Towards Interoperability with Ontologies and Semantic Web Services in Manufacturing Domain

Nelson Rodrigues[1,2]
[1] Faculty of Engineering of the University of Porto
Rua Dr. Roberto Frias, s/n 4200-465, Porto, Portugal
[2] Polytechnic Institute of Bragança, Campus Sta Apolonia, Apartado 1134,
5301-857 Bragança, Portugal
nrodrigues@ipb.pt

***Abstract*—Nowadays, companies are very dynamic, thus increasing their competitiveness is mandatory. This competitiveness is associated with dynamics and flexibility, a fast product changeover is needed. Interoperability has an important role to implement these features that companies need, in order to reduce time, effort and money. This paper describes how the production process can be improved with semantic models. With technical and methodological review through re-configuration low-level devices with Service Oriented Architecture and explores how Semantic Web Services can assist on this domain. The paper reviews the literature and points out the current research focusing Semantic Web Services and Ontologies applied to the manufacturing domain, and how interoperability in this field has proven to be essential.**

Keywords—Interoperability, Semantic Web Services, Ontologies, Manufacturing

I. INTRODUCTION

Currently, in the field of manufacturing, companies spend a lot of money and time installing new products and changes in production, such as configuration. It is necessary to adapt and optimize these processes to make them more dynamic in its reconfiguration. Nowadays, companies are very dynamic, thus increasing their competitiveness is mandatory. This competitiveness is associated with dynamics and flexibility, which are reached through solutions and interoperable infrastructures. Interoperability has an important role here and a huge impact, so creating dynamic, interoperable systems will bring several benefits, such as saving companies money, the ability to produce products in mass and quickly, and leaving an open door to, in the future, be able to integrate new processes without effort. *"it is estimated that 70% of the engineering teams' effort is involved in re-implementing the control"* [1].

In this way, technologies based on Services Oriented Architecture (SOA) become quite important to perform this passage. Collaboration between entities can be reflected in diverse domains. The current research direction is focused on developing semantic contracts among collaborating partners. Nevertheless, interoperability brings some problems, and it is necessary to identify them. There is the necessity to have a type of contract/meta-document among the entities, or different domains in the same company. This meta-document has the description and the semantics of the terms used. This problem can be solved with services and ontologies, which can clearly distinguish the semantics of terms. The Semantic Web Services is the connection of interoperable services with the semantics of the terms in a specific domain. With this expected dynamic it is necessary to re-think the companies IT architectures.

The remainder of this paper is as follows: Section II it briefly introduces the ontologies moreover their components and methodologies. Section III discusses the methodologies of SOA mechanisms. Section IV introduces the case study domain**.** Section V focuses on the engineering methodology on manufacturing according to Semantics Web Services. Finally, Section VI presents the paper results and section VII rounds up the paper with the conclusions.

II. ONTOLOGIES

The term "ontology" originates from the philosophy domain that has been adopted in Computer Silences, even though vague and not precise. Ontologies have been gradually used because of the need to represent knowledge in an area that has gained more interest in Semantic Web. Among the several definitions of ontology that can be found in the literature, the following one can be pointed out as the main definition:

An ontology "*is a formal, explicit specification of a shared conceptualization*". [2]

In face of this definition two different modelling layers can be described. The conceptualization level defines the concepts and relations among them, i.e., a way how to view a model from one perspective. The specification level specifies the conceptualization, in other words, is how formally (formal language) specifies how the world is seen.

In the computational world, ontologies are one way to describe, computationally, processable knowledge, but also to increase communication between computers and humans. There are three main reasons for using ontologies [3]:

1. Assist in communication between humans and computers.
2. Achieve interoperability between software systems.

3. Help improve the quality of design and system architecture software.

To accomplish the previous reasons, ontologies are developed taking into consideration knowledge reuse, sharing:

A. *Components*

Basically the smallest ontology is defined by a triple, namely the subject, predicate and object or in other terms concepts, relations and attributes.

Concepts are expressions that indicate domain entities with a complex structure that can be defined in terms of classes or objects, e.g., *Product*: (is a finished or semi-finished entity that is produced by the enterprise in a value-adding process). *Relations* or predicates establish the relationships among the concepts, e.g., *hasOperation (x, y): (*process plan x contains operation y). *Attributes* are values relative to properties of concepts, e.g., productID: a non-negative integer number that provides the unique identification of the product. *Restrictions* are conditions that should be satisfied when instantiating a class. Restrictions can be applied to the predicates, defining the range, domain and cardinality of the classes involved in the relation; and to the attributes of one class, defining the range and domain. In Fig 1. the ontology components are described, as well as how they are related.

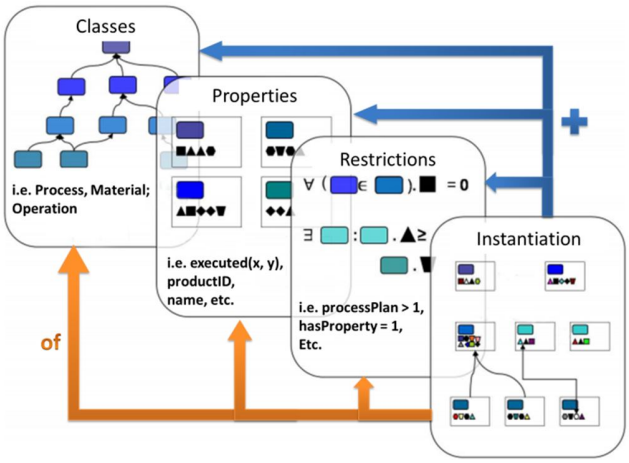

Fig.1. Ontology representation levels [47]

In the next sections, the several ontological components will be analysed. A crucial point is how to represent the ontology knowledge, i.e., the definition of classes, properties and relationships among classes. For this purpose a methodology should be followed.

B. *Methodology*

In literature it is possible to find particular frameworks that describe the stages step-by-step. Noy and McGuinness propose a methodology for the development of ontologies [4], for instance, Gruber proposed some principles [5]; the terms used in the ontology must be clear; the ontology should avoid doubts and misunderstandings about the terms used; the ontology design should support an easy expansion; among others.

The main idea in the development process of an ontology is to verify if existing ontologies can accomplish the proposed requirements, aiming to reuse ontologies; if the requirements are not accomplished, the option is to move to the next phases as illustrated in Fig 2.

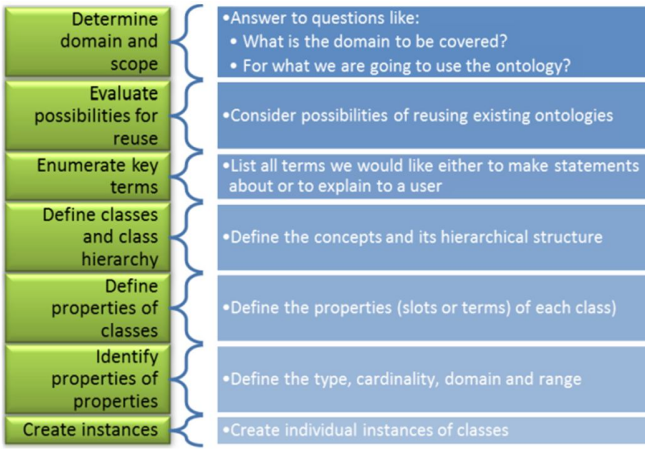

Fig.2. Methodology to build proposed ontologies [4].

As the ontology development evolves, it should exist a need for continuous evaluation of the ontology. At the end of a good agreement between the domain expert, users, and ontologies engineers, the ontology is concrete for that domain and generic to future improvements.

C. *Ontology Languages*

Nowadays, there are several languages to describe ontologies, giving here more attention to the more recent ones.

Resource Description Framework (RDF) [6] is one language used to develop ontologies based on the markup languages, e.g., the SGML (Standard Generalized Markup Language) and the XML (eXtensible Markup Language) [7]. Since XML is a declarative language, being quite limited, RDF appears to overcome these limitations, e.g., in terms of relations. RDF is used for representing information about resources on the web, thus constituting a basic ontology language. In RDF, the statements used to describe resources are represented as triples, consisting of a subject, predicate and object, i.e., {S, P, O}. The RDF(S) (Resource Description Framework Schema) is a semantic extension of RDF, which allows describing taxonomies of classes and properties, supporting the demand for creating a schema. The Web Ontology Language (OWL) [8] is another markup language that semantically extends RDF and RDFS, it derives from the DAML + OIL (DARPA Agent Markup Language - Ontology Inference Layer) [9]. OWL has a rich set of modelling constructors, offering improved pre-defined

templates, e.g., supporting the inclusion of restrictions in the concepts and predicates. OWL has a reasoning layer that allows representing an ontology in a more expressive manner.

D. *Ontology Frameworks*

The development of ontologies is a complex task that requires the support of proper frameworks which assist the creation or manipulation of ontologies and are able to express ontologies in one of many ontology languages. The use of these tools may lead to a more productive task in the design of ontologies, supporting the concurrent work of the ontology engineers and the domain experts. Several frameworks are currently available, namely OntoEdit [10], WebODE [11], Protégé [12] and Hozo [13]. Even the Protégé API can be used just like an API (Application Programming Interface). This API is implemented in Java and is essentially the same as Protégé, only without the graphic component; this API is to be used in conjunction with JENA [14]. Jena is a common framework that can be used in several approaches. It can be used individually but, it is explicitly used as the basis of Protégé API.

E. *Ontologies for manufacturing*

Ontologies are used in several and divers domains. In this paper the field is limited to the manufacturing domain.

In EU FP7 GRACE (inteGration of pRocess and quAlity Control using multi-agEnt technology) project, an manufacturing ontology was developed [15] to handle the knowledge exchanged among the Multi-agent system [16]. The EU FP6 PABADIS'PROMISE project proposed a reference meta-ontology for manufacturing [17]. ADACOR (ADAptive holonic COntrol aRchitecture for distributed manufacturing systems) for the manufacturing control domain, which was formalized with the DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering) language [18]. MASON (Manufacturing's Semantics Ontology) introduces an ontology, in order to unify the ontologies using cognitive architectures, leaving to an implementation of a generic manufacturing ontology [19]. Other attempts to establish generic manufacturing ontologies are the NIST's description of shop data model [20], the Automation Objects [21], OOONEIDA focusing on the infrastructure of automation components by applying the semantic web technologies [22], and TOVE (Toronto Virtual Enterprise Ontology) that describes an ontology for virtual enterprise modelling [23]. The ISO 15926 standard [24] aims to support the integration of industrial automation systems. The challenge in manufacturing capability modelling lays in developing conceptual capability models that characterize several features of manufacturing, in terms of abstraction as well as formalization of the model.

III. SOA

This section describes the methodologies in Services-Oriented-*. SOC (computing) and SOA increase dramatically the services interoperability applied at inter-enterprises or intra-enterprises layers. The key concepts about SOA, are integration and reuse. SOA became very popular due to its features, which are very easy to implement and expand.

A. *SOA Components and methodology*

Nowadays SOA is a very popular architecture, due to an excellent solution to the many challenges of the current business, namely: providing a large component of agility through a quick response, and adaptability to changes, allowing companies to save time and money.

In SOA, one of the features is to minimize the relation of dependencies. This stateless services need to be dynamic. SOA follows certain principles, such as interoperability among the systems, reuse, granularity, modularity and componentization. Also it offers several services as: standardized service contract, loose coupling, service abstraction, service reusability, service statelessness, among others. Commonly there are three actions associated to a SOA: discovery, request and response, as it is described in Fig. 3, where three major entities are illustrated as well their own actions. The first action is the registration process (step 1), where the provider registers the services that can be performed. A discovering process of finding the service that provides the functionality that is required to the discovery service is usually called UDDI (*Universal Description, Discovery and Integration*) (step 2).

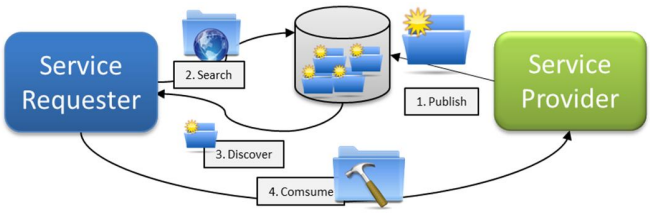

Fig.3. Service Oriented Architecture

The service provider receives from the UDDI Registry the interface needed (step 3) to invoke the service of the provider (step 4); the reply yields the output from the service (step 5).

B. *SOA Languages*

In the publish step, the protocol Web Services Description Language (WSDL) is used, in the following SOAP (Simple Object Access Protocol) messages are used, which are XML-based protocols that allows applications to exchange information. Also a XML-based protocol (the WSDL language) is responsible for describing Web services and how to access them, the definitions of ports, service name, operation, message, bindings and types. The message is well separated from its concrete instance, at the end is an interface description prepared to be reused. Thus an agreement, known as SLA (Service Level Agreement), is

necessary, responsible to handle the negotiated agreement between two entities; the service contract is then strictly defined. In what regards to the service contract, its anatomy is described in a WSDL, XML schema, and WS-Policy definition.

These concepts are well integrated due to the fact that there is good incorporation with collaborative automation, in the sense of self-governing, reusable and loosely-coupled distributed components. Also, due to this effort, modeling tools with Web Services protocols was developed to deal in a more abstract way, from the BPEL **(**Business Process Execution Language) [25] to the WSBPEL (Web Services Business Process Execution Language) [26], and WSFL (Web Services Flow Language) [27] proposed by IBM. The purpose is to assist on the modeling, which should be so abstract that if we put a new device and new processes in the automation system, they can integrate in order to achieve the objective: total integration. Some process can became automated to assist on the modelling, such as:

- **Orchestration** is an automatic and coordinated management of services taking into account a set of centralized services into a single one. In other words, consists on the combination of services to produce a more complex and useful services.
- **Choreography** describes each service as a service that knows exactly when to become active and with whom to interoperate, in a collaborative way. Both specifications should be implemented to make a more autonomous system.

C. *REST Web Services vs SOAP Web Services*

The major applications that fulfil the requisites of SOA use Web Services, which can be implemented in SOAP, REST and WSDL. However, there are several platforms to use/implement interoperability, e.g., REST, SOAP, XML-RPC, among others. Obviously there are some platforms that are more absorbed by the industry than others, perhaps because they bring short-term benefits, or because they are more easily implementable. REST is the simplest of all, being well regarded by the community for its simplicity. However the most significant difference is in relation to the interface definition, in RESTful systems is not necessary to describe one. This is the main reason why this implementation is simpler to use. Nevertheless there is some discrepancy regarding SOA implemented through REST, if it is, or not, a proper fulfilment of SOA; RESTful systems attempt to be implemented according SOA paradigm as seen in [28], it is alleged to be a Resource-Oriented Architecture (ROA) paradigm [29] and not SOA, where services are replaced by resources. One of the reasons of this paper is to identity which is the best platform, in a long term. REST technologies and SOAP, even if they are or not SOA compliant, both have their merits, but SOA becomes more semantic.

D. *SOA in manufacturing domain*

There is already some work in the manufacturing field, several European research project initiatives are available for consulting, based on the migration of industrial processes into service oriented architectures [30]: a FP7 project IMC-AESOP (www.imc-aesop.eu) [31], SOCRADES is a FP6 project addressing SOA-based in manufacturing (www.socrades.eu), focused on coupling web service enabled devices with enterprise applications [32]. Also in project SIRENA (www.sirena-itea.org), SOA is extended to a low-level domain such as embedded-devices (sensors and actuators) [33]. How to implement service-orientation through Multi-Agent Systems in industrial automation is described in [34]. A survey of the engineering of SOA is described in [35]. At [36] a practical example of device-level SOA is given.

IV. MANUFACTURING

Although companies realize the benefits of SOA implementation, they are still very apprehensive. Since usually when something is working well the main idea is to not change anything, but a restriction of evolution is placed every day on the company financial equation. As already demonstrated, the company must evolve. Then, why not implement it?

However industry has been slow when applying the agility and dynamic that SOA methodology offers, mainly because of the cost of replacing or develop from scratch their IT architecture, since most of the manufacturing companies have invested a considerable amount of money in manufacturing devices to handle the IT architecture. The massive computational power that has been developed in recent years is viewed as a disadvantage in addressing the problems in the companies today. Solutions were installed over time, of several application types, which enhance the automation or processing of each company's domain. Computational power is no longer so important. Nowadays the problematic is the integration/deployment of interoperable services. The developed solutions must assure trust and support along the time.

Thus, there is a problem in this temporal validation. After a system is developed and implemented in the production line, it takes years for refinement, validation and verification, thereby creating a dual problem. Firstly, the system must be generic as possible to leave open interoperability insights; secondly, the system must be specific enough to be able to accomplish the purpose for which it was developed. The process reengineering should be transparent, which it is not. In a long term, when a change is needed in a real model that has been used for years, according to the views of interoperability, it will be the most crucial test.

A. SOMAS

Manufacturing systems can be defined as "*a collection or arrangement of operations and process [...] to make (a) desired product(s) or component(s)*" [37]. To accomplish this concept a collaborative work must be performed between entities. The automation literature is replete of examples with Multi-Agent systems (MAS), which represent each entity in order to offer a solution to increase flexibility, distributed control, reduced complexity, etc. Currently trends in Service Oriented Multi-agent Systems (SOMAS) are being explored more often to increase interoperability, semantic descriptions, composition of services, among others, for further detail see [34] [35]. Additionally, some work related with agents based on ontology-based services to achieve interoperability is described in [46].

This way, such systems must assure modular capabilities, which mean that a system component can be divided into smaller components and mixed and matched in a variety of configurations. If the modularity is guaranteed by the system it is a good asset, and for the future one can implement new systems.

B. Manufacturing Standards

With the fast advance in technologies, the way how interoperable systems are developed should be rethought, because systems are developed taking into account present technology. In the future, technology will improve, and more systems and standards will appear, the question is how to create a system today that can be interoperable with the systems of yesterday and tomorrow. This is probably one of the major reasons why companies are so septic to implement such systems. The financial impact is very high to simple implement a system that only works in the manufacturing domain during one or two decades.

It is necessary to create rigid standards that assure this problem, in order to convince the companies that the system that they are using follows the standard points. The academic community is behind some standards to support and try to increase the implementation of SOA in the manufacturing domain. Also it is necessary to understand the industrial problems and solutions. On other hand, the industry must understand clearly the benefits of SOA and the openings that will appear with the academic research involvement, since industrial standards will be created.

These standards are also based on communication protocols, and to the message content specifications frameworks and architectures. However, there are many standards that already exist (as it will be seen next), but they do not consider the different knowledge entities. The problem with standards occurs when it is necessary to create two entities automatically interoperable; this approach is understandable in theory, but in practice it is very difficult to put two entities that were created from different approaches, and even without any of the methodologies, created without any standards.

V. Semantic Web Services & Ontologies

It is necessary to guarantee common understanding and data semantics among distributed entities (also reuse and share of knowledge). Ontologies can increase how the knowledge is expressed as it was seen in the first sections of this document. Moreover, ontologies can increase the semantic of the services, like the processes, how the knowledge is exchanged between two entities where both can understand the meaning [38]. It is mandatory to use a meta-model to exchange interoperability, and these Meta-Models can be made in several formats, namely in XML [7], RDF or OWL (Web Ontology Language) [8]. But in order to create interoperability this is not enough: it is necessary to implement services that express more than simple functions.

A. Semantic Web Services Motivation

In previous chapters it was mentioned that Web Services can offer features such as modular, self-describing, self-contained applications that are accessible over the Internet, being these, also, some of SOA characteristics.

Sometimes Web Services are confused with SOA, but SOA does not specifically mean Web Services. As an alternative, Web Services can be realized as a specialized SOA implementation. However, Web Services Description Language (WSDL) does not contain semantic descriptions of the operations, the notion is simple: join ontological notions in Web Services (WSDL). The combination of these two concepts makes Web Services more semantic, capable of expressing the semantics of their services. An evolution, taking into account this initiative, is illustrated in Fig. 4.

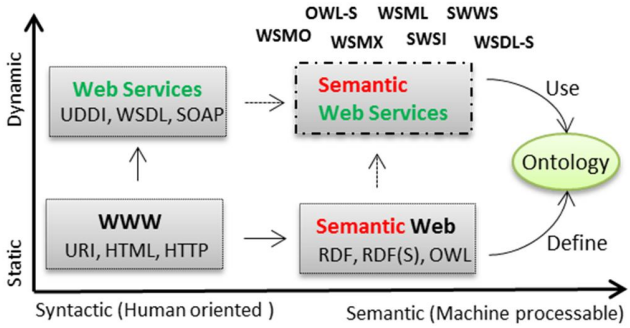

Fig.4. Evolution of the WEB. (adapted from [39])

The Web has experienced changes in its anatomy, becoming more dynamic and semantic.

B. Semantic Web Services Languages and Protocols

Approaches and initiatives which aim to specify Web Services using semantics and ontologies include: OWL-S [40], the SWSI, SWWS, WSML, WSMO and WSMX that can be view in more detail in [41]. WSDL-S [42] defines new elements and annotations for already existing elements, which offer a great potential to implement SWS.

C. Semantic Web Services in Manufacturing Domain

In the industry domain the Implementation of SOA can exist at different levels: on high-level and very similar to other companies (in the field of industry or not), or low-level, on devices of their own manufactures [42].

An IT infrastructure for the heterogeneous message communication available is Enterprise Service Bus (ESB) [43]. This infrastructure is probably the most used approach when a communication channel for SOA is needed. However this communication layer does not represent the meaning of each message. An FP6 project, SUPER (www.ip-super.org) [44], aims to achieve a higher degree of automation in discovery and mediation of co-operating services. The goal can be described by a "*Semantic Business Process Modeling"* [45], which is to follow a usage of semantic technologies, as Semantic Web Services, in the process modeling phase, creating a Semantic Service Bus (SSB) as an enhancement of the general ESB. Some projects offer semantically enhanced business process modeling and design of semantic ESB, such as: Object Management Group (www.omg.org), FP6 R&D project STASIS (www.stasis-project.net), and OPUCE (www.opuce.tid.es). The concept of SSB was also adopted in the SPIKE project (Secure Process-oriented Integrative Service Infrastructure for Networked Enterprises). The objectives of these projects are to recognize and provide observation in the manufacturing domain of the interoperable systems integration.

1) OWL-S

Mapping services and processes at a low-level domain in WSDL files, which describe the operations process is simple, the difficult part is to describe in semantics such services/operations in the WSDL file. One advantage of OWL-S is the specification of semantic concepts. OWL-S is represented by three main concepts (grounding, service and profile). It can automatically discover, invoke, compose and monitor resources, allowing then the offering of services. The challenge of integrating the approach involves two services that complement each other. The *ServiceProfiler* is responsible to fully describe the request service, namely what the service does. *ServiceGrounding* specifies how to use the service, in order to execute it. And *ServiceModel* gives information on how the service works.

2) WSDL-S

But WSDL-S can also be used to do this process. Having some benefits compared to the OWL-S, in particular the simplicity in the implementation transition and a wider range on what regards the types of modeling. WSDL-S is based on mapping annotations of a WSDL file, see Fig. 5 as an example. Therefore the selection process of services, discovering, description of services, invocation and composition, becomes more automatic and not dependent on the interpretation that each engineer intendeds to give, thus solving some terminology ambiguities that might exist.

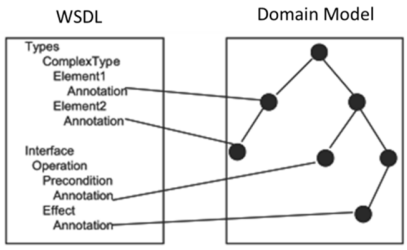

Fig.5. Association between WSDL and Ontologies from [42].

When using either WSDL-S or OWL-S it is necessary an automatic composition service. Web Services in WSDL can be matched against an announced OWL-S Web Service in an autonomous way [1]. A mediator is necessary to perform these and have the decision support centralized.

D. Practical example Mediators

To understand how the ontology concepts are associated with services and semantic used by the agents, let's get back to the example of SOA in the automation domain. One of the research paths that was followed, when placing in the services way the goals and functionalities, was to make this automatic switch, in order to achieve a machine-interpretable and human-interpretable transition, by defining the features and services of each device through Services notions. A Mediator should thus be used to aggregate services in SOA systems, applied to this domain.

The mediator is responsible for solving some mismatches in order to give to the systems the interoperability that they need. Another type of mediator is the "OO Mediator" that is responsible for mediating the ontologies, to merge, align and map, in order to retrieve integrated and homogeneous solutions. For example, if SOMAS is used, an agent can very easily provide a new service or a new process. In Fig. 6 the *Agent A* can easily reasoning that the service "*Dispatch pallet type B*" from the *Agent* B has similar features as the service "*Dispatch pallet A*", so if it is more rentable for divers variables to use different services, the agent can match, merge, discover, monitor or infer new services. The inference can be performed through the ontologies with the SWRL (Semantic Web Rule Language), where rules are used to assert a specific combination, e.g., the combination of the *hasParen*t and *hasBrother* properties implies the *hasUncle* property.

$$hasParent(?x1,?x2) \wedge hasBrother(?x2,?x3) \Rightarrow hasUncle(?x1,?x3)$$

In the manufacturing domain, some rules could be executed as some of them could be created implicitly during the process. Through SWS (Semantic Web Services) this step can be more easily automatized. The idea is to use a Mediator to take advantage of semantics. The composition of new services is according to the semantics of each one. Creating a semi-automatic composition performed by the *Mediator*.

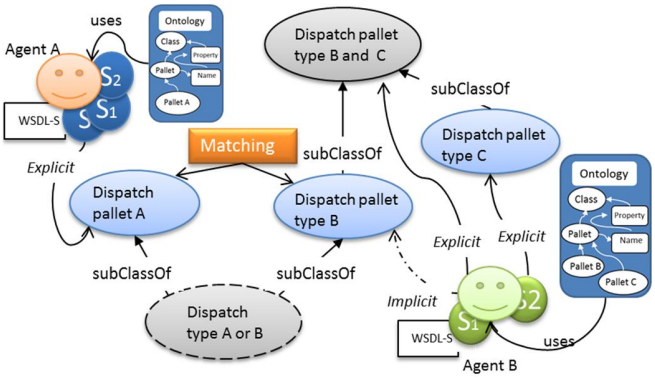

Fig.6. An example of reason matching services.

Centralized system has access to services in order to control the creation and integration of services.

E. *New knowledge for the services*

One very important point is the implementation of inference, know implementing the methods/techniques which will infer new facts or rules, making this inference an obvious reason in the context of this article, brings new relations and describes the best services. Thus the entities in charge of this service will emerge in new services, making them more semantic. To achieve that goal, some inference skills are necessary. Techniques asserted (asserted triples are those that were populated by triples from merging several sources) or inferred (are triples that were inferred by inference rules) can bring this type of new knowledge to create new or better services.

VI. Discussion

Implementing SOA architecture can be very difficult. As already seen, if well implemented it brings plenty of benefits, however if poorly implemented can harm a company financially and structurally. To achieve this, it is vital to follow some type of guidelines in order to not make mistakes. It is important to recognize the benefits, but just as important to know where the failures occur when doing this integration on a real company.

A. *Semantic benefits in Automation*

Semantic Web Services can be implemented in several domains, with several profits, but the key is highlighting the potential benefits of SWS in manufacturing. In an abstract way it is possible to simplify the development of flexible reducing development costs and time; create more robust systems; because is simpler, the software system maintenance will be easier as aggregation processes. In a low-level perspective allows assisting in automating service selection, fast reconfiguration, more agile automation, flexibility, without the need for system re-engineering.

B. *Before SOA, After SOA*

Right after implementation of a SOA system, it is expected to be more dynamic, to reuse and share services, more collaborative, very integrated and interoperable scenario. Adding SOA to the automation domain has clear benefits. Collaborate with industrial companies is mandatory to achieve a conclusion about SOA in industry. The industry, in order to be able and conscious that SOA brings benefits horizontally and vertically, must see the results, being these all about numbers and costs, and SOA can has a major influence in those results.

VII. Conclusions

This survey takes a trip along the current trends in manufacturing domain. By analysing the approaches of this paper, it is noticed that companies are ready to increase their responsiveness by changing from a static, inflexible and slow architecture to evolve into a dynamic, faster and agile one. The independencies in a typical operating model were tight coupled among systems of coordination and now, companies are improved to a loose coupling among systems of coordination.

Ontology and services can help on the heterogeneous conversations between the entities, creating then an interoperable system. SOA is perhaps the greatest revolution in business and industrial companies. That tends to link the functional processes of enterprises to the use of productive technologies. The sooner a company starts to use SOA, the sooner it will be ready to provide the best of services, thereby creating, in advance, more rivalry in relation to its competitors. In low-level SOA implementations, namely SOA in automation industry, it enables companies to perform an optimized business management, and a better final product allowing reconfiguration at real time. The engineers' efforts can be focus on dynamic systems in order to create such system that allows to infer new processes.

However this work is not finished, it is necessary to perform a validation. The interoperability of the system will be put into test when the re-engineering step arrives, so it is necessary to make sure that the system is interoperable. It is mandatory, in the future, the implementation of real scenarios to get real results, being a path to follow so that these approaches are absorbed by the manufacturing industry. Another path to follow is the large computational power spent to create this easy integration to be on a Cloud and profit from its benefits and reduced costs.

References

[1] A. W. Colombo, F. Jammes, H. Smit, R. Harrison, J. L. M. Lastra, and I. M. Delamer, "Service-oriented architectures for collaborative automation," in *Industrial Electronics Society, 2005. IECON 2005. 31st Annual Conference of IEEE*, 2005, p. 6 pp.

[2] T. R. Gruber, "Toward principles for the design of ontologies used for knowledge sharing," *Int. J. Hum.-Comput. Stud.*, vol. 43, no. 5–6, pp. 907–928, 1995.

[3] M. Uschold, V. R. Benjamins, B. Ch, A. Gomez-perez, N. Guarino, and R. Jasper, "A Framework for Understanding and Classifying Ontology Applications," 1999.

[4] N. F. Noy and D. L. Mcguinness, "Ontology Development 101: A Guide to Creating Your First Ontology," 2001.

[5] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowl. Acquis.*, vol. 5, no. 2, 1993, pp. 199–220.

[6] O. Lassila, R. R. Swick, W. Wide, and W. Consortium, "Resource Description Framework (RDF) Model and Syntax Specification." 1998.

[7] W3C, "XML. Extensible markup lan guage (XML) 1.0." .

[8] W3C, "OWL. OWL Web ontology language overview," 2004. [Online]. Available: www.w3.org/TR/2004/REC-owl-features-20040210/.

[9] I. Horrocks, "DAML+OIL: a Description Logic for the Semantic Web," *IEEE Data Engineering Bulletin*, vol. 25, pp. 4–9, 2002.

[10] Y. Sure, M. Erdmann, J. Angele, S. Staab, R. Studer, and D. Wenke, "Ontoedit: Collaborative ontology development for the semantic web," 2002, pp. 221–235.

[11] Óscar Corcho, M. Fernéndez-López, A. Gómez-Pérez, and Óscar Vicente, "WebODE: An integrated workbench for ontology representation, reasoning, and exchange," in *IN: PROCEEDINGS OF EKAW 2002. LNCS 2473*, 2002, pp. 138–153.

[12] J. H. Gennari, M. A. Musen, R. W. Fergerson, W. E. Grosso, M. Crubzy, H. Eriksson, N. F. Noy, and S. W. Tu, "The evolution of Protégé: an environment for knowledge-based systems development," *Int. J. Hum.-Comput. Stud.*, vol. 58, no. 1, pp. 89–123, 2003.

[13] K. Kozaki, Y. Kitamura, M. Ikeda, and R. Mizoguchi, "Hozo: An Environment for Building/Using Ontologies Based on a Fundamental Consideration of Role and Relationship," in *Proc. of EKAW2002*, 2002, pp. 213–218.

[14] HP, "Jena - A Semantic Web Framework for Java." 2002.

[15] P. Leitao, N. Rodrigues, C. Turrin, A. Pagani, and P. Petrali, "GRACE ontology inteGrating pRocess and quAlity Control," in *IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society*, 2012, pp. 4348–4353.

[16] P. Leitao and N. Rodrigues, "Multi-agent system for on-demand production integrating production and quality control," in *Proceedings of the 5th international conference on Industrial applications of holonic and multi-agent systems for manufacturing*, 2011, pp. 84–93.

[17] L. Ferrarini, C. Veber, A. Luder, J. Peschke, A. Kalogeras, J. Gialelis, J. Rode, D. Wunsch, and V. Chapurlat, "Control Architecture for Reconfigurable Manufacturing Systems: the PABADIS'PROMISE approach," in *Emerging Technologies and Factory Automation, 2006. ETFA '06. IEEE Conference on*, 2006, pp. 545–552.

[18] S. Borgo and P. Leitao, "The Role of Foundational Ontologies in Manufacturing Domain Applications," in *On the Move to Meaningful Internet Systems 2004: CoopIS, DOA, and ODBASE*, vol. 3290, R. Meersman and Z. Tari, Eds. Springer Berlin / Heidelberg, 2004, pp. 670–688.

[19] S. Lemaignan, A. Siadat, J.-Y. Dantan, and A. Semenenko, "MASON: A Proposal For An Ontology Of Manufacturing Domain," in *Distributed Intelligent Systems: Collective Intelligence and Its Applications, 2006. DIS 2006. IEEE Workshop on*, 2006, pp. 195–200.

[20] C. Mclean, Y. T. Lee, G. Shao, and F. Riddick, "Shop data model and interface specification," in *NISTIR 7198, National Institute of Standards and Technology*, 2005.

[21] O. J. L. Orozco and J. L. M. Lastra, "Using semantic web technologies to describe automation objects," *International Journal of Manufacturing Research*, vol. 1, no. 4, pp. 482–503, 2007.

[22] V. Vyatkin, J. Christensen, J. L. M. Lastra, and F. Auinger, "OOONEIDA: an open, object-oriented knowledge economy for intelligent distributed automation," in *Industrial Informatics, 2003. INDIN 2003. Proceedings. IEEE International Conference on*, 2003, pp. 79–88.

[23] M. S. Fox, "The TOVE Project Towards a Common-Sense Model of the Enterprise," in *Proceedings of the 5th international conference on Industrial and engineering applications of artificial intelligence and expert systems*, 1992, pp. 25–34.

[24] R. Batres, M. West, D. Leal, D. Price, K. Masaki, Y. Shimada, T. Fuchino, and Y. Naka, "An upper ontology based on ISO 15926," *Computers & Chemical Engineering*, vol. 31, no. 5–6, pp. 519–534, 2007.

[25] F. Curbera, R. Khalaf, N. Mukhi, S. Tai, and S. Weerawarana, "The next step in Web services," *Commun. ACM*, vol. 46, no. 10, pp. 29–34, 2003.

[26] M. Kloppmann, D. Koenig, F. Leymann, A. Rickayzen, C. von Riegen, P. Schmidt, and I. Trickovic, "WS-BPEL Extension for People - BPEL4People," 2005.

[27] F. Leymann, "WSFL. Web services flow language," 2001.

[28] T. Erl, B. Carlyle, C. Pautasso, and R. Balasubramanian, *SOA with REST: Principles, Patterns & Constraints for Building Enterprise Solutions with REST*. 2012.

[29] Leonard Richardson and Sam Ruby, *RESTful Web Services Web services for the real world*. O'Reilly Media, 2007, p. 454.

[30] T. B. Jerker Delsing, Fredrik Rosenqvist, Oscar Carlsson, Armando W. Colombo, "Migration of Industrial Process Control Systemsinto Service Oriented Architecture," in *IECON*, 2012.

[31] S. Karnouskos, A. W. Colombo, F. Jammes, J. Delsing, and T. Bangemann, "Towards an architecture for service-oriented process monitoring and control," in *IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society*, 2010, pp. 1385–1391.

[32] A. Cannata, M. Gerosa, and M. Taisch, "SOCRADES: A framework for developing intelligent systems in manufacturing," in *Industrial Engineering and Engineering Management, 2008. IEEM 2008. IEEE International Conference on*, 2008, pp. 1904–1908.

[33] F. Jammes, H. Smit, J. L. M. Lastra, and I. M. Delamer, "Orchestration of service-oriented manufacturing processes," in *Emerging Technologies and Factory Automation, 2005. ETFA 2005. 10th IEEE Conference on*, 2005, vol. 1, p. 8 pp. –624.

[34] J. M. Mendes, F. Restivo, P. Leitão, and A. W. Colombo, "Injecting service-orientation into multi-agent systems in industrial automation," in *Proceedings of the 10th international conference on Artifical intelligence and soft computing: Part II*, 2010, pp. 313–320.

[35] J. M. Mendes, P. Leitão, F. Restivo, A. W. Colombo, and B. A, "Engineering of service-oriented automation systems: a survey," in *3rd I*PROMS Virtual International Conference on Innovative Production Machines and Systems*, 2007.

[36] L. M. S. De Souza, P. Spiess, D. Guinard, M. Köhler, S. Karnouskos, and D. Savio, "SOCRADES: a web service based shop floor integration infrastructure," in *Proceedings of the 1st international conference on The internet of things*, 2008, pp. 50–67.

[37] J. T. Black, *The Design of the Factory with a Future*. 1991.

[38] L. Obrst, "Ontologies for semantically interoperable systems," in *Proceedings of the twelfth international conference on Information and knowledge management*, 2003, pp. 366–369.

[39] F. Curbera, W. A. Nagy, and S. Weerawarana, "Web Services: Why and How," in *In OOPSLA 2001 Workshop on Object-Oriented Web Services. ACM*, 2001.

[40] *OWL-S. OWL-based Web service ontology*. 2004.

[41] J. Cardoso, *Semantic Web Services: Theory, Tools and Applications*. Hershey, PA, USA: IGI Publishing, 2007.

[42] R. Akkiraju, J. Farell, J. A. Miller, M. Nagarajan, A. Sheth, and K. Verma, "Web Service Semantics - (WSDL-S)," in *{W3C} Workshop on Frameworks for Semantics in Web Services*, 2005.

[43] D. Chappell, *Enterprise Service Bus: Theory in Practice*. 2004.

[44] D. Karastoyanova, B. Wetzstein, T. van Lessen, D. Wutke, J. Nitzsche, and F. Leymann, "Semantic Service Bus: Architecture and Implementation of a Next Generation Middleware," in *Proceedings of the 23rd International Conference on Data Engineering Workshops, ICDE 2007, 15-20 April 2007, Istanbul, Turkey*, 2007, pp. 347–354.

[45] M. Hepp, F. Leymann, J. Domingue, A. Wahler, E. Wahler, and D. Fensel, "Semantic Business Process Management: A Vision Towards Using Semantic Web Services for Business Process Management," in *In Proceedings of the IEEE ICEBE 2005*, 2005, pp. 535–540.

[46] A. Malucelli, "Ontology-based services for agents interoperability", Doctoral thesis, University of Porto, 2006.

[47] Z. Sun (2009) "Using ontology and semantic web services to support modeling in systems biology", Doctoral thesis, University College London, 2009.