A First-Order Logic Formalization of the Industrial Ontologies Foundry Signature Using Basic Formal Ontology

Barry Smith[a], Farhad Ameri[b], Hyunmin Cheong[c], Dimitris Kiritsis[d], Boonserm Kulvatunyou[e], Dusan Sormaz[f], Evan Wallace[e], Chris Will[g], and J. Neil Otte[h, 1]

[a]*National Center for Ontological Research*, University *at Buffalo*

[b]*Engineering Informatics Lab, Texas State University*

[c]*Autodesk Research*

[d]*EPFL, ICT for Sustainable Manufacturing*

[e]*National Institute of Standards and Technology (NIST)*

[f]*Ohio University, Department of Industrial and Systems Engineering*

[g]*Dassault Systèmes*

[h]*Johns Hopkins University Applied Physics Laboratory*

Abstract. Basic Formal Ontology (BFO) is a top-level ontology used in hundreds of active projects in scientific and other domains. BFO has been selected to serve as top-level ontology in the Industrial Ontologies Foundry (IOF), an initiative on the part of representatives from a number of branches of the advanced manufacturing industries to create a suite of ontologies to support digital manufacturing. We here present a first draft set of axioms and definitions of an IOF upper ontology descending from BFO that is designed to capture the meanings of principal terms used in manufacturing and related areas. This set of terms can be viewed as the signature common to the modules of the IOF ontology suite.

Keywords. Basic Formal Ontology (BFO), Industrial Ontologies Foundry (IOF), advanced manufacturing industry, top-level ontology, first-order logic.

1. Introduction

Basic Formal Ontology (BFO) is a small, top-level ontology that is used in a wide range of projects focusing especially on information-driven sciences. BFO provides ontology developers with a common, tested starting point for the formulation of definitions in a way that is designed to promote interoperability. BFO has been used in this manner for some fifteen years in the domain of biomedical ontology, where it serves as the top-level ontology of the Open Biomedical Ontology (OBO) Foundry. The latter is an initiative established in the wake of the Human Genome Project to coordinate the development and use of high-quality ontologies in the domain of biology and biomedicine. To be admitted to the OBO Foundry, biomedical domain ontologies are required to conform to a series of principles relating to accessibility, quality, scientific accuracy, and consistent development.

[1] Corresponding Author: neil.otte@jhuapl.edu

In more recent years, BFO has been applied in other areas, including intelligence, defense, and security, and as a consequence of these developments, BFO is in the final stages of review to become international standard ISO/IEC: 21838-2.

In addition, BFO is being applied in a series of industrial engineering-related projects, including those documented in [4]–[17] in what follows. Following a lengthy evaluation and selection process, BFO was adopted in the spring of 2019 to be the top-level ontology of the Industrial Ontologies Foundry (IOF), an ecosystem of ontology resources designed to promote interoperability in digital manufacturing and related fields.

Like the OBO Foundry, the IOF promotes a principles-based approach to the design of ontologies. Our initial focus is on support for the manufacturing domain, and more specifically on ontologies for design, maintenance, supply chain, production, and lifecycle management. In due course, it is anticipated that the service, construction, and extraction industries will be included also.

It has been clear for some time that the task of developing a coherent set of ontologies covering the manufacturing domain will present a considerable challenge. Manufacturing is not only highly multidisciplinary, but it is affected also by concerns of manufacturing enterprises, who gain commercial benefits when their data is held in proprietary siloes. On the other hand, the increasing importance of long supply chains of outsourcing provide incentives also for the sharing of data and for the coordinated development of interoperable software, and it is to address these needs that the IOF was formed.

Table 1. List of terms provided to us by members of the IOF

IOF Top Twenty(-Six) Term List	
[1] Assembly	[14] Manufacturing resource
[2] Assembly process	[15] Manufacturing tool
[3] Business process	[16] Material resource
[4] Component	[17] Operation specification
[5] Customer	[18] Piece of equipment
[6] Design	[19] Piece of manufacturing equipment
[7] Feature description	[20] Plan
[8] Input material role	[21] Product quality
[9] Manufacturing machine	[22] Product
[10] Manufacturing process	[23] Quality specification
[11] Manufacturing process plan	[24] Task
[12] Raw material role	[25] Transport process
[13] Supplier	[26] Step

The IOF initiative was launched with a proof-of-concept project described in Kulvatunyou et al. (2018), which summarizes the goals and organizational set up of the IOF, and presents the results of an initial scope-determining experiment consisting of the selection of some twenty (in the end: twenty-six) representative terms compiled by subject-matter experts within the IOF community (see Table 1). The goal of the present

paper is to provide an early draft formal representation of the definitions of these terms and of associated axioms within the BFO framework.[2]

1.1. Universals and Defined Classes

The formalization in what follows employs standard first-order logic (FOL) notation for negation, conjunction, disjunction, material implication, biconditional implication, universal, and existential quantification, using, respectively: ¬, ∧, ∨, ⊃, ≡, ∀, ∃. Variables $t, t', \ldots$, range over temporal regions.

For the sake of readability, initial universal quantifiers are suppressed. Hence, instead of writing:

$\forall (x)$ [business-process(x) → planned-process(x)]

we write:

business-process(x) → planned-process(x)

Terms from the list in Table 1 appear in **bold**. Other terms necessary for the representation of these terms are also axiomatized and these appear in roman face. Some terms are primitive – which means that they are too basic in our vocabulary to receive definitions because there are no more basic terms which could be used to define them. Two sorts of primitive terms will appear in the formalization that follows. First are primitives, such as disposition, role, and material entity—the treatment of which may be found in the BFO 2.0 Specification and User Guide (Almeida, et al.). Other terms, such as 'artifact' and 'information content entity', are treated in the Information Artifact Ontology (IAO)[3] and in the Common Core Ontologies (CCO), a suite of mid-level ontologies conformant to BFO 2.0.[4]

1.2. Universals and Defined Classes

Some terms in what follows refer not to universals but to what BFO calls 'defined classes'. Consider, for example, the term 'lawyer'. This does not represent an extra entity instantiating a universal in its own right. Rather, it connotes that some already existing entity (some instance, in this case, of the universal *homo sapiens*) has a certain *lawyer role*. The latter *is* an extra entity, and thus BFO admits a corresponding role universal. The term 'lawyer' then represents the defined class consisting of all those entities (human beings) that have the *lawyer role*. Defined classes may also be defined disjunctively – for example, in the definition of 'agent' as 'person or organization'.

[2] The material presented here is intended for review only and is extracted from a more extensive set of formal proposals that is available at https://buffalo.box.com/v/IOF-Signature. The latter will be subject to periodic updates during June-August 2019, and critical comments are welcome.

[3] http://www.obofoundry.org/ontology/iao.html

[4] The Common Core Ontologies (CCO) are available at: https://github.com/CommonCoreOntology.

2. BFO-IOF-FOL

2.1. Section One: Taxonomy

In this section we provide a representation fraction of the *is-a* (or *subclass of*) relations in our first-order logic axiomatization based on BFO. A somewhat compressed version of the is-a hierarchy is presented in Figure 1, which corresponds to some seventy axioms in the BFO-IOF-FOL formalization. (The version here leaves out IAO and CCO terms for the same of readability. The fuller version is online includes these terms.)

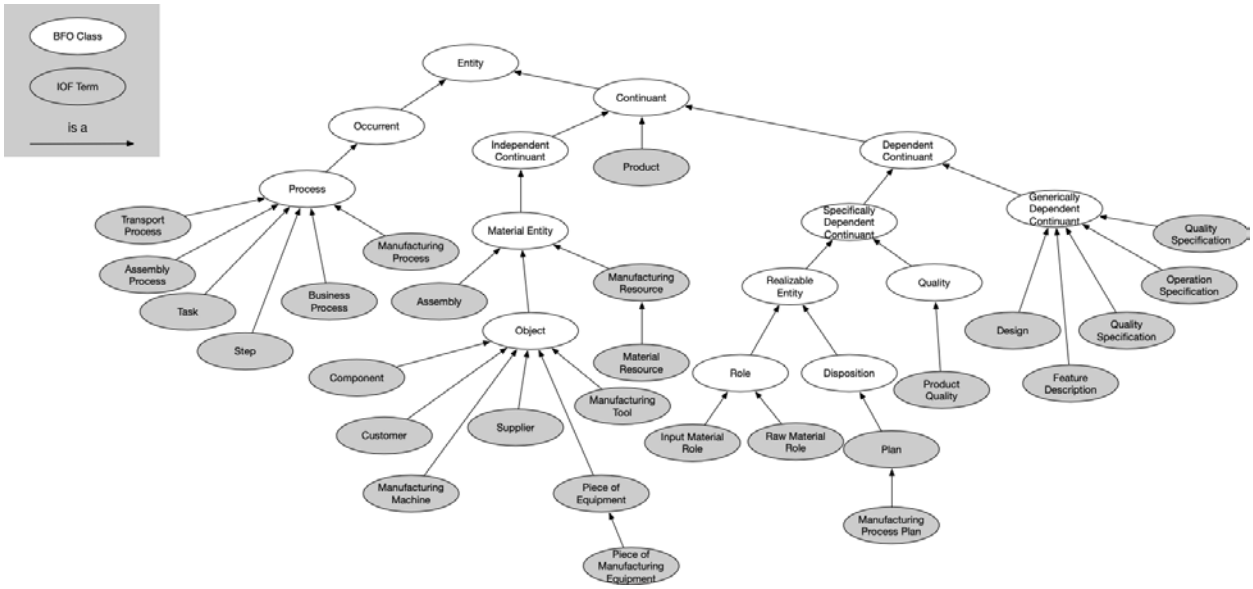

Figure 1. Fragment of the Basic Formal Ontology class hierarchy aligned with classes bearing labels from the top twenty-six terms provided by the IOF. Terms from IAO and CCO are provided in the expanded figure at https://buffalo.box.com/v/IOF-Signature

Examples of such axioms are as follows:

2.1.1. capability subclass of disposition.

capability(x) → disposition(x)

Comment: The BFO term 'disposition' is used to refer to all tendencies, powers, habits, skills, potentials, and so forth, which a material entity may possess. A capability is (simplifying somewhat) a disposition that in normal circumstances brings benefits to its bearer, user, or owner.

2.1.2. intention subclass of disposition.

intention(x) → disposition(x)

2.1.3. intention to perform subclass of intention.

intention-to-perform(x) → intention(x)

2.1.4. material artifact subclass of object.

material-artifact(x) → object(x)

Comment: The BFO term 'object' comprehends material entities possessing one or other type of causal unity. In addition to material artifacts such as laptops, objects include: solid portions of matter, such as a lump of iron, and biological entities such as cells and organisms.

2.1.5. organization subclass of object aggregate.

organization(x) → object-aggregate(x)

Comment: BFO distinguishes objects from object aggregates, where the latter include organizations and systems. An organization, from the BFO perspective, is an aggregate of persons with specified roles.

2.1.6. **product** subclass of continuant.

product(x) → continuant(x)

Comment: 'product' is a defined class combining manufactured products and information products. The latter will be addressed in a later version of this axiomatization.

2.1.7. information content entity subclass of generically dependent continuant.

information-content-entity(x) → generically-dependent-continuant(x)

Comment: Generically dependent continuants, in BFO, are entities that depend for their existence on some carrier or bearer, but in such a way that they can be transferred (for example through a process of copying) to some second carrier or bearer.

2.1.8. **material resource** subclass of manufacturing resource.

material-resource(x) → manufacturing-resource(x)

Comment: This term is here defined as meaning, not raw material, but rather resources available to the enterprise that are made of matter (thus buildings, vehicles, equipment, and so forth, as contrasted with intellectual property, software, and so forth).

2.2. Section Two: Definitions and Axioms

A definition of a term T is a statement of jointly sufficient and individually necessary conditions which an entity must satisfy if it is to be an instance of the universal or class referred to be T. In the ideal case we provide equivalent definitions of all terms both in natural language (using '=def.') and in FOL syntax (using the biconditional ' ≡ '). Primitive terms, which are marked as such, are terms for which we can provide only necessary (but not sufficient) conditions (labelled using '→'). In some cases we can provide English language definitions but not equivalent FOL defintions (for instance because FOL does not include the resources to capture possibility or necessity).

2.2.1. product =def. continuant that has a product role.

product(x, t) =def. instance-of(x, continuant, t) and $\exists r$ product-role(r) &
(has-role(x, r, t)

Comment 1: This definition leaves open the possibility that generically dependent continuants may be products.

Comment 2: The BFO term 'role' is used extensively in this formalization to do justice to those cases where general terms are *phase sortals*, which means they hold only for certain phases in the existence of the relevant entities. Thus, for example, a given material entity may be correctly describable as a *prototype* at one phase in its existence and as a *product* in a later phase.

2.2.2. has-agent: a primitive relation between a process and an agent.

x has-agent $y \rightarrow$ (instance-of(x, agent, t) & instance-of(y, process, t))

Comment: The inverse of has-agent is agent-in.

2.2.3. action =def. process that has an agent.

$\exists$t (instance-of(x, action, t)) $\equiv$ $\exists$y, t (instance-of(y, agent, t) & has-agent (x, y, t))

2.2.4. Only agents have intentions.

intention(z, y, t) $\rightarrow$ instance-of(z, agent, t)

2.2.5. Every action realizes some intention that precedes it or starts simultaneously with it.

instance-of(x, action, t) $\rightarrow$ $\exists$y, t'(instance-of(y, intention, t') & realizes (x, y) &
(precedes(t', t) $\vee$ first-instant(t', t)))

2.2.6. agent =def. person or organization.

instance-of(x, agent, t) $\equiv$ (instance-of(x, person, t) $\vee$ instance-of(x, organization, t)

2.2.7. Every agent bears some intention at some time.

$\exists t$ (instance-of(x, agent, t)) $\rightarrow$ $\exists$y, t'(instance-of(y, intention, t') & bears(x, y))

Comment: Every agent has the capability to bear intentions.

2.2.8. uses-in: a primitive relation between an agent, an entity, and a process.

uses-in (x, y, z) $\rightarrow$ agent(x) & entity(y) & process(z)

2.2.9. act of artifact employment =def. process in which an agent uses some artifact.

act-of-artifact-employment(x) $\equiv$ process(x) & $\exists y \exists z$(artifact(y) & agent(z) &
uses-in(z, y, x))

2.2.10. Every **manufacturing tool** bears a function that, if realized, is realized in a manufacturing process.

manufacturing tool(x) ≡ material-entity(x) & ∃f(function(f) & has-function(x, f) & ∀y(realizes(y, f) → manufacturing process (y)))

2.2.11. **planned process** =def. process that occurs as the result of one or more intentions to realize a plan and where the process successfully realizes that plan.

instance-of(x, planned process, t) ≡ instance-of(x, process, t) & ∃y(instance-of(y, plan, t) & realizes(x, y))

Comment 1: This definition implies that every planned process is an actual process (though not vice versa). Thus, 'planned' is here functioning as a specifier (rather than as a modifier analogous to 'cancelled' or 'averted'). Therefore, to say that a process is planned is not to say that it has not yet taken place; rather, it is to say that it is (was or will have been) protocol-driven, instruction-driven, command-driven, or software-driven (or some combination thereof). A planned process remains a planned process even after it has occurred. Hence, the contrast is with *accidental* processes or with processes that did not turn out as planned (e.g. because a fire broke out in the engine room) or with incidental processes not part of the realization of a plan (such as, the lathe operator smoking a cigarette while operating her lathe).

Comment 2: 'Planned' means 'protocol driven'. Protocols may be written, spoken, or simply thought – as when upon waking up, we plan, for instance, what to eat for breakfast.

2.2.12. manufacturing enterprise =def. an organization whose function is to engage in manufacturing processes.

manufacturing-enterprise(x) ≡ organization(x) & ∃f(function(f) & has-function(x, f)) & ∀y(realizes(y, f) → manufacturing enterprise process (y))

2.2.13. manufacturing enterprise process =def. planned process that has a manufacturing enterprise (including an employee of such an enterprise) as agent that supports the manufacturing processes of the enterprise.

Comment: manufacturing enterprise processes include manufacturing processes, maintenance processes, transport processes, design processes, business processes, and so on.

2.2.14. manufacturing resource role =def. role that inheres in an independent continuant where that continuant bears a disposition that, if realized, is realized in a manufacturing enterprise process.

instance-of(x, manufacturing-resource-role, t) ≡ ∃y(instance-of(y, continuant, t) & has-role(y, $\underline{x}$, t) & ∃d(instance-of(d, disposition, t) & has-disposition(y, d, t)) & ∀p((process(p) & realizes(p, d)) → manufacturing-enterprise-process(p)))

2.2.15. **manufacturing resource** =def. continuant that bears a manufacturing resource role.

instance-of(*x*, manufacturing resource, *t*) ≡ ∃y(manufacturing resource role(*y*) & has-role(*x*, *y*, *t*))

2.2.16. **material resource** =def. manufacturing resource that is a material entity.

instance-of(*x*, material-resource, *t*) ≡ instance-of(*x*, manufacturing-resource, *t*) & instance-of(*x*, material-entity, *t*)

2.2.17. **input material role** =def. role that inheres in a material entity that has the capability to serve as a specified input of a manufacturing process.

instance-of(*x*, input-material-role, *t*) ≡ role(*x*) & ∃*y*,*w*,*z*(instance-of(*y*, material-entity, *t*) & has-role(*y*, *x*, *t*) & has-capability(*y*, *w*) & ∀z(realizes(*z*, *w*) → & manufacturing-process(*z*) & has-specified-input (*z*, *y*)))

2.2.18. portion of input material =def. material entity that has an input material role.

instance-of(*x*, portion-of-input-material, *t*) ≡ ∃y(instance-of(*y*, input-material-role, *t*) & has-role(*x*, *y*, *t*))

raw material role =def. role that inheres in a material entity that is a specified input of a manufacturing process and is not the output of any manufacturing process.

instance-of(*x*, raw-material-role, *t*) ≡ role(*x*) & ∃*y*(instance-of(*y*, material-entity, *t*) & has-role(*y*, *x*, *t*) & ¬ ∃*z*(manufacturing-process(*z*) & has-specified-input(*z*, *y*))

2.2.19. **piece of equipment**: primitive term that refers to a material artifact that is used in an operation or activity.

Comment**:** 'Equipment' is a synonym of '**piece of equipment**'.

2.2.20. **piece of manufacturing equipment** =def. piece of equipment that bears a function where any process that realizes that function is a manufacturing process.

piece of manufacturing equipment(*x*) ≡ piece of equipment(*x*) & ∃*f*(has-function(*x*, *f*) & ∀p(process(*p*) & realizes(*p*, *f*) → manufacturing process(p))

2.2.21. member-of-organization =def. member of an organization and the organization itself.

member-of-organization(*x*, *y*) ≡ organization(*y*) & member-part-of(*x*, *y*) & (person(*x*) ∨ organization(*x*))

Comment 1: a division within an enterprise is an example of a member part of an organization that is also an organization.

Comment 2: the ISO formalization of BFO is consistent with a scenario under which an organization loses all but one of its member parts, for example because all the members

of a fire brigade retire before replacements can be recruited. Every organization, however, is then such that it contained multiple member parts at an earlier stage.

2.2.22. supplier role =def. role inhering in an agent that, if realized, is realized in some act of selling.

supplier role(x) ≡ ∃y(agent(y) & has-role(y, x) & ∀p((process(p) & realizes(y, p)) → act-of-selling(p))

2.2.23. **supplier** =def. agent who bears a supplier role.

supplier(x) ≡ ∃y(supplier role(y) & has-role(y, x))

Comment: 'provider' is a synonym of 'supplier'.

2.2.24. customer role =def. role inhering in an agent and realized in an act of purchasing, and which comes into exist at the point in time when a purchasing act has been initiated through to completion.

instance-of(x, customer-role, t) ≡ ∃y, z(agent(y, t) & has-role(x, y, t) & ∃w(instance-of(w, act-of-purchasing, t) & agent-in(y, w, t)))

2.2.25. **customer** =def. agent who bears some customer role.

customer(x) ≡ ∃y(customer role(y) & has-role(x, y))

2.2.26. prospective customer =def. agent capable of performing an act of purchasing.

prospective-customer(x) ≡ ∃y(capability(y) & has-capability(x, y) & ∀p(realizes(y, p) → act-of-purchasing(p))

2.2.27. has-specified-output: a primitive relation between a planned process and an entity where the entity satisfies the process endpoint specification in the plan specification.

2.2.28. manufactured product =def. material artifact that is the specified output of a manufacturing process.

manufactured product(x) ≡ product(x) & ∃p(manufacturing process(p) & has-specified-output(p, x))

2.2.29. **business process** =def. planned process that realizes a business function of an organization.

business-process(x) → planned-process(x) & ∃y∃f(organization(y) & has-function(y, f) & realizes(x, f))

Comment: business functions are functions realized in the financial processes of an enterprise or non-profit organization.

2.2.30. **manufacturing process** =def. planned process that is an occurrent part of a product production process in which one or more material entities that will be part of a manufactured product are modified.

manufacturing-process(x) → ∃y(product-production-process(y) & occurrent-part(x, y))

2.2.31. product production process =def. planned process that has specified output some product, where the product did not exist prior to the planned process.

product-production-process(x) ≡ ∃y(continuant(y) & has-output(x, y) &
∃t∃t'(exists-at(y, t) & precedes(t', t) & ¬exists at(y, t')

Comment: a production process is distinct from a maintenance process in that, in the latter case, the product exists both before and after the process occurs.

2.2.32. product role =def. role inhering in an entity that is the specified output of a product production process.

Comment 1: naturally found entities such as seashells are not products, according to this definition; they become products only if, for example, they are packaged in a certain way.

Comment 2: Parcels of real estate, according to this definition, are excluded from the realm of products.

2.2.33. measured product quality =def. specifically dependent continuant that inheres in some product and is measured by some measurement information content entity.

measured-product-quality(x) ≡ specifically-dependent-continuant(x) & ∃y(product(y) & inheres-in(x, y)) & ∃z(measurement-result(z) & is-measured-by(x, z))

2.2.34. maintenance process (primitive): planned process that has the same manufactured product as both specified input and specified output.

maintenance process(x) → planned process(x) & ∃y(manufactured product(y) &
has-specified-input(x, y) & has-specified-output(x, y)

2.2.35. **transport process** (primitive): planned process in which a material entity is moved from one site to another.

transport process(x) → planned-process(x) & ∃y∃t∃t'∃s∃s'(material entity(y) &
occupies(x, t) & occupies(x, t') & occupies(y, s, t) & occupies(y, s', t')) &
earlier-than(t, t')

2.2.36. requirement (primitive): an entity that is specified in a requirement specification

Comment: This is a defined class (almost certainly to be defined by enumeration).

2.2.37. product requirement: requirement whose specification specifies a product

2.2.38. **design** (primitive): directive information content entity that has product requirements as parts.

design(x) → directive information content entity(x) & ∃x_1, x_2, …, x_n∀i(product requirement(x_i) & part-of(x_i, x)))

Comment 1: Prescribes is a primitive relation. A prescribes B means: A is some information content entity that tells us how the world has to be for it to conform to A. For example, a command prescribes how you should behave in order to conform to the command. A quality specification prescribes how a product has to be in order to conform to the quality specification.

Comment 2: 'Prescribes can be understood in terms of Searle's idea of a mind to world direction of fit – where prescribing occurs there is a portion of reality that involves something like an attempt by an agent to make the world fit what the agent intends. World to mind direction of fit occurs where a mind tries to make itself fit – for example, in its assertions or beliefs – the world [19].

2.2.39. feature: characteristic of a continuant or a process (primitive)

Comment: "feature" (like "characteristic") is an umbrella term including in its coverage domain: qualities, parts of a material product (for example, chromium plating), a hole within a material product (for example, a button hole), as well as information entities, as well as metalevel characteristics such as availability, reliability, average dimensions, as well as characteristics of processes such as rate, continuity, and so forth.

2.2.40. **feature description** =def. descriptive information content entity that describes some feature.

2.2.41. specification =def. directive information content entity that prescribes some part or feature or some outcome of a planned process.

2.2.42. specifies: primitive relation between a specification and that which it species.

specifies(x, y) ≡ specification (x) & prescribes(x, y)

Examples: action specification, objective specification, plan specification, quality specification, requirement specification.

2.2.43. **quality specification** =def. specification that prescribes some quality that an object must have.

quality-specification(x) ≡ specification(x) & ∃Q(specifies(x, Q) & ∀q(instance-of(q, Q) → quality(q)))

Comment: A quality specification specifies a quality as something that is to come into existence as a result of a planned process.

2.2.44. action specification =def. specification that prescribes a type of action

2.2.45. objective specification =def. specification that specifies an intended process endpoint.

Comment: when an objective specification is part of a plan specification, the concretization of the latter is realized in a planned process in which the bearer tries to effect the world so that the process endpoint is achieved. This is another example of mind to world direction of fit.

2.2.46. plan specification =def. specification with action specifications and objective specifications as parts.

Comment: When concretized, may be realized in a process performed by some agent to achieve the prescribed process endpoints by taking the prescribed actions.

2.2.47. **plan** =def. intention-to-perform processes on the part of an agent as prescribed by a plan specification.

Comment 1: if the agent of a plan =def. person-aggregate (for example an enterprise, a team), then the intentions will be intentions of the corresponding persons. Where persons of different levels of authority are involved, the subplans will be correspondingly nested.

Comment 2: A plan is a concretization of a plan specification, namely the concretization in which the intention-to-perform (disposition) is first established.

2.2.48. For every plan there is some plan specification that precedes the plan.

instance-of(x, plan, t) → ∃y, t'(instance-of(y, plan-specification, t') & (precedes(t', t))

Comment: the plan specification may be written down, or inferred on the fly by a human agent from what is written down, or it may exist as a spoken command that is communicated to those required to realize it.

2.2.49. production plan specification =def. plan specification that prescribes a manufacturing process.

2.2.50. **production plan** =def. plan that is specified by a manufacturing process plan specification.

Comment: **'manufacturing process plan'** is a synonym of 'production plan'.

2.2.51. **operation specification** =def. specification that specifies some manufacturing process and the resources required to perform the process by specifying some ordered sequence of steps

2.2.52. **step** =def. action that is specified by a corresponding part of an operation specification.

instance-of(x, step, t) → instance-of(x, action, t) & ∃y(operation-specification(y) & prescribes(y, x))

Comment: A step is a minimal action as specified in a plan.

2.2.53. fixture =def. piece of equipment used to locate or support material, workpieces, or tools during machining operations.

2.2.54. **assembly process** =def. planned process of joining two or more objects together to form a new object.

instance-of(*x*, assembly-process, *t*) → planned-process(*x*) & ∃*y*, *z*(object-aggregate(*y*) & has-specified-input(*x*, *y*) & object(*z*) has-specified-output(*x*, *z*)) & exists-at(*z*, *t'*)

∀*t'* (exists-at (*z*, *t'*) → *t* = *t'*) & immediately-precedes(*t*, *t'*)

Comment 1: Simplifying somewhat, an assembly process has two kinds of specified outputs – objects (which are not finished artifacts but work-pieces) and material artifacts (which are end-products).

Comment 2: A subassembly is a kind of workpiece that is a partially finished material entity.

2.2.55. **assembly process** =def. planned process whose plan specification specifies an endpoint that includes a completed assembly as output.

2.2.56. completed assembly =def. specified output of an assembly process that achieves its objective.

instance-of(*x*, completed-assembly, *t*) ≡ instance-of(*x*, material-artifact, *t*) & ∃*y*(instance-of(*y*, assembly-process, *t*) & has-specified-output(*y*, *x*) & ∀*z*(has-specified-output(*x*, *z*) → *z* part-of *x*))

2.2.57. subassembly =def. assembly that is not a completed assembly, and which is intended to become part of a further assembly.

∃*x*(instance-of(*x*, subassembly, *t*) → instance-of(*x*, assembly, *t*) & ¬instance-of(*x*, completed-assembly, *t*)

2.2.58. component role =def. role inhering in a material artifact that is designed to be a proper continuant part of some material artifact.

2.2.59. **component** =def. material artifact that bears a component role.

instance-of(*x*.component, *t*) ≡ instance-of(*x*, material-artifact, *t*) & ∃y(instance-of(*y*, component-role, *t*) & has-role(*x*, *y*))

Comment: All components exist initially as self-standing products, since they have to be manufactured.

2.2.60. machine =def. material artifact that has a mechanical system as part.

2.2.61. mechanical system =def. engineered system that realizes its function through the use of power to apply forces.

2.2.62. system =def. object aggregate whose member parts are causally integrated.

Examples: solar system, digestive system, forest ecosystem, hydraulic system, subway system.

2.2.63. manufactured system =def. engineered system that is the specified output of a manufacturing process.

Comment 1: a manufactured system is not a manufactured product, since all manufactured products are objects and all systems are object aggregates.

Comment 2: 'engineered system' is broader than 'manufactured system' – the Milan subway is an engineered system but in addition to manufactured parts it includes also personnel and real estate.

2.2.64. **manufacturing machine** =def. machine whose function is realized in a manufacturing process.

instance-of(x, manufacturing-machine, t) ≡ instance-of(x, machine, t) &
∃y(has-function(x, y) & ∀z(realizes(f, z) → manufacturing-process(z)))

2.2.65. task specification =def. specification that is part of a process plan and specifies one or more actions.

2.2.66. **task** is an action or unified action complex that is specified in a task specification.

References

[1] Almeida, M. et al. *Basic Formal Ontology 2.0: Specification and User's Guide*. Published June 26, 2015. Available at: https://github.com/BFO-ontology/BFO/raw/master/docs/bfo2-reference/BFO2-Reference.pdf. [Online; accessed 03-Mar-2019]

[2] Arp, R., Smith, B. & Spear, A. D. (2015). *Building Ontologies with Basic Formal Ontology*, Cambridge, MA: MIT Press.

[3] Munn, K. & Smith, B. (eds.). 2008. *Applied Ontology: An Introduction*, Frankfurt/Lancaster: ontos/Walter de Gruyter.

[4] Otte, J. Neil, Dimitris K, Munira Mohd A, Yang, R, Zhang B, Rudnicki R, Rahul R, Smith B). An Ontological Approach to Representing the Product Life Cycle. *Applied Ontology* (2019. 14 (2):1-19.

[5] Smith, B. et al. (2007). The OBO Foundry: Coordinated Evolution of Ontologies to Support Biomedical Data Integration. *Nature Biotechnology* 25.11: 1251.

[6] Rudnicki, R. (2016). *Modeling information with the Common Core Ontologies*. Available at: https://github.com/ CommonCoreOntology. [Online; accessed 25 April 2019].

[7] Kulvatunyou B, Wallace E, Kiritsis D, Smith B, Will C. (2018) The Industrial Ontologies Foundry proof-of-concept project, *APMS: Advances in production management systems. Smart manufacturing for industry 4.0* (IFIP International

conference on advances in production management systems, 536). Springer. p. 402-409.

[8] Furini F, Rai R, Smith B, Colombo G & Krovi V. (2016) Development of a Manufacturing Ontology for Functionally Graded Materials. *Proceedings of International Design Engineering Technical Conferences & Computers and Information in Engineering Conference* (IDETC/CIE), Charlotte, North Carolina, USA.

[9] Suwelack S, Markus Stoll M, Serf M, Bursac N, Albers A, Bendl R, Dillmann R & Speidel S (2016). Towards Cognitive Computer Aided Engineering, *Proceedings of the International Association for the Engineering Modeling, Analysis and Simulation Community*, NAFEMS Americas Conference, June 2016.

[10] Correia A, Stokic D, Siafaka R, Scholze S. (2017) Ontology for Collaborative Development of Product Service Systems based on Basic Formal Ontology, *International Conference on Engineering, Technology and Innovation* (ICE/ITMC).

[11] Hagedorn TJ, Krishnamurty S & Grosse IR (2018) A Knowledge-Based Method for Innovative Design for Additive Manufacturing Supported by Modular Ontologies], *Journal of Computing and Information Science in Engineering*, 18(2).

[12] Bone M, Blackburn M, Kruse B, Dzielski J, Hagedorn T & Grosse IR (2018) Toward an Interoperability and Integration Framework to Enable Digital Thread, *Systems*, 6 (4), 46.

[13] Mesmer L & Olewnik A, Enabling Supplier Discovery Through a Part-Focused Manufacturing Process Ontology], *International Journal of Computer Integrated Manufacturing*, 31(1), 87-100.

[14] Arena D, Ameri F, Kiritsis D, Skill Modelling for Digital Factories (2018) *IFIP International Conference on Advances in Production Management Systems*: *Advances in Production Management Systems. Smart Manufacturing for Industry 4.0* (APMS 2018), 318-326.

[15] Ali MM, Rai R, Otte JN & Smith B (2019_ A Product Life Cycle Ontology for Additive Manufacturing, *Computers in Industry* 105, 191-203.

[16] Otte JN, Kiritsis D, Ali MM, Yang R, Zhang B, Rudnicki R, Rai R, Smith B (2019) An Ontological Approach to Representing the Product Life Cycle, *Applied Ontology*, 14 (2), 155-177.

[17] Karray H, Ameri F, Hodkiewicz M, Louge T (2019) ROMAIN: Towards a BFO compliant reference ontology for industrial maintenance, *Applied Ontology*, 14 (2), 179-197.

[18] Hagedorn TJ, Smith B, Krishnamurty S, Grosse IR (2019) Interoperability of disparate engineering domain ontologies using Basic Formal Ontology, *Journal of Engineering Design*, June 2019, https://doi.org/10.1080/09544828.2019.1630805.

[19] Searle, J.R. (1985) *Expression and Meaning: Studies in the Theory of Speech Acts*, Cambridge University Press, Cambridge.