Maracay, Venezuela
.

Ellaborated in January, 2016//

Onto*Smart* Applied in the Manufacturing Domain

Luis Ramos

Document Version: 0.1

Contents

1 Changes . . . 3
2 Report Scope . . . 4
3 Brief Methodology Description . . . 5
4 Results . . . 9
4.1 Initial Activities . . . 9
4.2 Domain Specifications . . . 9
4.3 Competency Questions . . . 10
4.4 Knowledge Acquisition . . . 11
4.5 Finding and Measuring Ontology Quality . . . 11
4.6 Finding Ontologies . . . 11
4.7 First Quality Control . . . 13
4.8 Performing Competency Questions . . . 14
4.9 Mapping, merging and importing . . . 20
4.10 Hyper Modules Extraction . . . 28
5 Between Light Weight and Heavy Weight Ontologies . . . 34
6 Implementing Heavyweight Ontologies . . . 37
7 Extending Ontologies through Heterogeneity . . . 42
8 Discussion of this Research . . . 49
9 Opportunities for Future Research . . . 50

1 Changes

Date	**Author**	**Description**
January 4, 2022	Luis Ramos	Initial implementation of Onto*Smart*

2 Report Scope

Within this report we pretend to present the results of implementing Onto*Smart*[1], a methodology for the development of network of heterogeneous (logically speaking) ontologies, with the aim of overcoming limitations of ontology languages like Web Ontology Language (OWL).

[1] https://github.com/luisenriqueramos1977/OntoSmart

3 Brief Methodology Description

For developing ontologies into the scope indicated in the previous Section, we propose Onto*Smart*, a methodology depicted in Fig.1. This methodology is explained in details in its Wiki[2], however in this Section we provide a brief description.

Onto*Smart* is divided into two main horizontal sections that identify two activity groups. The former correspond to development activities and the latter to implementation activities. The inclusion of an implementation layer implies that ontologies will be developed in an application centered context; therefore these ontologies can be evaluated by an effectiveness criterion.

Going into a first abstraction level, **Initial Activities** are those where the ontological engineer or working team roughly defines the target domain to be modeled. Meaning that the domain shall be outlined and general objectives and purpose(s) listed. **Domain Specification** activities are then a continuation of the previous activity. In this stage the ontological engineer or working team has to focus on detailed, measurable sub-objectives and specific goals. In other words, objective should include measurable elements , which will permit us to measure our success level. Candidate sources of knowledge have to be proposed at this stage as well. The type and amount of documentary sources, their size and formats have to be listed; this information is later used to define development and implementation software tools. In most ontology development methodologies, this stage commonly rests on the definition of competency questions, to which the developed or chosen ontology has to provide answers. Competency tasks that will likely require ontology (reasoning) can also be included.

We included the **Reusability**[3]as a subactivity of **Modularity** because ontologies promised to be artifacts for interoperability. But to achieve this, ontologies should be managed as standardized artifacts (i.e., pieces of software). However, in most cases the use of ontologies is developed from scratch, limiting reusability and interoperability as well. In this vein, we consider worth including activities reuse and an indicator to measure reutilization level. These activities reuse consist on the following subtasks:

- **Finding ontologies related to the target domain**[4]
- **Quality assurance from the information point of view**[5]
- **First check of ontologies against requirements**[6]

From the above described scenario we have the following possible outcomes: first, to find a fully reusable ontology is an ideal situation; second, to require development from scratch, third to develop an ontology using parts of others and fourth to reuse modular ontologies.

To analyze the fourth scenario we require:

1. Ontologies related to a common domain as input.
2. A software tool for mapping and merging ontologies.

Then, the current task consists in mapping all ontology against each other, so we can obtain ontological commitments amongst them. The resulting mappings per pair of ontologies will be recorded and are maintained as a hyper-ontological structure over modules.

At this point we should have one of the following options available:

1 . One ontology, if we have developed one from scratch,

[2]https://github.com/luisenriqueramos1977/OntoSmart
[3]https://github.com/luisenriqueramos1977/OntoSmart/wiki/Reusability
[4]shorturl.at/gDFKZ
[5]shorturl.at/bimL6
[6]shorturl.at/dfgxT

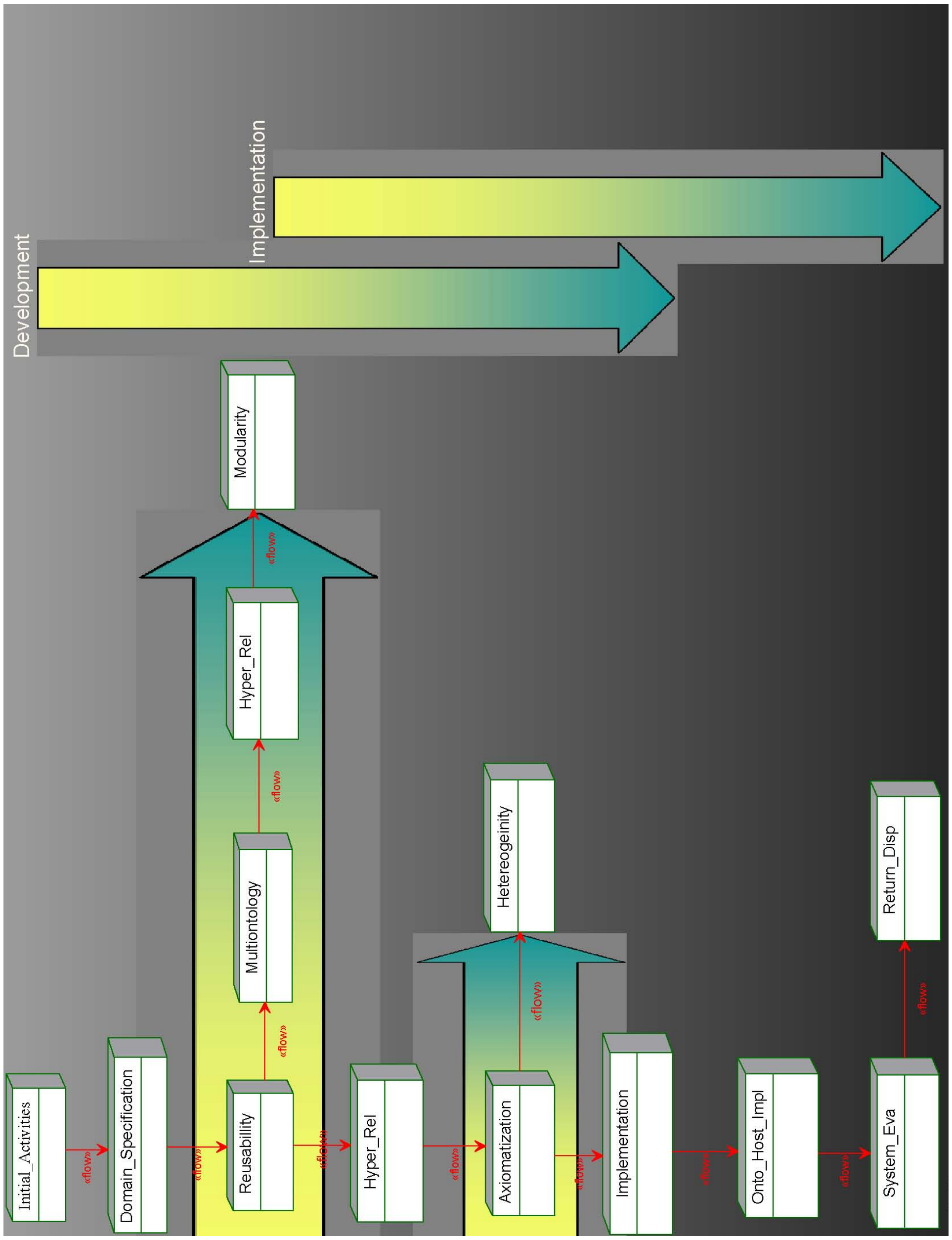

Figure 1: My Methodology

2 .- One ontology developed from a group of ontologies, or

3 . A set of linked (or to be linked) ontologies for reuse.

After having a modular structure, we have to evaluate the necessity of **axiomatization**. That is because from the intended use of ontology, it is possible to draft the axiomatization requirements for our ontology or ontologies. In this case we need to define whether or not we need lightweight or a heavyweight ontology[7]. As we indicated above, the existence of heavyweight ontologies indicates that the axiomatization process should be carefully considered in order to determine if the target language constructors are expressive enough to reach the expressiveness level required to fulfill our requirements. We have to be aware of the presence of n-ary relations (higher than binary), mereotopological relations, procedural reasoning and different unit systems. The aforementioned requirements could possibly enhance the expressiveness level of any language using them in a higher level. If any of them are present in the ontology or ontologies, limiting the full implementation of our model, then we should proceed with the **hetereogenity**[8] Building Block to determine if a heterogeneous layer is required. Otherwise, we can proceed with system implementation. In the **Implementation** Building Block, as most of Ontology methodologies, we should implement the developed ontology. However in this case, given that this methodology is centered in reutilization, it is highly likely that we should just have our ontology or ontologies developed in an ontology language. Finally, in the **System Evaluation** Building Block, we aim at assigning a quality attribute to our ontology as we currently do with any other artifact, offering the final user a reference of the quality of the ontology as a reusable product. Here, we propose to divide our evaluation process as follows:

1 . Evaluation based on the inputs, product and development process This evaluation is self-subdivided into the following stages:

   a. Input quality assurance, through the input ontology evaluation procedure.

   b. Structure measurement, implementing the metrics recommended by [18]. These authors are of the criteria that empirical studies are needed to validate how different metrics are capable of judging deciding about quality properties and their interpretation. They developed a tool called Ontometrics, which considered metrics indicators such as Number of classes (noc), Number of Instances (noi), Number of Properties (nop), Number of Root Classes (norc), Number of leaf Classes (nolc), Averagea Population (ap), among others.

   c. Modularity , as a criterion to classify patterns, is mostly to provide judgment about quality. Considered that implanting wrong patterns will affect the quality of the system as a whole. [25].

   d. Reusability, as an average metric for reutilization, will indicate how interoperable the system is by using inherited terms as a reference. For this task we will follow a criteria similar to the proposed by [27], with the difference that we considered no pair, but networks of ontologies.

2 . Evaluation related with functionally and implementation

   This evaluation is, in our opinion, the most crucial one, but it is subject to the system development. In other words, this evaluation will take place only when some functional part of the system is able to carry out certain tasks. For a successful evaluation, developers should have previously categorized activities, their complexity level, similar to having a list of competency questions to be answered. The criterion to define complexity will be:

   a. If the answer to a query, or to carry out some task can be done through ontology, or consultation is required for other ontologies in order to provide the required answer, or carry out the task.

[7] https://github.com/luisenriqueramos1977/OntoSmart/wiki/Axiomatization
[8] https://github.com/luisenriqueramos1977/OntoSmart/wiki/Heterogeneity

b. If a decidable language is required to execute the corresponding tasks, but the chosen languages are not expressive enough to represent the required knowledge in the knowledgebase, such scenario will oblige us to consider a heterogeneous framework. .

In this Section we presented a new methodology that integrates two fundamental approaches into the Ontological Engineering, those are modularity and heterogeneity. This methodology initiates with a quality assurance procedure, followed by specific method to decide when modularity and heterogeneity are required. We provides tools to support the workflow along the methodology. Furthermore, we introduced some required technologies for implementing the proposed methodology.

In next Section, we will present the results of implementing the just described methodology in an use case of manufacturing.

4 Results

In this Section we will go throughout the steps proposed in the methodology depicted in Fig. 1 of Section 3. The general steps are domain specification or definition, deciding about modularity, deciding about heterogeneity, implementation and evaluation. We will address each in turn.

4.1 Initial Activities

Manufacturing is the target domain of this implementation. However, given that several types of products can be included under this topic, we consider it necessary to define the scope as accurately as possible. This work focuses on products that can be represented through a specific geometry. The products to be considered will be therefore goods (mechanical parts), not services, which consist of one, two or three-dimensional parts, and can be modeled with Computer Aided Design (CAD) tools, and manufactured with a set of automatic machine tools. This might require the interaction of various machines to obtain the finished product. Moreover, it should be possible to modify the final product in order to fulfill all the customers' possible demands.

The automation to be discussed will cover the automatic validation of designs based on machine and product features, as well as production restrictions in order to improve productivity and reduce time to markets [**?**].No code will be generated to program any specific commercial machine. Instead: the existing upper ontologies which have been developed and that can be useful for the stated objectives will be used following the basic ontology reuse principle. The complete life cycle of the product will also not be covered as well, where interaction with customers, distribution or logistics of the product and marketing behavior will not be addressed. These issues can be considered for future work where the complete life cycle of the product will be covered.

4.2 Domain Specifications

Given that we limited the type of products to mechanical parts, our target designs and features are those related to the machining process. These are: drilling, cutting, punching, and shearing among others.

We now present some examples of possible scenarios that can arise when dealing with Automated Features Recognition and Design validation, and which constitute the main target of our research.

- **Scenario 1**: When a designer creates a new product, it is a common fact that the designer is only focused on the functionality from the user's point of view. As soon as the product design is completed, it is sent to the manufacturing engineer who may determine that the product cannot be manufactured due to factory restrictions. Consequently, the designer has to modify the design in order to accommodate the given recommendation.

- **Scenario 2**: A manufacturing engineer needs to generate a process plan for a new or modified product, based on a digital design of the product itself. Most of the time engineers use their own experience and knowledge about the facility, machines and raw materials. For instance, the designer may decide to use a new raw material due to its higher corrosion resistance, but without having the possibility of indicating in the digital design that such raw material has a higher mechanical resistance. As the manufacturing engineer does not see any change in shape, producing waste of raw material, because of the lack of information exchange between designer and manufacturer. Furthermore, the workflow described in this scenario is mostly carried out in an automatic or semiautomatic manner, thus human intervention is reduced. Of course, reducing human intervention we increase productivity, although certain type of issues are harder to find by current information systems. Consequently, design mistakes can affect the manufacturing process production can waste raw materials.

- **Scenario 3**: A group of investors is interested in offering a new product because market research has demonstrated that the product is highly innovative, and is likely to be well accepted. These investors also know that the technical resources (raw material and machinery) are going to have a high cost, requiring a detailed cost evaluation to determine its profitability. Furthermore, investors also know that market competition has the same information, and they require to make their decisions on producing it, and if favorable, place the product on the market as soon as possible. Therefore, an economical evaluation of the project is urgently required, and gathering information from several distributed sources becomes indispensable.

Considering the three scenarios described above, we can declare our ontology specific objectives as follows:

- To enhance manufacturability evaluation of new products by integrating digital designs with raw materials specifications and manufacturing constraints of the latter.
- To improve concurrency of factory main components. This improvement is obtained by enabling virtual modeling and providing communication among them.
- To integrate products, processes and resources specification data into a digital production model, so that all data can be accessed and interpreted by existing software systems and tools.

4.3 Competency Questions

We found in previous researches many of the questions listed below, for instance in the research of [12] and [21]. Some further questions are integrated below based on the scenarios and objectives set out in Subsection 4.2:

Questions (CQ)1 : is my digital design topologically correct?

CQ2 : what manufacturing features are present in my product (design)?

CQ3 : what machinery will enable the performance of a given manufacturing operation?

CQ4 : what type of machinery is available in the target factory?

CQ5 : Can the features of a product candidate be manufactured in a target factory with the available machinery?

CQ6 : Is there any process restriction (e.g the occurrence of an activity A2 shall be preceded by an activity A1)?

CQ7 : Is the required operation available in a given time space?

CQ8 : In the case that no machinery is available for manufacturing a certain feature:

[CQ8.1]: where can it be obtained?

[CQ8.2]: what is the price?

[CQ8.3]: which are the corresponding features?

[CQ8.4]: Are they expressed in homogeneous units?

[CQ8.5]: what is its replacement cost?

[CQ8.6]: which is its official currency?

CQ9 : Which are the attributes of the raw material?

CQ10 : Which is the cost of performing certain machining operations?

In Subsection 4.8 we will implement these CQ's into the respective ontology language of every selected ontology. Thus, according to the number of CQ's answered by ontology a quality metrics will be proposed.

4.4 Knowledge Acquisition

1. Sources of manual knowledge extraction

   Knowledge was manually extracted from documentary sources, such as brochures and data-sheets. This type of documents is characterized because of their specificity and length, mostly short documents of no more than one or two pages with specific information of products. It is necessary to mention that, in the manufacturing domain, expert domain knowledge is required for interpreting and analyzing documents containing standard specifications.

   Consequently, in this case the implicit knowledge is made explicit through listings, tabling and defining ontological elements (types, individuals, properties, etc.). This knowledge is used to develop an ontology of CAD and sheet metal parts features.

2. Sources of automatic knowledge extraction.

   Knowledge automatically extracted from digital documents, which we divided into three categories:

   - Design standard files: Digital designs represented in Drawing Exchange Format (DXF), Initial Graphics Exchange Specification (IGES) and Standard for the Exchange of Product model data (STEP) standards were used to automatically extract data and to populate the ontology. A previous manual review of the respective standards was necessary.
   - Ontology standard files: The ontology languages like Resource Description Framework (RDF), Resource Description Framework Schema (RDFS), OWL, Knowledge Interchange Format (KIF) and Common Algebraic Specification Language (CASL), considered in the form of a specific ontology directly or indirectly related to the target domain. Thus, this analysis not only considered the encoded knowledge, but also the features of the respective language.
   - General purpose formatting files: These files comprised general text files, PDF, XML, HTML, and XHTML among other formats. These types of files mainly corresponded to products, machinery and other product descriptions. Because of their particular unstructured format and quantity, the use of specialized ontology tools was necessary, for example, like the ones listed in Table 1. There, Gate can be highlighted as a tool for extracting semantic information, and populating ontologies from text.

4.5 Finding and Measuring Ontology Quality

In Section 3 the steps of our methodology were mentioned as a possibility for the reutilization of existing ontologies. In comparison with the large number of ontologies that have been developed in other domains, such as medicine and biology among others, ontologies related directly or indirectly to manufacturing are fewer. However, modeling products and processes is a current concern for Ontological Engineering, as evidenced in the list of ontologies mentioned in Section 2. It is worth remarking that according to the description made in that section, the development of ontologies was not limited to only individual ones, but to propose ontology networks such as Toronto Virtual Enterprise (TOVE) and Semantic Web Open Engineering Platform (SWOP).

In the following subsection our search process is explained in detail.

4.6 Finding Ontologies

Finding Ontologies is a requirement for reusability that can be considered as a part of the "Reusability" activity shown in Fig. 1 in Section 3. There we made use of the Swoogle[9] and Watson[10]. The keywords used were the

[9] http://swoogle.umbc.edu/
[10] http://kmi-web05.open.ac.uk/WatsonWUI/

Table 1: Software tools and Technologies to be used in Ontologies and Modules development

Features / Editors	Protégé 3.x	Protégé 4.x	KAON	OntoStudio	Kojeware (Eclipse Plug - In)	Top Brain Composer	SWOOP (Not under development)
Collaborative	Webprotégé	x	x	Web Ontostudio			
Ontology Languages Supported	owl 1.1, rdf, Turtle, N-triple	owl 2.0, rdf		owl, rdf, rdfs, rif	X	owl, rdf, rdfs, rdfa,Turtle, N-triple	probando
Reutilization	Mapping and merging with prompt	merging		Mapping with Ontomap	x	merging, mapping	
Rules Languages	SWRL with build-ins Jess engine	SWRL without build-ins		RIF		SPARQL Rules	
Query Languages	SWQRL SPARQL Jess engine			SPARQL		SPARQL	
API's	ProtégéAPI	OWL- API			Samian ODE		
OL interface	GATE						
Ontology Importing							
Logic	DL-shoin	DL-shoir			Common Logic (CL)		
Visualization	OntoViz Jambalaya	Onto-Graph					
Reasoning	Fact++, Pellet, Racer Pellet	Fact++, Hermit, Pellet		OntoBroker (repository)			

Table 2: Ontologies Potentially Relevant to Manufacturing.

Ontology Name (Acronym)	Domain Subdomain	Language	Year
EngMath	Resource	KIF	1993
Resource Ontology (TOVE)	Resource	FOL	1994
Organization Ontology (TOVE)	Resource	FOL	1998
Port Ontology	Resources	OWL	2003
Process Specification Language (PSL)	Process	FOL	2004
unspcsOWL	Products	OWL	2004
SWEET Units	Resource	OWL	2004
Manufacturing's Semantic Ontology (MASON)	Resources, Product	OWL	2006
ADACOR	Resources, Product	FOL	2006
MSE	Resources, Product	OWL	2007
Beyond STEP Ontology	Product	OWL	2007
GoodRelations	Product	OWL	2008
SWOP Product Ontology	Product	OWL	2008
Features-Based Design Ontology	Product	OWL	2008
DFM e-Design	Product	OWL	2009
Cutting Process Ontology	Product, Process	FOL	2009
MTM	Resources	OWL	2009
ONTOMoPS	**Product, Process**	**OWL**	**2011**
PROduct ONTOlogy (PRONTO)	**Product**	**OWL**	**2011**
OntoSTEP	**Product**	**OWL**	**2012**

ones proposed by [19] and [16], who have discussed that the terms *product*, *process*, *resource* and *equipment* are considered as higher level concepts in the manufacturing domain. Every ontology reference we found through this tool was recorded, and the list of ontologies finally tabulated.

Table 2 lists ontologies found by this procedure explained above. Results are presented by publication year. The corresponding domain or concept covered by each ontology is indicated in the second column. Some ontologies were developed with a large scope, and due intended to cover many domains, therefore more than one domain is listed in the respective column. Another aspect worth mentioning is that most of the ontologies shown were implemented in OWL: more precisely 70%, while only 30% were implemented in First Order Logic (FOL). But, from the FOL ontologies, 50% were published over 15 years ago. In fact, before the publication of the first OWL version, ontologies were written in FOL and KIF. Since then OWL became the preferred ontology language.

At the bottom of the table several ontologies are highlighted because they were published during the production period of this research. It becomes evident that the process of considering ontologies has not been static but dynamic with continuous updates.

This list of ontologies requires a certain level of quality assurance in order to determine its reusability; in the following subsection this procedure is carried out and results are presented.

4.7 First Quality Control

The ontologies listed in Table 2 were submitted to the first quality assurance of Onto*Smart*[11]. In order to quantify those parameters we built and proposed a questionnaire[12]. There, every dimension is presented with a list of weighted scenarios. The value of the scenario is ranging from 1 on an ideal situation to a lower value given to worse or less advantageous scenarios. The results of applying this questionnaire permitted us determine the current quality level of each ontology. Fig. 2 outlines the output of performing this preliminary evaluation.

[11] shorturl.at/oBEIV

[12] https://github.com/luisenriqueramos1977/OntoSmart/wiki/General-Informational-Quality-Evaluation-Questionnaire

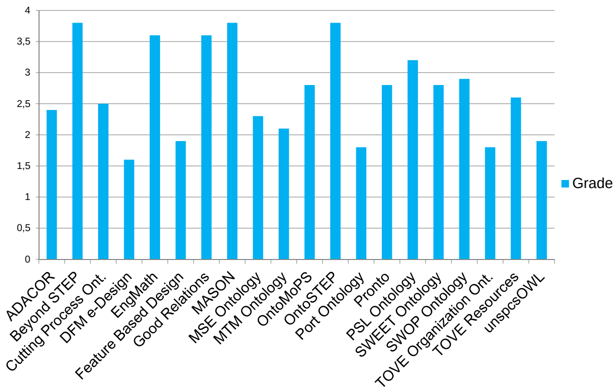

Figure 2: Results of Input Evaluation

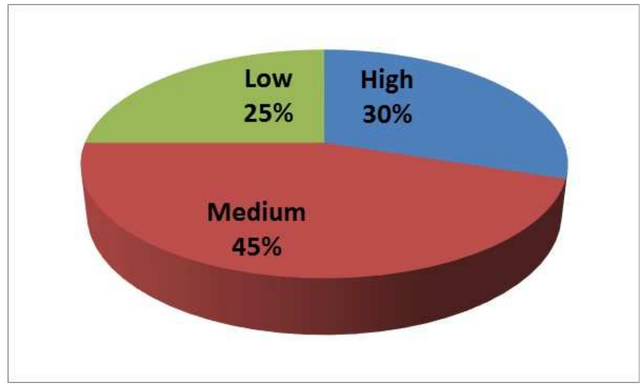

Figure 3: Grouping of Ontologies by Quality Level

According to the results shown here, Ontologies evaluated can be divided into three categories: those with a quality range [4, 3] were considered of high quality and passed to the next evaluation procedure. Ontologies with a quality range (3, 2] were considered of good quality and passed to the next stage. The ontologies within the range (2,0) were discarded.

A simplified view of these quality sets mentioned above is outlined in Fig. 3. From this figure, it can be considered that a reduced number of ontologies fulfill the reusability criterion we proposed in the questionnaire. That is 30% appears as highly reusable, while a larger set can be reused after certain intervention of the ontologists. A last set is of low quality and should not be reused.

Reasons leading to a larger set of low and medium quality ontologies are displayed in Fig. 4 and Fig. 5. Firstly, conceptually speaking, ontologies should be publicly available, and with minimum limitations for their use. However, as Fig. 4 shows, from our sample ontologies, only 55% were available for direct download or provided by authors, while the other portion was not. This issue limits the use of an ontology and also reduces the possibility of performing a more accurate evaluation.

Furthermore, as the next figure shows, explicit indication of rights to reuse and modify the ontology had also been omitted in most of the reviewed ontologies. In an ideal situation, this legal statement should be encoded within the ontology itself, and in the worst case, it should be specified on the site where an ontology is available for download. OWL has a mechanism that makes it possible to add such types of annotations to ontologies, but even in many of the ontologies written in OWL this possibility was not used.

Finishing this first evaluation procedure, ontologies were reordered according to the quality level previously given. Table 3 lists the order in which ontologies will be considered for subsequent uses and evaluation.

4.8 Performing Competency Questions

The use of Competency Questions to define the scope of ontologies, and additionally to validate them, was explained in Section 3. This technique was included as a part of the proposed methodology. Therefore, in Subsection 4.3 a set of competency questions for the manufacturing domain was proposed. Here we mention that if one of the ontologies listed in Table 3 provides appropriate answers to all Competency Questions, then we can proceed to hosting and implementing that ontology as given in our proposed methodology presented in Section 3. However, if no ontology fulfills this requirement, then this will require developing a newer ontology, although preferably reusing existing

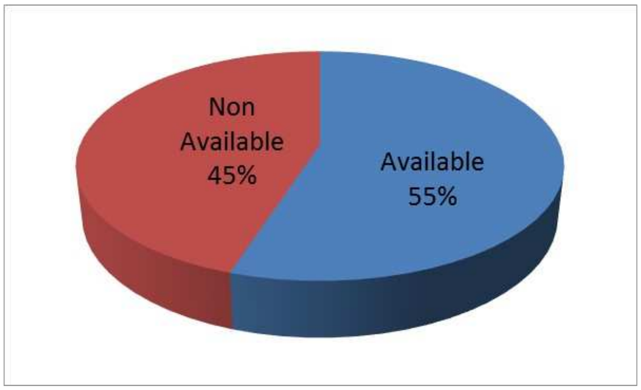

Figure 4: Ontology Availability Evaluation

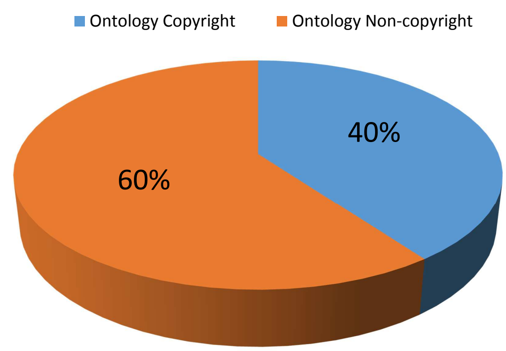

Figure 5: Ontology Intellectual Property Evaluation

Table 3: Quality Order

Quality Order	Ontology Name (Acronym)
1	**MASON**
1	**Beyond STEP Ontology**
1	**OntoSTEP**
2	**GoodRelations**
2	**EngMath**
3	**PSL Ontology**
4	SWOP Product Ontology
5	SWEET Units
5	PRONTO
5	ONTOMoPS
6	Resource Ontology (TOVE)
7	Cutting Process Ontology
8	ADACOR
9	MSE
10	MTM Ontology

content.

Competency Questions were performed on the OWL ontologies listed in Table 3 considering their quality order, specifically ontologies with a quality order in the range from 1 to 3. The Query Tab plugin of the ontology editor Protégé was chosen from Table 1. This choice was made because, first both the editor and the Plug-In support OWL, and second because they offer a friendly interface that makes interaction with the chosen ontology possible. To illustrate this example we chose CQ3 (Section 4.3), which in natural language is expressed as follows: which machinery enables the realization performance of a given manufacturing operation? It is worth mentioning that this query is concatenated with CQ2, which asks questions on product features. Furthermore, CQ3 can be considered as part of the context of scenarios 1 and 3 described in Section 4.2. In Equation 1, this query is formalized with Semantic Query-enhanced Web Rule language (SQWRL).

$$Machine\_Resource(?m) \wedge enablesRealizationOf(?m, \text{“}punching\text{”}) \longrightarrow sqwrl : select(?m, \text{“}punching\text{”}) \tag{1}$$

Fig. 6 illustrates interaction with the ontology. That is, starting with the first ontology presented in Table 3, that is the MASON ontology, our chosen ontology was manually populated with data. That means, several instances of the concept `mason:Machine_resource` were created, and properties of machines were encoded in the ontology, using information of commercial brochures. In this example, a `mason:Machine_resource` was related to a `mason:Punching` operation by a `mason:enablesRealizationOf` predicate. The Query Tab presented in Fig. 6 is organized according to Protégé vocabulary, therefore from left to right we can view the terms “Class” which is a concept (`mason:Machine_resource`), the term “slot” corresponds to a predicate or property (`mason:enablesRealizationOf`), and the condition “contains”. Within “contains” we will evaluate wheter or not a concept contains a given individual. The resulting individual has to satisfy the condition drawn by the query. In this case the query evaluates whether or not the given instance is “contained” by any individual in the target class by means of a property. In the illustrated case it was possible to find an instance that fullfils our requirement, that is the individual `mason:Punching_press`. The result is shown on the right of the figure in “Search Result”. In natural language we can say that we found machinery that enables performance of punching operations, which is a Punching Press. In other words, we found a result for our query. This procedure was followed with every ontology of Table 3

$$\begin{aligned} (defrelationis - occurring - at(?punching?p) :=\\ (and(activity - occurrence?punching)\\ (betweenEq(beginof?punching)?p(endof?punching)))) \end{aligned} \tag{2}$$

For ontologies written in other languages (FOL, KIF), questions were considered as answerable when the available terminology (concepts and predicates) could be structured to represent the requirement expressed in the query. For instance, within CQ7 it is required to answer whether or not an operation or activity is occurring in a given time space. This can represented in PSL as indicated in Equation 2. In PSL vocabulary an **activity-occurrence** or manufacturing operation **is-occurring-at** a timepoint **p** if and only if **p** is **betweenEq** the activity occurrence's begin and end points.PSL therefore supports the required representation of CQ7, and within it we can confirm if a given activity is taking place.

Fig. 7 shows how many competency questions were answered by each ontology, according to the procedure described above.

If we want to use the number of competency questions answered per ontology as a quality parameter, it could be considered that the quality of selected ontologies is low because most of them represent a quantity of knowledge only sufficient to answer less than 50% of the questions. Nevertheless, within Fig. 8 a distributed view can be presented. This graphic breaks down how the selected ontologies *as a whole* provide answers to 15 of 16 queries.

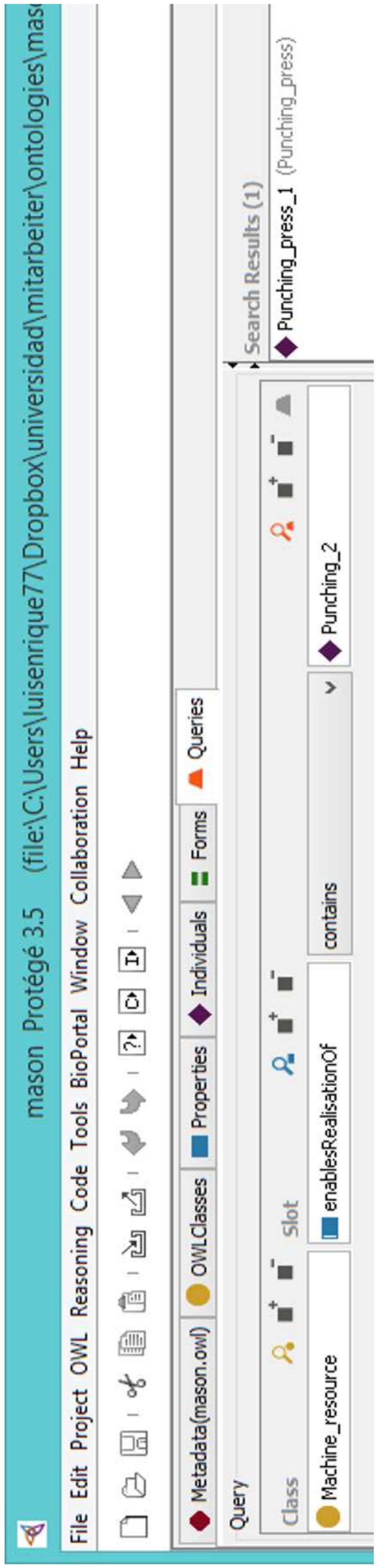

Figure 6: Ontology Query in Protégé

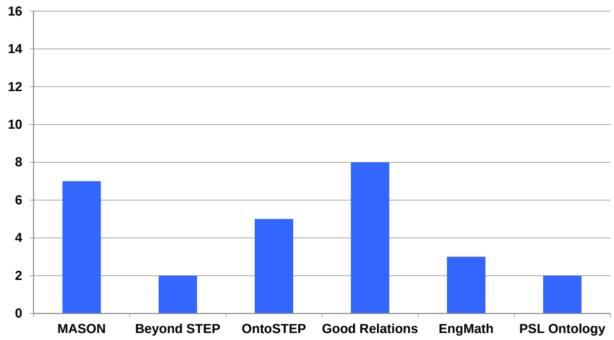

Figure 7: Queries Answered per Ontology

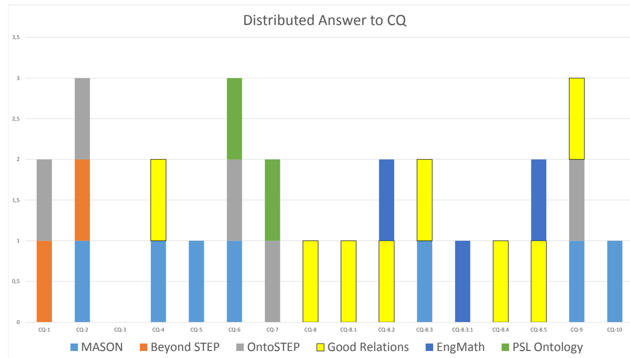

Figure 8: Answers Distribution between Ontologies

Every column represents the number of ontologies that provide adequate answers to the corresponding CQ. For instance, no ontology provides an answer to CQ-3, while CQ-2, CQ-6 and CQ-9 found answers from three different ontologies respectively.

From our point of view, answers to these CQ certainly support using *modularity*. That means working with a network of ontologies we can be more efficient in the sense of getting more answer to our CQ's than working with individual ontologies. In other words, while there is no ontology that provides answers for every CQ, many CQ can obtain answers from different ontologies. That means working with these ontologies in a modular architecture, we can obtain answers to 94% of the given queries, while just using such ontologies alone we would be able to provide answers to only 50% of the CQ.

In fact, more than obtaining an answer for most queries, several ontologies provide an answer to related sets of queries, while many others can be divided into those where the ontology provides an answer to only one query, and others that do not provide answers to any other query. Such an outcome indicates that, on the one hand, there is some common knowledge between these ontologies, and it would be possible for a question to obtain answers from different ontologies. On the other hand, there are some questions that obtain answers only from one ontology, indicating that there is certain localized and isolated knowledge in some of them as well.

Fig. 9 represents another view of the scenario described in the previous paragraph. There, every circle represents one ontology and their overlapping regions indicate commonalities in answers to CQs. The position of circles in this figure is also meaningful: from the middle to the left side, there are ontologies related to the products represented as solid parts, and the manufacturing process representation as well. OntoSTEP nearly subsumes BeyondSTEP and PSL has commonalities with OntoSTEP and MASON, but not with BeyondSTEP. This occurs because BeyondSTEP does not mention any process in its terminology. On the right side of this figure is GoodRelations. This ontology is intended to represent products on the Internet, but it does not deal with representing manufacturing processes, therefore it is isolated from PSL. While it holds some relation with queries that can be answered by EngMath, given that the former highlights the use of International System Units in the definition of product features and parameters, which is wthin the scope of EngMath. For instance, in the case of the CQ-8.5 **What is the replacement cost?**, which is a query that comes from the fields of cost accounting, projects evaluation and insurance, concepts related to monetary units and physical units are required. Thus, for CQ-8.5 the concepts `gr:UnitPriceSpecification`, `gr:QuantitativeValue` (GoodRelations), and `system-of-unit` (EngMath) can e considered to provide an appropriate answer to this query. We can also mention that this result

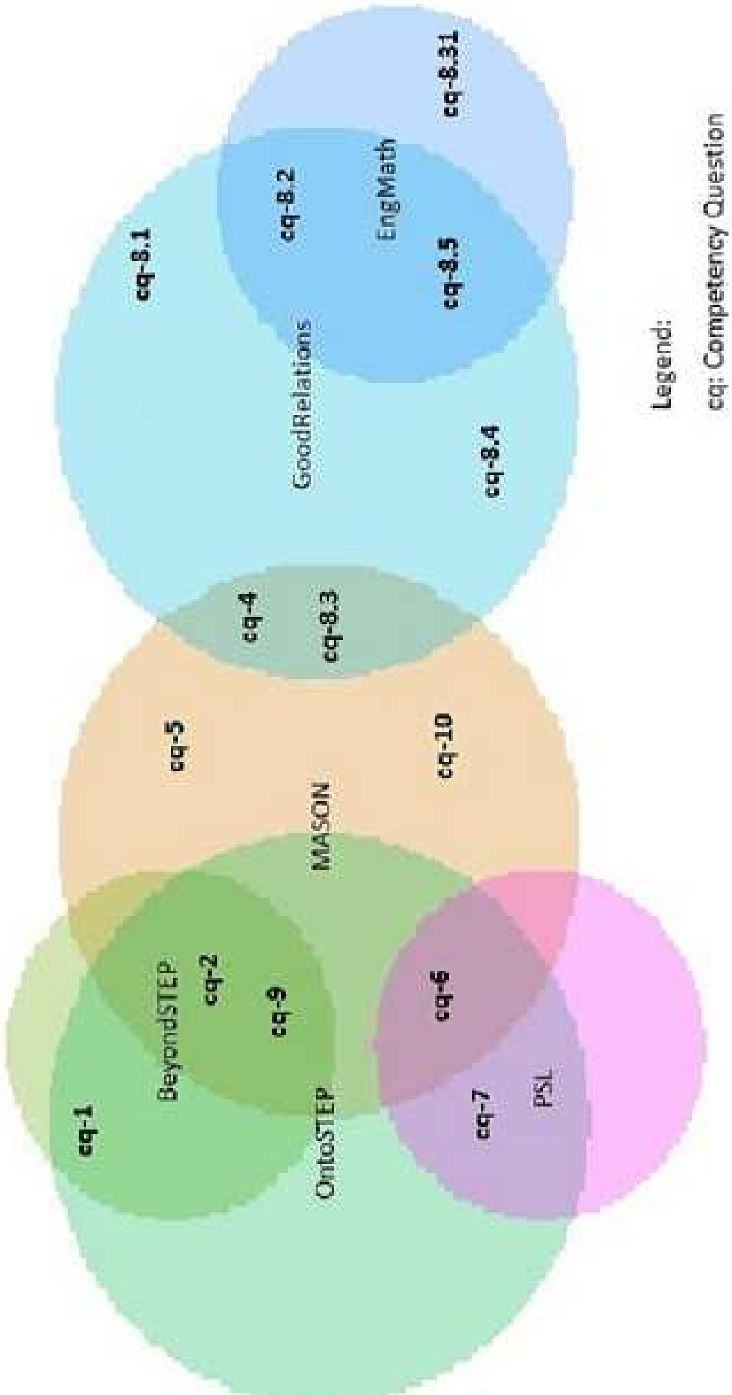

Figure 9: Ontological Commonalities in Answers to Competency Questions

differs from what would be expected from ontologies for manufacturing and engineering science in general, given that metric units, the definition and concepts are fundamental in these fields. However, metric concepts do not appear in most of them.

In order to continue with this subject, and to provide sufficient generalization, it is beneficial to make use of Onto*Smart*'s definitions[13].

Definition 4[14] can be visualized in Fig 9, where a modular domain was obtained.

In this case, considering the definitions proposed in Onto*Smart*, and the results obtained until now, which are represented as the number of answer to queries (see Fig. 7 and Fig. 8 and its distribution in the network of ontologies (see Fig. 9) we can proceed to summarize our findings in order to decide which way to go from here:

1. First, according to the results outlined in Fig. 7, where we can see no ontology provides answers to every CQ, a fully reusable ontology o_r, as defined in Definition 3[15] of Onto*Smart*, is not present in the manufacturing domain under study, and so a direct step to implementation is not possible for the given use case.

2. Second, development from scratch should be discarded, given that the current manufacturing ontologies under consideration provide enough information to answer most of the domain questions listed in Section 4.3, as we can observe in Fig. 7. Therefore, discarding these ontologies would make us lose time and other resources that could be used in developing a new ontology.

3. Third, developing a new ontology with the ones evaluated as parts is another possibility. This ontology is also likely to use one of the existing ontologies as subsumed and increase the knowledge encoded in the former by integrating the latter within it.

4. Fourth, finding one ontology to be used as an interoperability artifact between a set of ontologies would make it necessary to determine whether an ontology exists that could be categorized as an upper level ontology. The use of this ontology should be similar to that displayed in Multiontology[16] Building Block of Onto*Smart*, where one ontology is used with the sense of providing interoperability among domain ontologies.

5. Last, it is also possible to have separate ontologies that contribute to providing modeling structures and answers to queries distributed among subdomains. For instance, a scenario like the one depicted in Fig. 9, where answer to queries are distributed among several ontologies.

Option 3 corresponds to the most common approach adopted in Ontological Engineering, that is: enriching an ontology by importing or merging other ontologies into it. Options 4 and 5 correspond to two schools of thought in Ontological Engineering. The former corresponds to the upper level approach, and the latter corresponds to the hyper-ontological approach. In the next subsection these approaches, their implementations and issues within our domain of study are discussed.

4.9 Mapping, merging and importing

In this Subsection, we applied Onto*Smart*[17] techniques to ontologies listed previously in Table 3 in order to clearly define which of the scenarios described at the end of the previous section we are facing. In other words, we have to determine if there is an upper level ontology, or if we have a network of ontologies, such as a Hyperontology.

However, it is necessary to remark that these techniques are simple to implement when dealing with *homogeneous* ontologies written in the same implementation language. In our case most ontologies presented in Table 3 are written in OWL. Consequently, there were only four ontologies available for immediate mapping of this type; these

[13] https://github.com/luisenriqueramos1977/OntoSmart/wiki/Formal-Definitions
[14] shorturl.at/kBLNZ
[15] shorturl.at/alyH0
[16] https://github.com/luisenriqueramos1977/OntoSmart/wiki/Multiontology
[17] https://github.com/luisenriqueramos1977/OntoSmart/wiki/Hypermodules-Extractions

Table 4: Number of Ontology Mappings through Different Tools

Ontology 1	Ontology 2	Protégé 3.4.4 PromptTab* (Number of Alignments)	Falcon-AO 2010 (Number of Alignments)	NeonToolKit 2.3.1 Alignment Plug in! (Number of Alignments)
MASON	BeyondSTEP	18	NaN	34
MASON	GoodRelations	0	NaN	14
MASON	OntoSTEP	SC	0,85	101
BeyondSTEP	OntoSTEP	SC	NaN	209
BeyondSTEP	GoodRelations	0	NaN	21
OntoSTEP	GoodRelations	0	NaN	417

* With Lexical matching
!SMOA Name Alignment and trim 0.7
SC: System crashes
NaN: No Alignment found

were MASON, the BeyondSTEP Ontology, OntoSTEP and GoodRelations. The other high quality ontologies listed in Table 3, PSL and EngMath, were not considered for automatic mapping because of its technological limitation just mentioned, meaning that the ontology language of implementation differs from the language used in most high quality ontologies (OWL).

According to the software tools listed in Table 1, Protégé (prompt), Falcon and NeonToolKit are ontology editors that support OWL, thus they were chosen for mappings. Ontologies were mapped by pairs with every tool. The number of positive mappings were counted and tabled. Table 4 shows how this mapping experiment took place and which results were obtained by mapping the ontologies with different mapping tools and specific mapping techniques. In the third column mappings obtained by the PromptTab plug-in of Protégé are listed. In this case only mappings between pair of ontologies were found. In the other cases, no mapping was found. The mapping experiment was repeated with Falcon. In column four results mapping the target ontologies are listed. Experimenting with Falcon, mappings were found in only one of six cases.

A third mapping experiment was carried out by the NeonToolKit Alignment plug-in. The fifth column of Table 4 lists every time an alignment was found with this tool. This time, unlike the previous experiments, a positive mapping was found for each pair of ontologies in most cases. There was a fundamental difference with this tool, that is we had the possibility to set up a similarity measure or threshold. Threshold of 1.0 would let us align only terms exactly written, a lower threshold value (e.g: 0.9, 0.8, 0.7) would let us align similar terms like `paint` and `painting`, but a low threshold value (e.g: 0.3, 0.2) could drive have to obtain wrong alignments of term like `paint` and `point`. Thus, for our experiments we chose to set up a threshold of 0.7. As a result, mappings provided by NeonToolKit were considered for further analysis.

Although a performance evaluation of mapping is not in the scope in this research, it is worth remarking that different results were obtained by applying the different alignment algorithms available in each software tool. NeOn alignment plug-in provided the best results, thus these results were taken for working out the modularity.

Mappings obtained with NeonToolKit (fifth column) were carefully checked manually in order to remove mappings that we considered incorrect, thus False Positive (FP) mappings produced by the mapping system were removed and only True Positive (TP) mappings were included [15]. The cleaning process consisted in discarding every mapping falling in each of the following cases:

Table 5: Ontology Mapping in Manufacturing Domain (First Iteration)

Ontology 1	Concepts in source Ontology	Ontology 2	Number of Mappings		
			FP	TP	Total
MASON	222	BeyondSTEP	27	7	34
GoodRelations	37	MASON	14	0	14
OntoSTEP	1625	MASON	90	11	101
BeyondSTEP	114	OntoSTEP	104	105	209
BeyondSTEP	-	GoodRelations	21	0	21
OntoSTEP	-	GoodRelations	414	3	417

- Property–property mappings were discarded.
- Redundant mapping through subsumption relations, in other words a concept in source mapped to two or more similar concepts in target. For instance, we obtained mappings from `Bezier_Curve` in source ontology to `Rational_Bezier_Curve` and `Bezier_Curve` in the target ontology. That means, NeonToolKit provided two pairs of mappings: (`Rational _Bezier _Curve`, `Bezier_Curve`) and (`Bezier_Curve, Bezier_Curve`). Thus, the first pair was discarded, and the second was considered.
- Similar names, but with different meanings, e.g, mapping of `Paint` in source to `Point` in target, were discarded.

Furthermore, mapped concepts with a threshold lower than 1.0, but with similar meanings, were left in the mapping file, e.g., `Thread` in source mapped to `Threading` in target. Then, the total mappings were divided in FP and TP as depicted in Table 5. The output datasets corresponding to this experiment are available online (adding a link - I will publish the link before delivering the thesis).

Fig. 10 shows a graphical representation of the TP mappings. Only true positive mappings between selected ontologies related to manufacturing are shown. This TP was selected according to the criterion previously described with the intention of avoiding incorrect mappings. For instance, mappings of similar terms with different meanings, like point and paint. Similar mappings were considered as FP and were discarded. In the figure, we can observe that the number of mappings from BeyondSTEP to OntoSTEP (105) appears to be the most significant one in this set of ontologies, but when the number of concepts of Beyond STEP (114) were considered, and compared against the number of TP mappings, it can be observed that more than 92% of the concepts in BeyondSTEP were present in OntoSTEP. This result shows a scenario similar to the one shown in Fig. 9, where BeyondSTEP was mostly subsumed by OntoSTEP.

However, when mappings are reviewed in detail, we find that both BeyondSTEP and OntoSTEP share some common mappings with MASON. These are the ones related to geometric concepts. Also, the mappings MASON-OntoSTEP highlight mechanical feature concepts on the one hand, while on the other hand the mappings MASON-BeyondSTEP highlight more complex geometric concepts. This mapping situation shows a very common scenario in Ontological Engineering, i.e., the existence of overlapping ontologies O_1 and O_2 which describe independent aspects of a given domain, but with some overlap.

As our interest consists in finding answers to the competency questions previously listed, a first method would be to grant full access to the knowledge in those ontologies in order to answer these domain questions. This is equivalent to integrating them by merging. To proceed with this, Protégé 4 was selected from Table 1 as an ontology editor implementing support for merging ontologies. Thus, MASON and OntoSTEP were merged first,

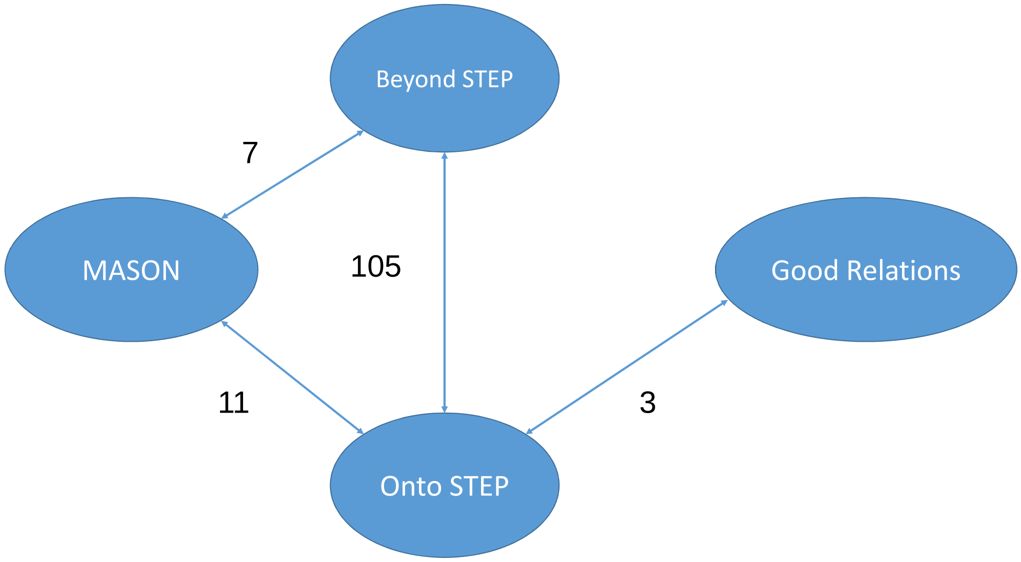

Figure 10: Ontological Commonalities in Answers to Competency Questions

obtaining a new ontology. However, after merging them and running the respective reasoner, it was found that the resulting ontology became inconsistent.

Much has been studied about the issue of inconsistency for ontology reutilization. For instance, [9] show that the appearance of inconsistencies when merging ontologies also depends on the logic of the ontology language in which the ontologies to be merged are implemented, in other words if we have two ontologies, one written in OWL, and another written in FOL, they cannot be merged, because of the language heterogeneity. To date some techniques to deal with inconsistent scenarios have been proposed by [11] and [31]; both authors considered the need for user intervention and judgment in order to extract those modules from O_1 which can be reused in O_2 avoiding inconsistencies. Others authors have recommended keeping ontologies separated, but with logical links among these ontologies (e.g [4] and [14]).

However, from the options given above, those considering human intervention are only feasible when dealing with pairs of small ontologies. In cases where a larger number of ontologies with a larger number of concepts are present, the implementation of appropriate algorithms for generating candidate modules is clearly going to be necessary as long as this proves possible. Thus, the human intervention possibility is discarded here. Before considering the last possibility (option 5, of the list depicted in Subsection 4.8, that is the hyper-ontology), we exhausted option 4: that is the possibility of finding one ontology of Upper Level as an interoperability artifact. Therefore, retaking Fig. 1 where our methodology is described, we decided on Reusability, because we found evidence that the ontologies under study (see Table 2), provide answers to some of the proposed CQ, that means we are in a Multiontology environment. Then, we have to evaluate the most commonly found approach in Ontological Engineering, which is using an ontology as an interoperability artifact among other ontologies. That means that we have to determine if one of the ontologies under study could receive a higher categorization of upper level.

In this vein the number of mappings in Fig. 10 is insufficient to make any conclusions, because most of the mappings are inconsistent in number, and similarly directed to one ontology or distributed among ontologies. Therefore, a second mapping iteration was carried out in order to obtain a larger network mapping, and obtain more accurate conclusions.

According to the quality order previously outlined in Table 3, the SWOP Product Ontology, SWEET Units, PRONTO and ONTOMoPS were considered for this second iteration. In this case, only NeonToolKit was used for further experiences because of the better performance shown in the results obtained from Table 4 from the previous iteration.

Fig. 11 represents the output of including the mappings of this second iteration of ontologies. Here, OntoSTEP appears to also have the most mapping sets to every ontology in this network and BeyondStep has six mappings, while SWOP and OntoMoPs have five mappings respectively. It is worth highlighting that these last two ontologies were not considered for the first iteration because of their lower quality evaluation. However with an automatic tool like NeonTooloKit we have found they have a large set of commonalities in the network of ontologies evaluated. The terms: `Product, Unit, Assembly, Process` and `Material` were some of those common terms found. SWOP and OntoMoPs were discarded at the begining because of preliminar evaluations, that did not consider the encoded knowledge in a first step. Those previous evaluation are ilustrated in Fig. 2 and Fig. 7. Nevertheless, at this stage an automatic tool was included, taking us to discard others and retake these ontologies.

In the specific case of the set of mappings between BeyondSTEP and OntoSTEP, this large set of mappings was expected due to both ontologies having a closely related scope, which is the STEP standard. Although slightly different in the parts of the standard they modeled, the relation among the number of true positive mappings and the number of concepts of BeyondSTEP (92.1%) allows us to affirm that BeyondSTEP is redundant compared to OntoSTEP.

Discarding the set of mappings between BeyondSTEP and OntoSTEP, a more complex scenario was obtained. In this case, from the 28 mappings 19 were sets of true positive mappings ranging from only one (7 times) to 16 individual mappings. The sets of only one mapping contained the terms Product and/or Unit. The remainder sets contained terms related to geometry features of products, mechanical features, and other terms repeated less frequently, such as assembly, material and process. Table 6 presents the details of those terms and their frequency. Here we can retake the list of terms we mentioned in Section 4.6, proposed by [19] and [16] as significant for the manufacturing domain. Those are `Product`, `Process`, `Resource` and `Equipment`. All of them except **Equipment** are present in this table. The obtained mapping terminology was grouped according to the proposal of the mentioned authors. For instance, the terms `Assembly`, `Part` and `Set` were considered as representations of `Product`. Likewise the terms `Operation, Change, Transformation, Milling` and `Drilling`, were grouped as `Process`.

There are two additional aspects to comment from this table. At first they appeared as new terms that can be considered significant as well. Those that were grouped as features, which included specific mechanical features, were obtained by machining process, and the other term was Unit. This last term was not grouped with other mappings, however it presents the most frequent after the term `Product`. Consequently this term has to be considered as significant as the ones proposed by the authors mentioned above.

We consider that because of the number of mappings shown in Fig. 11, which is larger than the number of mappings shown Fig. 10, we had to continue working with the network of ontologies shown in the later.

Of course, it is necessary to remember that our goal is to determine whether or not in this network we can identify one of the following patterns:

a An ontology that could be used as an interoperability artifact between other ontologies, or

b Ontologies that could be used as separate modules in a network of ontologies.

Here we observe the issue that, besides having a network with significant terms, to date there is no systematic and objective procedure to determine when an ontology can be considered as upper level. Moreover, we have to remark that some authors have proposed their manufacturing ontologies as Upper Level Ontology (ULO), without providing a proper reason, analysis or methodology to support such a statement. Therefore, it is necessary to accurately define when an ontology is "Upper Level". In Onto*Smart* [18], we proposed a group of metrics and a criteria to determine when such ULO scenario appears.

[18] https://github.com/luisenriqueramos1977/OntoSmart/wiki/Hypermodules-Extractions

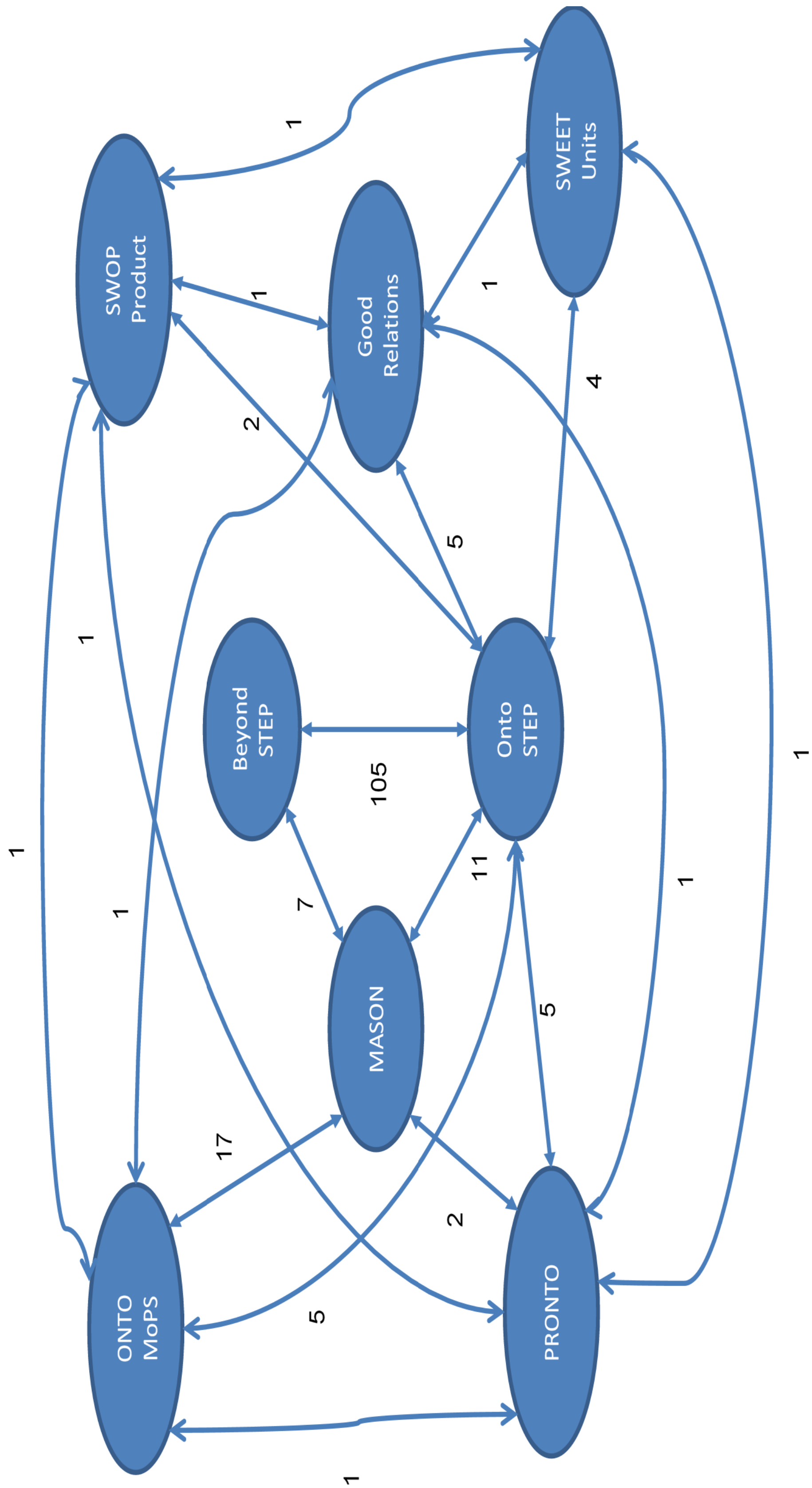

Figure 11: Network of Mappings in Manufacturing Ontologies

Table 6: Ontology Mapping in Manufacturing Domain (First Iteration)

Terms 1	Mapping Frequency	Grouping
Product	7	Product
Assembly	3	Product
Part	1	Product
Set	1	Product
Circular Slot	2	Features
chamfer	1	Features
slot	1	Features
pocket	1	Features
Line	2	Features
Operation	1	Process
change	1	Process
Transformation	1	Process
Process	1	Process
Milling	1	Process
Drilling	1	Process
Event	1	Process
Resource	1	Resource
Tool	1	Resource
Machine	1	Resource
Lathe	1	Resource
Organization	1	Resource
Person	1	Resource
Material	2	Resource
Unit	6	Unit

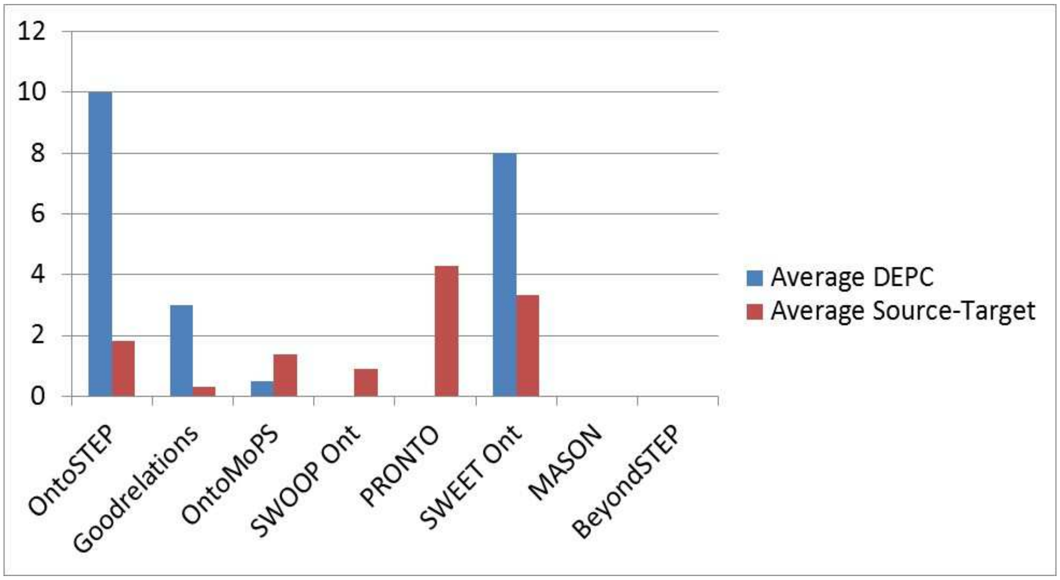

Figure 12: Relative Upper Relationship among Ontologies

With the definition of the previous metrics, we calculated the results for the adopted ontologies, and display the results in Fig. 12. As we stated in the previous paragraphs, when the concept of deployment measurement was introduced, this evaluation is intended to determine: first how an upper ontology is compared to other pairs of ontologies that have mappings among them and, second, the interoperability level that could be provided by this ontology in a hypothetical network of systems where the other ontologies are implemented. For authors like [5], this interoperability is granted by means of shared ontologies, but while they provide a subjective method for alignment of ULO´s with manufacturing ontologies, and other authors like [17] declare their manufacturing ontologies as ULO´s without providing enough support for such a statement, we consider that more objective metrics are required. In this vein the figure depicted above breaks down how the results were obtained applying the metrics to the network of ontologies shown in Fig. 11. That is, the upper relationship between ontology and its respective mapping environment. Unlike the previous views provided in Fig. 10 and Fig. 11, with the results depicted in Fig. 12, we have a criterion to determine how upper an ontology is compared to others in a network.

Fig. 11 introduced above is interpreted as follows: Every ontology evaluation yields two columns. The left column corresponds to the average of deployment of concepts among the ontology evaluated as an upper one, and the right column corresponds to the average of mapped concepts in the source and the target ontology. To consider one of the evaluated ontologies as relatively upper in comparison to the others within the network under study, the two following assumptions are made:

a The column on the left should be smaller than the column on the right for the ontology under evaluation. When this occurs, it means that the deployment of mapped concepts in source and target ontologies is larger than the deployment in the upper ontology.

b The column on the left is greater than 2. This value for the source and target indicates that for each mapping concept at least two new concepts are linked to it by arcs. This means that, as in the previous points, the mapped concept has been deployed in the source and target ontologies, and that it is not a mapping with a leaf node.

According to the criterion indicated above, and considering the results depicted in Fig. 12, we can mention that in the manufacturing network of ontologies shown in Fig. 11 there is a lack of any single ontology that could be considered to occupy a relatively upper level in the network. In other words, according to our metrics, none of the manufacturing ontologies evaluated would serve as an interoperability artifact among other ontologies.

The relevance of this discussion on the presence of ULO´s in the network of ontologies presented in Fig. 11 lies on one hand in the fact that some of the ontologies presented used in this evaluation have been declared as ULO´s by their proponents without supporting their statements in a detailed analysis of previously existing ontologies in the domain. MASON and SWOP are some of these ULOs. On the other hand, although the concept of ULO is clearly defined from the philosophical point of view and many ULO´s are clearly identified, the category of ontologies according to the hierarchy indicated in [10] is still subjective. This issue may cause interoperability misinterpretations when using an ontology in a hierarchy higher than where it should be according to its *ontological contribution* to the given target domain, manufacturing in our case. Consequently, with this partial result of our research, in Fig. 12 we highlight the necessity of evaluation of ontologies in a given network of ontologies prior to declaring them as ULOs. If we declared ontology to be upper level for a given domain, to our knowledge it means it can be used to enable interoperability between systems that use lower level ontologies. This "level" of the ontology is not related to quality, but with reusability. This means the more, *upper the ontology* the more reusable it is, because it is more general, however the upper is the ontology the less usable it is, which is a disadvantage. Therefore, Ontological engineers shall be able to manage such a categorization.

From our point of view, what some authors have tried to express when declaring their ontologies as ULO´s, is that they assume their ontologies have a higher level of reusability than the other ontologies they compared in their studies. Such a feature would make their ontologies more reusable, but according to our evaluation it is not simple to highlight one of the evaluated ontologies for permitting interoperability among the others.

Because it was not possible to find the ontology category of Upper Level in the evaluated network. In the following subsection, we will proceed to describe the next steps of our methodology in order to determine whether or not it is possible to establish other kinds of relations between those manufacturing ontologies.

4.10 Hyper Modules Extraction

As discussed in previous Subsections, modularity can be pursued as an alternative to the ULO approach, intended to enable maintenance, publication, validation and processing of ontologies [8]. Two main modularity approaches have been distinguished from each other in the literature, the former corresponds to the division of a large ontology into modules, thus rendering such an ontology to be more manageable, and the latter corresponds to the generation of small modules from links between some collections of ontologies in an storage. Defining so-called 'hyperlinks' between ontologies is intended to enable the interconnection of related ontologies, but without merging the logical content of those ontologies.

In addition to mappings between concepts that are considered equivalent, which is the foundation of hyperotological networks, we argue that the identification and consideration of subsumption relations between equivalent concepts in this Hyperontology will allow us to identify *hypermodules*. Onto*Smart* provides algorithms 1[19] and 2[20] to identify such modules.

As we indicated above, with the implementation of our algorithm in the network of ontologies presented in Fig. 11 it was also possible to find a group of six hyper-modules of multi-module type. Every multi-module, contains several concepts subsumed by a root concept which in turn is connected to another concept in another ontology through mapping. These modules were named `Unit, Product, Process, Features, Resources` and `Geometry`. Regarding the specific concepts, it is necessary to remember the list of concepts mentioned in Subsection 4.9 proposed

[19]shorturl.at/aknIM
[20]shorturl.at/htKTW

Table 7: List of Monomodules

Monomodules
Event
Material
Set
Organization
Person

by [19] and [16] as significant for the manufacturing domain. Those terms are `Product, Process, Resource` and `Equipment`. Moreover, we are including the root concepts `Unit, Features, Geometry` and the concepts `event, material, set, organization` and `person` (`mono-modules`). With this information we proceeded to present every resulting hyper module of multi-module type, at first for every-module evaluating graphics is presented, and then an ontological view of the module is shown, this view was obtain after editing the hierarchical view of the hyper module in an ontology editor, Protégé for this case. After describing every hyper module, they will be included in a single view that integrates the hyper modules of the Hyperontology with the network of ontologies.

Hyper module of Unit

In Subsection 4.9 we introduced Table 6 to present the most frequent terms found in the network of ontologies shown in Fig. 11. With the application of our proposed algorithm, the concept `Unit` appears again as a part of a hyper module of multi module type. This concept is integrated with the concepts `BaseUnit`, `Mega`, `micro`, `meter` and `degree Celsius`. It is necessary to mention the relevance of these terms for Science and Engineering, given that metrology, as the science of measurement, takes part in those fields. Nevertheless, besides the importance of the term `Unit` for manufacturing, it does not appear in every ontology of the network; moreover the other terms that form part of this hyper module appears in only 2 of those ontologies.

In Fig. 13 we show the results obtained calculating the values of Boundary Strength (BS) and Domain Strength (DS) of the concept `Unit` and some of its subsumed terms among ontologies included in network shown in Fig. 11. There, the concepts present in this multi-module are evaluated in two steps: first an evaluation of the ontologies that contain the concept `Unit` and, second, around every term subsumed by `Unit`. Most of the terms presented there have a BS of 0.4 and a DS of 0.26. This result means that, although the BS is closer to 50%, when measuring the domain under study, the average DS indicates a weakness more than a strength on the terms related to `Unit`. Going into details, the term `Unit` is presented in 5 of 8 ontologies of the network, while the other terms are present only in 2 of 8.

Fig. 14 shows the concepts and instances that appear mentioned in Fig. 13. This set of concepts was obtained as a partial result of executing the `compute_hypermodule` routine which is part of our algorithm. This figure shows the result as an ontology.

Hyper module of Product

Similarly, and following the procedure described above, the Hyper module `Product` was obtained. This hyper module is integrated by the terms `Product, Part` and `Assembly`. It is worth noting the relevance of these terms in Semantic Manufacturing because, on the one hand the Product can be considered the center of this conceptualization, and products many times are integrated by some type of parts, generating an assembly. On the other hand the presence of the concepts `Part` and `Assembly` in this hyper module indicates the existence of a relation that comprises the product, and this type of relation is known as part-hood relations. Later, as part of our methodology, when

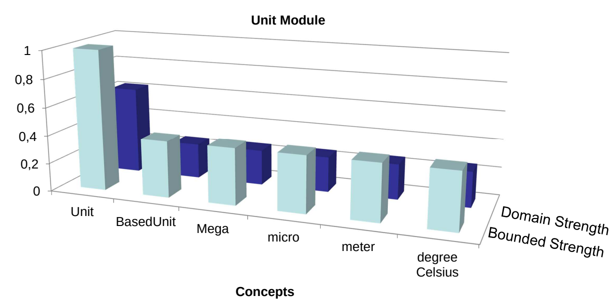

Figure 13: Strength of Concepts Bounded by Unit

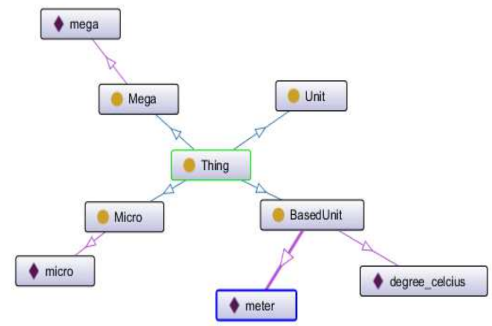

Figure 14: Partial Hyperontology for the term `Unit` graphed with Protégé

Axiomatization and Heterogeneity activities take place, the part-hood relations issued by OWL, will be taken up and discussed. Below we present the analysis related to this hyper module.

Fig. 15 shows the results of the statistical analysis we performed, which is the term `Product` has a BS of 0.83, followed by `Assembly` with a BS of 0.5; and the term `Part` with a BS of 0.33 Moreover `Product` has a DS of 0.625 followed by `Assembly` and `Part` with a DS of 0.375 and 0.25 respectively. In the case of the term `Part`, and following a criterion similar to the one used for the hyper module `Unit`, we can state that there is a weakness in it, which means there is a low likelihood of achieving interoperability through it. After analyzing this hyper module according to the proposed BS and DS metrics, we proceed to represent it as an ontology in Fig. 16. This figure corresponds to a simplified view given that the corresponding ontologies, and other hyper-modules have to be integrated in a complete view, present in the hyper ontology as integrated by hyper modules in an ontologies network.

Hyper-module of the Process

Following the statistical analysis of results, Fig. 17 introduced the hyper module of `Process`. `Process` and `Operation` are meaningful concepts for the manufacturing domain. In this vein, the concept `Process` has been mostly adopted to define ontologies related to this terminology. Nevertheless, from the graphic it can be concluded that in this network the concept `Operation` is found with more strength (BS: 0.6) than the concept `Process` (BS: 0.4). Other terms found in this Hypermodule were `Change`, related to `Process` and `Operation`, and the terms `Cutting, Drilling, Milling` and `Addition`.

The last four terms mentioned above are closely related (`Cutting`, `Drilling`, `Milling` and `Addition`), given that the three terms correspond to a manufacturing application domain named machining, and all of them belong to MASON. We go into details of these results because the authors of MASON declared it is an Upper Level Ontology. These concepts have a low BS and DS (0.2 and 0.125 for each respectively) which makes them highly usable, but less reusable because of their specificity. This result is not necessarily related to quality, because the effectivity of this ontology will depend more on the skills of the ontologist than the ontology itself. But, the resulting values of BS and DS of those MASON concepts serves to categorize MASON as an Application Domain Ontology. This is not an isolated result. In previous Section we mentioned the research of other authors who have made similar remarks about their work, but with the implemented metrics, procedures followed, and results obtained, the classification

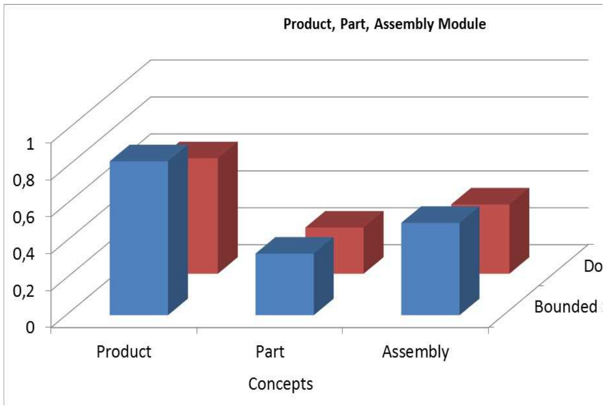

Figure 15: Product Hyper Module

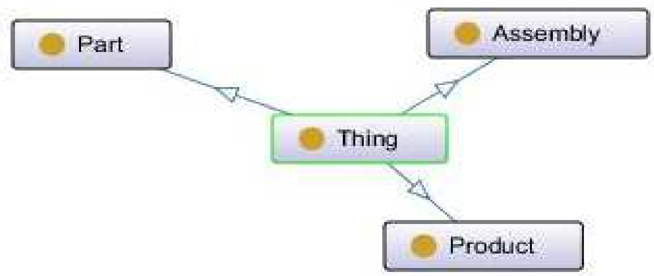

Figure 16: Members of the Hypermodule Product

made by these authors on ontologies as ULO´s can be questioned.

In Fig. 18 we introduce an ontological view of the Hyperontology of the `Process` as resulting from Fig. 18. The concept `Operation` was used as root concept for specific operations due to its higher DS (0.375), while `Process` and `Change` remained at the same `Operation` level. **meaning**

Hyper module of Features

Continuing with our Hypermodule extraction, we processed a hyper module containing some features commonly found in mechanization processes. We named this hyper module of features. Some of the concepts contained by this module are `Circular_Pattern`, `Threat`, `Slot`, `Pocket`, and `Chamfer`. Most of the terms were found in MASON and Onto STEP, which will allow information exchange among both ontologies. It is necessary to highlight that the features mentioned below are common in mechanization processes, but they are not the only ones and they were also not found in the other 6 ontologies. For instance, PRONTO was proposed to represent products and their features, however no hyper-mapping was found in the network, because they referred to different Application Domains: MASON corresponds to machining processes, and PRONTO to breaking up or separating processes.

Fig. 20 shows the Hyperontology of Features, where concept `Features` is the root concept in this ontology. These Features mostly apply to mechanization procedures.

Continuing with the description of the hyper modules obtained by the execution of the proposed algorithm, we obtained `Resource, Machine, Tool, Lathe` and `Drill` concepts. All these concepts were found in the MASON and OntoMoPS ontologies, while in OntoSTEP only 2 of them were found.

If we return to the results previously obtained during the evaluation of the Hyper module of Features, we found that the features and the resources to get the features manufactured on the raw material are present in the MASON ontology as well. For instance, with a Lathe we can make circular patterns. This is a commonality that until now has only appeared in this case. Where MASON appears with common concepts that relates two pairs of ontologies: a first pair OntoStep-MASON where the concept `Feature` is developed, and a second pair MASON-OntoMoPS where the concept `Resource` is developed. Some of the resources mentioned in the latter pair are required as machinery for manufacturing the Feature mentioned in the former pair.

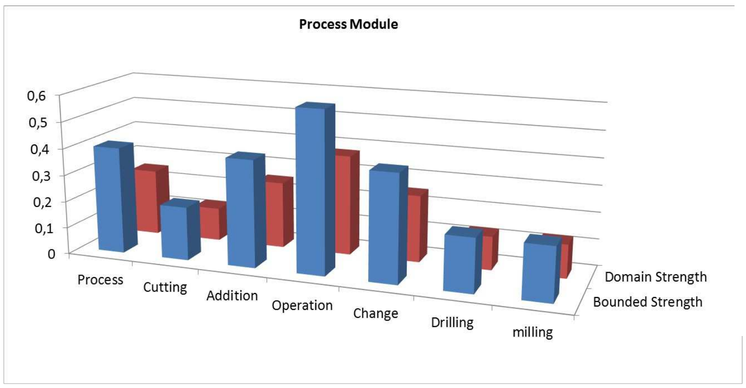

Figure 17: Hyper Module Process

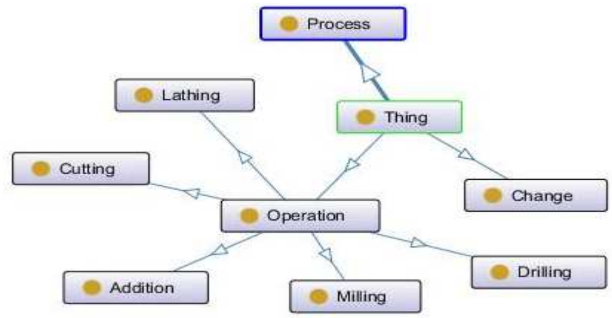

Figure 18: Hyperontology of Process

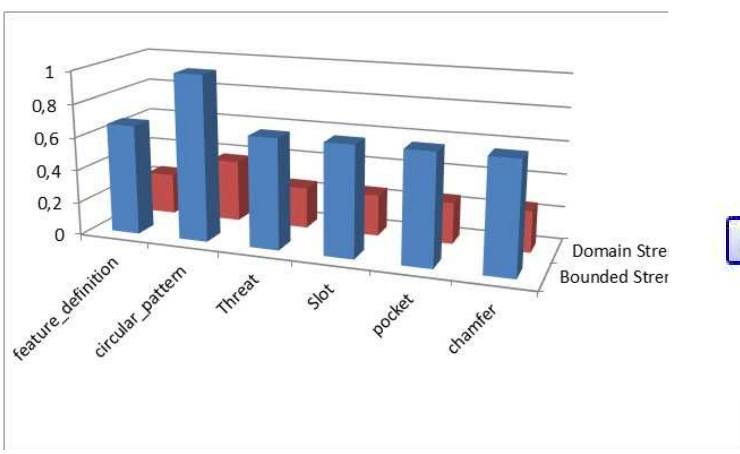

Figure 19: Hypermodule of Features

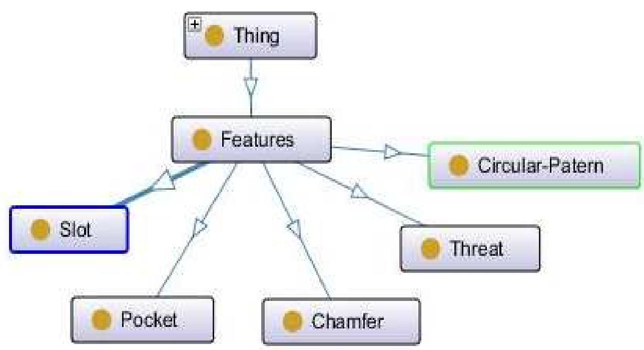

Figure 20: Hyperontology Features

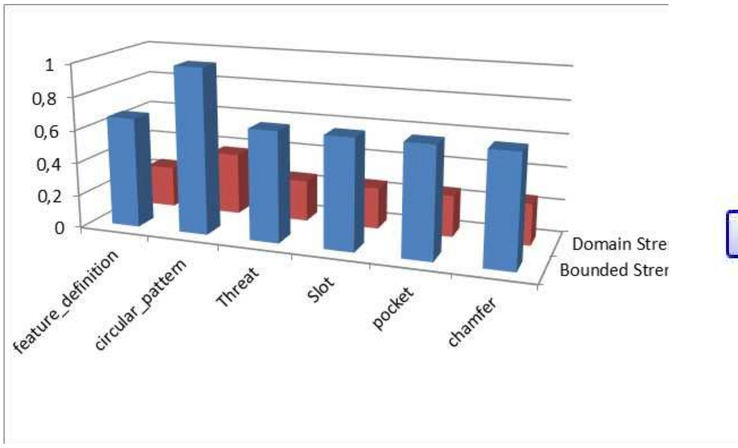

Figure 21: Hypermodule Resources

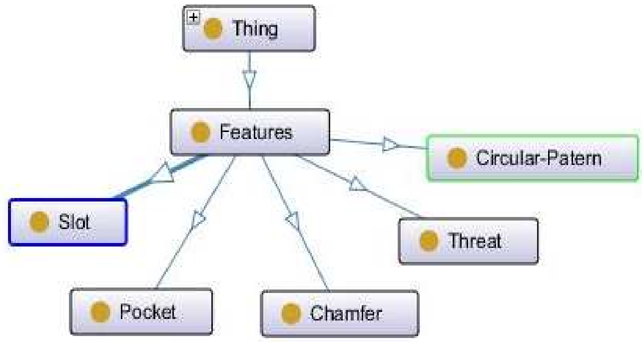

Figure 22: Hyperontology of Resource

In this vein, Fig. 21 contains the result of applying the BS and DS parameters to the network where these `Resource` and `Drill` concepts have a BS of 1, while the others have a BS of 0.66. Besides having the same BS value, `Resource` is more general than `Drill`, where `Drill` is a type of machining. The resulting ontology is shown in Fig. 22.

Hyper module of Curve

The last hyper module we found following the proposed algorithm, corresponds to the `Line, Conic, Curves, Circle, Hyperbola, Ellipse` and `Parabola` concepts. The mappings found correspond to 2 ontologies only, those are OntoStep and BeyondSTEP, while the other ontologies of the network remain isolated.

In this case, the BS of every concept is equal to 1, meaning that every concept is in all ontologies of the boundary, and DS is equal to 0.25 which means that these concepts make up less than 25% of the ontologies. Fig. 24 introduces the categorization of these concepts. Considering `Line` and `Conic` as `Curves`, and `Circle`, `Hyperbola`, `Ellipse` and `Parabola` as `Conics`.

With the hyper module of curves we complete the application of the algorithms proposed in Onto*Smart*. The obtained hyper modules now have to be integrated with the network of ontologies shown in Fig.11. At this point it is worth remarking that our hyper ontology will be formed by some hyper modules, and in the case of our algorithm we have proposed two types, mono-modules and multi-modules. This Hyperontology is an interoperability artifact that is located on top of the network of ontologies, and the mappings obtained from the algorithm serve as hyperlinks. With these notions we proceed to summarize our findings in Fig. 25, which at the same time serves as the representation of our complete Hyperontology. At the bottom of the figure, we show the network of ontologies from Fig. 11. At the top, every Hypermodule is drawn; the mono-modules containing only one concept in yellow, and the multi-modules containing several concepts in black.

This figure illustrates the complexity of representing a domain of discourse, concretely considered in this case for manufacturing. For instance, a product can be goods or service. If we choose to represent goods, some products could be represented as a CAD drawing, while others, like liquids, cannot be represented in this way. In the case of the resulting hyper ontology, it will only serve for representing solids according to the Features terminology it

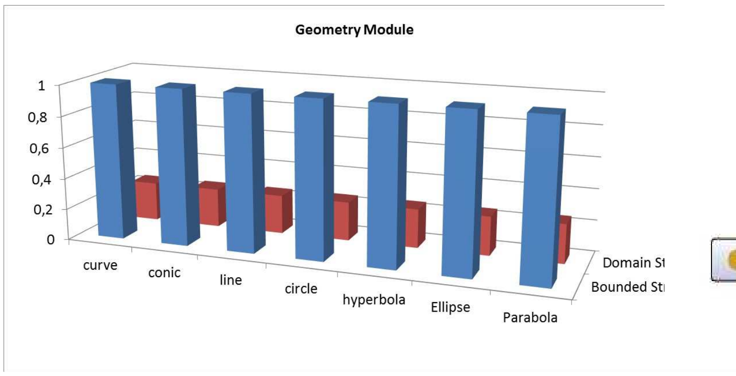

Figure 23: Geometry Hypermodule

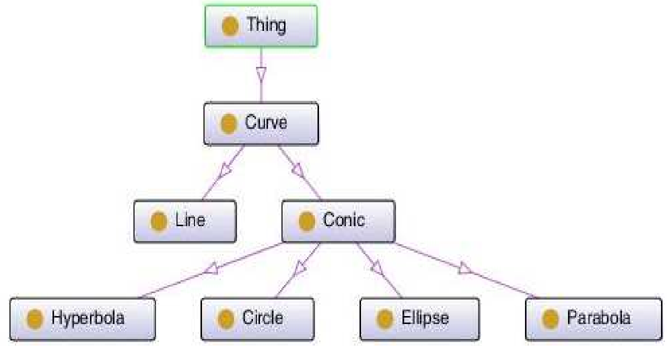

Figure 24: Geometry Ontology

contains.

Furthermore, it is likely to find at least two ontologies with restricted scope. Those include, for example, the Sweet Ontology, which only has a mapping through the hyper module of `Unit` toward 4 more ontologies, and BeyondStep which has two mappings through the concepts `Features` and `Geometry` toward 3 more ontologies. Such mappings allow us to extend those apparently restricted scope ontologies through these mappings toward more complex ontologies.

Moreover, if we consider ontologies with an apparently different scope in a manufacturing domain, like MASON (manufacturing) and GoodRelations (ecommerce), their commonalities allow certain information exchange through this network, in this specific case information about the product. In contrast, information of the manufacturing process is not exchangeable because these concepts are not part of GoodRelations terminology.

Here we complete the activities of modularization planned in our methodology, before proceeding onto the steps of Axiomatization and Heterogeneity. Moreover in Section 6 an application example with the obtained hyper ontology in Fig. 25 will be introduced.

5 Between Light Weight and Heavy Weight Ontologies

Continuing with the development of our methodological approach as depicted in Fig. 1 of our Proposed Methodology, we have to proceed to determine whether or not heavyweight ontologies are required, following the procedure indicated in Subsection 3. In this stage we applied the metric *al* which lets us know the average of defined concepts in an ontology; moreover we proposed a criterion to determine whether or not a heavyweight ontology is required in our application[21].

Following the proposed criteria, first it is necessary to determine whether heavyweight ontologies are needed, which depends of the application requirements. Thus we have to consider the following aspects: All ontologies considered in Fig. 25 were written in OWL and OWL is based in Open World Assumption (OWA), which means an individual is considered as a member of a concept, and it is necessary to provide a specific definition of the concept

[21]https://github.com/luisenriqueramos1977/OntoSmart/wiki/Axiomatization

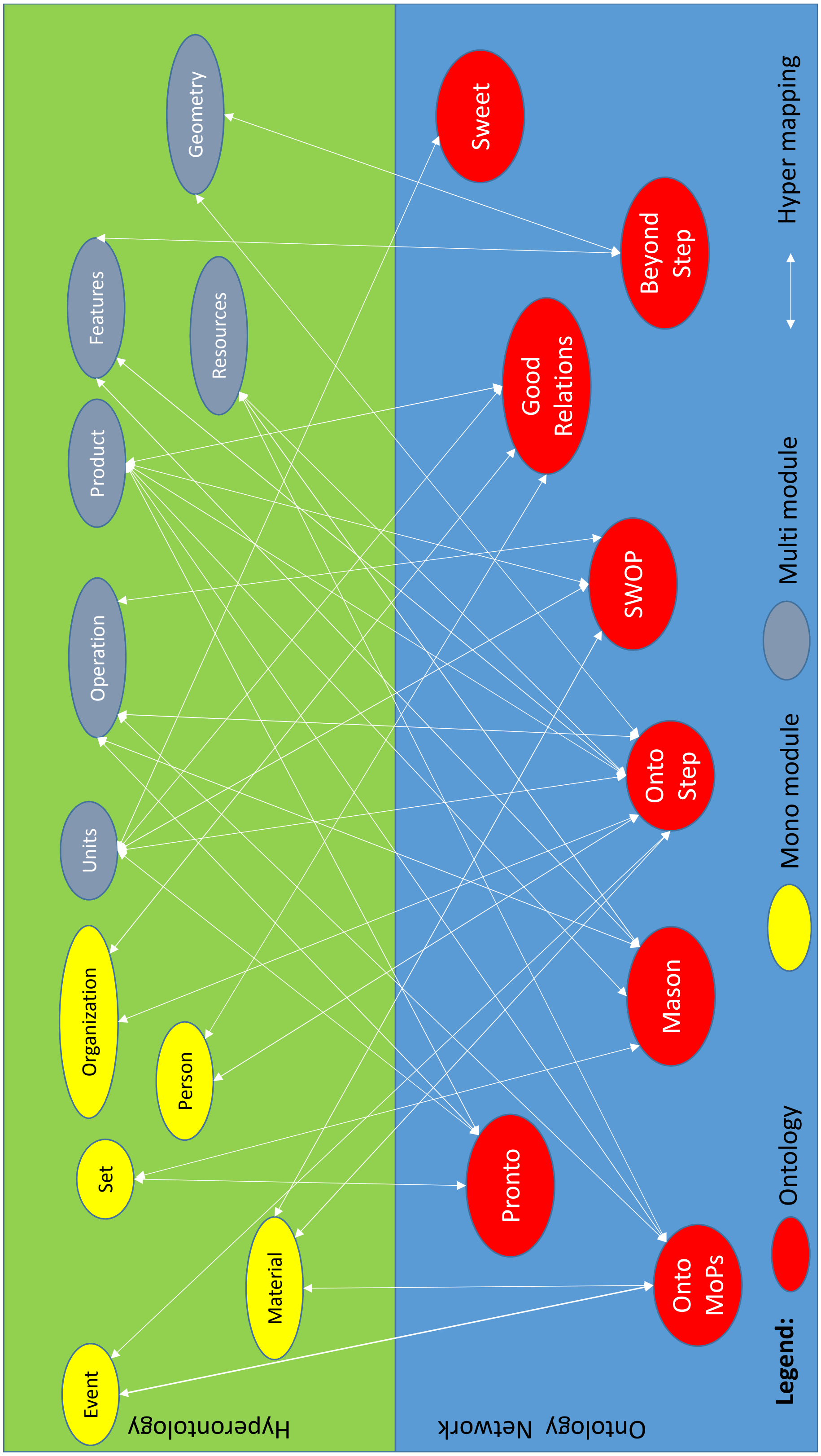

Figure 25: Resulting Hyper ontology

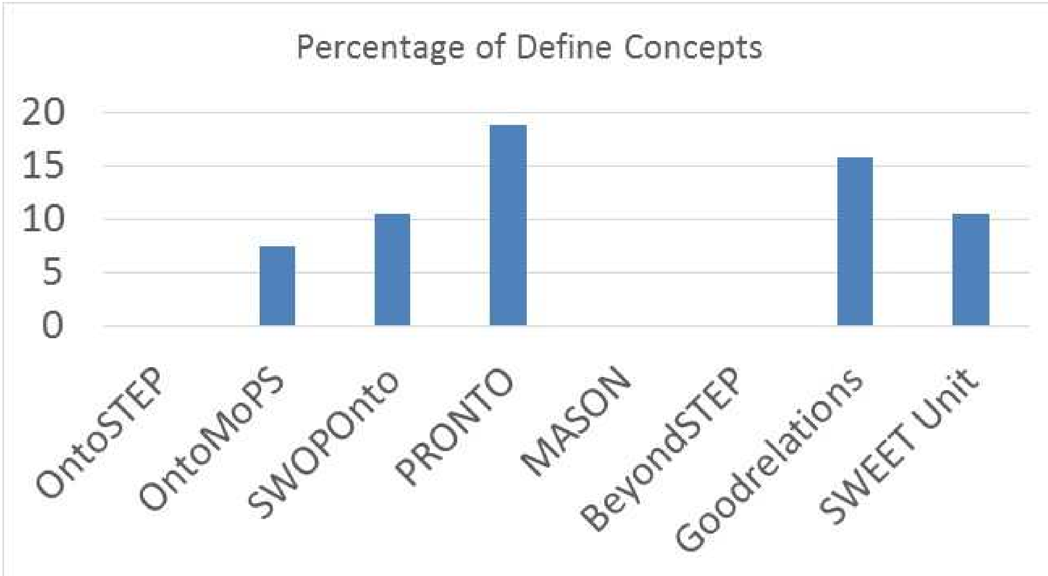

Figure 26: Defined Concepts in Network

or restrictions through closure axioms. These closure axioms allow the classification of individuals as members of a class. This classification feasibility is a fundamental characteristic in Ontological Engineering.

To illustrate this classification need in engineering, we can consider the Automatic Features Recognition (AFR) approach, which consists in recognizing certain features from CAD files. In the hyper module of Features, some of the most common mechanical features mentioned in the literature are included. When AFR is executed on a digital design, the following classification takes place: a group of geometrical elements receive mechanical and manufacturing semantics when they are classified according to some specific features, defined by a set of axioms.

There is also the Flexible Manufacturing System (FMS) approach, which assumes that with an intensive utilization of Numeric Control (NC) techniques, automatic materials handling and computer hardware and software, the manufacturing time for customizing goods should be considerably reduced. Here a classification is also involved whenever from a group of resources (machinery) we have to decide which can fulfill our manufacturing requirements and how they should be organized in the factory.

Another example can be provided considering market segmentation principles. In this case customers are divided into groups or segments, based on their preferences, and products can be assigned to certain segments of customers according to the satisfaction levels they are able to provide for each group or segment.

Through the description of the three common scenarios mentioned above, we can consider that heavyweight ontologies are required in such scenarios in order to enable classification.

After defining the requirement of heavyweight ontologies for manufacturing, we applied the formula proposed in Onto*Smart*[22] to obtain the relation between defined and primitive concepts in an ontology, with the intention of defining the purpose of the evaluated ontology, controlling vocabulary or classification. Thus, we can confirm our position that manufacturing ontologies should have mostly defined concepts with the intention of making such ontologies usable.

[22]shorturl.at/oI347

Fig. 26 presents a view of the presence of defined concepts in the network of ontologies under evaluation. From the 8 ontologies considered, 5 have some defined concepts in different proportions, varying from 7 to 19% of the total. Here, an apparent contradiction appears: that is while in the previous paragraph we stated that heavyweight ontologies are needed for applications like AFR and FMS, Fig. 26 shows that in many ontologies like OntoSTEP, MASON and BeyondStep, there are no defined concepts. However, in other ontologies like PRONTO and GoodRelations there are defined concepts present, ranking from 15% to 19%.

To relate usability with heavyweight ontologies, we consider the result obtained in the case of the GoodRelations ontology, because from ontologies mentioned in Fig. 26 only this has shown use outside of academia. This ontology is used in some of the most important search engines and it is well accepted in the ecommerce community [12]. Moreover it has an appropriate platform for making it available to a community of users. Approximately 16% of this ontology is made up of defined terms, and its basic use consists in allowing goods selection according to user criterion.

Another necessary consequence of this result can be concerning OntoStep and BeyondStep. From the percentage of defined concepts in OntoStep and BeyondStep shown in Fig. 26, we can assure they are lightweight ontologies because they contain no defined concepts. Thus, if we wish to perform AFR, many terms in these ontologies shall be defined, in other words OntoStep and BeyondStep, are incomplete for AFR and FMS, and the Ontological Engineer will need to perform additional work defining concepts to make these ontologies usable with this type of tasks.

In this Subsection we discussed axiomatization requirements, and showed that from the list of ontologies we have evaluated, many were lightweight. This result does not make these ontologies useless, but it does make necessary additional work to define appropriate concepts in order to make them usable in tasks like FMS and AFR. For these tasks, in our opinion, heavyweight ontologies are required.

Following with our methodology shown in Fig. 1, we proceed to the heterogeneity activities. However as we will later demonstrate, this decision is based on application requirements, so in the next Section we present an application example where domain requirements will take us back to heavyweight ontologies in order to fulfill the application goals.

6 Implementing Heavyweight Ontologies

At the beginning of report, Computer Aided Systems (CAx) systems were introduced as an attempt to improve manufacturing automation, optimization and acceleration levels. For example, during this research we observed how AutoDesk developed its DXF, a *de facto* standard for the exchange of CAD data facilitating reading a CAD design previously deployed using other software tools such as AutoCAD® of AutoDesk. A detailed explanation of this standard can be found in the DXF Reference Manual [3]. This standard defines geometric primitives such as LINE, CIRCLE, ARC and ELLIPSE entities. A group of codes is specified for each one, indicating what type of data value or feature they follow. It is also possible to extract from a DXF file descriptions of text, surfaces, color and texture, but the information on solids is encrypted [7] and limits data information exchange among CAD design. In the same way, the work flow starting from the manufacturing design is restricted to the AutoDesk family of products. Therefore, besides the important advantage of CAx systems, these technologies have shown patent shortcomings, one of these is the lack of interoperability. This interoperability issue occurs because most vendors allow interoperability along the workflow of the tools they offer, but interoperability between tools of different vendors tends to be limited in some way.

In recent years there has been a movement toward the utilization of the ontological approach in engineering applications for the representation of CAD models to capture feature semantics and to use such models among different systems maintaining the designer's purpose. [1] presented an architecture for a Data Exchange among different CAD software tools, where ontologies are proposed to represent terminologies of several commercial CAD software tools, and a main ontology would serve as a Common Design Feature Ontology. [1] proposed to write and store ontologies of each CAD system using OWL, generating ontologies of such systems. These ontologies have to be mapped into a Common Design Ontology to make them interoperable across different software applications.

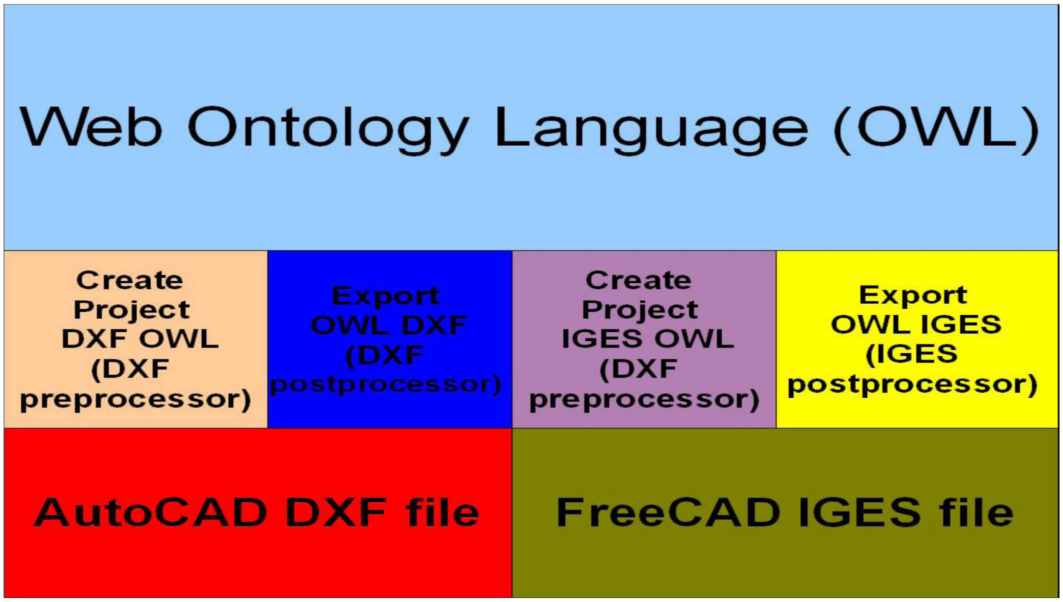

Figure 27: OWL Based CAD Exchange Framework

Similarly, [2] proposed an ontology of CAD model information; this proposal is described as an introduction to ontologies and shapes representation and deals only with the STEP standards as also done by [1], presenting a taxonomy of terminologies included in the STEP standard.

In contrast, building on the proposal of [1] we have proposed a different data exchange approach [22]. There, intermediate ontologies were avoided and a general CAD terminology was used without adopting specific standard CAD terminology. We considered this could be a disadvantage for an approach that, on the one hand, claims to be ontological while, on the other hand, it is only being related to one CAD standard. Fig. 27 presents our first attempt to deal with this interoperability issue in CAD. In this figure, we exemplify the architecture with Drawing Exchange Format (DFX) and IGES standards. Consequently, in the figure there are four parsers: Two preprocessors parsers for exchanging DFX into OWL, and two postprocessors parsers for exchanging OWL in DFX, similarly in IGES. As soon as a standard is converted into OWL, the reasoning framework of the Semantic Web becomes available. With a CAD design based on OWL, it is possible to provide reasoning on the CAD ontology, in order to exchange from one format to another.

In order to assess the usability of the Semantic Web in manufacturing, other uses additional to exchanging data of specific CAD formats like DXF and IGES shall be considered. The issue of representing and reusing the designer's purpose in a CAD designs is one of those. This issue is illustrated in Fig. 27 and Fig. 28. The former shows a common mechanical piece used in engineering, better known as a flange. The latter shows the same piece obtained by automatically parsing a DFX file into a CAD–OWL file generated by means of a plug-in called "CAD Viewer Tab", which we developed as part as our previous work related with this research. This plug-in was

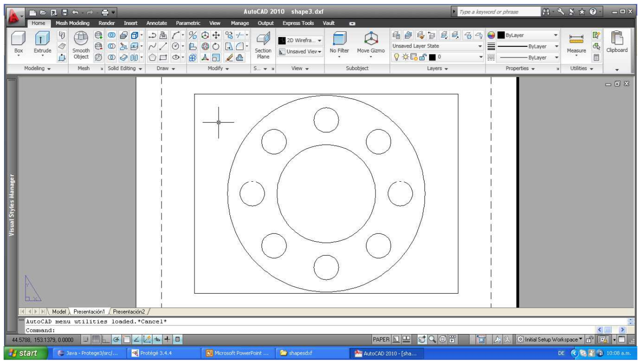

Figure 28: Shape in AutoCAD

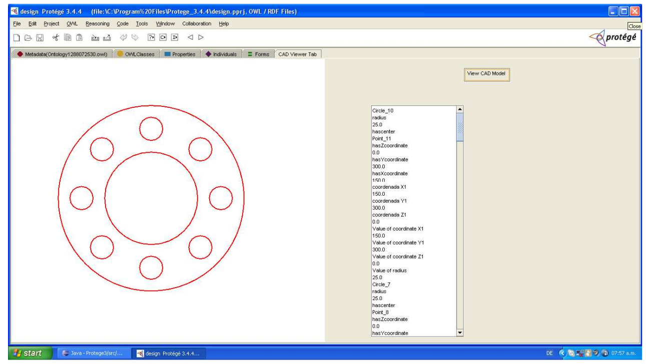

Figure 29: Shape in OntoCAD

integrated in Protégé. This plug-in implements the procedure of our proposed architecture, specifically the "Create Project DFX OWL" parser shown in Fig. 27.

Even though both figures seem to have the same representation of a mechanical piece, they are actually not the same. Only primitives (circles) are what have been actually exchanged from AutoCAD into DXF, and so into OWL. This means that while the viewer may perceive a flange, for the computer they just remain as circles. Thus design purpose (knowledge) is lost during the exchange due to missing semantics.

In order to recover the designer intent, it is necessary to provide tools or representations to capture the designer's knowledge. Moreover, it would be appropriate to support the knowledge of the manufacturing engineer, quality control, and other related tasks as well.

Fig.30 illustrates how we consider Semantic Web Technologies should be ordered in a general framework to make the requirements mentioned in the previous paragraph. Going into details, in Fig.30 we propose a foundation of the initial architecture.

In the middle of the architecture, we placed Semantic Web technologies for 2D, and a 2D features ontology. Thus, primitives can be interpreted as features, and quality control can be applied to them. At the top of the architecture, Semantic Web technologies adapted for 3D interpretation are included. A GUI was developed in order to allow visualization of the design. Although this framework apparently covers the aforementioned requirements, we want to introduce a more technical example to view detailed limitations of the OWL language for representing certain manufacturing scenarios, and to link it to our evaluation and definition of the above hyperontology.

As we mentioned in the previous paragraph, we want to introduce a more technical example from the manufacturing point of view, in order to require the major expressiveness of OWL. Such an expressiveness level demand can be found even in the manufacturing process of 2D design. Sheet metal parts are some of these. These elements are commonly used in several products manufactured in modern industry: aerospace, automotive, appliances, machine tools, etc. are only some of these. Although various software tools are available for making digital designs for such metal parts, trying to use those designs as input for process planning and manufacturing is not simple. The majority of standards for CAD do not represent the particular features required when manufacturing. For instance, declaring when a circle corresponds to an inner edge or an outer edge has significant consequences during manufacturing, although this may not be necessary when displaying designs. Allowing interoperability across the CAD-Computer Aided Manufacturing (CAM) process should enable designers to select optimal manufacturing conditions. For instance, to choose a sheet thickness that would prevent crashes when punching holes.

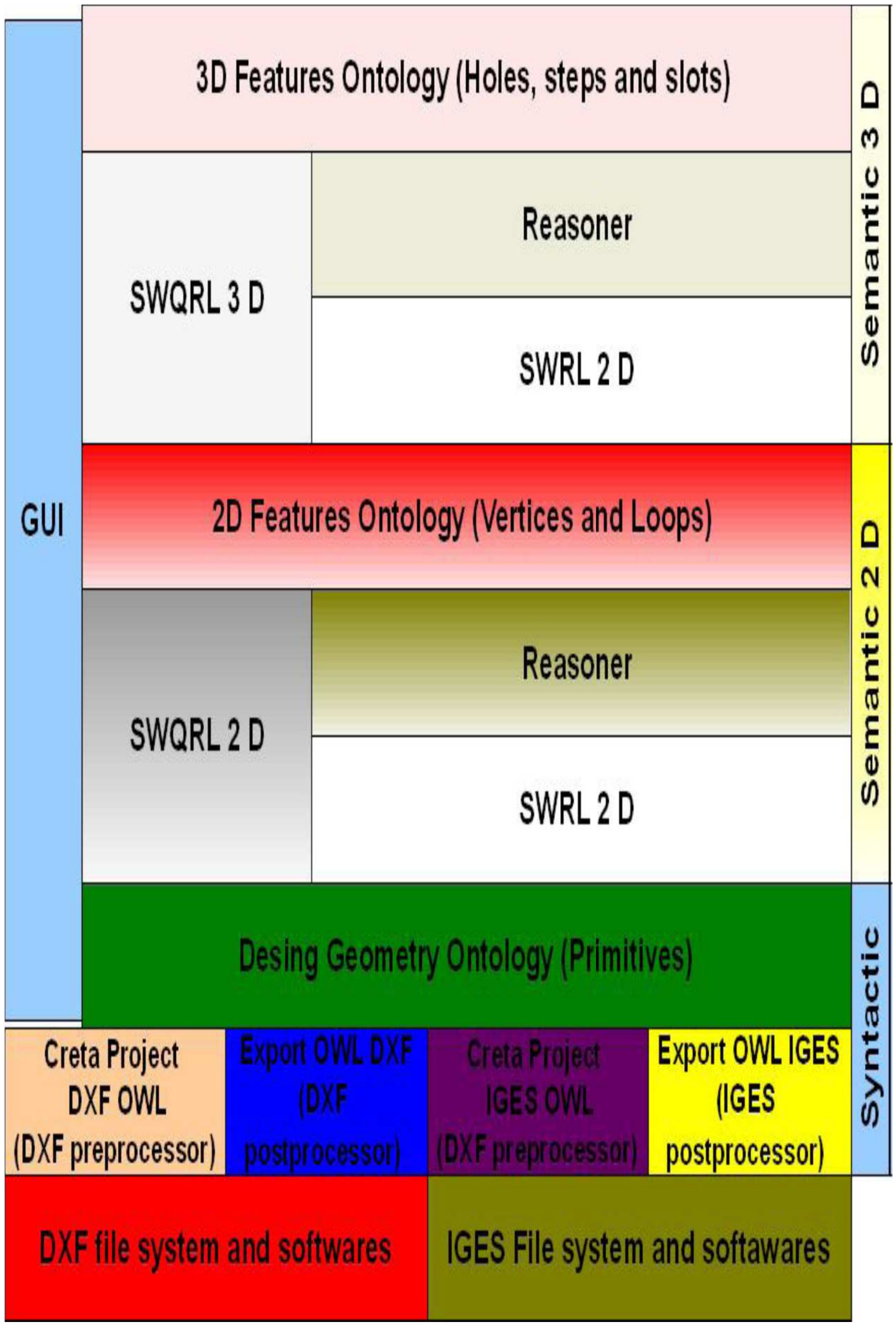

Figure 30: OntoCAD Framework

In order to achieve such interoperability, manufacturing features must be extracted from CAD files. Extracting these features demands identifying certain patterns in the CAD files that shall receive additional manufacturing-relevant enrichment. Information about features is further used to determine the machining tools and manufacturing processes required to manufacture a given design [6]. Feature information can also then be used to pre-check designs in order to detect production rule violations. If these violations are not detected in an early stage of design, the life cycle of development increases, raising production costs and time to market [21]. Manual recognition of the targeted features is not a viable alternative, however: AFR is required. Nevertheless, even nowadays AFR is not fully integrated with CAD software tools. It is applicable only to parts with relatively simple geometry and still requires human intervention to obtain the features identified. Moreover, such techniques are generally supported only in expensive CAD software tools that are beyond the reach of small scale industry [13].

In [24] we proposed the partial application of the framework presented in Fig. 30 in order to deal with the shortcomings of AFR from sheet metal designs and designs checking. This ontology-based framework should facilitate interoperability across the entire CAD-CAM process. Our system integrates both a CAD and a features ontology; these ontologies were written in OWL (Web Ontology Language) and represent 2D primitives as lines, arcs and circles in the CAD ontology and as edges, slots, tab and holes in the features ontology. In order to provide rules and query support, OWL was complemented with SWRL and SQWRL. The result is an ontology-based system that allows us to automatically extract the most common manufacturing features referred to in the literature.

One of the main issues of AFR is the abstraction level involved in the manufacturing domain [26]. On the one hand, designers make designs from the functionality and usability point of view, with its own restrictions and rules. On the other hand, manufacturing engineers evaluate a design from the manufacturing point of view, considering factory constraints. This is mainly the reason why quite distinct standards, formats and tools have been developed. In the two domains, two different interpretations of the requirements take place for the same purpose, so an interoperability channel between them is needed. This channel needs to enable the interchange of information of products as well as the evaluation of designing and manufacturing restrictions.

In Fig. 31, we introduce the mapping architecture implemented for dealing with this situation. Knowledge transfer between domain ontologies was enabled by a third, mapping ontology, which keeps track of the related concepts of both domains. This mapping ontology was developed using the Prompt plug-in [20]. This latter architecture can be considered as a simplified and specialized view of Fig. 25, where two terminologically similar ontologies, MASON and OntoStep were hypermapped with BeyondStep within a feature's Hypermodule. Terms present in those ontologies are naturally related to the 3 dimensional domain given the relations found by the results obtained from the extraction of the feature's hyper module, which was described in Section 4.10. We want to remark that the application, based on the architecture depicted in Fig.31, has a limited scope and it was developed as part of our initial experiences, while the Hyperontology proposed in Fig.25 has a large scope within the manufacturing domain.

Moreover, Fig. 31 makes evident how we deal with one of OWL's shortcomings, which was the OWL limitation in performing inference over inferred knowledge. In other words, if in a given Knowledge Base (KB) we applied certain rules to determine the quality of a design, and we inferred the design or some parts of it are technically correct, the inferred design or its parts cannot be further reused, unless we record them as asserted knowledge into the same KB.

This OWL issue is also depicted in Fig. 30 when we try to move upstream through the architecture, given that in the case of DXF only primitives are exported in this format. Thus 2D features can be inferred in a first stage, but if we try to go forward for a 3D inference, it is necessary to recreate an environment to enable reusing the previously inferred knowledge. Such an environment implies mapping between several ontologies as in the case displayed in Fig. 31.

For larger sets of ontologies, and with more complex tasks to perform, we can say that we are in the presence of a hyper-mapped architecture. For instance, manufacturing Operations, mentioned in Fig. 25, could be introduced in Fig. 30 in order to deduce such operations for manufacturing the inferred features. Going into detail, Fig. 32 lists most of the features mentioned so far, as well as the restrictions required to make them appropriate for

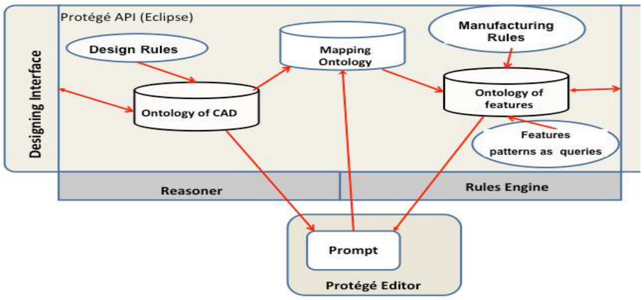

Figure 31: Mapping Architecture

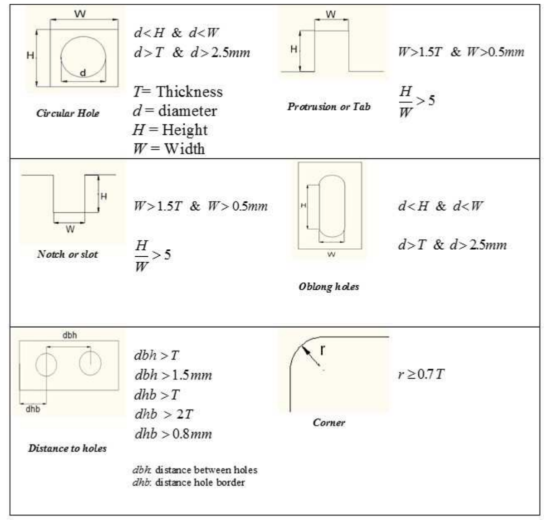

Figure 32: Manufacturing Features with their Constraints

manufacturing. Thus, after features are deduced, through the architecture suggested in Fig. 31 inferred knowledge as features can be available to evaluated them in order to determine their accuracy with quality restrictions. Every restriction illustrated in Fig. 33 was successfully represented with OWL, and in all restrictions, except Distance between holes, quality could be deduced.

This last constraint, Distance between Holes, has a peculiarity. While the others involve binary relations (e.g $has\_Diameter(h_1, d)$), this one involves a ternary relation (e.g $distance\_Between\_Holes(h_1, h_2, d)$), which is not possible to be expressed in OWL and is therefore one of the restrictions of this language. In the next section we explain how the heterogeneity approach will help to overcome this issue.

7 Extending Ontologies through Heterogeneity

Given that in the previous section, a shortcoming of OWL was identified during the procedure of trying to represent certain manufacturing restrictions, and in order to provide inferences for it, it is necessary to recall Fig. 1 to determine which course of action to take. In the methodology set out there, a branch toward heterogeneity was included. Heterogeneous ontologies has been proposed as a further way to deal with the limitations of certain logics, making available other logics that provide more reasoning power.

In [23] we introduced an architecture to deal with the validation of features such as those represented in Fig. 33. These in fact correspond to the Distance between holes example mentioned in previous section, and illustrated in Fig. 32. That is, there are two holes, h_1, h_2, and a restriction of distance d between them . We can represent such a restriction by means of the following ternary relation: $distance\_Between\_Holes(h_1, h_2, d)$. Furthermore, we add a new restriction called $distance\_Hole\_Border$, which is also mentioned in the literature. These distance features are considered during manufacturing in order to avoid metal deformations when the hole is too close to the border, or when holes are too close to each other as well.

Each of these restrictions involves two instances of the sheet metal issue, and they become related by a valued number restriction. For representing these features, we proceed as follows: in equation 3, an intra-feature restriction is present, while in equations 4 and 5 inter-feature restrictions are introduced.

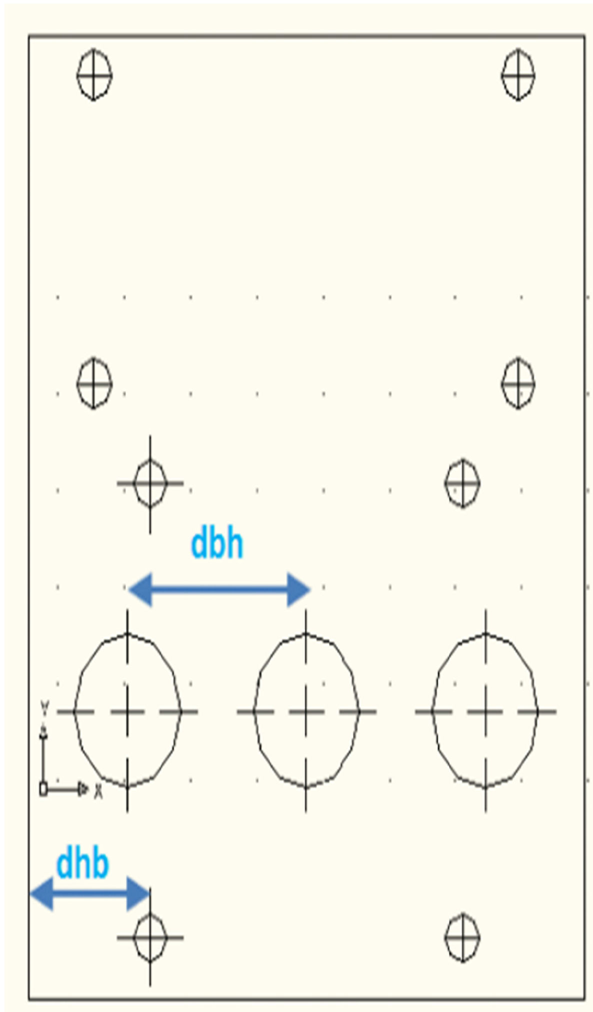

Figure 33: Restrictions in Sheet Metal Parts Fabrication

$$has\_Diameter(h_1, d) \tag{3}$$

$$distance\_Between\_holes(h_1, h_2, d) \tag{4}$$

$$distance\_Hole\_border(h_1, b_2, d) \tag{5}$$

The restriction described in equation 3 can be represented in OWL and conclusions about the quality of the feature itself can be deduced by reasoning. Restrictions expressed in equations 4 and 5 can be indirectly expressed in OWL, but no conclusions about the quality of this design can be obtained from the OWL model because OWL reasoning is limited to binary relations, while equations 4 and 5 illustrate ternary relations. It is also worth mentioning that the value of these features (dbh and dhb) shown in Fig. 33 also depend on the materials and other features, for instance thickness. Moreover, we have also avoided mentioning metric units in order to simplify the exposition at this point. This example is an illustration of a type of issue that involves higher predicate arity in a specific task where OWL expressiveness is insufficient to reach the validation of certain mechanical features.

To deal with a scenario such as the one previously described, and to fulfill the modeling requirements of engineering, we at least need an ontology language with a higher expressiveness level. But, when we move from OWL to a more expressive ontology language, we also face the risk of falling into undecidable scenarios. This is a common trade off that has been previously studied and for which frameworks have been proposed. In this vein, if we can precisely divide the scenarios when OWL is expressive enough for our purposes from the ones where OWL is not enough, then we can introduce a formalism to represent our requirements and evaluate its decidability level.

Therefore, we propose a heterogeneous architecture to bridge the gap between representing and reasoning over manufacturing requirements in the Semantic Web. Fig. 34 presents our architecture. It is a modified view of the architecture presented in [29]. The main difference is the inclusion of a heterogeneous layer. Such a layer allows us to face engineering requirements written in different logics. Our architecture includes:

1. A Product Ontology (PO) that contains product definitions and features that can be represented in a given language and where conclusions concerning their accuracy can also be obtained.

2. Exemplifications of this PO that can be performed by introducing specific designs into it.

3. A heterogeneous bridge for when higher expressiveness with proof capabilities is required.

4. Quantities, Units and Scale are also included as elements of this architecture, because they are a fundamental aspect in product descriptions. There are links to the Heterogeneous Bridge because we do not want to employ a fixed standard system, but leave open the possibility of assigning the one preferred by the product user.

Considering the last mentioned aspect related to unit systems, in this case we decide to validate a library of CAD designs. The features listed in Fig.33 could be represented in different metric systems, and even in heterogeneous metric systems (e.g International Metric System, British Metric System). OWL would not be expressive enough for representing and reasoning on the accuracy of such features. Consequently, in order to simplify the scenario we have considered homogeneous designs at this stage, which means a unique homogeneous measurement system. Heterogeneous Tool Set (HETS) and CASL are the basement of our architecture (See Heterogeneity Building Block[23] of Onto*Smart* for a detailed explanation of HETS and CASL).

Fig. 35 shows the CASL code corresponding to `My_Checker` specification. In the upper part, a Sheet Features Ontology (SFO) is imported with its terminology into My_Checker, and the natural numbers library (Nat). Then, a group of ternary predicates are defined: designDistance, standardDistance and properDistance. Finally, we specify that there will be a proper distance if the design distance is greater than the standard distance.

Fig. 36 introduces this proof obligation for a set of instances from the SFO ontology. There, the `properDistance` predicate is used to evaluate the values of `hole_u1` and `hole_u2` from SFO.

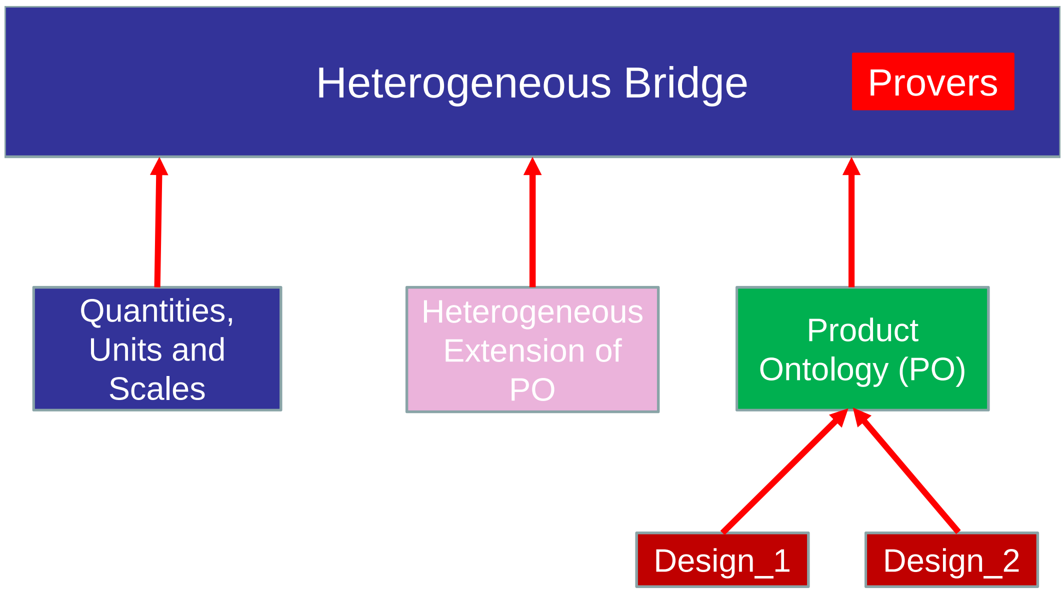

Figure 34: Heterogeneous Ontological Manufacturing Architecture

```
spec My_Checker =
          SFO and Nat
then
          sorts integer < Nat; Nat < DATA
          pred design_Distance: Thing * Thing * Nat
          pred standard_Distance: Thing * Thing * Nat
          pred proper_Distance: Thing * Thing * Boolean
          foralla,b:Thing
          .exist c:Nat
          Hole(a) /\ Hole(b) /\ c in Nat => standard_Distance(a,b,c)
          foralla,b:Thing
          .exist c:Nat
          Hole(a) /\ Hole(b) /\ c in Nat => design_Distance(a,b,c)
          foralla,b:Thing;c,d:Nat
          .design_Distance(a,b,c) /\ standard_Distance(a,b,d) /\ (d < c) => proper_Distance(a,b,True)
```

Figure 35: Partial View of `My_Checker` Spec

Then %implies

forall a,b: Thing; c,d: Nat

proper_distance(hole_u1, hole_u2, True) %(true-proof)%

Figure 36: Instantiated Proof

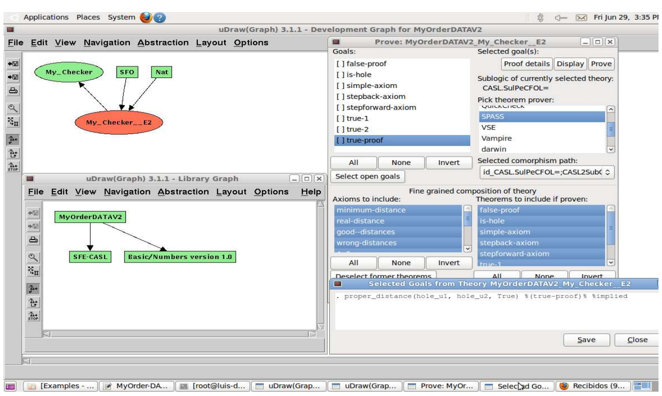

Figure 37: `My_Checker` HETS view

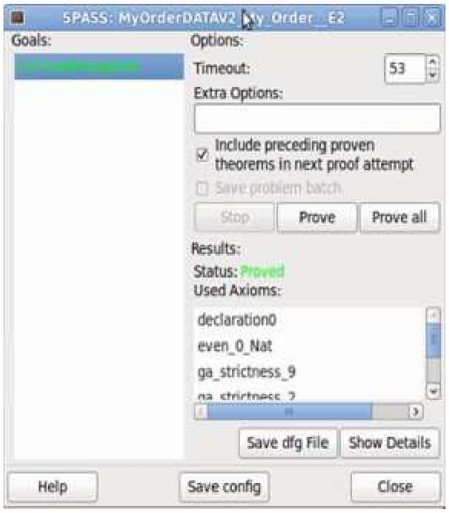

Figure 38: Proof of Sample provided

After loading the specification in HETS we obtained the windows shown in Fig. 37. On the left side, at the top and bottom, the development graphic is represented. There, nodes named Nat, SFO and `My_Checker` correspond to given specifications; such nodes are shown in green color in HETS. More specifically, Nat comes from the library of Numbers (CASL), SFO comes from the SFO (OWL) and `My_Checker` imports both (SFO and Nat). The node `My_Checker_E2`, shown in red by HETS, represents proof obligations. On the right side of the same figure, in the top view we can observe the axiom to be proved representing the proof obligation. Finally, in the middle of the window, there is a list of all axioms present in the specification and the available theorem provers. They are identified as "Axioms to Include" and "Theorems to Include if Proven".

To complete our task, the theorem prover SPASS [30] was run in proof node to assure the correctness of our instantiation. Fig.38 depicts the result, which was proved in this opportunity, confirming that our design complies with the technical requirements.

This then overcomes the reasoning OWL restriction, as demonstrated by implementing the architecture proposed in Fig.34.

We have to follow up with the final implementation of the products obtained from the systematic application of the methodology proposed in Fig. 1. But in fact, this implementation was carried out progressively from the modularization stage onward. This criterion was captured in our methodology as an `Implementation` arrow that starts in `Modularization` and ends in `Return_Disp`. In other words, since the very beginning a list of competency questions were defined in Subsection 4.3, and since then we followed a course of action in order to provide answer to such questions.

Throughout this Section we captured the implementation of the methodology we proposed in Onto*Smart*. Our main motivation for this proposal was the ongoing discussion in ontological engineering which includes topics such as reusability, ULO, modularity and heterogeneity. We proposed some specific metrics and methods to serve as criterion in a development workflow. A group of ontologies related with the manufacturing domain was selected because in this domain the manufacturing community has shown interest in Ontological Engineering. These ontologies passed an evaluation procedure, based in information quality and Competency Questions. Given that no ontology provided answers to all competency questions, we tried to make reusable the knowledge encoded in these ontologies. Our first attempt was using one of the ontologies as an interoperability artifact, however according to this proposed metrics and the obtained results, no upper level ontology was found. This result is worth mentioning because some authors have stated that their ontologies were of upper level, and for us these statements can be questions. Due to these results it was necessary to continue with our methodological workflow, in order to fulfill our requirement. Whereby some hyper-modules were extracted from the network ontologies under study. The resulting hyper-modules were integrated in an Hyperontology shown in Fig.25. Later, during the implementation phase, the OWL restrictions as ontology language were evidenced to represent the proposed scenario, therefore it was necessary to proceed with the heterogeneous activities proposed in our methodology. The inclusion of an heterogeneous layer made possible to complete the workflow, and check the restriction in the product shown in Fig.34. With this result we consider to have complete the methodological workflow we proposed.

In next Section we present our conclusion and discussion future work.

[23]https://github.com/luisenriqueramos1977/OntoSmart/wiki/Heterogeneity

8 Discussion of this Report

From the very beginning of this research, we have searched for an implementation approach, which means that we made efforts to validate by implementation the conclusions made in this Section. We now go through each of these, discussing our findings.

OWL is not expressive enough for representing knowledge, rules and restrictions of the manufacturing domain:

We initiated our research with the assumption that OWL would be enough for our general purpose. Very early in the research process it became clear that, although OWL provides a suitable level of decidability, in our practical application of design validation, conclusions from certain kinds of constraints would not be possible. This happened in the given examples because OWL does not support ternary relations, limiting the expressiveness of this language. In a more complex scenario, which could include parthood relations (e.g. assemblies), and heterogeneous metric systems, OWL could therefore become useless. Moreover, it is necessary to highlight that OWL is frequently mentioned as the chosen language to develop manufacturing ontologies. But in most research we found that no statement on OWL limitations was provided. Therefore we can also conclude that this very important flaw has been largely neglected. According to our results this may happen because in most cases the proposed ontologies were only lightweight ontologies, lacking the axiom definitions required by heavyweight ontologies. From our point of view, given that most researchers only developed ontologies at the lightweight level, the reasoning ability of OWL was not considered, consequently the flaw of this language was not evidenced in those studies. However, there are enough studies that refer to the limitation of OWL for knowledge representation, even in products and processes.

Hyperontology, within ULO and Modularity:

In most of the studies reviewed, we also realized a lack of proper methodologies for ontology development. Some authors highlight their proposed ontologies as ULOs, and as we have stated in the previous Sections, this is a result of a misconception. Concerning modularity, or a modular framework, we considered the early work of TOVE and SWOP. However, most other research has made no mention of this method. Therefore we could state that they consider an individual ontology sufficient to reach their goals. In our case, both approaches were studied, intending to make use of existing ontologies. Then a methodology was proposed which included an algorithm that allowed us to generate a new type of module called hyper-modules. This group of modules constitutes what we have now called a Hyperontology. For us, this is an empirical result in the sense that we did not expect to find a consistent structure of modules or Hyperontology in the ontology network that we evaluated. However, this result was found on our proposal based on a clear algorithm that could be validated in relation to other sets of ontologies selected from a consistent domain. We would expect that similar results will be obtained in other domains.

Heterogeneity, as a fact:

As mentioned in the first part of this Section, from the very beginning we were interested in making the most intensive use of the reasoning power of OWL within the Semantic Web framework. This intention led us to discover the flaws of this approach at an early stage as well. Therefore, we had to perform a deeper search into logics, languages, and related software tools. We found that HETS and its support of CASL provided an initial and basic support to extend the OWL language and its logics in relation to other logics, which would permit the achievement of more expressiveness. We cannot omit the fact that, when increasing expressiveness, we lose decidability, and so when working in cases related to manufacturing it is necessary to carefully evaluate each use in order to decide if the use case is restricted enough so that moving onto another logic can be avoided. For instance, if we consider heterogeneous metric systems, it is most probable that we have a situation where we will need more expressive languages and logics than OWL. However, if we decide to use only one metric system, then we can avoid this scenario.

Manufacturing processes are more than prismatic parts and shape fabrication:

Another point worth mentioning is that from our review of the literature, we found that most studies had centered their attention on prismatic parts and the shape fabrication procedure. However, manufacturing has other processes such as assembling, disassembling, forming, folding, and chemical procedures, among others, which to date have not been properly studied. Only one of the studies found mentioned assembly and disassembly [28], whereas others did not make mention of this. Our research, as most other research mentioned here, was centered on prismatic parts, and its intention was not to demonstrate omissions on the part of other studies – nevertheless, we considered it necessary to mention in these conclusions with the intention of identify open discussions for future studies.

Methodological Integration is the key:

Besides the limitations of OWL and its logics, OWL has considerable advantages, so it should not be left aside. On the contrary, we have integrated it into our proposed methodological framework. This methodological framework additionally includes methods for generating sets of hyper-modules constituting our hyper-ontology. This structure intends to use the knowledge inferred from one source ontology as the asserted knowledge of a target ontology. Furthermore, given that reasoning is a highly demanding task in Semantic Manufacturing, it is possible that OWL expressiveness may prove insufficient, and so a heterogeneous branch was included in our methodology in order to provide enough expressiveness to our framework when needed. The result of this integration was, first, a consistent Hyperontology, or network of modules, that linked most of the selected ontologies. Most of the concepts obtained by the proposed methodology were mentioned previously by other authors, moreover there were some other concepts that we also considered significant for the manufacturing domain. Second, we also validated our methodology with the use case of providing a basic CAD design quality checker.

9 Opportunities for Future Research

We consider that future research originating from our study can be performed in two ways: first, given that we have observed that the modeling pursued within the community of Semantic Manufacturing has omitted most of the manufacturing process, we recommend to move on to the production process of prismatic parts. This means that other processes, such as disaggregation, parts assembly, chemical, services, among others, should be included. On the other hand, from the Ontological Engineering point of view, the metrics and algorithm proposed by us for modules should be tested in other scenarios in order to determine if similar results can be obtained. Ontological Engineers should take special attention to the heterogeneity framework presented in this research, its CASL language and HETS tool, because the shortcoming of OWL can appear in other domains other than manufacturing. Ontologists should therefore be prepared to make use of this framework when necessary.

Finally, more complex products and processes than the examples provided in Sections 6 and 7 have to be studied. It is important to consider that from our Hyperontology depicted in Fig. 25, only one hyper module out of 12, and one ontology out of 8 were considered for this example. This represents less than 10 percent of the whole network. Thus, more complex scenarios need to be addressed in order to explore and evaluate the inclusion of more ontologies.

Acronyms

CAD Computer Aided Design. 9, 11

CASL Common Algebraic Specification Language. 11

CQ Competency Questions. 10, 17

DXF Drawing Exchange Format. 11

FOL First Order Logic. 13

IGES Initial Graphics Exchange Specification. 11

KIF Knowledge Interchange Format. 11, 13

OWL Web Ontology Language. 4, 11, 13, 15, 17

RDF Resource Description Framework. 11

RDFS Resource Description Framework Schema. 11

SQWRL Semantic Query-enhanced Web Rule language. 17

STEP Standard for the Exchange of Product model data. 11

SWOP Semantic Web Open Engineering Platform. 11

TOVE Toronto Virtual Enterprise. 11

Bibliography

[1] Samer Abdul-Ghafour, Parisa Ghodous, Behzad Shariat, and Eliane Perna. A Common Design-Features Ontology for Product Data Semantics Interoperability. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence*, pages 443–446. IEEE Computer Society, 2007. Available at http://portal.acm.org/citation.cfm?id=1331888&dl=GUIDE&coll=GUIDE&CFID=86829175&CFTOKEN=89177860.

[2] Odd Andersen and George Vasilakis. Building an ontology of cad model information. In Geir Hasle, Knut-Andreas Lie, and Ewald Quak, editors, *Geometric Modelling, Numerical Simulation, and Optimization*, pages 11–40. Springer Berlin Heidelberg, 2007. Available at http://www.springerlink.com/content/qx4r21362gu59632/abstract/.

[3] Autodesk. Dxf reference. Available at:http://images.autodesk.com/adsk/files/\acad_dxf1.pdf, 2009.

[4] Jhon Bateman, Alexander Castro, Immanuel Normann, Omar Pera, Leyla Garcia, and Jose-Maria Villaveces. OASIS common hyper-ontological framework (COF), January 2009. Available at:http://www.oasis-project.eu/docs/OFFICIAL_DELIVERABLES/SP1/D1.2.1/OASIS-D121.pdf.

[5] Stefano Borgo and Paulo Leitão. The role of foundational ontologies in manufacturing domain applications. *LNCS*, 3290/2004:670–688, 2004. Available at http://www.springerlink.com/content/rg5le0daa3vpd5w0/.

[6] Ibrahim Cayiroglu. A new method for machining feature extracting of objects using 2d technical drawings. *Comput. Aided Des.*, 41(12):1008–1019, 2009. Available at http://portal.acm.org/citation.cfm?id=1663670.1664169.

[7] Guk-Heon Choi, Duhwan Mun, and Soonhung Han Han. Exchange of cad part models based on the macro-parametric approach. *International Journal of CAD/CAM*, 2(1), December 2002. Available at:http://www.ijcc.org/ojs/index.php/ijcc/article/view/8.

[8] Mathieu D'Aquin, Anne Schlicht, Heiner Stuckenschmidt, and Marta Sabou. Criteria and evaluation for ontology modularization techniques. In Heiner Stuckenschmidt, Christine Parent, and Stefano Spaccapietra, editors, *Modular Ontologies*, pages 67–89. Springer-Verlag, Berlin, Heidelberg, 2009. Available at http://dx.doi.org/10.1007/978-3-642-01907-4_4.

[9] Silvio Ghilardi, Carsten Lutz, and Frank Wolter. Did I Damage my Ontology? A Case for Conservative Extensions in Description Logics. In *In Proceedings. OF KR2006*, pages 187–197. AAAI Press, 2006.

[10] Asunción Gómez-Pérez, Mariano Fernández-López, and Oscar Corcho. *Ontological Engineering: With Examples from the Areas of Knowledge Management, e-Commerce and the Semantic Web. (Advanced Information and Knowledge Processing).* Springer-Verlag, Berlin, Heidelberg, 2007.

[11] Bernardo Cuenca Grau, Ian Horrocks, Boris Motik, Bijan Parsia, Peter Patel-Schneider, and Ulrike Sattler. OWL 2: The next step for OWL. *Web Semantics: Science, Services and Agents on the World*

Wide Web, 6(4):309–322, November 2008. Available at http://www.sciencedirect.com/science/article/B758F-4TP1FC8-1/2/9d2f647c7ac874b8f8baa9cf92cf73a3, doi = 10.1016/j.websem.2008.05.001.

[12] Martin Hepp. References goodrelations wiki. http://wiki.goodrelations-vocabulary.org/References, 2013.

[13] S. Kumar and R. Singh. Trends and Developments in Intelligent Computer Aided Design of Progressive Dies. *Advanced Materials Research*, 6-8:241–248, 2005. Available at http://www.scientific.net/AMR.6-8.241.

[14] Oliver Kutz, Till Mossakowski, and Dominik Lücke. *Carnap, Goguen, and the Hyperontologies Logical Pluralism and Heterogeneous Structuring in Ontology Design.* 2010.

[15] Oliver Kutz, Immanuel Normann, Till Mossakowski, and Dirk Walther. Chinese whispers and connected alignments. In Pavel Shvaiko and Isabel Cruz, editors, *Proceedings of the 5th International Workshop on Ontology Matching*. ceur-ws.org, 2010.

[16] Jose L. Martinez Lastra and Ivan M. Delamer. Ontologies for Production Automation. In *Advances in Web Semantics I: Ontologies, Web Services and Applied Semantic Web*, pages 276–289. Springer-Verlag, 2009. Available at http://portal.acm.org/citation.cfm?id=1505694.

[17] 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*, pages 195–200, 2006. Available at 10.1109/DIS.2006.48.

[18] N. Manouselis, M.A. Sicilia, and D. Rodríguez. Exploring ontology metrics in the biomedical domain. *Procedia Computer Science*, 1(1):2319–2328, May 2010. Available at http://www.sciencedirect.com/science/article/pii/S1877050910002619.

[19] Patrick Martin and Alain D'Acunto. Design of a production system: an application of integration product-process. *International Journal of Computer Integrated Manufacturing*, 16(7):509, 2003. Available at http://www.informaworld.com/10.1080/0951192031000115831.

[20] Natalya F. Noy and Mark A. Musen. The prompt suite: interactive tools for ontology merging and mapping. *International Journal of Human-Computer Studies*, 59(6):983–1024, 2003. Available at http://www.sciencedirect.com/science/article/B6WGR-49S75R9-4/2/3fce33d1e0c00d144a96fcbeb54266ce.

[21] Raj Radhakrishnan, Araya Amsalu, Mehran Kamran, and B. O. Nnaji. Design rule checker for sheet metal components using medial axis transformation and geometric reasoning. *Journal of Manufacturing Systems*, 15(3), 1996. Available at http://www.sciencedirect.com/science/article/B6VJD-3WCSYR5-4/2/d9445ad0e020aaf6eddebabf222d77dd.

[22] L Ramos. Ontological cad data interoperability framework. Florence, Italy, 2010. Available at http://www.iaria.org/conferences2010/ProgramSEMAPRO10.html.

[23] Luis Ramos. Hetereogeneous Logics for Semantic Manufacturing. In *OSEMA 2012 Ontology and Semantic Web for Manufacturing*, volume 886, page 73, Graz, Austria, July 2012. CEUR. Available at http://ceur-ws.org/Vol-886/.

[24] Luis Ramos, Alexander García, and John Bateman. Ontology-Based Features Recognition and Design Rules Checker System. Bonn, Germany, 2011. Available at http://iswc2011.semanticweb.org/fileadmin/iswc/Papers/Workshops/OCAS/99990048.pdf.

[25] Gabriele Randelli and Daniele Nardi. Introducing Ontology Best Practices and Design Patterns into Robotics: USAREnv. In *Proceedings of the 2010 Conference on Modular Ontologies: Proceedings of the Fourth International Workshop (WoMO 2010)*, pages 67–80, Amsterdam, The Netherlands, The Netherlands, 2010. IOS Press. Available at http://dl.acm.org/citation.cfm?id=1804760.1804768.

[26] J.J. Shah and M.T. Rogers. Functional requirements and conceptual design of the feature-based modelling system. *Computer-Aided Engineering Journal*, 5(1), 1988.

[27] Vassilis Spiliopoulos, George A. Vouros, and Vangelis Karkaletsis. On the discovery of subsumption relations for the alignment of ontologies. *Web Semantics: Science, Services and Agents on the World Wide Web*, 8:69–88, March 2010. Available at http://www.sciencedirect.com/science/article/B758F-4Y52R2D-2/2/2ac5cd1dde9112fa9da924f714d939e4.

[28] Marcela Vegetti, Horacio Leone, and Gabriela Henning. Pronto: An ontology for comprehensive and consistent representation of product information. *Engineering Applications of Artificial Intelligence*, 24(8):1305–1327, 2011. Available at http://www.sciencedirect.com/science/article/pii/S0952197611000388.

[29] W3C. Product Modelling using Semantic Web Technologies, 2009. Available at:http://www.w3.org/2005/Incubator/w3pm/XGR-w3pm-20091008/.

[30] Christoph Weidenbach. Chapter 27 - Combining Superposition, Sorts and Splitting. In *Handbook of Automated Reasoning*, pages 1965–2013. North-Holland, Amsterdam, 2001. Available at http://www.sciencedirect.com/science/article/pii/B9780444508133500291.

[31] Zuoshuang Xiang, Mélanie Courtot, Ryan Brinkman, Alan Ruttenberg, and Yongqun He. OntoFox: web-based support for ontology reuse. *BMC Research Notes*, 3(1):175, June 2010. Available at http://www.biomedcentral.com/1756-0500/3/175/abstract.