWRL: A Semantic Web Rule Language Combining OWL and RuleML”, May 2004

[SWS-1] The Open University & Wright State University, “Semantic Web Services”, 2010

[SWS-2] Yuxiao Zhao, “Combining RDF and OWL with SOAP for Semantic WebServices”, Programming Environment Laboratory (PELAB), N/A

[SWSL-1] W3C Member Submission, “Semantic Web Services Language (SWSL)”, September 2005

[TML-1] OGCI – Open Geospatial Consortium Inc., “Transducer Markup Language (TML) Implementation Specification”, 2007

[UDHR] The Universal Declaration of Human Rights of 1948, available online in the United Nations Web site: http://www.un.org/en/documents/udhr/

[Vera2012] Juan Vera del Campo, Josep Pegueroles, Juan Hernández-Serrano, Miguel Soriano: Design of a P2P content recommendation system using affinity networks. Computer Communications 36(1): 90-104 (2012)

[Vera2013] “DocCloud: a document recommender system on cloud computing with plausible deniability” Information Sciences. Volume 258, 10 February 2014, Pages 387–402

[W3C-1] W3C Website. All Standards and Drafts: http://www.w3.org/TR/#tr_P3P

[WSPL] Anne H. Anderson (2004). An Introduction to the Web Services Policy Language (WSPL).

[WSAN-1] Research Institute of Posts and Telecoms, Post and Telecommunications Institute of Technology, Hanoi University of Technology, “Wireless Sensor Actor Networks and routing performance analysis”, N/A

[WSAN-2] MDPI sensors, “Wireless Sensor/Actuator Network Design for Mobile Control Applications”, October 2007

[WSAN1] G. F. Coulouris, Distributed Systems: Concepts and Design. Addison Wesley, 2005.

[WSAN2] M. M. Wang, J. N. Cao, J. Li, and S. K. Dasi, "Middleware for Wireless Sensor Networks: A Survey," Journal of Computer Science and Technology, vol. 23, pp. 305-326, May 2008.

[WSDL-S-1] W3C Member Submission, “Web Service Semantics – WSDL-S”, November 2005

[WSML-1] W3C Member Submission, “Web Service Modeling Language (WSML)”, June 2005

[WSMO-1] W3C Position paper, “Web Service Modeling Ontology (WSMO) – An Ontology for Semantic Web Services”, June 2005

[WSMO-2] OASIS Symposium 2006, “Applying Semantics to Service Oriented Architectures”, OASIS Semantic Service Oriented Architecture Tutorial, The Meaning of Interoperability, May 2006