View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Loughborough University Institutional Repository

Verification of Knowledge Shared across Design and Manufacture Using a Foundation Ontology

Najam Anjum, Jenny Harding, Bob Young, Keith Case, Zahid Usman and Tish Changoora

Wolfson School of Mechanical and Manufacturing Engineering, Loughborough University
Loughborough, LE11 3TU

Abstract

Seamless computer-based knowledge sharing between departments of a manufacturing enterprise is useful in preventing unnecessary design revisions. A lack of interoperability between independently developed knowledge bases, however, is a major impediment in the development of a seamless knowledge sharing system. Interoperability, being an ability to overcome semantic and syntactic differences during computer-based knowledge sharing can be enhanced through the use of foundation ontologies. Foundation or core ontologies can be used to overcome differences existing in more specialized ontologies and to ensure a seamless sharing of knowledge. This is because these ontologies provide a common grounding for domain ontologies to be used by different functions or departments. This common bases can be used by mediation and knowledge verification systems to authenticate the meaning of knowledge understood across different domains. For this reason, this research proposes a knowledge verification framework for developing a system capable of verifying knowledge between those domain ontologies which are developed out of a common core or foundation ontology. This framework makes use of ontology logic to standardize the way concepts from a foundation and core-concepts ontology are used in domain ontologies and then by using the same principles the knowledge being shared is verified.

Keywords: Foundation ontologies, domain ontologies, knowledge verification, ontology mediation, Common Logic, Manufacturability analysis

1. Introduction

The European Interoperability Framework (EIF) defines interoperability as “the ability of information and communication technology (ICT) systems and of the business processes they support to exchange data and to enable the sharing of information and knowledge” (European Communities, 2004). Networked businesses today encounter recurring difficulties due to limitations in interoperability between enterprise systems (Panetto and Molina, 2008). Similar interoperability problems occur when departments within a manufacturing enterprise share information and knowledge among themselves through ICT-based knowledge management systems. Such systems assist in organizing information and management, but their ability to represent and share manufacturing knowledge is very limited (Young et al, 2010).

The necessary functionality of interoperable systems for 'enabling the sharing of information and knowledge' requires the knowledge management systems to overcome several types of incompatibilities and heterogeneities between sets of knowledge residing in independently developed computer-based knowledge management systems. These differences occur because engineers working in different parts of the organization or different groups, with time, develop their own vocabulary for particular issues, elements or activities and this results in different information models (Lin and Harding, 2007). To overcome these differences, ontologies are used. Ontologies, provide the basic structure or armature around which knowledge bases can be built (Devedzic, 2002). The use of ontologies does not completely alleviate the problem of semantic heterogeneity but it helps in designing systems which can do so. Knowledge bases are therefore built around ontologies and in the event of knowledge sharing, these ontologies first interact to reach an agreement on the meaning of the terms used in the knowledge bases. Once that agreement is reached, the knowledge is shared seamlessly. Reaching an agreement over the meaning of a term, however, is a contentious issue. In the field of knowledge management the resolution of this issue is known as knowledge verification and it is this area of knowledge management where this paper makes a contribution.

The research explained in this paper proposes a unique framework for matching two domain ontologies for the purpose of knowledge verification. The proposed framework is tested on domain ontologies of manufacturing production and design concepts specialized from a manufacturing foundation ontology. This paper is arranged as follows: the way in which the concept of ontology is used in this paper is discussed first, and this is followed by an explanation of the research area of knowledge verification through ontologies. Next a brief review of existing related work is presented preceding the explanation of the proposed knowledge verification framework along with an implementation scenario. Finally, conclusions are provided and possible further research is discussed.

2. Literature Review

The way the concepts of ontology and knowledge verification are used in this research is briefly defined here while a detailed discussion is presented on foundation ontology based matching of domain ontologies, which is the main focus of this paper.

2.1. Ontologies

Ontologies are most frequently quoted as an explicit and formal specification of a shared conceptualization' (Studer et al, 1998).The intent of thissection, however, is not to discuss the concept of ontology but to explicitly state this concept as it is understood and used in the presented research. Ontology in this sense is considered, as defined by Gruninger, a hierarchical arrangement of terminologies aimed at defining concepts, their mutual relationships and constraints on those relationships(Gruninger et al, 2000).

This hierarchical arrangement of concepts is used in this research to verify knowledge shared across diverse domains. For this purpose,ontologies carrying very general concepts are used as mediators. These ontologies are commonly known as foundation ontologies. The significance of foundation ontologies with respect to knowledge sharing and verification is discussed next.

2.2. Foundation ontologies

Ontologies, being the explicit and formal specification of a conceptualization, provide a good platform for building shareable and interoperable knowledge repositories. They not only provide a way to preserve knowledge but also enable one to produce pre-packaged sets of information and knowledge available for individual use or for constructing large knowledge sets by using them as building blocks (Neches et al, 1991). Therefore, if these building blocks are made available in the form of a shared or foundation ontology, they can assist ontology builders to construct their own ontologies. Researchers agree that to make knowledge bases more shareable and expandable, instead of building them from scratch, it is more appropriate to develop them out of a single agreed upon foundation or standard (Neches et al, 1991). Foundation ontologies, as their name suggests, provide the basis for this standard. They 'describe very general concepts and provides general notions under which all root terms in existing ontologies should be linked' (Gomez-Perez et al, 2004).These ontologies make the expansion and integration of knowledge bases easier. This is because if two system builders build their knowledge bases on a common ontology, the system will share a common structure, and it will be easier to subsequently merge and share the knowledge bases (Swartout et al, 1997).

Some of the most famous foundation ontologies include Standard Upper Ontology – SUO (Niles and Pease, 2001), Suggested Upper Merged Ontology – SUMO (Niles and Pease, 2001), WordNet(Deng et al, 2009), DOLCE (Gangemi et al, 2002), and Cyc Ontology (Matuszek et al, 2006). Foundation ontologies like these may help to reduce semantic heterogeneity by restricting domain ontology builders to match their own conceptualisations against a common foundation, so that all communication is done according to the constraints derived from the ontology (Schorlemmer and Kalfoglou, 2005). These constraints, in a way, serve as a means of binding domain ontology builders to an ontological commitment. Ontological commitment is the process in which interested parties agree on the use of terminologies in an ontology. It helps in defining precisely the meaning of a term (Gomez-Perez et al, 2004) and thus helps in sharing knowledge accurately with minimal misinterpretation. It is this agreement, in the form of constraints, which is the basis of the mediation approach proposed in this paper. In the following sections, the term Core Concepts Ontology is also used which is comparatively a more general form of foundation ontology.

Along with foundation ontologies, domain ontologies are also an important part of this research and therefore are briefly looked at next.

2.3. Domain ontologies

Domain ontologies 'provide vocabularies about concepts within a domain and their relationships, about the activities taking place in that domain, and about the theories and elementary principles governing that domain'(Gomez-Perez et al, 2004). In the presented research, two domain ontologies belonging to the domains of Engineering Design and Manufacturing are featured and details of these ontologies can be found in section 5.1.2.These two ontologies are assumed to have heterogeneities among them and these heterogeneities are shown to be resolved through the proposed knowledge verification framework. The sense in which knowledge verification is used in this research is explained next.

2.4. Knowledge Verification

When formalized knowledge is shared between different domains, prevention of its subjective interpretation becomes necessary. This process of authentication of the interpretation, here, is referred to as knowledge verification. The description which endorses the sense in which the term verification is used here is the one given by Gupta (1993) where he mentions that knowledge verification involves the checking of completeness, consistency and correctness of knowledge. For this to happen during the cross domain knowledge sharing between ontology-based knowledge bases, ontology heterogeneities need to be overcome first. Semantic and syntactic heterogeneity was defined in the introduction section. Verification of knowledge here is, therefore, defined as the process of ensuring the correct understanding of knowledge built out of concepts taken from ontologies carrying semantic and syntactic heterogeneities. Two main steps involved in this kind of knowledge verification are:

1. The overcoming of mismatches existing in heterogeneous ontologies and
2. Correcting the knowledge built out of these ontologies.

Overcoming of mismatches is usually studied under the topics of ontology matching, alignment and mapping(Klein, 2001;Noy, 2004;Anjum et al, 2010). It is evident from the existing ontology research that ontology matching mainly involves the establishment of similarities between the ontologies to be matched. This is usually done in one of the following two ways. In the first case a general upper ontology (also called a foundation ontology) is agreed upon by ontology developers, who then extend this general ontology with concepts specific to their field. The extensions of this foundation can be called domain ontologies if the concepts they contain are more specific to a particular domain of interest. Finding correspondence between these domain ontologies is easier if their development is performed in a way consistent with the definition in the upper or foundation ontology (Noy, 2004).The other method of finding similarities involves the analysis of various characteristics of the two ontologies. These characteristics may include the structure of the ontology, definitions of concepts, and instances of classes (Noy, 2004). Ontological concepts having similar characteristics are then declared similar.

The existing tools for ontology matching are mostly based on the second method. These tools, however, are limited in their automation and accuracy of matching results. As a result, a significant level of human intervention is required to successfully match ontologies which is extremely cumbersome and time consuming (Anjum et al, 2010). The foundation or upper ontologies, on the other hand, provide a basic platform for the ontology builders to commit to and this makes the process of ontology matching more automatic and accurate, which is the main assumption of this research.

Since the main focus of this research is on foundation ontology based ontology matching, a detailed review of this research area is presented next.

2.5. Foundation ontology based ontology matching

The communication between independently developed domain ontologies through a foundation ontology is done by first aligning domain ontologies with concepts in the foundation ontology and then based on these alignments, the similarities between the domain ontologies are established as shown in figure 1. Some examples of this use of foundation or upper ontologies can be seen in the literature. In allcases, independently developed heterogeneous ontologies are provided with a pathway to communicate with each other by using an upper ontology as a semantic bridge. For example, Changoora and Young (2010) propose a multilayer Semantic Manufacturing Interoperability Framework (SMIF) which contains a foundation ontology layer to give domain ontologies, aligned with it, a pathway to communicate seamlessly. This framework assumes that domain ontologies are formed by specializing concepts from the foundation ontology. An important feature of this proposed framework is the use of logic to retrieve mappings between the concepts in

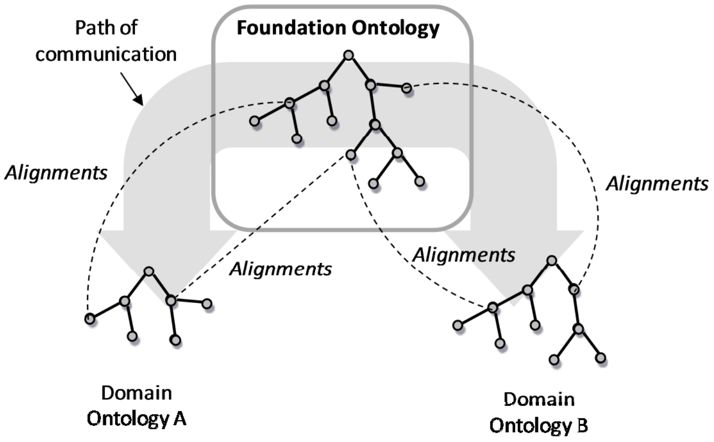


Figure 1: Ontology matching through a foundation ontology

domain and foundation ontologies. Another example can be found in the form of the Fishery Ontology Services (FOS) project (Gangemi et al, 2004). Here the authors developed a Core Ontology of Fishery (COF) in order to reengineer, align, refine and merge the fishery knowledge organization system. The COF is developed by specializing the DOLCE-Lite-Plus that is an extension of DOLCE foundation ontology containing ontologies of Descriptions and Situations. The part of their work that is relevant to this research comprises the mapping of domain ontology concepts with the COF. The core ontology in this way provides a bridge for the domain ontologies, or proto ontologies as the authors call it, to become semantically interoperable and thus communicate with the minimal chances of misinterpretation. In another work, Swartout et al (1997) use SENSUS for aligning some specific ontologies belonging to the domain of air campaign planning. SENSUS is a foundation ontology developed by extracting and merging existing electronic resources and contains very general terms (Swartout et al, 1997). The DOGMA (Developing Ontology-Guided Mediation for Agents) framework is another example where a centralized shared ontology is used to define more specific application ontologies to be used by certain applications (Jarrar and Meersman, 2009). In this way the centralized ontology provides a basis of shared meaning.Similar to the SMIF approach described above, the use of logic based assertions is demonstrated in this approach to definethe meaning of a term at different levels of ontologies. These logic based assertions are named as axiomatizations. The authors argue that a complete and exact meaning of vocabulary existing in the ontologies at levels below the foundation is impossible to define with the help of axiomatizations existing at the foundation level. This is because the use of a concept in different domains or at different levels of ontology may be different and a strict definition may restrict the use of that concept or vocabulary for different domain ontology builders, thus causing difficulties.Other examples of the use of WordNet, SUMO and MILO by Li (2004) in the Lexicon-based Ontology Mapping tool (LOM), the use of DICE ontology as background knowledge by Aleksovski et al (2006) and the experimentation of Mascardi et al (2008, 2010) with OpenCyc, SUMO-OWL and DOLCE for the provision of a confidential mediation system for businesses to share information and knowledge without compromising their business secrets.

2.6. A brief analysis of the existing work

The work related to the use of foundation ontologies, reviewed here, shows that usually a vocabulary is developed relevant to a specific domain to provide a shared platform for building ontologies at more detailed or specific levels. This shared vocabulary is given the name of Core ontology. It can be seen that this approach prevents any ontological mismatches occurring because the domain or application ontology builder provides a mapping during the ontology building stage. This is done by attaching concepts in the newly constructed ontology to a similar concept in the core ontology subsumed under a foundation or upper ontology. This attaching of domain concepts, however, may cause inconsistencies as different people may interpret, and thus use, concepts and terms differently. This inconsistency can be prevented by using logic-based axioms at the core and domain levels. The DOGMA framework provides a very useful insight into the way axioms have to be defined at different levels in an ontology-based knowledge sharing system. This is specifically relevant from the point of view of knowledge verification because a logic-based verification of shared knowledge comprises of a set of rules and axioms that need to be defined. It is also learnt from the review that care needs to be taken when defining axioms, as a too tight definition of a concept at the core or foundation level may hamper its use at the domain level. This can be prevented by allowing, to some extent, the domain ontology builders to define the use of a concept themselves when building ontologies.

The most important fact, from the point of view of this research that can be derived from the review of foundation based techniques is that the *selection of concepts from the core concepts ontology to build domain ontologies actually decides if the resulting knowledge is correct or not.*

For example, consider a case where a knowledge expression states a fact about the concept 'bank'. The word 'bank' may refer, either, to the bank of a river or a bank as a financial institution. It is important to check that the correct concept is selected from the core concepts ontology when stating facts in the knowledge base. The emphasis of knowledge verification techniques using upper ontologies, therefore, has to be on defining ways to prevent an incorrect use of concepts from the core concept ontology. Jarrar and Meersman (2008), in the above presented review, show that axiomatizations can be used for this purpose. For this reason, knowledge verification in foundation ontology based systems should comprise of a mechanism that specifies, with the help of axiomatizations, a standard way of specializing concepts from the core concepts or foundation ontology. A similarity finding technique then needs to be developed that uses the principles of that standard to discover correspondences between ontologies committed to the same core.

The following conclusions, therefore, can be drawn from the above review:

1- Foundation ontology based techniques resolve the issue of mismatches but introduce the problems of inconsistencies when concepts from the core concept ontology are chosen to build domain or application ontologies and associated knowledge bases.

2- A plausible solution for these inconsistencies appears to be the use of logical theories or axiomatizations at different levels of the foundation and core concepts ontology.

3- The design and development of a mechanism capable of finding similarities between two ontologies by using their inheritance in the foundation or core ontology is needed.

A description of such a mechanism cannot be found in the tools, techniques and frameworks reviewed in the previous section. Such a framework, therefore, is proposed in this research and a detailed description of this is provided in the next sections. This framework establishes similarities across two domain ontologies automatically by following concepts in domain or application ontologies to their origin in the core concept or foundation ontology. This framework is then tested and validated by using real industrial examples taken from a case study conducted with an aerospace engine manufacturer.

3. Findings of the case study

The theoretical findings presented in the last section needed some experimental evidence to prove their validity. Real time data was needed to perform experiments, and for that reason, a case study was conducted in an aerospace engine manufacturing plant. The findings are briefly presented here, but a detailed version of this case study can be found in Anjum et al (2012).

There were two main dimensions in this 12 weeks long case study. The first dimension dealt with the study of communication needs between design and production departments in order to produce a manufacturable design. As this research is all about computer-based knowledge sharing, the second dimension entailed the study of the criteria for modeling an engineering component in the form of an ontology to which knowledge can be attached for seamless sharing. The primary sources of data were individual one-to-one interviews but the company intranet and archival information also contributed to this 12 week case study. The study revealed the following findings:

1- The information flow between design and manufacture is mainly regarding the manufacturability of a component,

2- The Design and Manufacturing departments perceive engineering components differently depending upon whichaspects are important to them,

3- In the interpretation of different features of a component, some terminological differences also exist in addition to the perceptional differences across two departments,

These three requirements, obtained from the literature review shown in the previous sections, when viewed in the light of the findings of the industrial case study highlight the need for *a knowledge verification system, specializing in manufacturability analysis, capable of automatically resolving the semantic and syntactic inconsistencies resulting from the perceptional differences between design and manufacture through the use of logic in foundation ontology based knowledge bases.*

Some of the keywords above are underlined to emphasize the fact that the proposed solution has to *automatically resolve the perceptional differences through the use of logic*. How this is achieved by the proposed knowledge verification framework is explained next.

4. **Ontological formalism used in the presented research**

Before the description of the proposed verification framework is given it is important to briefly discuss the 'Knowledge Frame Language (KFL)' which is the ontological formalism used for performing the experimentation in this research. KFL is a syntactic layer sitting on top of an extension of Common Logic (CL) which is a family of logic-based languages standardized under ISO (ISO/IEC 24707:2007(E), 2007). This extension of CL is called ECLIF (Extended Common Logic Interchange Format)([Anonymous], 2010).It is to be noted that CL has been selectedto build ontologies in this researchinstead of OWL which is the most frequently used formalism for building and experimenting with ontologies. This selection is made because of the high expressiveness of common logic which is important to capture the complex relationships which exist in some manufacturing contexts. As compared to other formalisms which are usually restricted to the creation of binary relations, CL provides the user with the capability to define ternary (three places), quaternary (four places) and even quinary (five places) relations ([Anonymous], 2010). In addition to that CL also provides a highly powerful syntax for logical expressions. The scope of this paper does not allow a detailed description of this formalism. A relatively detailed discussion, however, can be found in Anjum et al (2012). In the sections to follow, the working of the proposed knowledge verification framework is explained with the help of examples written in KFL. Every possible effort has been made to remove the subjectivity of the coding for the readers not familiar with this formalism.

5. A novel knowledge verification framework

Keeping in view the requirements set by the above presented findings, a novel knowledge verification framework is proposed in this research. A schematic of this framework can be seen in figure 2. This verification framework proposes the use of logic not to overcome differences in domain ontologies during knowledge sharing but to prevent them from happening in the first place. This is done firstly at the domain ontology level i.e. during the process of domain ontology development and secondly at the knowledge base level i.e. during the process of knowledge base population. To materialize this idea into an understandable form, a verification framework is proposed to contain a set of ‘inconsistency preventing axiomatizations’ (shown as axioms in figure 2) and a ‘verification mediator’.

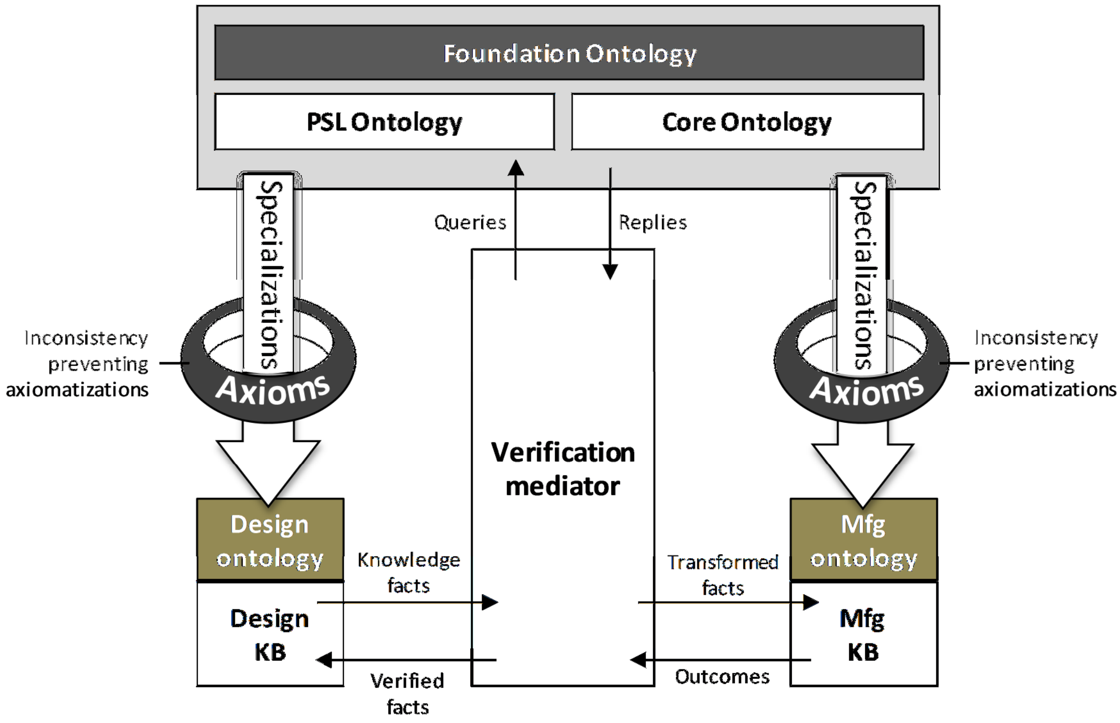


Figure 2. The verification framework

The proposed axiomatizations are aimed at standardizing the way concepts from the foundation and core-concepts ontology are specialized in the domain ontology. This standard way is needed to ensure that no inconsistencies are created during the process of concept specialization due to subjective interpretations of foundation and core concepts by the domain experts. These constraints are proposed to either exist in the foundation and core-concepts ontology or as a separate attachable ontology. The 'verification mediator', which is the second part of the proposed verification framework, is designed to detect similarities on the basis of the assumption that the specializations in the domain ontologies are created by following the 'standard way'.

The working of the verification mediator explained below involves some terminologies which need to be defined first. This is done below.

Fact: a fact is a sentence or an assertion which creates, relates or quantifies one or more ontological concept. For example the sentence '`(hasLength cylinder_1 (mm 150))`', is a fact that uses the ontological concepts of `hasLength`, `cylinder_1` and `mm` to state that a certain cylinder has a length of 150 mm.

Queries: these are the assertions written to investigate a certain ontological concept, relation, function or fact. For example the query '`(hasLength ?c (mm 150))`', enquires if there exist any components with length of 150 mm.

Replies: these are the data obtained in response to a query after the investigation of the ontology or knowledge base in question. For example, the query illustrated above may result in a list of components that exist in a knowledge base with a length of 150 mm.

Having explained some useful terminologies, the design and implementation details of the proposed framework are presented next.

5.1. Design of the verification framework

Figure 2 illustrates the setting in which this verification mechanism is designed to work. Six main components of this framework are:

1- Foundation and core-concepts ontologies,
2- Domain ontologies,
3- Knowledge bases,
4- A set of inconsistency preventing axiomatizations and
5- The verification mediator
6- The Java API

A detailed design of each of these components can be a research project in its own right. For that reason, the research presented in this paper only focuses on the last three components which specifically deal with the verification of knowledge shared between foundation ontology based domain ontologies. In the following text, a brief description is given of the first three components while a detailed design description of the last three components is presented.

5.1.1. Foundation and core-concepts ontologies

At the top of figure 2, two layers of ontologies can be seen. The first layer is the foundation ontology which then subsumes in the second layer - the PSL ontology and a core ontology of manufacturing concepts. The development of full scale core or foundation ontologies was beyond the scope of this research. However, this research was a part of a bigger project entitled Interoperable Manufacturing Knowledge System (IMKS) which does include the development of ontologies up to a certain level as a complementary research project (Usman, 2012). In order to test the proposed knowledge verification framework, small fragments of experimental ontologies were created which are illustrated in figure 6 in section 6.1.

The use of PSL ontology is also proposed here. PSL or process specification language is an ontology designed to aid semantically sound exchange of process information between different manufacturing setups (Gruninger, 2004). PSL is used here to provide concepts for defining processes. The manufacturing core-concepts ontology, on the other hand, provides concepts for defining products. Together, these two ontologies provide the vocabulary for building domain ontologies through a controlled specialization of their concepts and subsequently the knowledge base through controlled modeling of design and manufacturing processes. Although the whole upper block of ontologies may be referred to as a foundation, different sections shown help in describing the way knowledge can be created by using the available concepts in this block.

5.1.2. Domain ontologies

The foundation and core-concepts ontologies hold very general concepts. Which means, although these concepts can directly be used to build the knowledge base, only a very general level of modeling can be done without going into the low level details of manufacturing processes. For this reason, domain ontologies need to be built, with idiosyncratic and specialized concepts, before knowledge is modeled in the knowledge bases. These idiosyncrasies depend upon the preferences of a certain domain. Two domains studied during the case study were design and manufacture and therefore two domain ontologies belonging to these domains are included in the framework.

To explain the proposed framework, those examples from the case study are used where the domain ontologies are needed to cater for different perceptions and terminologies existing in different departments. In these cases, therefore, domain ontologies are used as a source of localized vocabulary to model knowledge in departmental knowledge bases. These domain ontologies, however, are assumed to have connections in the foundation and core-concepts ontology with concepts and terms of similar meaning. To make sure that these connections are built whilst building the domain ontology, it is proposed, that some axiomatizations and constraints should be used. These are called here the '*inconsistency preventing axiomatizations*' and will be explained later.

5.1.3. *Knowledge bases*

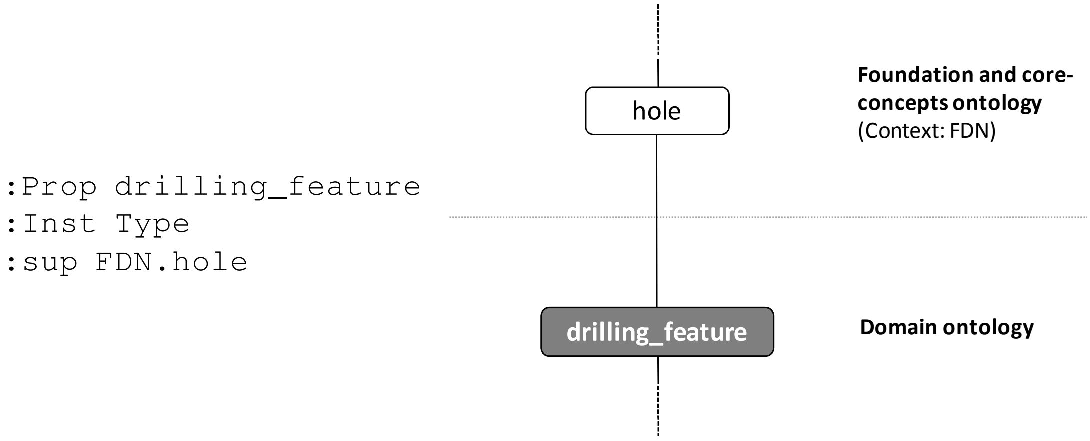
At the bottom of all the layers of foundation, core-concepts, and domain ontologies lie the knowledge bases as shown in figure 2. These knowledge bases are assumed to be built by using concepts from the domain ontologies of respective departments of an engineering firm. The two departments chosen in this research are design and manufacture. Since there are likely to be differences in the terminologies used by the design and manufacturing domain ontologies, the knowledge facts built under these ontologies may also be different. It is the work of the '*verification mediator*' then to identify similarities and verify knowledge. In the scenario considered, it is assumed that the design knowledge base contains the facts that model a component. It is proposed in this research that these design models of components are examined by the manufacturability constraints existing in the manufacturing domain ontology before they are finalized. This is done through the verification mediator as can be seen in figure 2. It is shown that, through the use of queries, the verification mediator verifies facts by first sending them to the manufacturing knowledge base to be checked by the manufacturability constraints existing in the manufacturing domain ontology. If no objections are raised by the manufacturing domain ontology the verification mediator sends the verified facts back to the design knowledge base for them to be finally asserted in the form of a product or feature model. In this way the knowledge base is populated with verified knowledge.

5.1.4. *Inconsistency preventing axiomatizations*

The design and testing of the framework proposed in this research has been based on possible inconsistencies identified through the industrial case study work reported earlier. Since the concepts from the foundation and core-concepts ontology are used first to build the domain ontology and then the knowledge base, two types of axioms need to be written. The first type is proposed to scrutinize the concept specialization in the domain ontology and the second type is aimed at examining the completeness of the knowledge facts in the knowledge base. These two types are explained below with the help of examples. The word 'axiom' is used, instead of axiomatization, from now on for the sake of brevity.

5.1.4.1. *Axioms for the domain ontologies*

The most vital issue for the foundation ontology based domain ontologies is the linking of concepts in the domain ontologies to similar concepts in the foundation and core-concepts ontology. Two types of cases may exist in this regard. In the first case, an independently developed domain ontology may need to be linked to a foundation by mapping similar concepts. In the second case, a domain ontology is built by using the concepts from the foundation ontology as building blocks. In both of these cases, there needs to be a standard way of linking together the concepts in the two ontologies such that the structure of the domain ontology is consistent with the foundation ontology. The compliance to this standard way can be ensured through axioms (or integrity constraints as they are called in KFL) present in the domain ontology. The existence of these axioms is critical for the functioning of the verification mediator which is the second main part of the verification framework proposed in this research. The reason being that the main assumption on which the verification mediator works is that the domain ontologies are consistent with the foundation and core-concepts ontologies in terms of their structure if not content. Hence, the method of subsuming the domain concepts under similar foundation or core concepts needs to be defined explicitly. As far as this research is concerned, the standard way of doing this is defined to be the super-concept relation. This is depicted in figure 3 where a concept named 'drilling_feature' in the domain ontology is shown to subsume a foundation or core concept of 'hole'. The KFL assertions needed for this subsumption are also shown. The 'sup' directive in these assertions is responsible for the establishment of a link between



```
:Prop drilling_feature
:Inst Type
:sup FDN.hole
```

Figure 3. Foundation concept subsumption

the domain and foundation concept. In this case it subsumes the domain concept under ‘FDN.hole’. The prefix FDN here represents the context in which concepts in the foundation and core are defined. The inclusion of this prefix shows that the class named ‘hole’ exists in the foundation and core-concepts ontology. The verification mediator works on the assumption that a ‘sup’ directive for every concept in the domain ontology exists. It is therefore very important that the inclusion of this directive is ensured. This can be done by writing the following axiom.

```
(=> (and (RootCtx.Property ?prop1)
                 (RootCtx.Context ?ctx1)
                 (RootCtx.contextFor ?prop1 ?ctx1)
                 (UserContext ?ctx1)
                 (not (= ?ctx1 MLO))
                 (not (= ?ctx1 MFG)))
     (exists (?prop2) (and (RootCtx.Property ?prop2)
                          (RootCtx.contextFor ?prop2 MFG)
                           (sup ?prop1 ?prop2))))
:IC hard "Every class in the domain ontology needs to be subsumed to
a class in the foundation and core-concepts ontology."
```

The above axiom says what is shown in the ‘IC hard’ directive. This axiom makes sure that no class in the domain ontology is left unidentified and thus provides a solid platform for the verification mediator to work from. This is, however, not the only type of inconsistency that may occur during domain ontology building and linking.

One other type of inconsistency can be the incompleteness of specialization. So if a concept **A** is to be specialized from a foundation or core which also requires the concepts of **B** and **C** to be specialized for its complete definition, then the incompleteness will occur when only **A** is specialized while **B** and **C** are not. A typical example in case of manufacturing can be the specialization of a geometrical shape feature, like a ‘hole’ or a ‘cylinder’ without the specialization of its dimensional characteristics, like diameter and length. A domain ontology, therefore, needs a check through axioms and constraints on the completeness of concepts specialized from the foundation and core-concepts ontology. Consider the segment of a core-concepts ontology with a domain specialization as shown in figure 3. It can be seen that the concept named ‘hole’ in the core-concepts ontology is specialized in the domain ontology with the name ‘drilling_feature’. For this specialization to be correct and complete, the foundation concept of diameter and depth also need to be specialized. To make sure that this happens during domain ontology building, the following axiom can be written.

```
(=> (and (RootCtx.Property ?p1)
                 (RootCtx.Context ?c)
                 (RootCtx.sup ?p1 MFG.hole)
                 (RootCtx.contextFor ?p1 ?c)
                 (not (= ?c FDN)))
(exists (?p2)(and (Property ?p2)
                 (sup ?p2 FDN.diameter) (contextFor ?p2 ?c))))
:IC hard "If there is a specialization of the 'hole' concept then a
specialization for the ‘diameter’ should also exist."
```

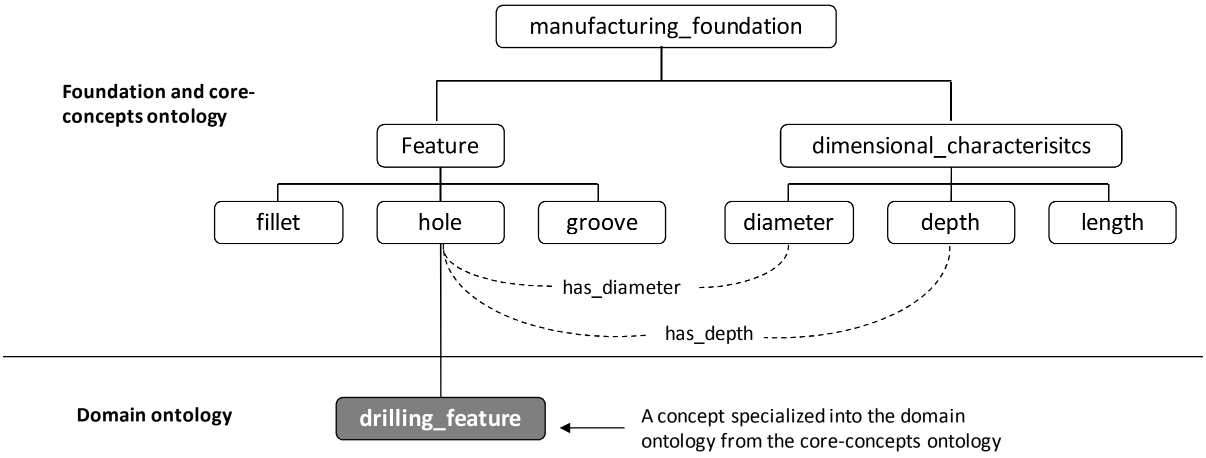
This axiom says that if there exists a class in the domain ontology and that class is a specialization of the foundation concept ‘hole’, then another class as a specialization of the foundation concept ‘diameter’ should also exist in the domain ontology.

The above example explains just a few of the many possible inconsistencies that may occur during the foundation or core concept specialization in the domain ontology. These examples, however, demonstrate the use of axioms to prevent inconsistencies, which was the main aim of their explanation. The methodology to prevent any known inconsistency, therefore, is the same. Axioms can specifically be written for a possible error and hence inconsistencies can be prevented.

5.1.4.2. Axioms for the knowledge base

When a domain ontology is constructed, the knowledge base is built though the population of concepts introduced in the domain ontology. There can, however, be a case where concepts from the foundation and core-concepts ontology are directly used to build a knowledge fact. To prevent inconsistencies in such a circumstance, axioms are proposed to exist. For example to make sure that all the essential dimensional parameters of a circular disc are defined in the knowledge base, the following axiom can be written.

```
(=> (disc ?x)
           (exists (?d ?h)
           (and (has_diameter ?x ?d)
                      (has_height ?x ?h))))
:IC hard "For a complete description of a 'circular disc' both
'diameter' and 'height' are needed."
```



manufacturing_foundation
Foundation and core-concepts ontology
Feature
dimensional_characterisitcs
fillet
hole
groove
diameter
depth
length
has_diameter
has_depth
Domain ontology
drilling_feature
A concept specialized into the domain ontology from the core-concepts ontology

Figure 4. A specialization example

This axiom says that if there exists a hole, then its diameter and depth should also exist and since it is a '*hard*' integrity constraint, it does not allow the user to proceed without correcting the error. Axioms like these can be written to prevent inconsistencies at the knowledge base level.

The axioms presented above specifically target certain inconsistencies, generally however, they attempt to maintain a certain structure of domain ontologies committing to the foundation. In such a case, the verification system needs to be designed according to the structure controlling constraints of the foundation and core-concepts ontology. The verification mediator explained in the next section, works on the assumption that the concepts in the domain ontologies are connected to similar concepts in the foundation or core through the '*sup*' directive as shown in figure 4. It is this link which is used by the verification mediator to establish similarity between independently developed domain ontologies. The working of this mediator is explained next.

5.1.5. The verification mediator

The verification mechanism described in this section is essentially an ontology mediation mechanism that uses the inheritance of domain ontology concepts in the foundation and core-concepts ontology to establish similarities. Keeping in mind the capabilities and shortcomings of existing ontology mediation tools, the new tool needs to have the capability to make the similarity detection process more automatic and accurate when two foundation ontology based domain ontologies are matched. Figure 5shows the working of this mechanism.

This verification mediator works with the help of four modules named (A) Source ontology inheritance identifier, (B) Concept matcher, (C) Target ontology inheritance identifier, and (D) Fact builder and asserter. The roles of these modules in ontology mediation and knowledge verification aredescribed in detail below.

5.1.5.1. Source ontology inheritance identifier

The inheritance identifier is responsible for finding the super-class of a domain ontology concept in the foundation and core-concepts ontology. This is done by sending queries to

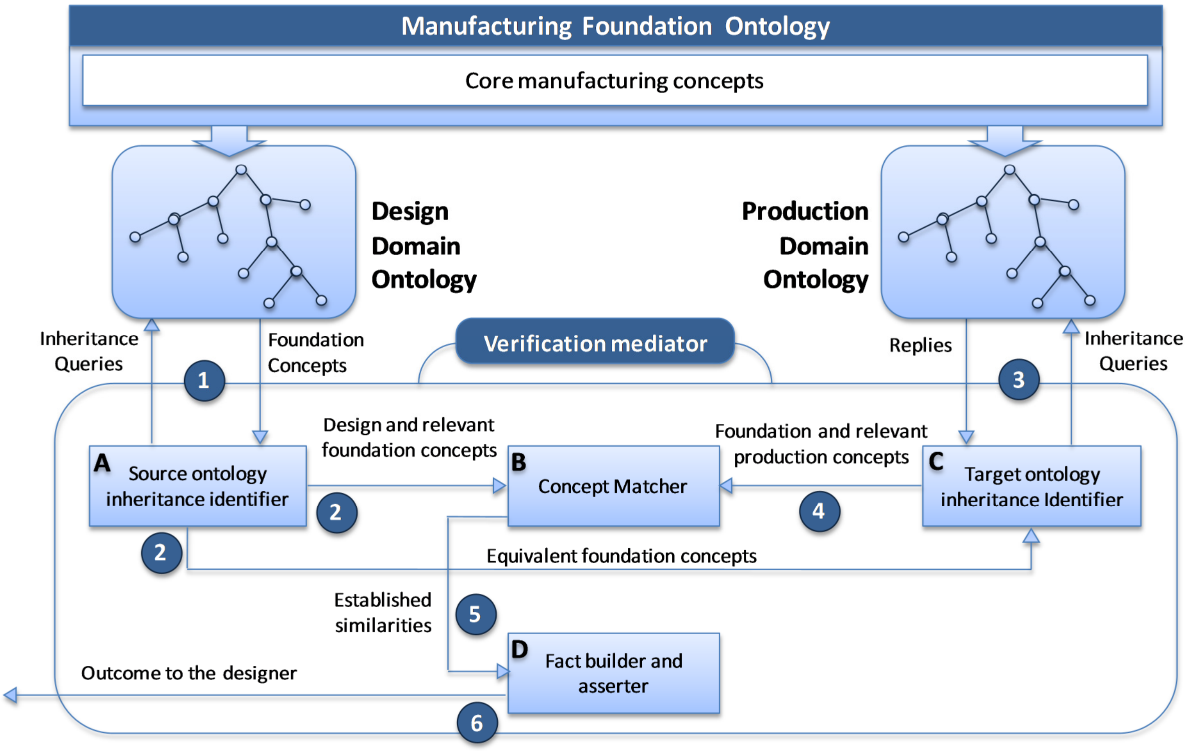

Manufacturing Foundation Ontology
Core manufacturing concepts
Design Domain Ontology
Production Domain Ontology
Inheritance Queries
Foundation Concepts
Verification mediator
Replies
Inheritance Queries
1
3
Design and relevant foundation concepts
Foundation and relevant production concepts
A Source ontology inheritance identifier
B Concept Matcher
C Target ontology inheritance Identifier
2
2
4
Equivalent foundation concepts
Established similarities
5
D Fact builder and asserter
Outcome to the designer
6

Figure 5. Working of the verification mediator

the foundation. It takes a concept from one domain ontology, writes a query to find its foundation inheritance, receives replies and sends the replies to the 'domain concept identifier and concept matcher' module.

5.1.5.2. Target ontology inheritance identifier

The functioning of this module is similar to the inheritance identifier. However, instead of finding the super-class of a domain concept, it searches for the sub-class of a concept in the foundation and core-concepts ontology. This is done by first locating the class names (received from the source ontology inheritance identifier) in the foundation and core-concepts ontology and then querying for their sub-classes in the target domain ontology.

5.1.5.3. The concept matcher

The concept matcher takes input from the inheritance identifier and domain concept identifier modules and determines their similarity by analyzing their foundation inheritances. The rule to follow here is that if the foundation inheritances of two classes in two domain ontologies are the same then these domain ontology classes are also the same. Once the similarity is established the similar class names are forwarded to the fact builder and asserter module.

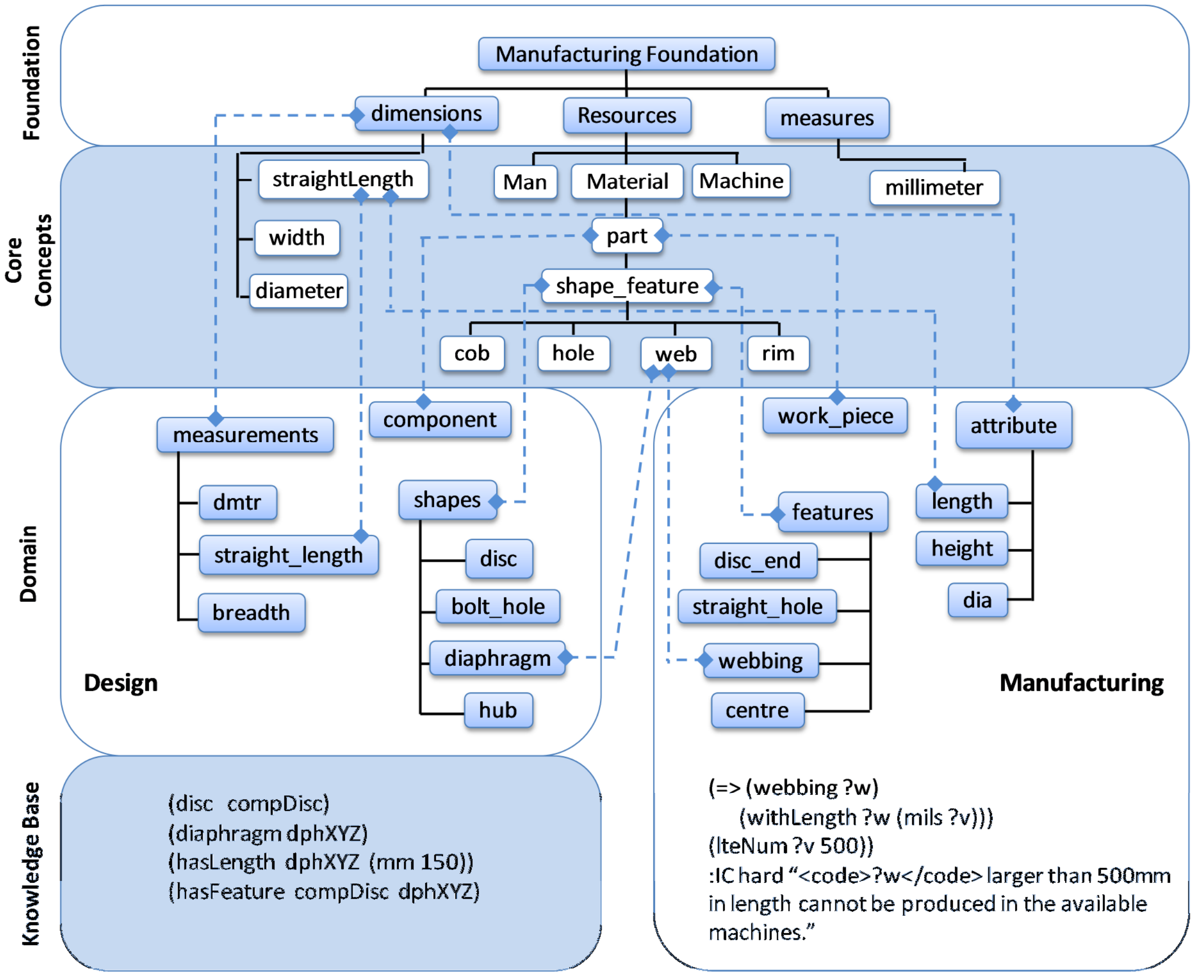
5.1.5.4. The fact builder and asserter

This module starts functioning once the similarities are established. It is, therefore, technically not a part of the ontology mediation system but does play a role in the verification of knowledge. The task of fact building involves writing a knowledge fact in an ontological formalism and then asserting it in the knowledge base built out of the domain ontology concepts.

The functioning of this module can be better understood when the functioning of the verification mediator is explained. This is done in the next section where, with the help of an industrial scenario, the implementation of the proposed verification framework is explained.

6. Implementation of the verification framework

The basic principle of the working of the verification mediator is based on the exploitation of the link between the domain concepts and their equivalent concepts in the foundation or



```
(disc  compDisc)
(diaphragm dphXYZ)
(hasLength  dphXYZ  (mm 150))
(hasFeature  compDisc  dphXYZ)
```

```
(=> (webbing ?w)
    (withLength ?w (mils ?v)))
(lteNum ?v 500))
:IC hard "<code>?w</code> larger than 500mm
in length cannot be produced in the available
machines."
```

Figure 6. A specialization and knowledge population example

core-concepts ontology. In the case of this research, this link is the inheritance of concepts in the foundation or core-concepts ontology. The functioning of the verification mediator is therefore now explained with the help of a real industrial example.

6.1. The industrial scenario explained

Figure 6 illustrates a scenario where differences exist in two domain ontologies committed to a common foundation and core ontology. This is a realistic scenario built out of the understanding developed during the case study explained in section 3 of this paper. Further details of this case study can be found in Anjum et al (2012). This scenario assumes that for the sake of knowledge transfer between the design and manufacturing departments of a company, domain ontologies are built out of a common foundation and core ontology bearing concepts needed to represent shape features which combine to form the component shown in figure 7. It was found during the case study that the manufacturing

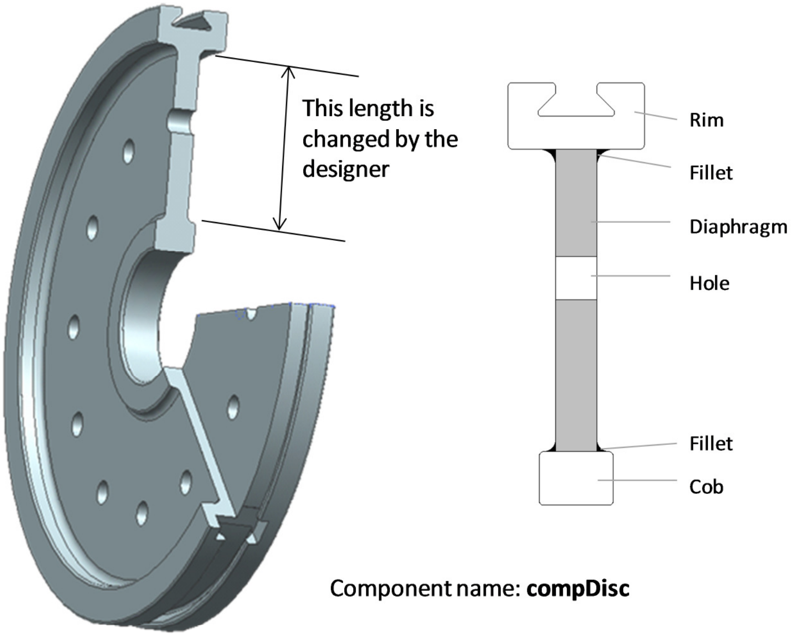

This length is changed by the designer
Rim
Fillet
Diaphragm
Hole
Fillet
Cob
Component name: **compDisc**

Figure 7. The example component

and design departments may have different understandings and thus different terminologies for same concepts. Some differences in the domain ontologies have, therefore, been introduced to represent this fact. It can be seen in figure 6 that there are four layers in this model of an ontology based knowledge sharing system. The first layer contains very general foundation concepts while the second layer holds core manufacturing concepts to be committed to by the concepts in the domain ontologies existing in the third layer. The fourth layer is that of the knowledge base which contains ontological models of the components formed by using shape features from the domain ontologies above. The manufacturability rules aimed at governing the creation and modification of these models lie in the manufacturing domain ontology. In figure 6, the dotted lines show that the terms in two domain ontologies are connected to their equivalents in the core-concepts ontology. It can be seen that the two domain ontologies use different terminologies to represent the same entity in the core-concepts ontology. This has implications for the facts asserted in the knowledge base. The knowledge facts in the design knowledge base, in such a situation, need to be translated into the manufacturing ontology language for the rules existing there to make sense. This is what the verification mediator does.

The implementation of the framework will now be demonstrated with the help of a formalized example. The ontological formalism used for the ontology formalization is KFL as explained earlier.

It is assumed that the design knowledge base contains the following facts:

```
(component compDisc)
(diaphragm dphXYZ)
(hasLength  dphXYZ  (mm 150))
(hasFeature compDisc dphXYZ)
```

These lines say that an instance of the class 'disc' exists with the name 'compDisc'. This instance has a feature named 'dphXYZ' which is an instance of the class 'diaphragm' and the length of 'dphXYZ' is '150mm'. In simple words, these facts say that there is a component which has a *diaphragm* feature with length 150mm. In an ideal case, these facts, before they are used to model the component in the design knowledge base, are to be inspected by the manufacturability rules existing in the manufacturing ontology. One such rule is shown in figure 6 which says that the length of the disc arm (which is *diaphragm* in the design domain ontology and *webbing* in the production domain ontology) should not exceed 500mm for it to be produced in the available machining facility. The rule is as follows:

```
(=> (webbing ?w)
      (withLength ?w (mils ?v)))
(lteNum ?v 500))
:IC hard "<code>?w</code> larger than 500mm in length cannot be
produced in the available machines."
```

In the start of the IC hard statement, it says '<code>?w</code>' which enables this statement to be modified according to the name of the feature represented by the variable ?w. Writing the IC hard statement in this way makes it understandable to the designer as the '<code>?w</code>' is replaced by the name of the feature used by the designer. This will be shown at the end of step 6 of the verification process. The coding of this rule, however, is purely in the manufacturing ontology language and is not comprehensible by the computer system in the design knowledge management system. This industrial scenario poses a problem that needs an ontology mediation mechanism capable of finding similarities across two ontologies. Following is the description of the verification mediator which is designed to do exactly this.

6.1.1. Six steps of verification mediation

As mentioned earlier, in order for the rule to understand the design component models, the facts used to build that model need to be translated into the manufacturing ontology language. This is precisely what the verification mediator is designed to do. It first translates the facts asserted in the design knowledge base into the language of the manufacturing domain ontology, it then asserts the translated facts into the manufacturing knowledge base. Since these facts are now in the manufacturing ontology language, all the manufacturability rules existing in the manufacturing ontology become valid and thus a response according to the validity of changes made by the designer is obtained. The knowledge facts written by the designer for modifying an existing model or for creating a new model are now asserted or denied depending upon the response obtained from the manufacturing ontology.

In order to translate the knowledge facts from the design into the manufacturing terminology, the verification mediator first needs to establish similarities between the concepts across two ontologies. A typical kfl fact consists of concepts, relations and functions (used here to define units of measurement), three types of similarities are to be established named:

1- Concept similarity,
2- Relation similarity and
3- Function similarity.

Once these similarities are established, the facts are translated from one form into another. Figure 5 shows the process of verification taking six steps to complete. These six steps are explained next by using the example of figure 6.

Step 1 – Inheritance queries and foundation concepts

In this step, the 'source ontology inheritance identifier' module of the verification mediator first decomposes the facts asserted by the designer into its elementary concepts. It then queries the foundation ontology for the inheritances of these concepts one by one. As mentioned earlier, three types of similarities have to be established i.e. the concept, relation and function similarity, the ‘source ontology inheritance identifier’ module divides these three components of the ontology into its constituent elements or concepts.

In the example of figure 6, the following design facts are asserted:

```
(componentcompDisc)
(diaphragmdphXYZ)
(hasLengthdphXYZ  (mm 150))
(hasFeaturecompDiscdphXYZ)
```

In step 1, these facts are decomposed as follows:

Concept 1: `component`

Concept 2: `diaphragm`

Relation 1: `hasLength`

Relation 2: `hasFeature`

Function 1: `mm`

These relations are further divided into the concepts they hold between.

Relation1: `hasFeature`

- Relation 1 - concept 1: `component`
- Relation 1 - concept 2: `shapes`

Relation2: `hasLength`

- Relation 2 - concept 1: `shapes`
- Relation 2 - concept 2: `straight_length`

In the same way, the functions are also divided into the entities they hold between.

Function 1: `mm`

- Concept measured: `straight_length`

Once the facts are decomposed, queries are then generated by the inheritance identifier to find the foundation inheritance of these concepts. One such query is shown here which results in the foundation inheritance of the concept ‘component’:

```
(and (sup DSN.component ?f) (contextFor ?f FDN))
```

This query says, ‘find that super-concept of DSN.component which has a context of FDN’. DSN here denotes the design ontology context while FDN is the context for the foundation ontology.

Similar queries written for all the concepts return the answers shown in table 1.

Table 1: Foundation inheritances of design concepts

Design Concept	Foundation Inheritance
component	part
diaphragm	web
shapes	shape_feauture
straight_length	straightLength

Step 2 – Storage of design and relevant foundation concepts

The results obtained in step 1, as shown in table 1, are forwarded in this step simultaneously to the two modules of the verification mediator named ‘concept matcher’ and ‘target ontology inheritance identifier’. The ‘concept matcher’ stores these results for later comparison with the results obtained in step 4 from the ‘target ontology inheritance identifier’.

Step 3 – Inheritance queries for the target ontology concepts

In this step, the ‘target ontology inheritance identifier’ takes the results received from the ‘source ontology inheritance identifier’ and generates queries to track down the sub-classes of the foundation concepts in the manufacturing domain ontology. One such query is given below which finds the manufacturing ontology subsumption of the foundation concept ‘part’.

```
(and (Property ?m) (contextFor ?m MFG) (sup ?m FDN.part))
```

Table 2. Foundation inheritances of manufacturing concepts

Foundation Concepts	Manufacturing Subsumpitons
part	work_piece
web	webbing
shape_feauture	features
straightLength	length

Similar queries are written for other foundation concepts identified in step 1. In the case of the example knowledge sharing system presented in figure 6, this step will obtain the results shown in table 2.

Step 4 – Sending foundation and relevant manufacturing concepts

In this step, the results obtained in step 3 are sent to the 'concept matcher module' for this module to compare the foundation inheritances of concepts in the design and manufacturing domain ontologies.

Step 5 – Sending similarity information to the 'fact builder and asserter module'

In the fifth step, the information received from the 'target ontology inheritance identifier' module is used to establish similarities by the 'concept matcher' module. This is done by comparing the foundation inheritances of design concepts received in step 2 from the 'source ontology inheritance identifier module' with those received in step 4. In other words, the information in table 1 is compared with that in table 2. The results obtained are shown in table 3.

Table 3. Established similarities according to the foundation inheritances.

Design Concepts	Foundation Inheritances	Manufacturing Concepts
component	part	work_piece
diaphragm	web	webbing
shapes	shape_feauture	features
straight_length	straightLength	length

This comparison of two sets of concepts results in the discovery of similarities across two domain ontologies. These similarities are shown in table 4 below.

Table 4. Established concept similarities

Design Concepts	Equivalent Manufacturing Concepts
component	work_piece
diaphragm	webbing
shapes	features
straight_length	length

Once similarities between concepts are established, relations and functions existing in the manufacturing ontology binding the same concepts as in the design ontology are also declared similar. The relation similarities found in this case are shown in table5.

Table 5. Relation similarities across two ontologies

Design relation	Holding between	Manuf. equivalent	Manuf. relation
hasFeature	component	work_piece	hasAttribute
	shapes	features	
hasLength	shapes	features	withLength
	straight_length	length	

In the same way, the function similarity is shown in table 6 below.

Table 6. Function similarities across two ontologies

Design function	Measures	Manufacturing equivalent	Manufacturing function
mm	straight_length	length	mils

The complete ontology containing all these relations and functions cannot be shown here due to the limited number of pages of this paper.

The steps explained up to this point fulfill the requirements of ontology mediation and resulting knowledge verification when text-based knowledge is shared between the communicating parties. The case considered here, however, is different. In this case, the manufacturing knowledge to be shared exists in the form of integrity constraints or manufacturability rules in the manufacturing ontology. To make these rules meaningful, the established similarities now have to be used to translate the entire design model in to a manufacturing model to make use of the manufacturing knowledge. Step 6performs this task.

Step 6 – Translating facts and asserting to verify manufacturability

Since all three types of similarities i.e. concept, relation and function, have now been established for the facts to be asserted, the facts are translated into manufacturing language as shown in table 7.

Since these facts are now in a comprehensible form for the manufacturing knowledge base, their assertion results in the manufacturability rule getting activated in case the length is changed to a value greater than 500mm. The facts modeling the length of the 'diaphragm' less than 500mm are successfully asserted in the design knowledge base as they have now been approved by the manufacturability rules existing in the manufacturing knowledge base. In case of a length value more than 500mm, the integrity constraint statement is passed on to the designer which says:

Table 7. Design and translated manufacturing facts

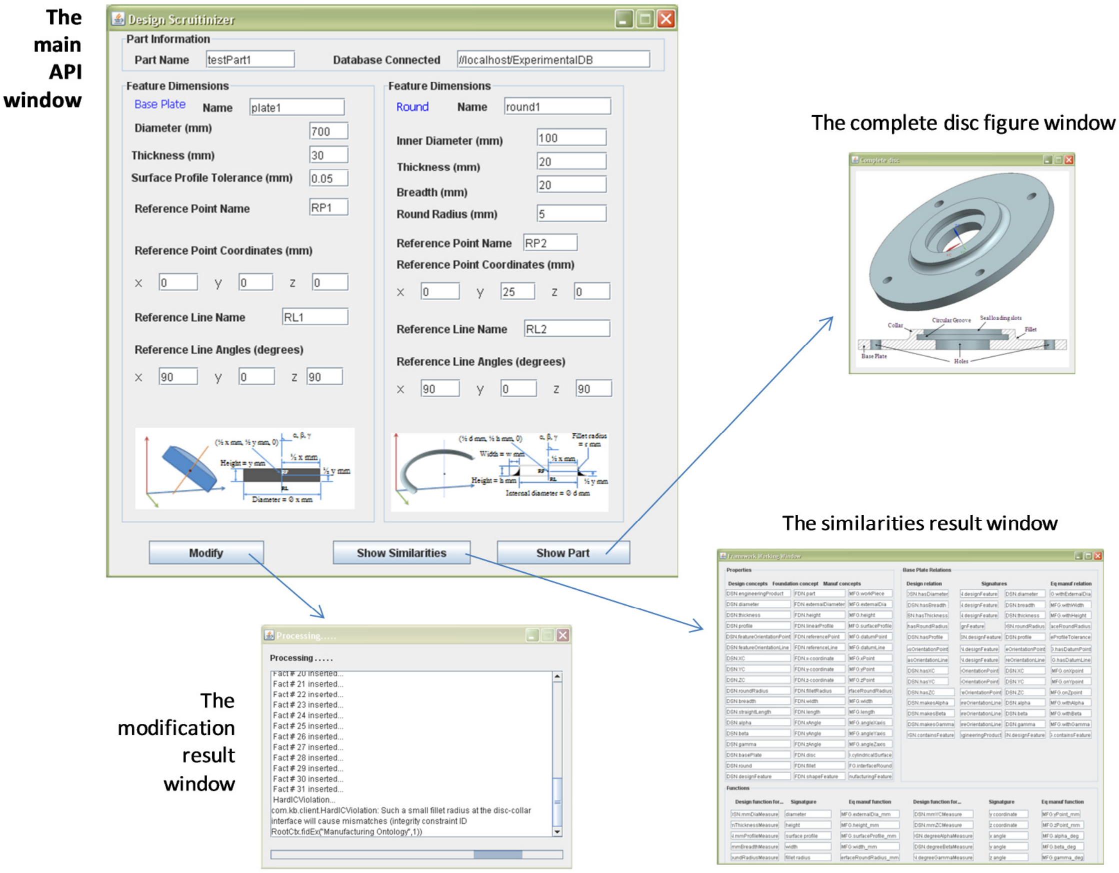
Design facts	Translated manufacturing facts
`(component compDisc)`	`(work_piececompDisc)`
`(diaphragm dphXYZ)`	`(webbing dphXYZ)`
`(hasLengthdphXYZ (mm 150))`	`(withLengthdphXYZ (mils 150))`
`(hasFeaturecompDiscdphXYZ)`	`(containsFeaturecompDiscdphXYZ)`

```
"dphXYZ larger than 500mm in length cannot be produced in the
available machines."
```

dphXYZ in the above statement is the name of the feature which the designer gave while asserting facts. It is to be noted that the process explained above, broadly, is manufacturability verification. The whole procedure to verify manufacturability, however, involves the process of similarity finding and knowledge verification which is done in steps 1 to 5 up to the point where similarities are established.

7. The Java API

This API or the Application User Interface was specifically developed to just deal with one design scenario but it demonstrates how a more comprehensive software application may work to verify the knowledge being shared which in this case is the checking of the manufacturability of a product being designed. However, the capability of this API to automatically detect similarities across two ontologies is more important from the point of view of this research.. It does this by working on the principles of the verification mediator explained earlier. This API is designed to take input from the design user in the form of mathematical values for the geometrical dimensions and positional parameters of a



Design Scruitinizer

Part Information

Part Name testPart1 Database Connected //localhost/ExperimentalDB

Feature Dimensions

Base Plate Name plate1
Diameter (mm) 700
Thickness (mm) 30
Surface Profile Tolerance (mm) 0.05
Reference Point Name RP1
Reference Point Coordinates (mm)
x 0 y 0 z 0
Reference Line Name RL1
Reference Line Angles (degrees)
x 90 y 0 z 90

Feature Dimensions

Round Name round1
Inner Diameter (mm) 100
Thickness (mm) 20
Breadth (mm) 20
Round Radius (mm) 5
Reference Point Name RP2
Reference Point Coordinates (mm)
x 0 y 25 z 0
Reference Line Name RL2
Reference Line Angles (degrees)
x 90 y 0 z 90

Processing.....

Processing .....

```
Fact # 20 inserted...
Fact # 21 inserted...
Fact # 22 inserted...
Fact # 23 inserted...
Fact # 24 inserted...
Fact # 25 inserted...
Fact # 26 inserted...
Fact # 27 inserted...
Fact # 28 inserted...
Fact # 29 inserted...
Fact # 30 inserted...
Fact # 31 inserted...
HardICViolation...
com.kb.client.HardICViolation: Such a small fillet radius at the disc-collar
interface will cause mismatches (integrity constraint ID
RootCtx.fidEx("Manufacturing Ontology",1))
```

Framework Working Window

Properties

Design concepts	Foundation concept	Manuf concepts
DSN.engineeringProduct	FDN.part	MFG.workPiece
DSN.diameter	FDN.externalDiameter	MFG.externalDia
DSN.thickness	FDN.height	MFG.height
DSN.profile	FDN.linearProfile	MFG.surfaceProfile
DSN.featureOrientationPoint	FDN.referencePoint	MFG.datumPoint
DSN.featureOrientationLine	FDN.referenceLine	MFG.datumLine
DSN.XC	FDN.x-coordinate	MFG.xPoint
DSN.YC	FDN.y-coordinate	MFG.yPoint
DSN.ZC	FDN.z-coordinate	MFG.zPoint
DSN.roundRadius	FDN.filletRadius	rfaceRoundRadius
DSN.breadth	FDN.width	MFG.width
DSN.straightLength	FDN.length	MFG.length
DSN.alpha	FDN.xAngle	MFG.angleXaxis
DSN.beta	FDN.yAngle	MFG.angleYaxis
DSN.gamma	FDN.zAngle	MFG.angleZaxis
DSN.basePlate	FDN.disc	).cylindricalSurface
DSN.round	FDN.fillet	FG.interfaceRound
DSN.designFeature	FDN.shapeFeature	nufacturingFeature

Base Plate Relations

Design relation	Signatures		Eq manuf relation
)SN.hasDiameter	N.designFeature	DSN.diameter	G.withExternalDia
DSN.hasBreadth	N.designFeature	DSN.breadth	MFG.withWidth
SN.hasThickness	N.designFeature	DSN.thickness	MFG.withHeight
hasRoundRadius	gnFeature	)SN.roundRadius	aceRoundRadius
DSN.hasProfile	SN.designFeature	DSN.profile	eProfileTolerance
sOrientationPoint	N.designFeature	eOrientationPoint	).hasDatumPoint
asOrientationLine	N.designFeature	reOrientationLine	G.hasDatumLine
DSN.hasXC	OrientationPoint	DSN.XC	MFG.onXpoint
DSN.hasYC	OrientationPoint	DSN.YC	MFG.onYpoint
DSN.hasZC	eOrientationPoint	DSN.ZC	MFG.onZpoint
DSN.makesAlpha	reOrientationLine	DSN.alpha	MFG.withAlpha
DSN.makesBeta	reOrientationLine	DSN.beta	MFG.withBeta
DSN.makesGamma	reOrientationLine	DSN.gamma	MFG.withGamma
)SN.containsFeature	gineeringProduct	N.designFeature	).containsFeature

Functions

Design function for...	Signatgure	Eq manuf function	Design function for...	Signatgure	Eq manuf function
)SN.mmDiaMeasure	diameter	MFG.externalDia_mm	DSN.mmYCMeasure	y coordinate	MFG.yPoint_mm
nThicknessMeasure	height	MFG.height_mm	DSN.mmZCMeasure	z coordinate	MFG.zPoint_mm
I.mmProfileMeasure	surface profile	MFG.surfaceProfile_mm	)SN.degreeAlphaMeasure	x angle	MFG.alpha_deg
mmBreadthMeasure	width	MFG.width_mm	DSN.degreeBetaMeasure	y angle	MFG.beta_deg
oundRadiusMeasure	fillet radius	erfaceRoundRadius_mm	I.degreeGammaMeasure	z angle	MFG.gamma_deg

Figure 8. The API and its components

component. These values are entered in the text boxes shown in the 'main API window' in figure 8. Figure 8 also shows the other components of the API behind which the software based on the verification mediator mechanism works. By following the first five steps of verification mediation, as explained earlier, the software discovers similarities across two ontologies and then displays them in the 'similarities result window' as shown in the figure. These similarities are then used by the software to form facts understandable by the constraints existing in the manufacturing domain ontology through the methodology explained earlier. The 'modification result window' shows progress on facts assertion both in the manufacturing knowledge base and then in the design knowledge base if the facts are found correct i.e. the created features or components are manufacturable. If the manufacturability is found difficult or impossible, the result is shown in the 'modification result window' and the fact assertion in the design knowledge base is cancelled.

Ideally, this knowledge verification system is to be connected to a CAD software where creation of drawing entities are to be examined by the knowledge existing in the manufacturing knowledge base. The API developed, however, is a first step towards that task and some further work may lead to the ultimate target of CAD based manufacturability verification system.

8. Discussion and Conclusions

This paper presents the design and working of a novel knowledge verification framework. This framework is specifically designed to work for a setting in which domain ontologies are committed to a common foundation and are built by using the concepts from a core-concepts ontology. This framework is proposed after a comprehensive study of the available literature and a case study at an aerospace components manufacturing company. The validity of this framework was tested through the development of a prototype user interface and afterwards a more detailed version was used to demonstrate more complex industrial examples, and to verify the design of the proposed framework. The novelty and uniqueness of this work in comparison with the existing tools and techniques of ontology mediation and knowledge verification lies firstly in its use of axiomatizations at the foundation and core-concepts level in order to maintain the consistency of domain ontologies, secondly, in its unique use of these same consistency principles (laid by the axiomatizations) to mediate between two diverse domain ontologies for knowledge verification, and thirdly, in the unique mechanism of similarity finding. The proposed framework is also a significant step towards increasing the level of automation and accuracy of the process of ontology mediation and knowledge verification. This is evident from the fact that no human intervention was required during the six steps of knowledge verification. There are, however, certain assumptions on which this framework works. For example, the manufacturing features used to demonstrate the working of the verification mediator are very simple and elementary. More complex cases also need to be considered, for example where domain ontology builders have to create more complex features through the aggregation of the available simple and basic features from the core-concepts ontology. Further work can also be done to increase the number of options available for the domain ontology builders to connect the newly created concept with the concepts already existing in the foundation and core-concept ontology. At the moment only the parent-child relationship is used. However despite these limitations, this work provides impetus for more developed and enhanced mechanisms of knowledge verification, especially in the design and manufacturing domain, where modeling of components is based on the creation of shape features.

9. Acknowledgements

This research was funded by the Loughborough Innovative Manufacturing and Construction Research Centre (IMCRC) and is being conducted in partnership with our industrial collaborators.

10. References

[Anonymous], 2010.KFL Reference, Ontology Works Inc, 3600 O'Donnell Street, Baltimore MD 21224, USA, www.highfleet.com

Aleksovski, Z., Klein, M., Kate, W.T. and Harmelen, F.V., 2006.Matching Unstructured Vocabularies using a Background Ontology, *In: Proceedings of Knowledge Engineering and Knowledge Management (EKAW,* 2006, Springer-Verlag, pp. 182-197.

Anjum, N., Harding, J., Young, B. and Case, K., 2010. Gap Analysis of Ontology Mapping Tools and Techniques.*Enterprise Interoperability IV,* , pp. 303-312.

Anjum, N., Harding,J., Young, B. And Case, K., 2012. Manufacturability verification through feature based ontological product model.*Journal of Engineering Manufacture IMECH-E Part B. Manuscript accepted.*

Changoora, N. and Young, R.I.M., 2010. The configuration of design and manufacture knowledge models from a heavyweight ontological foundation. *International Journal of Production Research,* 25(1).

Deng, J., Dong, W., Socher, R., Li, L.-., Li, K. and Fei-Fei, L., 2009. ImageNet: A Large-Scale Hierarchical Image Database, *CVPR09,* 2009.

Devedzic, V., 2002.Understanding ontological engineering.*Commun.ACM,* 45(4), pp. 136-144.

European Communities, 2004.*European Interoperability Framework v1.*European Communities.

Gangemi, A., Guarino, N., Masolo, C., Oltramari, A. and Schneider, L., 2002. Sweetening Ontologies with DOLCE, *13th International Conference on Knowledge Engineering and Knowledge Management (EKAW02),* 2002, pp. 166-182.

Gangemi, A., Fisseha, F., Keizer, J., Lehmann, J., Liang, A., Pettman, I., Sini, M. and Taconet, M., 2004.A Core Ontology of Fishery and its use in the Fishery Ontology Service Project, *EKAW-CorOnt,* 2004.

Gomez-Perez, A., Fernandez-Lopez, M. and Corcho, O., 2004.*Ontological Engineering: with example from the areas of Knowledge Manaement, e-Commerce and the Semantic Web.* London: Springer-Verlag.

Gruninger, M., Atefi, K. and Fox, M.S., 2000.Ontologies to Support Process Integration in Enterprise Engineering.*Computational & Mathematical Organization Theory,* 6, pp. 381-394(14).

Gruninger, M., 2004.Ontology of the Process Specification Language. In: S. STAAB and R. STUDER, eds, *Handbook on Ontologies.* 1 edn. Germany: Springer.

ISO/IEC 24707:2007(E), 2007. Information technology — Common Logic (CL): a framework for a family of logic based languages.

Jarrar, M. and Meersman, R., 2009.Advances in Web Semantics I. In: T.S. DILLON, E. CHANG, R. MEERSMAN and K. SYCARA, eds, Berlin, Heidelberg: Springer-Verlag, pp. 7-34.

Klein, M., 2001. Combining and Relating Ontologies: An Analysis of Problems and Solutions.

Li, J., 2004. LOM: A Lexicon-based Ontology Mapping Tool, *Proceedings of the Performance Metrics for Intelligent Systems (PerMIS),* 2004, pp. 2004.

Lin, H.K. and Harding, J.A., 2007. A manufacturing system engineering ontology model on the semantic web for inter-enterprise collaboration.*Computers in Industry,* 58(5), pp. 428-437.

Mascardi, V., Locoro, A. and Rosso, P., 2010. Automatic Ontology Matching via Upper Ontologies: A Systematic Evaluation. *IEEE Trans.onKnowl.and Data Eng.,* 22(5), pp. 609-623.

Mascardi, V., Rosso, P. and CordÃ¬, V., 2008. Enhancing Communication inside Multi-Agent Systems â‹† An Approach based on Alignment via Upper Ontologies.

Matuszek, C., Cabral, J., Witbrock, M. and DeOliveira, J., 2006.An Introduction to the Syntax and Content of Cyc.*AAAI Spring Symposium*.

Neches, R., Fikes, R., Finin, T., Gruber, T., Patil, R., Senator, T. and Swartout, W.R., 1991.Enabling technology for knowledge sharing.*AI Mag.,* 12(3), pp. 36-56.

Niles, I. and Pease, A., 2001. Towards a standard upper ontology, *FOIS '01: Proceedings of the international conference on Formal Ontology in Information Systems,* 2001, ACM, pp. 2-9.

Noy, N.F., 2004. Semantic Integration: A Survey Of Ontology-Based Approaches. *SIGMOD Record,* 33, pp. 2004.

Panetto, H. and Molina, A., 2008. Enterprise integration and interoperability in manufacturing systems: Trends and issues. *Comput.Ind.,*59(7), pp. 641-646.

Schorlemmer, M. and Kalfoglou, Y., 2005. Progressive ontology alignment for meaning coordination: an information-theoretic foundation, *AAMAS '05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems,* 2005, ACM, pp. 737-744.

Studer, R., Benjamins, V.R. and Fensel, D., 1998. Knowledge engineering: Principles and methods. *Data & Knowledge Engineering,* 25(1-2), pp. 161-197.

Swartout, B., Ramesh, P., Knight, K. and Russ, T., 1997. Towards Distributed Use of Large Scale Ontologies, A. FARQUHAR, M. GRUNIGER, A. GOMEZ-PEREZ, M. USHOLD and P. VAN-DER-VET, eds. In: *AAAI'97 Spring Symposium on Ontological Engineering,* 1997, pp. 138-148.

Usman, Zahid, 2012. A manufacturing core concepts ontology to support knowledge sharing. *Doctoral thesis*, Loughborough University, UK

Young, R., Chungoora, N., Usman, Z., Anjum, N., Gunendran, G., Palmer, C., Harding, J., Case, K. and Cutting-Decelle, A., 2010.An exploration of foundation ontologies and verification methods for manufacturing knowledge sharing, *Workshop on Interoperability for Enterprise Software and Applications (I-ESA Workshop, 2010),* 2010.