Computer-Aided Design and Applications

http://www.cadanda.com

Ontology and Assembly Joint Topology Representation

Kyoung-Yun Kim[1]

[1]Wayne State University, kykim@eng.wayne.edu

ABSTRACT

This paper presents an ontology-based assembly joint topology representation framework, which is illustrated with a fixture assembly case study. Joints within the physical structure of an assembly are inevitable because of the limitations of component geometries and the associated, required engineering properties. While joints themselves may have similar geometrical configurations, the physical implications of the selected joining processes vary significantly. The presented ontology framework and model are based on an understanding of the morphological characteristics of an assembly and its physical effects. Even if geometry is of great importance during assembly design, the morphological characteristics are consequences of the principle physical processes and of the design intentions. In this paper, an assembly topology is represented as a mereotopology, which is a region-based theory for the parts and associated concepts. This formal ontology is used to differentiate often ambiguous assembly and joining relations. The mereotopology for assembly design is implemented in SWRL (Semantic Web Rule Language) rules and OWL (Web Ontology Language) triples.

Keywords: mereotopology, ontology, topology, SWRL, and collaborative design.
DOI: 10.3722/cadaps.2008.630-638

1. INTRODUCTION

Heterogeneous tools and multiple designers are frequently involved in this network-based collaborative product design [4], [18]. In order to persistently capture designers' intentions and the semantics of the designers' terms, a standardized data format is prerequisite. One designer's terms and definitions for aspects of a product's design could potentially be different, and therefore an obstacle exists to sharing design information with multiple designers. Furthermore, appropriate design knowledge should be provided during all design processes in order to make proper design decisions. Within the physical structure of engineered products, joints are inevitable because of the limitations of component geometries and the required engineering properties. This implies that a framework is needed to capture and propagate assembly design and joint information in a robust assembly model throughout the entire product development processes, by an understanding of assembly geometry and its physical effects. However, the physical effects of joining processes are still a major problem related to quality and productivity [1],[8] in manufacturing industries, even more so in mechanized and automated industries. Furthermore, current solid modelers and simulation software cannot provide either complete product definitions or the semantic content, since traditionally the geometric model was in the center of product models. Even if geometry is of great importance during assembly design, the morphological characteristics are consequences of the principle physical processes and of the design intentions [11]. Therefore, this assembly design and its associated morphological characteristics should be represented in a standard way and be transparently shared throughout the entire product development, while maintaining the association of assembly joint design and morphological characteristics that includes geometry and topology.

Reusing existing models of components and sub-assemblies is important for reducing model development time, while allowing cross-fertilization of knowledge across multiple product development projects. The role of the semantic web is to operate as one large virtual model base, because it is not a web of documents, but rather a web of relations among resources. It denotes that product models—including data, information, and objects—can be connected to the Internet, and that design collaborators can have ubiquitous access to a product model through a client interface, while understanding the semantics and context of information. In this manner, existing product models and associated knowledge can be reused. Even though many researchers have presented various different ontologies, from a cooperate level ontology [9],[7],[17] to an engineering ontology [16],[11],[12], these all have limitations on representing mechanical assemblies and their complex morphological characteristics, including joining constraints. Neither has it been reported how the morphological characteristics of assembly design should be represented in ontology or be shared in the network-based assembly design in an integrated manner.

Therefore, this paper discusses how ontologies can be employed to represent assembly joint topology. A real fixture assembly is used to illustrate this paradigm. The assembly joint topology is represented as a mereotopology, which is a region-based theory for the parts and associated concepts. This formal ontology can be used to differentiate often ambiguous assembly and joining relations. Although joints may have similar geometrical configurations, the physical implications of the selected joining processes may vary significantly. The mereotopology for assembly design is implemented by using SWRL (Semantic Web Rule Language) rules and OWL (Web Ontology Language) triples.

2. BACKGROUND

2.1 Ontology

The broadest application domains of ontologies are upper-level ontologies that describe common sense-level knowledge (e.g., thing, individual, etc.) in a machine-interpretable manner. Cycorp's CYC is a commercial ontology containing over 200,000 terms and assertions. Its goal is to define high-level, common sense-type of concepts in a machine-interpretable manner. CYC's potential applications include online brokering of goods and services, enhanced virtual reality, improved machine translation, improved speech recognition, data mining, true language processing, etc. (Cycorp). However, since CYC is still a high-level ontology, it has not had a strong impact in the mechanical design domain. Nonetheless, in 1999, the National Institute of Standards and Technology (NIST) chose CYC as an ontology for further investigation in the manufacturing domain [20]. The results from this investigation led to the development of Process Specification Language (PSL), a language that is generic enough to represent discrete manufacturing and construction process data [10]. Narrower in scope than upper-level ontologies, enterprise-level ontologies attempt to formalize the practices and processes that occur within an organization. The level of the concepts is enterprise-specific, and the concepts are meant to promote knowledge reuse with regard to business decisions and transactions [23], [6], [7].

Some ontological research has been applied at both the conceptual and detailed design levels. Kitamura et al. [14] and Horváth et al. [11] attempt to develop ontologies to represent functional design and design features. In particular, the work of Kitamura et al. captures the flow of fluids or the flow of parts in manufacturing. However, these research works have limitations on representing mechanical assemblies and their joining constraints. Kim et al. [12][13] showed feasibility of ontological representation of assembly relations and associated constraints. This paper further enhances the scope of the explicit representation to various joining methods to properly represent topological information associated to assembly joints.

2.2 Mereotopology

Mereology is the name of the formal theory of parts and associated concepts developed by Leśniewski [15]. To distinguish his theory from other mereological theories, the theory of Leśniewski is referred to as “Mereology” written with a capital M. Just as mereology is a theory for the part-whole relation in general, there is also a theory of the “is-connected-to” relation in general. This theory is called topology. Region-based theories of mereotopology [19], [5] suggest that it can be used to represent entities that exist in other spaces besides the usual physical one. A region is a portion of a space, typically the portion occupied by some entity, whether material (e.g., a physical part) or otherwise (e.g., a hole). The overarching goal of any mereotopology theory is to describe the nature of regions and the entities that occupy them, and the interrelations between regions. Regions themselves, however, need not be primitive entities within the theories. There are many mereotopology theories that do not take regions as primitive entities, such as Casati and Varzi [3] and Smith [21]; rather they build them up from more primitive entities. Mereology claims to be not only a theory about the part-of relationship between components, but also between a large class of “things” for which parts and wholes are relevant. To represent assembly joints, connectivity of individuals to which the part-whole relation applied is obviously important. There are two approaches to develop such a theory. One is to extend an existing theory of mereology with the topological is-connected-to relation. The other is to integrate mereological and topological concepts and relations into one mereotopological theory. While the first approach is arguably more intuitive; however the second usually leads to more compact theories because axioms can be combined. This research uses Smith's mereotopology [22], which can represent regions in physical domains (e.g., parts and products), as well as non-physical domains (e.g., form feature) that are associated to assembly design formalisms.

3. ASSEMBLY DESIGN ONTOLOGY IMPLEMENTATION

In this research, Web Ontology Language (OWL) and Semantic Web Rule Language (SWRL) are used to represent assembly and joint design. OWL is designed for use by applications that need to process the content of information, versus simply presenting information [24]. SWRL is intended to be the rule language of the Semantic Web. The SWRL submission package contains three components in addition to the principal prose document: (1) an RDF Schema partially describing the RDF Concrete Syntax of SWRL; (2) an OWL ontology partially describing the RDF Concrete Syntax of SWRL; (3) an XML Schema for the SWRL XML Concrete Syntax [24],[25]. Through careful investigation, terms including Individual, Product, Assembly, Assembly Component, Part, Sub-assembly, Assembly Feature, Form Feature, Joint, Joint Feature, Mating Feature, and others are defined and used to construct an assembly design ontology as shown in Figure 1. The ontology contains detailed classes of joining processes. To implement an assembly topology using mereotopology, which we explain in the following section, the Individual class is defined. Smith's mereotopology is a theory about the part-of relationship between components, as well as a large class of "things" for which parts and wholes are relevant. Smith called "things" as "individuals" [22]. In this paper, we designate any physical components (e.g., part, assembly, etc.) and any geometric entities that are associated to assembly and joining as individuals. The Individual class does not include any information classes (e.g., spatial relationships, manufacturing processes).

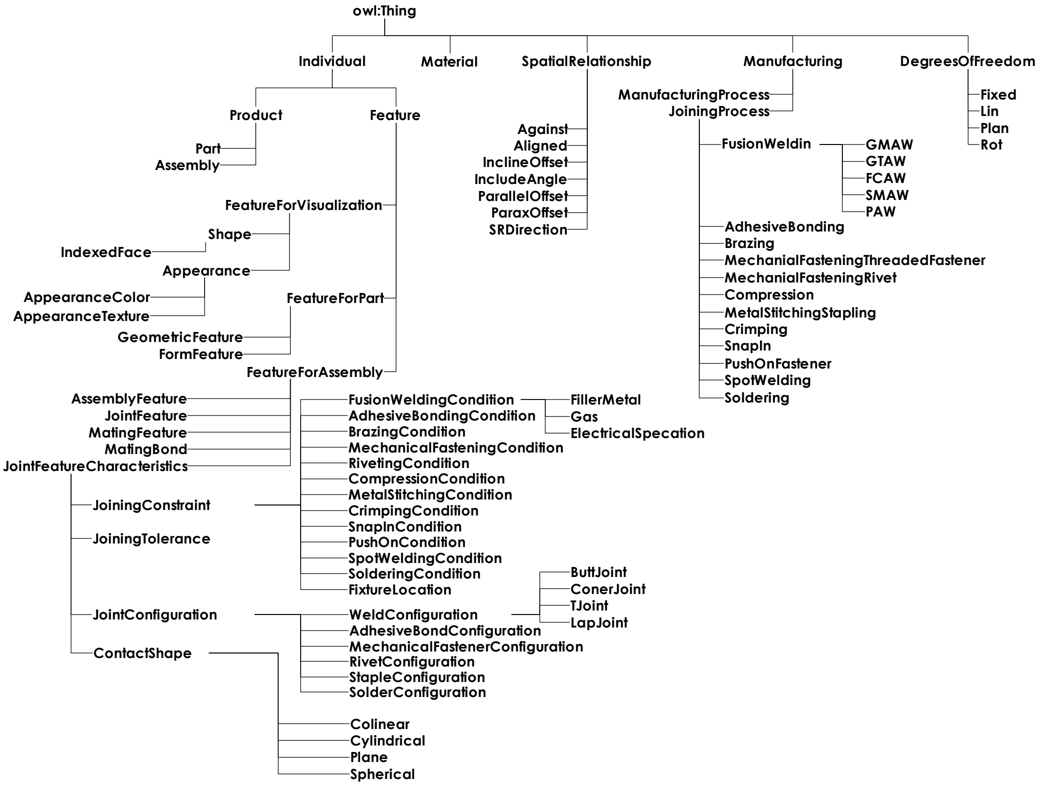

Fig. 1: Assembly design ontology class hierarchy.

4. ASSEMBLY JOINT TOPOLOGY REPRESENATION

An assembly is a collection of manufactured parts, brought together by assembly operations to perform one or more of several primary functions. To properly represent morphological characteristics (i.e., geometry, topology) of assembly design, we investigate two important relationships to represent topological relationships in this paper. One: assembly hierarchical relationships, including part-to-assembly, feature-to-feature, etc. Two: joining relationships. Joining finalizes the assembly operation and generates actual joints, while providing the primary function of the assembly (e.g., structural, mechanical, and electrical). We explain these relationships in following sub-sections.

4.1 Assembly Hierarchical Relationships

4.1.1 belongTo relations: A belongTo relation defines relations between a part and a form feature. The inferred constraints: 1) every form feature must belong to a part (C1-1); and 2) a form feature must not belong to two parts (C1-2), can be transformed to two asserted conditions (in Protégé 3.1). For the C1-1, an asserted condition, "∃ belongTo Part" is used and an asserted condition, "belongTo = 1" is used to represent C1-2. This cardinality condition means the property belongTo has exactly one value.

4.1.2 inter-featureAssociation relations: The inter-featureAssociation relation represents the relations between form features. The relational constraint stands for the relationship between two form features in the form feature hierarchy. For example, a block may have a blind hole at a certain location. The distance between the coordinates of the block and the blind hole is a dimensional constraint. Since the block form feature contains the hole form feature (since the block is a parent feature of the hole in the feature decomposition hierarchy), their relational constraint is 0. If a form feature has some form feature, then the relational constraint is 1. When two form features do not belong to each other (e.g., two holes in a block), the relational constraint is 2. The associated implied constraints: 1) the associated form features must not be identical (C2-1); 2) the associated form features must belong to same part (C2-2); and the relational constraint must be represented and included (C2-3), can be modeled in SWRL rules shown below.

SWRL rules:

- FormFeature(?x) ∧ FormFeature(?y) ∧ Part(?z) ∧ differentFrom(?x, ?y) ∧ belongTo(?x, ?z) ∧ belongTo(?y, ?z) ∧ relationalConstraint_0(?x, ?y) → inter-featureAssociation(?x, ?y)
- FormFeature(?x) ∧ FormFeature(?y) ∧ Part(?z) ∧ differentFrom(?x, ?y) ∧ belongTo(?x, ?z) ∧ belongTo(?y, ?z) ∧ relationalConstraint_1(?x, ?y) → inter-featureAssociation(?y, ?x)
- FormFeature(?x) ∧ FormFeature(?y) ∧ Part(?z) ∧ differentFrom(?x, ?y) ∧ belongTo(?x, ?z) ∧ belongTo(?y, ?z) ∧ relationalConstraint_2(?x, ?y) → inter-featureAssociation(?x, ?y)

4.1.3 Assembly/joining relations: The assembly/joining relations represent the relations between form features that belong to different parts. It implies two constraints: 1) the associated form features must belong to two non-equivalent parts (C3-1); and 2) the associated form features must be a joining pair (C3-2), and the implied constraints are represented in SWRL rule as shown below. This assembly/joining relation is extended in the following section to accommodate the complexity of joining relations.

SWRL rule:

- FormFeature(?x) ∧ FormFeature(?y) ∧ Part(?z) ∧ Part(?a) ∧ belongTo(?x, ?z) ∧ belongTo(?y, ?a) ∧ differentFrom(?z, ?a) ∧ isJointPair(?x, ?y) → assemblyJoiningRelationship(?x, ?y)

4.2 Mereotopological Representation of Assembly/joining Relations

To represent assembly joints, connectivity of individuals for which the part-whole relation applied is obviously important. There are two approaches to make such a theory. One is to extend an existing theory of mereology with the topological is-connected-to relation. The other is to integrate mereological and topological concepts and relations into one mereotopological theory. While the first approach is more intuitive, the second usually leads to more compact theories because axioms can be combined. This paper follows the second approach.

In order to define various assembly joints, some of mereotopological definitions, axioms, and theorems are adopted. Adopting the mereotopology definitions, we need to understand the meaning of operators used in the mereotopology. According to mereological primitive the relation of parthood or constituency, Smith [22] addressed that x is a part of y, and write ‘x**P**y’, when x is any sort of part of y, including an improper part. From the part relationship, the author derived the following further purely mereological notions. The first definition is that x overlaps y, and write ‘x**O**y := ∃z(z**P**x ∧ z**P**y),’ when z is any sort of part x and y. The second one is that x discrete from y, and write ‘x**D**y := ¬x**O**y,’ when x does not overlap y.

Smith addressed that a condition ‘φ’ in a single free variable ‘x’ is satisfied *iff* the sentence ‘φx’ is true for at least one value of ‘x’. The sum of φers can be defined as that entity y which is such that, given any entity w, w overlaps with y *iff* w overlaps with something that φs, that is ‘σx(φx): = ιy(∀w(w**O**y ≡ ∃v(φv ∧ w**O**v)))’. And then the following theorem can be proved: y = σx(φx) → ∀x(φx → x**P**y). With this proven theorem, we say that *j* is defined as any geometric entity that can be mating boundary including line, point, and face, of two joining entities of x and y (*j* is often virtual geometric entity, such as datum plane in CAD), and write ‘J := σz(φz) → ∀z(φz → z**B**x ∧ z**B**y)’.

In order to show what it is boundary, ‘**B**,’ some understandings are required. When x is a part of y that is off the boundary of y (i.e., it is neither tangential nor itself a boundary), x is an interior part of y and write ‘x**IP**y’. As the first step towards defining what it is for x to be a boundary of y, we define ‘x**X**y’ (x crosses y) by: ‘x**X**y := ¬x**P**y ∧ ¬x**D**y’ or equivalently, for y ≠ 1, ‘x**X**y := x**O**y ∧ x**O**(1–y),’ that x overlaps both y and its complement. The relation of straddles is defined as x straddles y by ‘x**St**y := ∀z(x**IP**z → z**X**y)’. From the sense, the entities straddling a given entity can be divided, intuitively, into two classes. There are those, which include among their parts a boundary of the straddled entity; there are those – characteristically non-connected – which include no such boundary. The former group is called tangents defined by ‘x**T**y := ∃z(z**P**x ∧ z**B**y)’ and the later boundary by ‘x**B**y := ∀z(z**P**x → z**St**y)’.

We derived the definition of assembly joints based on mereotopology of Smith. In this section, two geometrically and topologically similar joint pairs (i.e., fusion welding and adhesive bonding, and mechanical fastenings with threaded fasteners and rivets) are presented to describe how the mereotopological representation can be defined.

4.2.1 Fusion Welding: The thermal energy required for these welding operations is usually supplied by chemical or electrical means. Filler metals, which are metals added to the weld area during welding of the joint, may or may not be used. When filler metals are not used, the melted material still generates a weld zone. Solid state welding (e.g., friction welding) also generate a weld zone. In mereotopology, fusion welding can be defined as $x\mathbf{J}_{\mathbf{fw}}y$:= ∃w(w**O**x ∧ w**O**y) ∧ ∃w(w**St**j ∨ w**X**j). This definition indicates that if there is an entity, w, associated to joining, we observe that the entity w (i.e., a weld) overlaps x and y, which are fusion welded. In addition, the w should either straddle or cross to j. When a groove (e.g., Butt-V, Butt-double V, Tee square groove) is used, we can say kDx ∧ kDy, where k is a groove.

4.2.2 Adhesive Bonding: Adhesive bonding joins components using an adhesive. Many types of adhesives provide adequate joint strength, including fatigue strength. The mereotopological definition of adhesive bonding is $x\mathbf{J}_{\mathbf{ab}}y$:= ∃g(g**St**j ∧ g**T**x ∧ g**T**y). As described in this definition, the characteristics of adhesive bonding are different from fusion welding. In other words, the entity associated to joining is a glue; it should straddle to mating boundary, j and overlap both of x and y, which are to be assembled. Note that after the joining process, the joint is identical to the mereotopological definition. Note also that surface reaction by filler metal or adhesive treatment is not considered in this mereotopological representation.

4.2.3 Mechanical Fastening by Threaded Fasteners: Threaded fasteners are the most common method of mechanical fastening. Examples include bolds, nuts, screws, pins, and a variety of other fasteners. The mereotopological definition of mechanical fastening by threaded fasteners is $x\mathbf{J}_{\mathbf{jf}}y$:= ∃u,v{(u**X**x ∧ u**X**y) ∧ (v**T**x ∨ v**T**y) ∧ (u**P**fs ∧ v**P**fs)} ∧ fs**X**j. To represent this relationship in a mereotopology, fasteners are classified into two sections: u = threaded body; v = head of fastener and/or nut. This definition indicates that the body of any threaded fastener should cross joining parts, and the head/nut should be tangent with one of the joining parts (Figure 2). Sometimes, the threaded fastener is used to clamp a part, rather than joining two separate parts. Even though the threaded fastener for clamping is not joining two parts, it provides similar functionality with joining, since it combines two section of the parts. The mechanical fastening for clamping has a similar definition with the definition above except that x’ and x” should be part of x, $x'\mathbf{J}_{\mathbf{jfs}}x''$:=

$\exists u,v\{(u\mathbf{X}x' \wedge u\mathbf{X}x'') \wedge (v\mathbf{T}x' \vee v\mathbf{T}x'') \wedge (u\mathbf{P}fs \wedge v\mathbf{P}fs)\} \wedge fs\mathbf{X}j \wedge (x'\mathbf{P}x \wedge x''\mathbf{P}x)$. This definition needs to be understood for the case study in Chapter 5.

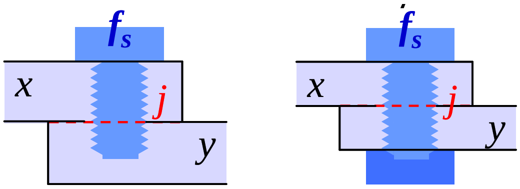

Fig. 2: Example of mechanical fastening with threaded fasteners.

4.2.4 Riveting or Mechanical Fastening by Rivets: Rivets are also another common mechanical fastening device. Installing a rivet consists of placing the rivet in the hole and deforming the end of its shank by upsetting or heading. The mereotopological definition of riveting is $x\mathbf{J}_{jr}y := \exists u,v\{(u\mathbf{X}x \wedge u\mathbf{X}y) \wedge (v\mathbf{T}x \wedge v\mathbf{T}y) \wedge (u\mathbf{P}fs \wedge v\mathbf{P}fs)\} \wedge fs\mathbf{X}j$. Similar to the threaded fastener, a rivet is divided into two sections for the definition: u = body of rivet; v = head and upset tail of rivet. By the definition, the rivet's body crosses the joining components, and the head and upset tail should be tangent to the joining components (Figure 3). It indicates a difference between the mereotopological definitions of riveting versus mechanical fastening with threaded fasteners. This definition is valid for mathematically differentiating joints; however the definitions should be modeled in a standard ontology to increase the universality of the semantic definitions.

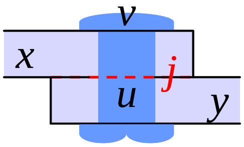

Fig. 3: Example of riveting.

4.3 SWRL Modeling of Assembly Joints

While a mereotopological ontology for assembly design provides a theoretical foundation, it lacks the universality of semantic definitions. Therefore, mereotopological notions, definitions, and primitives should be represented in a standard manner. The Semantic Web Rule Language (SWRL) is used in this research. In this paper, implicit assembly constraints of joining are derived from the mereotopological definitions described in the previous sub-section, and the constraints are explicitly represented using SWRL rules.

Joining methods	Mereotopological definition	SWRL rules
Fusion welding	$x\mathbf{J}_{fw}y := \exists w(w\mathbf{O}x \wedge w\mathbf{O}y) \wedge \exists w(w\mathbf{St}j \vee w\mathbf{X}j)$	individual(?x) ∧ individual(?y) ∧ individual(?w) ∧ is_joiningEntity(?j) ∧ overlap(?w, ?x) ∧ overlap(?w, ?y) ∧ straddle(?w, ?j) → fusionWelding(?x, ?y) individual(?x) ∧ individual(?y) ∧ individual(?w) ∧ is_joiningEntity(?j) ∧ overlap(?w, ?x) ∧ overlap(?w, ?y) ∧ cross(?w, ?j) → fusionWelding(?x, ?y)
Mechanical fastening with treaded fasteners	$x\mathbf{J}_{jf}y := \exists u,v\{(u\mathbf{X}x \wedge u\mathbf{X}y) \wedge (v\mathbf{T}x \vee v\mathbf{T}y) \wedge (u\mathbf{P}fs \wedge v\mathbf{P}fs)\} \wedge fs\mathbf{X}j$	individual(?x) ∧ individual(?y) ∧ individual(?u) ∧ individual(?v) ∧ individual(?fs) ∧ individual (?j) ∧ cross(?u, ?x) ∧ cross(?u, ?y) ∧ tangent(?v, ?x) ∧ is_part_of(?u, ?fs) ∧ is_part_of(?v, ?fs) ∧ cross (?fs, ?j) → threadedMechanicalFastening(?x, ?y) individual(?x) ∧ individual(?y) ∧ individual(?u) ∧ individual(?v) ∧ individual(?fs) ∧ individual (?j) ∧ cross(?u, ?x) ∧ cross(?u, ?y) ∧ tangent(?v, ?y) ∧ is_part_of(?u, ?fs) ∧ is_part_of(?v, ?fs) ∧ cross (?fs, ?j) → threadedMechanicalFastening(?x, ?y)

Tab. 1: Example of SWRL modeling for joining methods.

Mereotopological operators and definitions of joints are converted into SWRL rules based on the conversion rules: 1) OR relation is converted to a disjunctive normal form (DNF) and generate separated rules; 2) logical implication is converted to ~p ∨ q; and 3) existential quantifier is constrained with specific rules. For example, to define the straddle relation, $x\mathbf{St}y := \forall z(x\mathbf{IP}z \rightarrow z\mathbf{X}y)$, a logical implication (p→q) is equivalent to ~p ∨ q and the logical OR can be represented with two separated rules as shown in Table 1. For fusion welding, the mereotopological definition of a joint is $x\mathbf{J_{fw}}y := \exists w(w\mathbf{O}x \wedge w\mathbf{O}y) \wedge \exists w(w\mathbf{St}j \vee w\mathbf{X}j)$. To represent this in a SWRL form, the definition should be converted to a disjunctive normal form (DNF), that is, $x\mathbf{J_{fw}}y := \exists w(w\mathbf{O}x \wedge w\mathbf{O}y \wedge w\mathbf{St}j) \vee \exists w(w\mathbf{O}x \wedge w\mathbf{O}y \wedge w\mathbf{X}j)$. Table 1 presents two examples of SWRL modeling from the mereotopological definitions. In SWRL, all variables are assumed to be universal and an existential variable is explicitly represented with the rules. For example, the existential condition of variable z in overlap is restricted by using strict rules (i.e., is_part_of(z,x) and is_part_of(z,y)).

5. CASE STUDY

To validate the presented method, we used a fixture, which is used to hold parts to be machined in-place inside a machining center, as a case study (shown in Figure 4). Most components are made of high strength steel, and clamps to hold the parts are operated using hydraulics. This fixture includes four clamping arm as indicated by the four circles. These clamping arms will be used to illustrate ontology implementation. Figure 4 shows form features and their relationships as related to this case study of the clamping arm. The clamping arm (A_5^1) is a sub-assembly of the overall product, the fixture (A_1^1). The clamping arm has nine parts; in other words, nine parts belong to (belongTo) the clamping arm. This belongTo relation is indicated by an arrow. Figure 4 displays only five parts to increase visibility. In Figure 4, duplicated parts are omitted (e.g., equivalent fasteners and holes). The lines with solid circle ends indicate inter-FeatureAssociation. When relational constraint (RC_{pq}) is 0 or 1, a line with a single circle end is used. For example, FF_{71}, which is the base extrusion of the cap, includes two sections of extrusion (FF_{78} and FF_{79}). When RC_{pq} is 2, a line with two solid circle ends is used. An example is FF_{11} (base extrusion) and FF_{12} (middle extrusion) of the arm. These two form feature are related to each other; however one does not include the other.

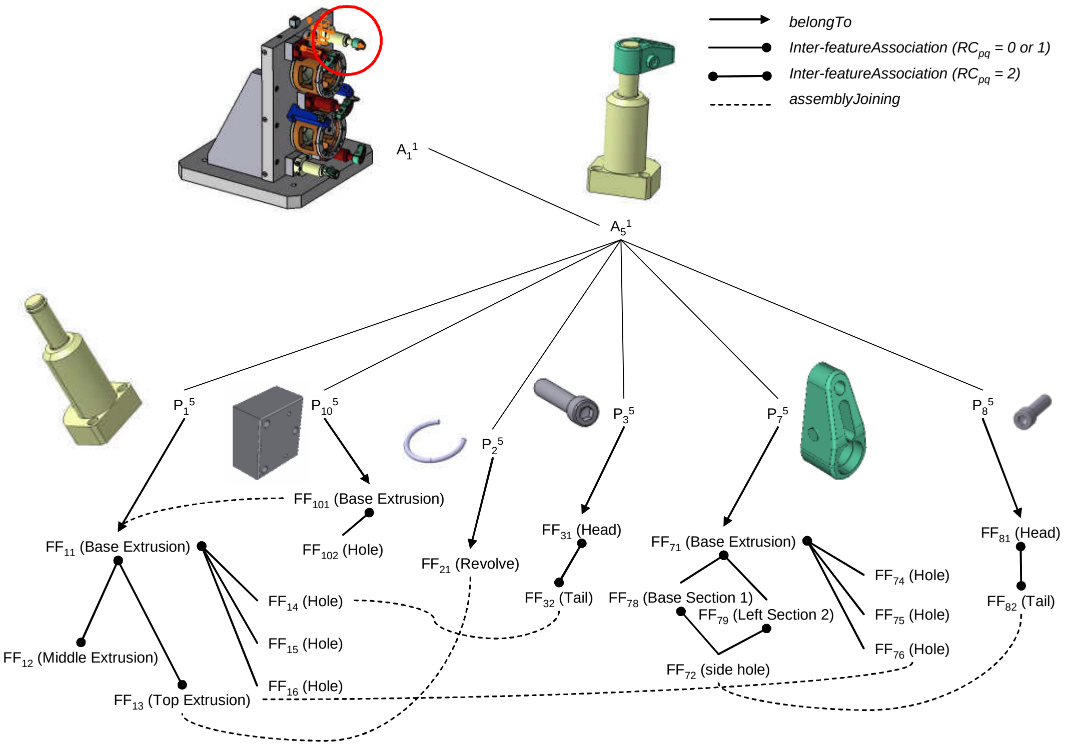

Fig. 4: Form features and relationships of clamping arm.

In this clamping arm assembly, five different types of assembly/joining relations are indicated: 1) mechanical fastening by threaded fastener between FF_{11} (base extrusion of arm) and FF_{101} (base extrusion of connecting boss); 2) mechanical fastening by threaded fastener between FF_{14} (hole of base extrusion of arm) and FF_{32} (Tail of a threaded fastener; 3) snap-fit between FF_{13} (top extrusion of arm) and FF_{21}(revolved ring); 4) simple insertion indicated between FF_{13} (top extrusion of arm) and FF_{76} (hole of cap); and 5) mechanical fastening with threaded fastener for clamping between FF_{72} (hole of cap) and FF_{82} (tail of threaded fastener). These assembly/joining relations are generated by joining the arm and the connecting boss (which is not a part of the clamping arm) by means of a mechanical fastening with threaded fastener; the arm and the cap by a ring; and the cap and a fastener for clamping the cap's two sections. The clamping is required to fasten the cap on the arm. Interestingly, this clamping of the cap uses a threaded fastener similar to the one for the arm and connecting boss. These two different joining methods are, however, geometrically very ambiguous.

SWRL rules	• Individual(?x) ∧ Individual(?y) ∧ Individual(?u) ∧ Individual(?v) ∧ Individual(?fs) ∧ Individual(?j) ∧ cross(?u, ?x) ∧ cross(?u, ?y) ∧ tangent(?v, ?x) ∧ is_part_of(?u, ?fs) ∧ is_part_of(?v, ?fs) ∧ cross(?fs, ?j) → threadedMechanicalFastening(?x, ?y) • Individual(?x1) ∧ Individual(?x2) ∧ Individual(?u) ∧ Individual(?v) ∧ Individual(?fs) ∧ Individual(?x) ∧ Individual(?j) ∧ cross(?u, ?x1) ∧ cross(?u, ?x2) ∧ tangent(?v, ?x1) ∧ is_part_of(?u, ?fs) ∧ is_part_of(?v, ?fs) ∧ cross(?fs, ?j) ∧ is_part_of(?x1, ?x) ∧ is_part_of(?x2, ?x) → threadedMechanicalFasteningClamping(?x1, ?x2)
Facts found by ontology reasoning	• fact ns_0:Fact4426 is ns_2:threadedMechanicalFastening(ns_2:FormFeature_11,ns_2:FormFeature_101); • fact ns_0:Fact4970 is ns_2:threadedMechanicalFasteningClamping(ns_2:FormFeature_78,ns_2:FormFeature_79); • fact ns_0:Fact4428 is ns_2:threadedMechanicalFastening(ns_2:FormFeature_78,ns_2:FormFeature_79);

Tab. 2: Examples of SWRL rules and reasoned facts.

In this case study, the clamping arm assembly is implemented in the developed assembly design ontology, as are the associated SWRL rules. Table 2 shows examples of SWRL rules. These rules are based on the mereotopology of assembly design. It also includes the facts found by ontology reasoning. The assembly design ontology is established by using Protégé-3.1, and the SWRL rule reasoning is conducted by a Rete-based reasoning engine (Bossam) developed by Korean Electronics and Telecommunications Research Institute (ETRI) [2]. As shown in Table 2, the facts clearly indicate that: 1) FF_{11} (of P_1^5) and FF_{101} (of P_{10}^5) are joined by mechanical fastening with a threaded fastener; 2) FF_{78} (of P_7^5) and FF_{79} (of P_7^5) are clamped by a threaded fastener. FF_{78} (of P_7^5) and FF_{79} (of P_7^5) are also related to mechanical fastening with threaded fastener. This finding is due to the similarity of rules of two joining methods. Using the Assembly Design ontology and its reasoning capability shown above, the intended joining methods can be persistently maintained and the associated topological information can be used to analyze the assembly design. Further ontology reasoning for other joining methods will be reported in a separated article.

6. CONCLUSION AND FUTURE RESEARCH

This paper presented an ontology based representation for assembly joint topology information. Using ontology, geometrically similar joints are successfully differentiated in a standard and machine-interpretable manner. By relating concepts through ontology technology rather than just defining data syntax, assembly and joining concepts can be captured in their entirety or extended as necessary. Such representation significantly improves design collaboration as assemblies that appear geometrically similar, have engineering information attached that differentiate them. In recognizing the fact that product development requires tremendous amounts of information and corresponding decisions, it is obvious that the reasoning behind every decision can hardly be transferred together with the information. Consequently, the need for feedback and communication among product development stakeholders increases, which may lead to extremely complex and uncontrollable flows of information in product development collaboration. However, providing that product information generated by the different collaborators is attached to an overall and widely accessible model, this situation may change considerably. As shown in this paper, by utilizing ontology technology, clear relations among assembly components and form features are established and assembly knowledge can be systemized in various classes. This semantically valid information provides a great foundation to realize a lean and selective assembly design information sharing environment, which the author is currently developing.

7. ACKNOWLEDGEMENT

This research is supported by the ubiquitous Autonomic Computing and Network Project, through the Ministry of Information and Communication (MIC) 21st Century Frontier R&D Program in Korea.

8. REFERENCES

[1] Beeson, R.: Pipeline welding goes mechanized. Welding Journal, 78(11), November, 1999, 47-50.
[2] Bossam Rule/OWL Reasoner. http://machine-knows.etri.re.kr/bossam/.
[3] Casati, R.; Varzi, A. C.: Spatial entities. Spatial and Temporal Reasoning, Stock, O. (eds), Kluwer, 1997, 73-96.
[4] Engineous International Symposium & Workshop, Novi, MI, USA, October 10-12, 2005.
[5] Eschenbach, C.: A mereotopological definition of point. Topological Foundations of Cognitive Science, Papers from the Workshop at the FISI-CS, Buffalo, NY, 1994, 63-80.
[6] Fox, M. S.: The TOVE project: towards a common-sense model of the enterprise, Enterprise integration laboratory technical Report, 1992.
[7] Fox, M. S.; Gruninger, M.: Enterprise Modeling, AI Magazine, 1998, 109-121, AAAI Press, Fall.
[8] Graf, T.; Staufer, H.: Laser-hybrid welding drives VW improvements, Welding Journal, 82(1), 2003, 42-48.
[9] Gruber, T. R.: A Translation Approach to Portable Ontology Specification, Knowledge Acquisition, 5(2), 1993, 199-220.
[10] Gruninger, M.; Sriram, R. D.; Cheng, J.; Law, K.: Process specification language for project information exchange. Int J IT in Architecture, Engineering and Construction, 2003, 307-28.
[11] Horváth, I.; Pulles, J. P. W.; Bremer, A. P.; Vergeest, J. S. M.: Towards an Ontology-based Definition of Design Features, SIAM Workshop on Mathematical Foundations for Features in Computer Aided Design, Engineering, and Manufacturing, 1998.
[12] Kim, K. Y.; Manley, D. G.; Yang, H. J.: Ontology-based Assembly Design and Information Sharing for Collaborative Product Development, Computer-Aided Design, 38, 2006, 1233-1250.
[13] Kim, K. Y.; Wang, Y.; Muogboh, O. S.; Nnaji, B. O.: Design Formalism for Collaborative Assembly Design Environment, Computer-Aided Design, 36(9), 2004, 849-871.
[14] Kitamura, Y; Kashiwase, M; Masayoshi, F; Mizoguchi, R.: Deployment of an ontological framework of function design knowledge, Advanced Engineering Informatics, 18(2), 2004(b), 115-27.
[15] Leśniewski, S.: On the foundations of mathematics, Topoi, 2, 1982, 7-52. (abridged English translation of podstawach matematyki)
[16] Lin, J.; Fox, M. S.; Bilgic, T.: A requirement ontology for engineering design, Concurrent Engineering Research and Applications, 4(3), 1996, 279-91.
[17] Mizoguchi, R.: Tutorial on Ontological Engineering Part 1: Introduction to Ontological Engineering, New Generation Computing, 21(4), 2003, 365-384.
[18] Pegasus: NSF I/UCRC for e-Design, Strategic Planning Meeting, Pittsburgh, PA, USA, December 9-10, 2003. http://www.e-designcenter.info
[19] Randell, D. A.; Cui, Z.; Cohn, A. G.: A spatial logic based on regions and connection, Principles of Knowledge Representation and Reasoning, Nebel, B., et al (eds), Cambridge, Mass, 1992, 165-176.
[20] Schlenoff, C.; Ivester, R.; Libes, D.; Denno, P.; Szykman, S.: An analysis of existing ontological systems for applications in manufacturing and healthcare, NISTIR 6301 National Institute of Standards and Technology, 1999.
[21] Smith, B.: Boundaries: an essay in mereotopology. The Philosophy of Roderick Chisholm, Hahn, L. (eds), LaSalle: Open Court, 1997, 534-561.
[22] Smith, B.: Mereotopology: a theory of parts and boundaries, Data and Knowledge Engineering, 20, 1996, 287-303.
[23] Uschold, M.; King, M.; Moralee, S.; Zorgios, Y.: The enterprise ontology, Knowledge Engineering Review, Special Issue on Putting Ontologies to Us, 13, 1998.
[24] World Wide Web Consortium, OWL web ontology language guide, http://www.w3c.org/TR/owl-guide.
[25] World Wide Web Consortium: SWRL: A Semantic Web Rule Language, 2004. http://www.w3.org/Submission/2004/SUBM-SWRL-20040521/