Taxonomy of Manufacturing Joining Operations based on Process Characterization

Arkopaul Sarkar[1,*], Dusan Sormaz[2], and Hedi Karray[1]

[1] *Laboratoire Génie de Production de l'École Nationale d'Ingénieurs de Tarbes (LGP-INP-ENIT), Université de Toulouse, CEDEX, 65016, Tarbes, France*
[2] *Russ College of Engineering, Ohio University, Athens, Ohio, USA*

Abstract

Depending on the complexity of the assembly design and required production constraints, factories employ various types of joining operations as part of product fabrication. Manufacturers, who are in the business of assembling, gain their competence based on what types of processes and resources they use in joining operations. For data interoperability and exchange among the partners of distributed manufacturing, these joining operations need to be described formally to build a common set of vocabulary. Current ontologies in the related topics lack the details of the process characterization in their analysis and do not adopt foundational concepts to build such definitions. This paper presents an ontology-driven characterization of the joining operations to formalize a set of definitions on ontologically grounded concepts towards the construction of a taxonomy of the joining operations. Following a rigorous characterization, provided for these joining operations based on detailed analysis of the underlying mechanisms, and associated material resources and instruments, an OWL-based rendition is also proposed which is shown to be capable of automatically deriving the taxonomy of various types of joining operations through inference.

Keywords

Assembly, Joining, Welding, Ontology, Taxonomy

1. Introduction

The objective of this paper is to present a domain knowledge-based analysis of various joining operations, commonly performed during manufacturing assembly, formally expressed using foundational concepts to provide an ontologically grounded set of vocabulary for achieving intra- and cross-domain data integration and exchange for assembly design and planning across different industrial sectors.

In the context of Industry 4.0 (I4.0), the Product Development Process (PDP) demands a high degree of digitization in every facet of decision making from product designing to production planning. In these product life cycle stages, process selection plays a critical role [1]. While planning the assembly line, the most appropriate set of processes and compatible resources need to be determined first. In the context of distributed cloud manufacturing (CM), the production planning becomes collaborative and virtual due to the availability of diverse kinds of manufacturing resources and production methods, offered as services on a cloud platform [2]. For satisfying the goal of multi-tenancy (especially when manufacturing resources are provisioned as services [3]), the planning system for CM needs to accommodate a multitude of assembly resources, with varying degrees of capability. In this regard, the lack of interoperability for assembly process specifications and capability information brings challenges and high costs to the manufacturers in their effort in the diversification of products and processes. Although globally agreed standards are available for many facets of assembly operations, inconsistencies in the definitions of vocabularies can be found across different standards, and even in

FOMI'22: 12th International Workshop on Formal Ontologies Meet Industry, September 11-12, Tarbes, France
* Corresponding author.
EMAIL: asarkar@enit.fr (A. Sarkar); sormaz@ohio.edu (D. Sormaz); mkarray@enit.fr (H. Karray)
ORCID: 0000-0002-8967-7813 (A. Sarkar); 0000-0003-3726-3288 (D. Sormaz); 0000-0002-9652-5164 (H. Karray)

the same standard. In many cases, these discrepancies need to be clarified by domain experts with expensive Information Technology (IT) support.

Many of these problems arise from ontologically 'opaque' information modelling, where the base assumptions for the concepts and relationships among them remain grounded only in the personal interpretation of the modelers and developers. In contrast, this paper develops the necessary set of vocabularies for characterizing joining operations based on foundational concepts that are philosophically rigorous and provides their definitions using first-order logic formulas instead of semantically ambiguous natural language statements. In short, this study follows the fundamental pillars of the philosophical study of ontology, i.e., "first, say what there is, what exists, what the stuff the reality is made out of, secondly, say what the most general features and relations of these things are" [4].

Before the quality of assembly, process, control, timing, and costs of assembly processes as well as the capabilities of the associated resources can be modelled, the joining processes, which act as the atomic operations for the assembly processes, need to be defined. Therefore, this paper specially focuses on the joining processes and makes an effort to characterize them as rigorously as possible with the aim to distinguish different types of joining processes. Furthermore, the model of the joining processes developed in this study is aimed at inferring subclass relationships among different joining processes to form the taxonomy. For the sake of the brevity of the paper, no example of joining process instances modelled and validated by the axiom is provided. This also allowed us to accommodate as many elucidations of the definitions as possible. However, all the subclass relationships in the joining process taxonomy are derived using reasoning based on their intricate characteristics. The validity of the model is then checked by comparing the resultant taxonomy with common knowledge from academia and industry. This validation also proves the model presented in this paper.

In section 2, past research in the development of ontology in the topics of assembly and joining operations are presented. In section 3, the core definitions of joining operations are formulated based on process characterization. In section 4, some examples from various joining operations are modelled in a taxonomy.

2. Literature Review

Earlier research on ontology modelling in this area focused on representing the product assembly. Open assembly model (OAM) proposed by NIST, defines generic concepts to integrate requirement, functional design, kinematic synthesis, and tolerance analysis along with basic part and assembly information [5]. OAM extends object-oriented assembly representation by Fiorentini et al. [6], which in turn is based on the Core Product Model (CPM), an earlier model proposed by NIST [7]. Assembly Relation Model (ARM) proposed by Kim et al. describes several widely used assembly processes, such as joining, bonding, riveting, welding, and their sub-processes [8]. Kim later provided an impressive mereotopological representation of the assembly/joining relationship [9]. Focusing on integrated product design and assembly sequence planning, Gruhier combined spatiotemporal relationships with a mereotopological construct in the so-called 'JANUS' theory that represents the joining processes as the temporal evolution of the mere topological relationships among parts, tools, and other consumables [10]. Apart from these generic models of joining processes, ontology models specific to a particular type of joining process includes welding ontologies developed by both Solano [11] and Saha et al. [12] separately. FMEA (Failure Mode and Effects Analysis) procedure in the field of reflow lead-free soldering was presented using ontology models by Molhanec et al. [13]. Except for the work of Kim et al. and Gruhier, the major drawback of the other ontology models is that they do not provide definitions of the predicates based on the foundational concept. Without formalized definitions based on commonly accepted axioms, these ontology models have a local scope that cannot be used for global interoperability. On the other end, ARM provides a detailed account of product assembly and constructs the taxonomy of joining operations based on the type of joints, assembly features, mating constraints, and spatial relationships among parts. They do not consider the joining strategy (e.g., welding fuses two parts by melting) that is implemented for different types of joining operations. Gruhier's work considers the details of sub-processes that take place during a joining operation; however, they only capture the mereotopological changes of components over time without considering the underlying process characteristics. So far, no study is conducted to capture the details of joining methods including the

state change of material and their underlying cause. Furthermore, the process specifications and process level capabilities of joining operations are not found in any existing ontology published at the time of writing this paper.

3. Ontology Model of Joining Operation

While the standard notion of 'assembling' includes many other supporting processes, such as a timely supply of the parts to workstations, arrangement of workstations on a shop floor, part mating techniques, and secondary finishing operations, this paper focuses on the exact processes that produce a joint among some components, namely 'joining process(s)'. Multiple such joining processes are applied to assemble individual parts into a larger, more complex component or assembly. For the focus of this paper being on the individual joining process, the member entities of a joint are simply referred to as components, irrespective of whether they are individual parts or sub-assembly. From the product's point of view, the assembly processes influence aesthetics, manufacturability, repairability, reliability, inspectability, safety, maintainability, and unit cost of fabrication of the product. However, the concerns for a particular joining operation are narrower, e.g., material types, thickness, joint geometry, location, accessibility, jigging and fixturing, distortion control, productivity, and initial and recurring cost of the operation. To model data for these concepts, the investigation needs to start by analyzing the strategies of different joining processes. Such characterization of joining processes will also help in inferring the taxonomy of these processes based on the known hierarchies among associated resources and subprocesses.

In this section, we provide a deep analysis of basic types of joining and formulate their definitions based on foundational concepts borrowed from top-level ontology 'Basic Formal Ontology' (BFO) [14] adopting the philosophical grounding and primitive concepts from BFO as the backbone. As the topic of this study is related to industry, the adoption of BFO makes sense as it has also been adopted by Industrial Ontology Foundry (IOF) as the foundational ontology [15]. IOF has recently published the core ontology for the industry called IOF-Core[2], which we heavily adopt in this work. If not separately defined, every term in the formulas given in the following sections for definitions and axioms, is to be considered from BFO 2020[3] or IOF-Core. The formulations also use the mereotopological constructs given by Smith [16], e.g., *overlaps*, *straddles*, *crosses*, *boundaryOf*. Please refer to [16], [17] for their elucidation.

In this study, the characterization of joining processes is analyzed based on three primary traits. First, different types of joining operations produce different types of joints. Second, a set of sub-processes, specific to the type of joining, to happen in a prescribed order to produce a joint. Lastly, the specific resources (equipment and consumables) that are needed. A great amount of attention is given to define the types of joints first as the basic classification of joining processes are derived based on the type of joints that they produce.

In a typical joining process, the components meet at a joint, which puts constraints on their relative movement. Literature and instruction manuals [18], [19] often mention the ease of disassembling a joint as a prominent criterion to classify joining processes. However, only a few joining operations produce non-permanent joints and all of them use removable fasteners (e.g., screws, nuts, and bolts) to make disassembly easy. It is more fundamental to investigate the reasons that cause the restriction on the relative movement of the components. At a first glance, such constraint is typically provided by three means: 1) continuity of material is produced between two components, 2) an additional material is used to hold the components (bonding) together in the desired relationship, and 3) relative orientation of parts restricting their degrees of freedom due to mutual constraints by their unique shapes. The first method is solely provided by welding operations which are normally applied to join metal parts by fusing or coalescing them together. Several other types of joining operations follow the second method while differing in the type of additional materials (e.g., adhesive, fasteners) and how they are applied. Brazing, soldering, bonding, and fastening operations may be classified in this category as brazing and soldering some filler metal, some adherent (glue) for bonding and some fastener for mechanical fastening acts as the additional material. The last mean, mainly adopted by fitting type operations,

[2] https://github.com/iofoundry/Core
[3] https://github.com/BFO-ontology/BFO-2020

neither creates continuity in the materials of parts nor uses additional material but only depends on the shape of the part. Before defining these various kinds of joints, we present a generic way to describe these different types of joints and then specialize them for joints produced by different types of joining operations.

3.1. Ontology of Joints

An intuitive way of expressing these three means of joining is to say that *every joining operation introduces a new part in the resulting assembly that did not exist before the joining operation*. This notion is different from the standard notion of assembly, which is normally described as made of two or more components. The new part may be provided along with the components being joined, such as a screw or nut, or created from materials out of the components themselves, as in the case of welding. To define different types of joints produced by different joining processes, we first introduce the predicate *hasConnector* between two material entities. As stated in Axiom 1, *hasConnector* specializes *bfo:hasProperContinuantPartAtAllTimes*, which is an inverse proper parthood relation projected over the entire temporal interval in which the 'whole' exists.

Axiom 1: $hasConnector(a,b) \rightarrow hasProperContinuantPartAtAllTimes(a,b)$

As stated in Axiom 2, the *hasConnector* is a two-place predicate relating to two material entities, of which the first material entity has two components, *x* and *y*, as parts and joined by the second material entity b. Additionally, the first material entity is made of only these three parts (*x*, *y*, and *b*) and nothing else. In reality, there can be more than two components that can be joined, but for simplicity, it is assumed that only two components can be joined by using *join* predicate. Moreover, more than two parts may still be modelled by using *join* predicate recursively. It is not shown here for brevity. It is to be noted that two components *x* and *y* may refer to the same entity. This situation may occur in certain cases, e.g., two ends of a wire are joined to form a ring.

Def. 1: $hasConnector(a,b) \leftrightarrow MaterialEntity(a) \wedge MaterialEntity(b) \wedge \exists x,y \left(partOf(x,a) \wedge partOf(y,a) \wedge joins(b,x,y) \wedge \forall m\big(overlaps(m,a) \leftrightarrow overlaps(m,x) \vee overlaps(m,y) \vee overlaps(m,b)\big)\right)$

Next, we define the tripartite relation $joins$ using mereotopological relations given by Smith [16], e.g., *overlaps*, *straddles*, *crosses*, *boundaryOf*. Please refer to [16], [17] for detailed elucidation. The first axiom 3 mereotopologically relates the joining material to the components being joined and the second axiom 4 positions the joining material at the juncture of these two components. Axiom 3 expresses that the connecting material must have two different parts "touching" *x* and *y* respectively. Following axioms of *straddles* in Smith [16], either the part of the joining material is at the boundary of or at a tangent to the component. The first case is evident in most of the joints whereas the second case may occur in cases, e.g., two magnetic balls joined by magnetic force.

Axiom 3: $joins(b,x,y) \rightarrow \exists b', b'' \left((b' \neq b'') \wedge partOf(b',b) \wedge partOf(b'',b) \wedge \big(straddles(b',x) \vee straddles(b'',y)\big)\right)$

Lastly, the connector either crosses or straddles the mating boundary *j*. We borrow the concept of the mating boundary from Kim's formulation defining *j* as any geometric entity that can be a mating boundary including line, point, and face, of two joining entities of *x* and *y* (*j* is often a virtual geometric entity, such as datum plane in CAD) [9]. The definition of mating boundary is also used by Demoly as the foundation for mereotopological description of kinematic pairs [20]. We reified the original formulation of Kim with a ternary predicate *MatingBoundary* and classified the mating boundary as bfo:TwoDimensionalSpatialRegion (Axiom 5)[9]. Axiom 4 considers two possible scenarios: a) a welded or glued portion may only stay at part of the boundary of the $MatingBoundary$ and b) the weld or fasteners may go across the mating boundary, i.e., consumes part of the mating boundary.

Axiom 4: $joins(b,x,y) \rightarrow \exists j \left(MatingBoundary(j,x,y) \wedge \left(crosses(b,j) \vee straddles(b,j)\right)\right)$
Axiom 5: $MatingBoundary(j,x,y) \rightarrow TwoDimentationalSpatialRegion\ (j) \wedge \forall w \left(partOf(w,j) \rightarrow overlaps(w,x) \wedge overlaps(w,y)\right)$

With the predicate *hasConnector* we can finally define *Joint* as given in Def 2. We classify *Joint* as *bfo:FiatObjectPart* because of its dependence on the parent assembly for its existence and that no bona fide boundary can be discerned of the *Joint*. We use the term to disambiguate from other types of assemblies (e.g., group of people, set of computer instructions). We borrow the definition of *MechanicalAssembly* from IOF-Core ontology, which defines it as having some *MaterialComponent* as part and is output of some *AssemblyProcess*. As per Def. 2, there must exist a *MechanicalAssembly* for which the *Joint* is connector. Conversely, as presented in Axiom 6, every *MechanicalAssembly* has at least one *Joint* as its connector. Additionally, we define another predicate *hasConnectedComponent* in Axiom 7 for relating the *Joint* of an assembly to the component of the assembly, which is *joined* by that specific *Joint*.

Def. 2: $Joint(j) \leftrightarrow FiatObjectPart(j) \wedge \exists a\ (MechanicalAssembly(a) \wedge hasConnector(a,j))$
Axiom 6: $MechanicalAssembly(a) \rightarrow Assembly(a) \wedge \exists j\ \left(Joint(j) \wedge hasConenctor(a,j) \wedge \forall p\ (hasComponentPart(a,p) \rightarrow (p \neq j))\right)$
Axiom 7: $hasConnectedComponent(b,x) \rightarrow Joint(b) \wedge MaterialComponent(x) \wedge \exists y\ \left(MaterialComponent(y) \wedge joins(b,x,y)\right)$

The model of $Joint$ presented above may require some more clarifications. First, it is to be noted that the def. 1 does not include the identity of the components *x*, *y*. It is possible that an assembly, produced by some joining process, is not the correct assembly as specified by the requirement, for example, a carpenter mistakenly nailing one piece of wood to the floor still produces a joint however not the one she intended. To include the correctness of the assembly produced, additional clauses for identifying the components, referring to the materials of the components, identification number, or visual inspection, may be added to the definition as required.

3.2. Ontology Model of Basic Joining Types

With the help of the predicate *hasConnector* the broad types of joining processes are presented below. First, the generic definition of the joining process is given in Def. 3.

Def. 3: $JoiningProcess(p) \leftrightarrow Process(p) \wedge \forall j\ (hasOutput(p,j) \rightarrow Joint(j)) \wedge \exists j\ (hasOutput(p,j) \wedge Joint(j))$

As the basic types of joining processes are derived based on the types of joints they produce as primary differentia, different types of joints need to be defined first. For welding, some portion of the material of this new part belonged to the original components before assembly as expressed for JointType1 in Def. 4. It is to be noted that some types of welding employ filler metal that becomes part of the connector. In contrast, the material of the connector is entirely supplied from outside for brazing, soldering, bonding, and fastening types of joining processes. Def. 5 expresses that for this type of joint (type 2), some additional material (filler metal) is used in the connector that fuses with the base metal. To differentiate between brazing and soldering, further details of which are presented in Section 3.3. Lastly, fitting produces a unique type of joint (type 3 in Def. 6) in which the connector is embedded in the components as they are specially designed to provide shape constraints.

Def.4: $JointType1(j) \leftrightarrow Joint(j) \wedge \exists a,b\ hasConnectedComponent(j,a) \wedge hasConnectedComponent(j,b) \wedge \exists m\ \left(partOf(m,j) \wedge \left(partOf(m,a) \vee partOf(m,b)\right)\right)$
Def. 5: $JointType2(j) \leftrightarrow Joint(j) \wedge \exists a,b\ hasConnectedComponent(j,a) \wedge hasConnectedComponent(j,b) \wedge \exists z\ (MaterialEntity(z) \wedge (z \neq a) \wedge (z \neq b) \wedge \forall m(partOf(m,j) \leftrightarrow \neg partOf(m,a) \wedge \neg partOf(m,b) \wedge partOf(z))$
Def. 6: $JointType3(j) \leftrightarrow Joint(j) \wedge \exists a,b\ hasConnectedComponent(j,a) \wedge hasConnectedComponent(j,b) \wedge \forall m\left(partOf(m,j) \leftrightarrow properPartOf(m,a) \vee properPartOf(m,b)\right)$

Based on the three types of joints defined in Def. 4-6, we can define five basic types of joining processes by specializing in the type of Joint which is the output of the joining process (see Def. 3). *Welding* produces *JointType1* as expressed in Def. 7. *JointType3* is produced by *Fitting* as expressed in Def. 11. Joining processes e.g., *Brazing*, *Soldering*, *Fastening*, and *Bonding*, all produce *JointType2*. For *Fastening* (Def. 9) and *Bonding* (Def. 10), all material of the joint is part of a type of *Fastener* and *Adhesive*, respectively. For both *Brazing* and *Soldering* type processes, the joint is created by some *FillerMaterial* (Def. 8). To distinguish between these two types of joining, further characterization of the underlying processes is needed. This is tacked in the next section.

Def. 7: $Welding(p) \leftrightarrow JoiningProcess(p) \wedge \forall j\ (hasOutput(p,j) \rightarrow JointType1(j))$
Def. 8: $Brazing(p) \vee Soldering(p) \leftrightarrow JoiningProcess(p) \wedge \forall j\ (hasOutput(p,j) \rightarrow JointType2(j) \wedge \forall m\ (partOf(m,j) \rightarrow \exists f\ FillerMaterial(f) \wedge partOf(m,f))$
Def. 9: $Fastening(p) \leftrightarrow JoiningProcess(p) \wedge \forall j\ (hasOutput(p,j) \rightarrow JointType2(j) \wedge \forall m\ (partOf(m,j) \rightarrow \exists f\ Fastener(f) \wedge partOf(m,f))$
Def. 10: $Bonding(p) \leftrightarrow JoiningProcess(p) \wedge \forall j\ (hasOutput(p,j) \rightarrow JointType2(j) \wedge \forall m\ (partOf(m,j) \rightarrow \exists f\ Adhesive(f) \wedge partOf(m,f))$
Def. 11: $Fitting(p) \leftrightarrow JoiningProcess(p) \wedge \forall j\ (hasOutput(p,j) \rightarrow JointType3(j))$

In some of the above and many following definitions, various types of material entities, e.g., *FillerMaterial*, *WeldingMachine*, *Fastener*, *Adhesive*, and processes, e.g., *Heating*, *Melting*, *Solidifying*, *ApplyingPressure*, are used. These classes are not defined in this paper as they are general objects and processes that may be used in many other situations than just joining processes. Therefore, their definitions need to be handled by some other ontology. For example, a fastener or an adhesive is a type of artefact that is specifically engineered for connecting components, that they bear some type of *bfo:function* (e.g., $Adhesive(x) \leftrightarrow Artifact(x) \wedge \exists f\ (FunctionOfHoldingSurfaces(x) \wedge bears(x,f))$.

3.3. Characterization of Processes Underlying Joining Operations

The joining processes described in Def. 7-11 are characterized by the relationships, that the resultant assembly holds with the input components. However, each type of process is also composed of multiple sub-processes which transform and produce the joint in its final form. This reductive perspective is necessary for modelling the joining processes more accurately and thus classifying them further into sub-categories. These subcategories are distinguished by the type of tools used, the underlying causes of the transformation, and the associated consumable resources required by these processes.

For joining processes such as welding (except spot welding), soldering, and brazing, a typical run is composed of a cyclical pattern of processes that are repeated for the entire length of the joint. This is not true for other types of joining processes, e.g., joining two wooden boards with nails. However, one may need to place more than one nail to join them properly but driving each nail is discrete and fundamentally different from a continuous process, e.g., welding. For these types of continuous joining processes, the underlying processes cannot be part of the entire run of the process, but only part of each unit pattern which is repeated. In this paper, we provide two different ways to model the process characterization, focusing mainly on handling the continuous joining processes, but in such a way that each of the methods can also be used for the discrete type of joining processes without any special adjustment. In the first method, only the minimum unique pattern of the joining process is characterized, and, in the second method, every joining process is considered as a not actual but planned process and the characterization of the actual processes is expressed as the corresponding specification (which are some types of information – namely a 'plan' - that in 'aboutness' relation with the process).

In the following formulations, thematic roles [21] of the participants (i.e., some entity that participates in a process) are used for distinguishing different types of materials involved in a process, e.g., *isPatientOf* is used for denoting a material that is affected by the process and *isInstrumentOf* to denote the participating machine or tools. Trivially, these predicates are the specializations of *bfo:participatesIn*, however, not yet incorporated by BFO or IOF-Core.

3.3.1. Method 1: Characterization Using the Unit Joining Process

We use the predicate *hasUnitProcess* to express that the transformation scheme, to be characterized for each of these types of joining processes, is for each unit pattern (minimally homomeric). Axiom 8 is constructed following the formulation of atomicity given by Varzi [22] saying that process p is composed of all $\varphi(p')$, i.e., all and only those processes (here p') that satisfy the given condition φ, such that every p' is part of p, and every unit process of p overlaps some φ-er, i.e. they also satisfy the same given condition φ.

Axiom 8: $hasUnitProcess(p,p') \rightarrow partOf(p',p) \wedge \varphi(p') \wedge \forall p'' \left(hasUnitProcess(p,p'') \rightarrow \exists x(\varphi(x) \wedge overlaps(x,p''))\right)$

Based on the nature of condition φ different types of joining processes may be characterized. It is to be noted that the predicate *hasUnitProcess* may also be applied to fastening, bonding, and fitting, which are discrete by default. In that sense, every occurrence of fastening, bonding, and fitting is composed of a single occurrence of the corresponding unit process. The condition φ for welding, brazing, and soldering is expressed with the predicates *hasWeldingMethod*, *hasBrazingMethod*, *hasSolderingMethod* etc. No generic formulation of such a primary method is given as they are extremely varied for different kinds of joining processes that are presented below.

Speaking generally of the welding process, the base metals of the input components are melted at the mating boundary by the application of a high heat source to produce a pool of molten metals (the weld pool) from both the components and sometimes filler metals. This weld pool subsequently solidifies and creates the connector as the heat source is removed. In the def. 12 the welding method is a *FusingMaterial* process, which is composed of *Heating*, *Melting* and *Solidifying* as subprocesses, executed in that order. It also includes *WeldingMachine* as the instrument for the *Heating* type process (under variable *h*), expresses that part of both base metal and the filler metal is melted (sometimes only the base metal), which is input to the welding process (under variable mt), and what is melted solidifies in an environment of some type of inert gas or slug (under variable *s*).

Def. 12: $hasWeldingMethod(w,f) \leftrightarrow Welding(w) \wedge FusingMaterial(f) \wedge \exists h \left(Heating(h) \wedge \forall i \left(hasInstrument(h,i) \rightarrow WeldingMachine(i)\right)\right) \wedge \exists mt \left(Melting(mt) \wedge \forall c \exists b \left(hasPatient(mt,c) \rightarrow hasInput(w,b) \wedge partOf(c,b) \wedge \left(MaterialComponent(b) \vee FillerMaterial(b)\right)\right)\right) \wedge \exists s \left(Solidifying(s) \wedge \forall g \left(environs(s,g) \rightarrow InertGas(g) \vee slug(g)\right) \wedge \forall c' \, hasPatient(s,c') \rightarrow hasPatient(mt,c')\right)\right) \wedge occurrentPartOf(h,w) \wedge occurrentPartOf(mt,w) \wedge occurrentPartOf(s,w) \wedge precedes(h,mt) \wedge precedes(mt,s)$

Although both the brazing and soldering process also apply heat to melt metals, unlike the welding process, no material from the input components is melted during the brazing and soldering process. Def. 13 for the brazing method is formulated in a similar way to Def. 12, except stating that only filler metals are melted by the heating process (clause under m). Also, a separate clause using the predicate *hasTargetTemperature* is added to distinguish the brazing process from the soldering process, which uses much less temperature than the brazing process. As per standard, all such processes that exceed the temperature of 450˚ are considered brazing[4]. For elucidation, the definition of the predicate *hasTargetTemperature* is given in Def. 14, which says that the target temperature is the maximum temperature of the patient, i.e., the material that is being heated, during the heating process. We do not include the definition for *hasSolderingMethod* as it will be almost the same as Def. 13 except the target temperature being less than equal to 450˚.

Def. 13: $hasBrazingMethod(b,p) \leftrightarrow Brazing(b) \wedge BrazingMethod(p) \wedge \exists h \left(Heating(h) \wedge hasTargetTemerature(h,T) \wedge T > 450^{\circ} \wedge \forall i \left(hasInstrument(h,i) \rightarrow BrazingInstrument(i)\right)\right) \wedge \exists mt \left(Melting(mt) \wedge \forall c \left(hasPatient(mt,c) \rightarrow hasInput(w,c) \wedge FillerMetal(c)\right) \wedge \exists s \left(Solidifying(s) \wedge \forall g \left(environs(s,g) \rightarrow Flux(g)\right) \wedge \forall c' \left(hasPatient(s,c') \rightarrow hasPatient(mt,c')\right)\right)\right) \wedge occurrentPartOf(h,p) \wedge occurrentPartOf(mt,p) \wedge occurrentPartOf(s,p) \wedge precedes(h,mt) \wedge precedes(mt,s)$

[4] ISO 857-2:2005(en) Welding and allied processes — Vocabulary — Part 2: Soldering and brazing processes and related terms, 3.1.2

Def. 14:$hasTargetTemperature(h, tmp) \leftrightarrow Heating(h) \wedge Temperature(tmp) \wedge \exists m \left(isPatientOf(m, h) \wedge \exists t \left(occuresIn(h, t) \wedge inheresIn(tmp, m, t)\right)\right) \wedge \forall tmp', t'(occuresIn(h, t') \wedge inheresIn(tmp', m, t') \wedge Temperature(tmp') \rightarrow tmp' < tmp)$

Underlying methods for the fastening and bonding processes center around the placing of the fastener into and applying adhesive to the components being joined. In both processes, the components need to be positioned to align the mating surfaces. However, such alignment for fastening is performed before the fastener is placed, e.g., aligning the holes for inserting nuts, whereas the adhesive is applied to the mating surfaces before the components are aligned. For Fastening, different methods may be required to secure the fastener, e.g., tightening a bolt on the nut, driving of screw, hammering a nail, and bucking or upsetting the tail of a rivet. Similarly, pressure is applied on the mating surfaces to set the bonding. In the following definitions, Def. 15 for Fastening, and Def. 16 for Bonding, no preparatory phases are considered. Such preparatory phases may require drilling holes or slots for inserting the fastener and surface preparation for bonding.

Def. 15:$hasFasteningMethod(f, p) \leftrightarrow Fastening(f) \wedge FasteningMethod(p) \wedge \exists a \left(Aligning(a) \wedge \forall c \left(hasPatient(a, c) \rightarrow MaterialComponent(c)\right)\right) \wedge \exists i \left(InsertingFastener(i) \wedge \forall x \left(hasPatient(i, x) \rightarrow Fastener(x)\right) \wedge \exists s \left(SecuringFastener(s) \wedge \forall x' \left(hasPatient(s, x') \rightarrow hasPatient(i, x')\right)\right)\right) \wedge occurrentPartOf(a, p) \wedge occurrentPartOf(i, p) \wedge occurrentPartOf(s, p) \wedge precedes(a, i) \wedge precedes(i, s)$

Def. 16:$hasBondingMethod(f, p) \leftrightarrow Bonding(f) \wedge BondingMethod(p) \wedge \exists a \left(ApplyingAdhesive(a) \wedge \forall c \left(hasPatient(a, c) \rightarrow MaterialComponent(c) \vee Adhesive(c)\right)\right) \wedge \exists i \left(Aligning(i) \wedge \forall x \left(hasPatient(i, x) \rightarrow MaterialComponent(x)\right) \wedge \exists s \left(AppyingPressure(s) \wedge \forall x' \left(hasPatient(s, x') \rightarrow hasPatient(i, x')\right)\right)\right) \wedge occurrentPartOf(a, p) \wedge occurrentPartOf(i, p) \wedge occurrentPartOf(s, p) \wedge precedes(a, i) \wedge precedes(i, s)$

Fitting is the simplest of all joining types as the special shapes of the components that fit together to create the joint are created beforehand when the components are fabricated or moulded. As expressed in Def. 17, the actual fitting process requires only aligning the components at their mating surfaces and applying pressure of varying degrees (sometimes may be insignificant) to secure the joint.

Def. 17:$hasFittingMethod(f, p) \leftrightarrow Bonding(f) \wedge FittingMethod(p) \wedge \exists a \left(Aligning(a) \wedge \forall c \left(hasPatient(a, c) \rightarrow MaterialComponent(c)\right) \wedge \exists g \left(AppyingPressure(g) \wedge \forall x' \left(hasPatient(g, x') \rightarrow hasPatient(a, x')\right)\right)\right) \wedge occurrentPartOf(a, p) \wedge occurrentPartOf(g, p) \wedge precedes(a, g)$

3.3.2. Method 2: Characterization Using the Joining Process Specification

The formulation of the unit process, described in the last section, aims to solve the joining processes by focusing on some unique pattern that is repeated. However, the unit process may not be required for characterization if the joining process is modelled as a plan for some future process but not the actual process. The benefit of modelling the joining process as a planned process is that the length of the run for which the actual process will be performed may not be needed to be considered in the plan. The repeating patterns may be considered as different actualizations of the same plan that occurs subsequently. In this way, the characterization of the joining processes becomes simpler and may provide an opportunity to capture the formalizations in less expressive languages, e.g., OWL 2.0[5].

Much of the foundational work from IOF-Core may be used for expressing the plan for processes, in which a *iof:PlanSpecification* is defined as an *cco:InformationContentEntity* which has some *iof:ActionSpecification* and *cco:ObjectSpecification* as part and *cco:prescribes* some iof:*PlannedProcess*. To exploit the *PlanSpecification* and *ActionSpecification* class for representing the characterization of joining processes, we first need to adjust def. 3 as in def. 3a., by classifying *JoiningProcess* as a *PlannedProcess* which is *prescribedBy* a *JoiningSpecification* (sub-class of *PlanSpecification*)

[5] https://www.w3.org/TR/owl2-overview/

Def. 3a: $JoiningProcess(p) \leftrightarrow PlannedProcess(p) \wedge \exists j\ (hasSpecifiedOutput(p,j) \wedge Joint(j)) \wedge \forall s\ \big(prescribedBy(p,s) \rightarrow JoiningSpecification(s)\big)$

As more than one *ActionSpecifications* can be part of the *PlanSpecification*, using them is beneficial in expressing the composing of different underlying processes, which can then be described as part of some specific type of *PlanSpecification*. In this way the *bfo:occurrentPartOf* relations in def. 13-17 are expressed as *bfo:continuantPartOf* as *InformationContentEntity*. At the same time, the actual processes are prescribed by *ActionSpecifications*, along with the precedence among the underlying processes, along with different materials and instruments are suitably modelled by the *ActionSpecifications*. Additionally, the *ObjectiveSpecification* as part of the *PlanSpecification* is used to represent the outcome of the joining processes. For example, Axiom 11 shows that *WeldingObjective* prescribes only *WeldedJoint*.

Following this way of characterizing the joining processes, the *hasWeldingMethod*, *hasBrazingMethod*, *hasSolderingMethod* etc. may be translated using *prescribedBy* (inverse of *prescribes*) which relates some sub-type of *JoiningProcess* to some suitable sub-type of *PlanSpecification*. In this paper, we only provide the model for a generic welding method represented by Def. 12 in the following expressions. The other types of joining processes can be translated similarly.

Axiom 9: $WeldingProcess(w) \rightarrow PlannedProcess(w) \wedge \forall p\ \big(prescribedBy(w,p) \rightarrow WeldingSpecification(p)\big)$
Axiom 10: $WeldingSpecification(p) \rightarrow JoiningSpecification(p) \wedge \forall f\ \big(hasContinuantPart(p,f) \rightarrow FusingMaterialActionSpecification(f)\big) \wedge \exists o\ (WeldingObjective(o) \wedge hasContinuantPart(p,o))$
Axiom 11: $ObjectiveSpecification(o) \wedge prescribes(o,j) \wedge WeldedJoint(j) \rightarrow WeldingObjectuve(p)$
Axiom 12: $FusingMaterialActionSpecification(p) \rightarrow ActionSpecification(p) \wedge \exists h\ \big(HeatingActionSpecification(h) \wedge hasContinuantPart(p,h)\big) \wedge \exists mt\ (MeltingActionSpecification(mt) \wedge hasContinuantPart(p,mt)) \wedge \exists s\ \big(SolidifyingActionSpecification(h) \wedge hasContinuantPart(p,s)\big)$
Axiom 13: $HeatingActionSpecification(p) \rightarrow ActionSpecification(p) \wedge \forall h \exists mt,s\ \big((prescribes(p,h) \rightarrow Heating(h) \wedge (Melting(mt) \wedge precedes(h,mt) \wedge (Solidifying(s) \wedge precedes(mt,s)))) \wedge \forall i\ (hasInstrument(h,i) \rightarrow WeldingMachine)\big)$
Axiom 14: $MeltingActionSpecification(p) \rightarrow ActionSpecification(p) \wedge \forall mt \exists s\ \big((prescribes(p,mt) \rightarrow Melting(mt) \wedge (Solidifying(s) \wedge precedes(mt,s))) \wedge \forall c\ \big(hasPatient(mt,c) \rightarrow MaterialComponent(c) \vee FillerMaterial(c)\big)\big)$
Axiom 15: $SolidifyingActionSpecification(p) \rightarrow ActionSpecification(p) \wedge \forall s\ \big((prescribes(p,s) \rightarrow Solidifying(s)\big) \wedge \forall g\ (environs(s,g) \rightarrow InertGas(g) \vee slug(g)) \wedge \forall c\ \big(hasPatient(s,c) \rightarrow MaterialComponent(c) \vee FillerMaterial(c)\big)\big)$

4. OWL Implementation of The Taxonomy

In this section, we describe the taxonomy of the joining process encoded in OWL reusing IOF-Core and BFO 2020, recursively. The development is modularized as ***joining***[6], importing iof-core[7] along with bfo[8] and annotation vocabulary iof-av[9] (contains additional annotation properties), as its root, which is imported by modules dedicated to the taxonomy for each basic type of joining processes, e.g., *welding*, *soldering*, *brazing*, *bonding* etc. These ontology files are available through the GitHub repository (which has not been published yet).

The taxonomy produced under each basic type of joining process is vast and therefore it is not possible to provide the characterization of each subtype in this article. What we intend to show is that the taxonomy built is purely based on inferring subsumption instead of explicit *owl:SubClassOf* declaration. The second method, presented in Section 3.3.2, is adopted in the process characterization of the joining processes due to two reasons: 1) suitability for rendering in OWL, 2) availability of many base classes (e.g., *PlannedProcess*, *PlanSpecification*, *ActionSpecification*) in IOF-Core that can be reused.

[6] http://www.semanticweb.org/joining
[7] https://purl.industrialontologies.org/ontology/core/Core/
[8] http://purl.obolibrary.org/obo/
[9] https://purl.industrialontologies.org/ontology/core/meta/AnnotationVocabulary/

The groundwork for the basic classification of joining processes, presented in Section 3.2, is difficult to be rendered in OWL, due to the limitation in expressiveness of the language. This requires some the predicates defined in Def. 1 to 6 to be declared as primitive; however, the corresponding FOL formula is inserted as annotation using iof-av. *hasConnector* is declared as an sub-property of '*bfo:has proper continuant part at all times*' (Axiom 1), which has domain as *iof:Assembly* and range '*bfo:fiat object part*' and inverse of *isConnectorOf* as a brutal simplification of Def. 1. Appropriating Def. 2, *Joint* is classified under '*bfo:fiat object part*' that *isConnectorOf* some *MechanicalAssembly* and only *MechanicalAssembly*. Following Axiom 2, *MechanicalAssembly* is declared as an iof:assembly having at least one *Joint* as connector. Being a subclass of iof:assembly, it inherits the clause that it has some 'iof:material component' as part. Following Def. 3a, *JoiningProcess* is defined as a subclass of *'manufacturing process'* which has *Joint* as specified output. It is to be noted that *JoiningProcess* is not a subclass of 'assembly process' as the former is only a part of the latter along with many other processes such as materials feeding, material preparation, and post-joining processes, such as cleaning and deburring.

Three types of Joint(s) from Def. 4-6 are declared as primitive. For making the model intelligible, *WeldedJoint* is declared as a subclass of JointType1, *FittedJoint* as a subclass of JointType3, and *BrazedJoint*, *SolderedJoint*, *BondedJoint*, and *FastenedJoint* as subclasses of JointType2. These subtypes of Joint helps in classifying the basic types of *JoiningOperations*. Table 1 only shows the basic classes for *WeldedJoint* and *WeldingOperation*. Other types of joining operations and joints follow the same patterns.

Table 1 Definitions of Classes and Object Properties in OWL Functional Syntax

Class / Object Property	Definition
joining:hasConnector	***domain**:Assembly, **range**: 'fiat object part'*
joining:Joint	*'fiat object part' and (IsConnectorOf some MechanicalAssembly) and (IsConnectorOf only MechanicalAssembly)*
weld:WeldedJoint	*Joint and 'is output of' some WeldingProcess*
joining:MechanicalAssembly	*assembly and hasConnector some Joint*
joining:JoiningProcess	*'manufacturing process' and ('has specified output' only Joint)*
weld:WeldingProcess	*'manufacturing process' and 'has specified output' only WeldedJoint and ('prescribed by' only WeldingSpecification)*
Weld:WeldingSpecification	*'plan specification' and ('has continuant part at all times' only FusingMaterialSpecification) and 'has continuant part at all times' some WeldingObjective*
weld:WeldingObjective	*'objective specification' and (prescribes some WeldedJoint) SubClassOf WeldingObjective*
braze:Brazing	*'manufacturing process' and ('has specified output' only BrazedJoint) and ('prescribed by' only BrazingSpecification) and 'has continuant part at all times' some BrazingObjective*

The taxonomy of welding processes is complex as many variations are available for this joining process. In this section, we focus on showing the generic pattern of modelling the subtypes of welding processes using Metal-Inert-Gas (MIG) welding as an example. *MIGWelding* heats the base metals using an electric arc produced by a consumable wire electrode at the mating boundary. The base metals are melted along with the electrode which provides the filler metal, all of which then solidifies to produce the joint. A continuous stream of inert gas, dispensed by the welding torch shields the joint from oxidation and contamination. *MIGWelding* is inferred as a type of *ArcWelding* because its specification is inferred as a subclass of the specification of *ArcWelding*. *ArcWelding* is inferred as a type of *FusionWelding* because its specification is inferred as a subclass of the specification of *FusionWelding*, in turn, and being subsumed by *WeldingProcess*, ultimately. The subsumptions among the *PlanSpecifications* (e.g., *MIGWeldingSpecification*, *ArcWeldingSpecification*, *FusionWeldingSpecification*) are inferred based on the subsumptions among the different *ActionSpecification*(s) that are declared as part of them. For example, *ArcWeldingSpecification* is a subclass of *FusionWeldingSpecification* as one of the action specification parts, *HeatingByArcAction*, is subsumed by *HeatingAction*. Again, these *ActionSpecification*(s) are inferred as subclasses of each other because of the different types of heating processes, along with associated patients, agents, and

instruments, that they prescribe. For example, the heating process for the *HeatingAction*, specific to the *MIGWelding* uses some *SolidWireElectrode* which is a part of *MIGWeldingTorch* as an instrument. Needless to mention that different types of melting, solidification, and shielding mechanisms are also taken into account along with the heating in inferring the subsumptions. In this way, the inference of subsumption among welding processes traces back to the details of the underlying processes, their order, and the resources engaged in them. The inference route is shown in **Figure 1**.

However, most of the welding processes are fusion type as defined in Def. 12, cold welding process does not melt the base metals but uses pressure to deform the components to make a joint. Similar to MIG welding, other welding processes are also classified based on the type of Heating process, primarily based on how the components are heated at the mating boundary. Six types of such heating strategies are available, e.g., electric arc, resistance, thermochemical, pressure, radiation (laser and electron beam), and convection, which are applied to perform *ArcWelding*, *ThermoChemicalWelding*, *SolidStateWelding*, Laser Beam Welding (*LBW*), Electron Beam Welding (*EBW*), and *ThermoplasticWelding*. In the second level, every type of welding is performed by a specially designed welding machine (*hasInstrument* as differentia as in Def. 7). In the third level, some welding operation uses filler metal while others do not.

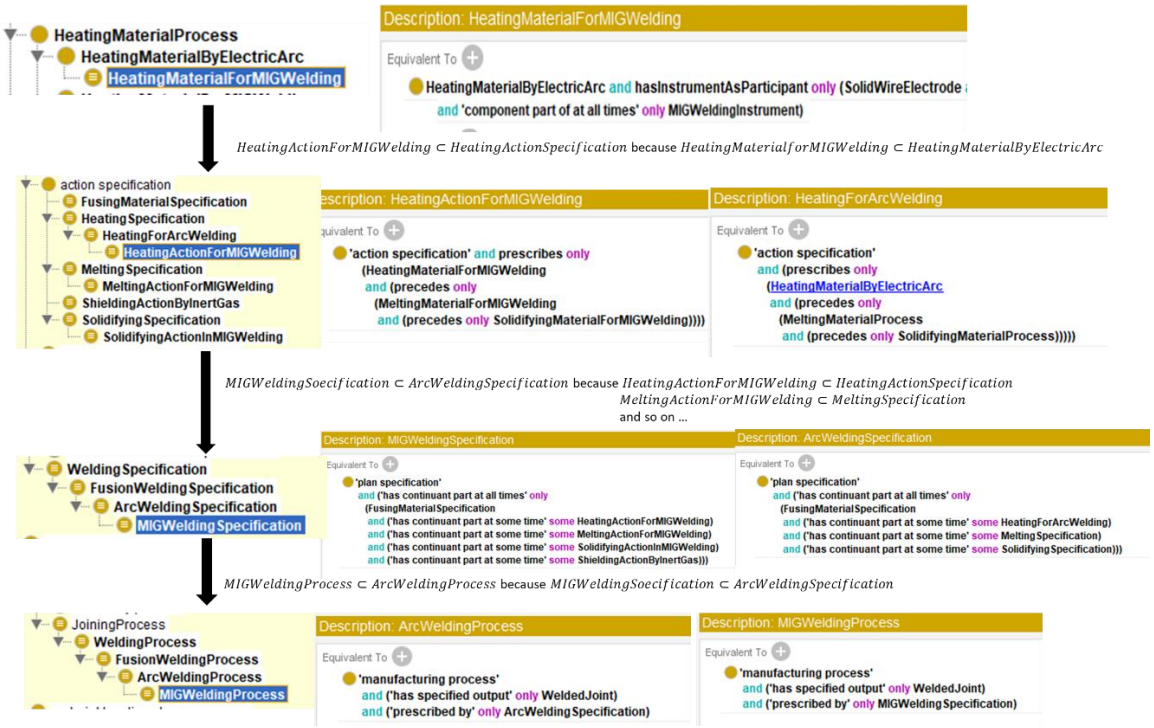

Figure 1: Taxonomic position of *MIGWelding* inferred based on underlying process specifications

Most of the sub-types of brazing and soldering processes can be classified similarly by primary methods as differentia, which are all some type of heating process, differentiated either by their source of energy, e.g., infrared, induction, resistance, or gas. Using the subtypes of primary methods, brazing can be classified into *InfraredBrazing*, *InductionBrazing*, *ResistanceBrazing*, and *GasBrazing*. Similar classification can be performed for soldering too. Another type of differentiae for both brazing and soldering is the environment in which the primary method takes place, similar to the treatment of shielding mechanism in welding. The environment for both *FurnaceBrazing* and *FurnaceSoldering* is a furnace and for both *DipBrazing* and *DipSoldering* is a bath of molten chemical or alloy covered with a suitable flux. Inference routes similar to *MIGWelding* can be derived for each subclass relationship that forms the taxonomy of brazing processes, but not given here due to the limitation of the length of this paper.

The taxonomy of both bonding and fastening may be derived only based on the type of connectors used. These connectors are subtypes of *Fastener*, which is the patient for the *InsertingFastener* process in Def. 15, and *Adhesive*, which is the patient for the *ApplyingAdhesive* process in Def. 16. Taxonomy

of various types of bonding may be defined based on the subtypes of *Adhesive*, e.g., *Anaerobic*, *Cyanoacrylate*, *Emulsion*, *EpoxyResin*, *HotMelt*, *Phenolic*, *Polyurethane*, *SolventBornRubber*, *Tape*, and *Polyimide*. Similarly, the taxonomy of fastening operations may be inferred based on subtypes of fasteners, e.g., *Rivet*, *Flange*, *Stake*, *Staple*, *Nail*, *Screw*, *Pin*, *RetainingRing*, *AnchorBolt*, and many more. Along with the types of connectors, different types of fasteners also demand different types of tools, especially some specialized instruments for *SecuringFastener* process, e.g., nailing needs a hammer. The complete taxonomy of all joining processes is given in **Figure 2**.

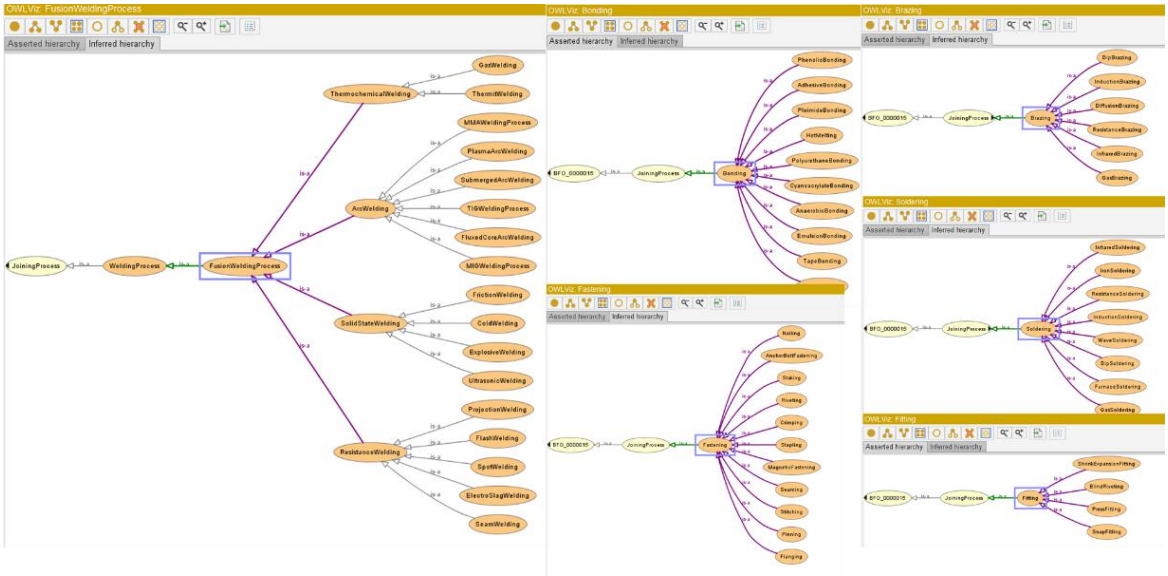

Figure 2: Complete taxonomy (inferred) of joining processes

The taxonomy of joining operations derived by reasoning may be compared with some benchmarks for validity. The taxonomy of joining operations in the handbook by Swift and Booker [23] agrees with the taxonomy in **Figure 2**, however, some detailed joining operations may be missing as our taxonomy stops at a higher level sometimes. The most important difference is that our taxonomy does not capture the distinction between permanent, semi-permanent, and non-permanent joining processes. In our model, most of the semi-permanent and non-permanent joining processes are classified as fitting and permanent joining processes as fastening.

5. Conclusion

In this paper, a set of definitions for manufacturing joining operations are provided based on foundational ontology concepts. The formulations of these definitions are derived from an in-depth analysis of the characteristics of joining operations. Finally, an OWL-based rendition of the model is built. Various types of joining operations are defined based on the model and a taxonomy is automatically derived by inference. In future efforts, a detailed account of the capabilities of the joining processes will be investigated including the geometry of the joint, materials of the input components, the thickness of the sheet they support, various costs incurred, level of operator's skill needed, and joint qualities, e.g., surface finish, fabrication tolerance, and distortion.

6. Acknowledgements

This research was partially financed by the OntoCommons project funded by the European Union's Horizon 2020 research and innovation programme under Grant Agreement no. 958371.

7. References

[1] D. Sormaz and A. Sarkar, “Distributed Integration of Design and Planning Activities in Manufacturing using Intelligent Agents,” in *TMCE*, 2014, pp. 19–23.
[2] A. Sarkar, “Semantic Agent Based Process Planning for Distributed Cloud Manufacturing.” 2020, Accessed: Dec. 16, 2020. [Online]. Available: http://rave.ohiolink.edu/etdc/view?acc_num=ohiou1578585210407386.
[3] X. Xu, “From cloud computing to cloud manufacturing,” *Robot. Comput. Integr. Manuf.*, vol. 28, no. 1, pp. 75–86, 2012, doi: 10.1016/j.rcim.2011.07.002.
[4] T. Hofweber, “Logic and ontology,” *The Stanford Encyclopedia of Philosophy*, 2021. https://plato.stanford.edu/archives/spr2021/entries/logic-ontology/.
[5] M. M. Baysal, U. Roy, R. Sudarsan, R. D. Sriram, and K. W. Lyons, “The Open Assembly Model for the Exchange of Assembly and Tolerance Information: Overview and Example,” 2004, vol. 2004, pp. 759–770, doi: 10.1115/DETC2004-57727.
[6] X. Fiorentini, I. Gambino, V.-C. Liang, S. Rachuri, M. Mani, and C. Bock, “An Ontology for Assembly Representation,” *National Institute of Standard and Technology*, 2007. .
[7] S. J. Fenves, S. Foufou, C. Bock, and R. D. Sriram, “CPM2: A Core Model for Product Data,” *J. Comput. Inf. Sci. Eng.*, vol. 8, no. 1, pp. 014501–014501, 2008, doi: 10.1115/1.2830842.
[8] K.-Y. Kim, D. G. Manley, and H. Yang, “Ontology-based assembly design and information sharing for collaborative product development,” *Comput. Des.*, vol. 38, no. 12, pp. 1233–1250, 2006, doi: 10.1016/j.cad.2006.08.004.
[9] K. Y. Kim, “Ontology and assembly joint topology representation,” *Comput. Aided. Des. Appl.*, vol. 5, no. 5, pp. 630–638, 2008, doi: 10.3722/cadaps.2008.630-638.
[10] E. Gruhier, “Spatiotemporal description and modeling of mechanical product and its assembly sequence based on mereotopology : theory, model and approach,” *Http://Www.Theses.Fr*, 2015, [Online]. Available: https://www.theses.fr/2015BELF0276.
[11] L. Solano, “Ontological modelling of welding processes,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 1193, no. 1, p. 012019, Oct. 2021, doi: 10.1088/1757-899X/1193/1/012019.
[12] S. Saha, Z. Usman, W. D. Li, S. Jones, and N. Shah, “Core domain ontology for joining processes to consolidate welding standards,” *Robot. Comput. Integr. Manuf.*, vol. 59, no. May, pp. 417–430, 2019, doi: 10.1016/j.rcim.2019.05.010.
[13] M. Molhanec, O. Zhuravskaya, E. Povolotskaya, and L. Tarba, “The ontology based FMEA of lead free soldering process,” *Proc. Int. Spring Semin. Electron. Technol.*, no. April 2014, pp. 267–273, 2011, doi: 10.1109/ISSE.2011.6053871.
[14] R. Arp, B. Smith, and A. D. Spear, “Introduction to Basic Formal Ontology II: Occurrents,” in *Building Ontologies With Basic Formal Ontology*, no. June 2017, The MIT Press, 2015, pp. 1–9.
[15] B. Smith *et al.*, “A first-order logic formalization of the industrial ontologies foundry signature using basic formal ontology,” *CEUR Workshop Proc.*, vol. 2518, no. October, 2019.
[16] B. Smith, “Mereotopology: A theory of parts and boundaries,” *Data Knowl. Eng.*, vol. 20, no. 3, pp. 287–303, 1996, doi: 10.1016/S0169-023X(96)00015-8.
[17] R. Casati and A. C. Varzi, *Parts and Places : the structures of spatial representation*. Cambridge, Mass.: MIT Press, 1999.
[18] K. G. Swift and J. D. Booker, *Manufacturing Process Selection Handbook*. Oxford: Butterworth-Heinemann, 2013.
[19] B. Lotter, “Manufacturing Assembly Handbook,” B. Lotter, Ed. Butterworth-Heinemann, 1986.
[20] F. Demoly, A. Matsokis, and D. Kiritsis, “A mereotopological product relationship description approach for assembly oriented design,” *Robot. Comput. Integr. Manuf.*, vol. 28, no. 6, pp. 681–693, Dec. 2012, doi: 10.1016/j.rcim.2012.03.003.
[21] C. Bonial, W. Corvey, M. Palmer, V. V. Petukhova, and H. Bunt, “A Hierarchical Unification of LIRICS and VerbNet Semantic Roles,” in *2011 IEEE Fifth International Conference on Semantic Computing*, Sep. 2011, pp. 483–489, doi: 10.1109/ICSC.2011.57.
[22] A. C. Varzi, “Mereology,” *The Stanford Encyclopedia of Philosophy*, 2019. .
[23] K. G. Swift and J. D. Booker, “Chapter 1 - Introduction to the Handbook,” in *Manufacturing Process Selection Handbook*, K. G. Swift and J. D. Booker, Eds. Oxford: Butterworth-Heinemann, 2013, pp. 1–20.