

An Application of Ontology Construction *

Pim Borst † Jan Benjamin § Bob Wielinga § Hans Akkermans ‡

‡University of Twente
Information Systems Department INF/IS
P.O. Box 217, NL-7500 AE, The Netherlands
E-mail: {borst, akkerman}@cs.utwente.nl

§University of Amsterdam, Social Science Informatics
Roetersstraat 15, NL-1018 WB Amsterdam, The Netherlands
E-mail: {benjamin, wielinga}@swi.psy.uva.nl

Abstract

An important recent idea to facilitate knowledge sharing is to provide libraries of reusable components (models, ontologies) to end users. However, when libraries become large, finding the right library components is a knowledge demanding task in itself. Our suggestion therefore is that methods will be needed that help the user to gradually *construct* such knowledge. This paper describes a framework how to do this for reasoning in technical domains. We then show how an ontology can be incrementally constructed with our framework, for the domain of physical systems. We will see that *mapping ontologies*, ontologies that define interrelationships between other ontologies, play an important role in this construction process.

1 Introduction

An important recent idea to support knowledge acquisition is to provide libraries of reusable knowledge components (ontologies). We find such an approach in very diverse projects such as CommonKADS (expertise models [15]), the ARPA Knowledge Sharing Initiative (sharable ontologies [6]), GAMES (medical ontologies [13]), OLMECO (mechatronic design components [1]), and KACTUS (technical domain ontologies [14]).

Although we believe that libraries of reusable components will prove to be a viable and highly useful approach to effective knowledge engineering, new problems arise with the advent of such libraries: their sheer size will cause problems related to structuring, retrieval, applicability and maintenance of the knowledge archived. A library will most probably contain several ontologies related to the same part of the world. Each of these ontologies may represent a different viewpoint, a different level of abstraction or may concern different aspects of the same object or phenomenon. For example, the Ontolingua library of ontologies contains an extensive set of concepts related to thermal systems, but it is difficult to decide which of these concepts are relevant in a particular context. The problem is that deciding which ontologies are relevant to solve a particular problem, requires knowledge of the relevance of certain distinctions that may only become available during the knowledge acquisition process itself.

*This work has been supported in part by the Commission of the European Communities as Esprit-III projects P6521 OLMECO (Open Library for Models of MEchatronic COmponents), and P8145 KACTUS (modelling Knowledge About Complex Technical systems for multiple USE). The partners in the OLMECO project are PSA Peugeot-Citroën (F), BIM (B), FAGOR (Sp), Ikerlan (Sp), Imagine (F), University of Twente (NL) and ECN (NL). The partners in the KACTUS project are: LABEIN (Sp), Lloyd's Register (UK), Statoil (N), CAP Programmatore (S), University of Amsterdam (NL), University of Karlsruhe (D), IBERDROLA (Sp), DELOS (I), FINCANTIERI (I) and SINTEF (N). This article expresses the opinions of the authors and not necessarily those of the consortia.

Hence, when libraries become large, finding the right library components is a knowledge-intensive task in itself. Our suggestion therefore is that methods will be needed that help the user to incrementally *construct* such knowledge: by starting from relatively commonsense information, and expanding this into the right technical concepts and choices, through the step by step addition of new ontological distinctions. This is the approach that we will elaborate in this paper. We describe a generic framework how to construct ontologies for reasoning in technical domains in an incremental manner. We then show how an application ontology can be constructed with our framework for the domain of physical systems.

We begin with an intuitive explanation of our example technical domain, viz., that of physical systems, and sketch what kind of considerations and (multiple) ontologies play a role in gaining a step-by-step understanding of the functioning of such engineering systems (Sec. 2). The subsequent sections discuss various aspects of knowledge and ontology construction in technical domains. Sec. 3 outlines our general framework for ontologies and stepwise ontology construction through mapping rules and other construction operators. This framework is then applied to the example ontology in Sec. 4, showing how application ontologies can be stepwise constructed. A concluding discussion is given in Sec. 5.

2 An example: the PHYSSYS ontology

As an example to demonstrate the principles of ontology construction the PHYSSYS ontology for physical systems will be used. PHYSSYS is a formal ontology based upon system dynamics theory as practiced in engineering modelling, simulation and design. It formed the basis for the OLMECO library, a model component library for physical systems like heating systems, automotive systems and machine tools. The ontology expresses different conceptual viewpoints upon a physical system. To demonstrate what these viewpoints are, we carry out the small exercise of determining the knowledge that is required to understand the formula $F = ma$.

Many people will immediately associate $F = ma$ with Newton's law stating that force is the product of mass and acceleration. But this is a highly non-trivial association, because it can only be made by invoking a lot of background knowledge. First, we have to know that we are dealing here with a mathematical expression, and we have to understand the related concepts of equations, parameters and variables. However, this is far from enough: the intended meaning of $F = ma$ may now still be that electrical voltage is the product of resistance and current (which, instead, many people would call Ohm's law but typically write as $V = IR$). To distinguish between such possible interpretations we need more knowledge about, for example, the concept of physical dimensions of variables. To capture the intended meaning of $F = ma$ in the context of its *use in problem solving*, we have to additionally invoke a significant body of expert knowledge. For example, we have to understand that in this context of problem-solving use, physical objects are abstracted to and parameterized in terms of a concept called 'mass', that this mass acts as a kind of storage place for movement, that this movement does not change when the mass is undisturbed, and that it does change under certain external influences and circumstances which are abstracted to and parameterized in terms of a concept called 'force', and so on. That is, in specifying intended meaning we unavoidably have jumped into a background body of specialist knowledge known as classical mechanics.

Accordingly, it is clear that three conceptual viewpoints on physical systems can be distinguished: (i) system layout, (ii) physical processes underlying behaviour and (iii) descriptive mathematical relations. The PHYSSYS ontology consists of three engineering ontologies formalizing these viewpoints. The interdependencies between these ontologies are formalized as ontology mappings. Furthermore, the concepts in each viewpoint can also be partitioned in smaller ontologies, some of which are very generic. Thus, a view is also gradually constructed from smaller parts. Before the way this has been done will be explained, the principles of ontology construction will be discussed in the next section.

3 General framework for ontologies

This section discusses our general framework for stepwise ontology construction in technical domains. We will show how it applies to constructing the ontology for physical systems discussed in Sec. 2. A central feature of our dynamic knowledge construction process is the idea that new ontologies can be generated by mapping a number of different general ontologies. Two types of ontologies are distinguished. These are primary ontologies, which model knowledge objects, and secondary ontologies, which describe properties, dimensions or theories concerning primary objects. This section describes the general structure of ontologies by discussing the two types of conceptualizations and ontology construction principles.

3.1 Dynamic knowledge construction

Reusability of knowledge is an important issue in recent knowledge engineering. Instead of recycling off the shelf knowledge components however, we propose to *construct* a knowledge base dynamically. Our dynamic knowledge construction approach lets the user fill in a framework of ontologies. This framework can describe technical devices (for instance) out of different viewpoints. It explicates the conceptual choices made, or in other words it represents the ontological commitments. Hence, previously modelled knowledge is reused, not by using application specific models, but by combining more general components of knowledge.

Although the first approach seems far more easy to follow, it does not suffice completely. A reusable model or ontology can contain some knowledge objects that are present in several domains, but it can not represent all objects in a particular (medical or technical) domain. According to Van Heijst [12], (medical) subdomains have domain-specific concepts that are often specializations of the basic (medical) concepts. In many application domains there are additional ontological distinctions that are specific for that domain. So, a method must be available to differentiate the general toplevel categories, according to the demands of a certain domain.

We propose a dynamic knowledge construction approach. It creates a framework by following a stepwise scenario. At every step an ontological commitment is added. The scenario starts with an informal description of a domain, which is ‘translated’ into a reusable model. This model is adapted according to the demands of the specific domain, for example by applying theories or adding dimensions.

3.2 Building blocks

Two types of building blocks constitute the general framework. These are primary knowledge objects, which model relevant elements in a certain domain, and secondary distinctions, which can be applied to the primary objects dynamically. The primary knowledge objects as well as the distinctions are modelled as *generic ontologies*. A generic ontology is a general, meta-level viewpoint on a set of domain theories [14].

The distinction between primary and secondary objects is extensively studied in philosophy. Sowa [11] gives an overview of the ideas about Firstness, Secondness as well as Thirdness. The philosopher Peirce was the first to actually mention the terms, being influenced by Kant and Aristotle. Peirce distinguishes Firstness, describing the object itself, Secondness, representing the role of the object, and Thirdness, the context. A woman is an example of Firstness, the roles mother or attorney are examples of Secondness and motherhood or the legal system are Third. At the moment we only distinguish primary and secondary objects. Whether we need a notion of context as a separate ontology is an issue for further research. The philosophical notion of Secondness and our secondary ontologies differ in the sense that the latter ones not only model simple distinctions like animate or inanimate (which Aristotle calls *differentiae*), but also complex theories like topology.

Guarino *et al.* describe in [8] a distinction that appears to correspond to the difference between primary and secondary concepts. According to Guarino, a sortal predicate (like *apple*) ‘supplies

a principle for distinguishing and counting individual particulars which it collects', while a non-sortal predicate (like *red*) 'supplies such a principle only for particulars already distinguished, or distinguishable, in accordance with some antecedent principle or method'.

The difference between sortal and non-sortal predicates or primary and secondary concepts is intuitively clear. However, there remains a choice as which type of ontology a certain predicate is represented. It is possible to *reify* secondary concepts into primary concepts (e.g. *redness*). The type of domain and application determine which concepts are primary and which secondary.

Primary ontologies Primary ontologies represent distinguished parts of the (concrete or abstract) world. They model general viewpoints on a domain, such as physical, functional or behavioural aspects of a domain. Primary ontologies consist of concepts and relations that are considered important and relevant items concerning a technical domain. Three examples of primary ontologies are presented:

- The *component ontology* describes physical items in the world. It models components, sub-components and connections.
- The *process ontology* contains knowledge objects like physical mechanisms, processes, energy flows and physical domains. It describes the viewpoint of physics on a domain.
- The *mathematical ontology* defines the mathematics used in engineering. It contains the required knowledge to mathematically describe measurable, quantifiable aspects of the modelled world.

The reason that these ontologies are considered to be primary is that each ontology introduces objects that can be thought of independent of the other ontologies. In the domain of physical modelling, it is quite customary to omit either the component view or the process view (sometimes even both). Therefore, we do not consider mathematics a secondary distinction to the primary objects of which it describes some properties, but think of it as primary. The fact that mathematics is a separate science that has theories about mathematical concepts, without relating to the things that they describe, also subscribes to this point of view.

Secondary ontologies Secondary ontologies introduce additional distinctions or typologies that can be applied to objects of primary ontologies. Or actually, they serve as meta-models for distinctions and typologies. A secondary ontology implements a particular dimension through which one can approach a domain. In this paper, two types of secondary ontologies are considered: ontologies that describe taxonomies or typologies of primary objects (e.g. mechanism typology) and ontologies which describe complex theories (geometry, mereology, topology). Examples of secondary ontologies are:

- The *mechanism-type ontology* models a typology of mechanisms, the building blocks of physical process descriptions. It contains mechanism types like *source*, *dissipator* and *converter*.
- The *topological ontology* describes a structural viewpoint on a domain. It represents the connections between knowledge objects. The topological ontology is a subset of graph theory [5].

The mechanism type ontology adds the distinction of different types to the otherwise uniform definition of mechanisms. One could say that it adds the concept of colour to the colourless lego blocks called mechanisms. The topological ontology is used in the component ontology. The component ontology defines components and a certain way to connect them, where connected means being able to exchange energy. The topological ontology adds the rules that have to be followed to make sensible energetic connections to the component ontology.

3.3 Ontology mappings

The dynamic knowledge construction approach requires that the generic ontologies must be used in a flexible, generative way. A domain- or application-dependent ontology can contain objects, that are not present in the general conceptualization. The addition and deletion of a generic ontology leads to an ontology for a specific purpose. Because of the generative character of the construction process certain extensions or inconsistencies can be added to the set of conceptualizations. Thus, the library of ontologies can be incoherent.

To create new objects, a differentiation of generic objects is needed. As stated by Van Heijst [12] (medical) subdomains have domain-specific concepts that are often specializations of the basic (medical) concepts. A way to specialize basic or primary concepts is to differentiate these according to a particular dimension. A mapping between a primary ontology and a secondary ontology, which explicitly specifies a dimension, creates a specialization of the primary elements.

There is no unique sequence in which a number of mappings must be applied. If one is only interested in the final result of the mapping process, it is less important which mapping is executed first. For example, mapping the dimension ‘type of process’ to the concept ‘machinery with pressurizing function’ creates an object called a ‘rotating machinery with pressurizing function. The mapping between the ‘function typology’ and the concept ‘rotating machinery’ would create the same object. But if it is important not only to consider the final result of the mappings, but also the intermediate concepts the order does matter. Hence, the tree of construction has no unique structure.

In theory it is possible to map all secondary dimensions on all primary ontologies. But by doing so, a lot of ontological mappings would construct ontological elements that are meaningless in most domains. When the *mereological ontology* is mapped onto the concept *substance* for example, a conceptualization is created that deals with the parts of substances. Only when one is interested in molecules and atoms this can be a useful ontology. Therefore, dimensions and secondary ontologies have certain constraints, that determine when these can play a role in an ontological mapping. A system can only be examined from a mereological viewpoint if the system has a fixed structure. For the topological ontology to be useful, there has to be a type of interaction between the objects in a domain.

A secondary ontology describes a viewpoint out of which one can approach a primary conceptualization. So by actually looking at a primary ontology out of a particular viewpoint a new ontology is formed. This is exactly how an ontological mapping works. A primary and a secondary ontology are combined to create a new ontology. Or in other words, a secondary ontology is mapped onto a primary one. The result of this mapping is a new conceptualization, that originates from the primary ontology and that is specified according to the secondary ontology.

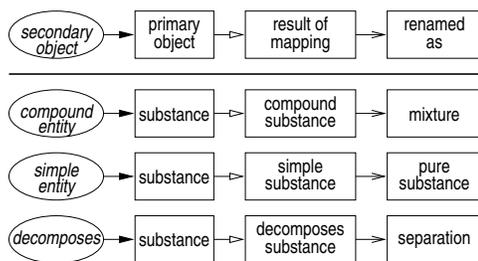


Figure 1: Ontological mapping between *decompositional ontology* and the concept *substance*.

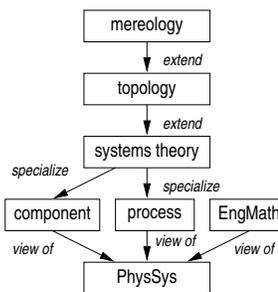


Figure 2: Inclusion lattice of the PHYSYS ontology.

An example is the mapping from the decompositional ontology onto the substance ontology. The decompositional ontology represents the ‘decomposable’-viewpoint. An entity is decomposable, if it has a relevant internal structure, and it is not decomposable, if it has no internal structure and can be considered a black box. The first concept is called a *compound entity*, while the latter is called a *simple entity*. Figure 1 shows the ontological mappings.

A description of substances in a domain can contain pure substances (simple entities) and mixtures (compound entities). The description could also contain a *separation* relation, that is a subtype of the *decomposes* relation.

4 Application of framework for ontology construction

Figure 2 gives an overview of the structure of the PHYSSYS ontology [3, 2]. Boxes represent separate ontologies whereas labeled arrows indicate ontology inclusion. The labels next to the arrows show the kind of inclusion. As can be seen in the figure, the PHYSSYS ontology consists of three primary ontologies which are formalizations of the three views. In the next sections these ontologies will be explained as well as the mereological, topological and system theory ontologies that are used in both the component and process ontologies. Special attention will be given to the ontology mappings.

4.1 Component Ontology

One particular viewpoint on a physical system is that it is a *system* in the sense of general systems theory. That is, it constitutes an entity that (i) can be seen as separate from the rest of the world—so it has a boundary and an outer world, the environment—and that (ii) has internal structure in terms of constitutive elements and subsystems maintaining certain mutual relationships.

For physical systems this implies that we focus on the *structural* aspects, and abstract from what kind of dynamic processes occur in the system and from how it is described in terms of mathematical constraint equations. Within such a purely structural view, we can express the following knowledge about the system:

- Mereological relationships: a system has a certain *part-of decomposition* into subsystems, which on their turn can be decomposed into more primitive components.
- Topological relationships: the various constituents of a system (subsystems, components) are linked to one another through certain *connections*. For a physical system, this provides information on the spatial topology of the system, but the connections additionally indicate the paths for physical interactions between the constituents.

An example of a structural-topological diagram for a physical system, i.c. an air pump, is shown in Figure 3. This structural view on physical systems is based upon what we call a *component ontology*.

Our component ontology is constructed from mereology, topology and system theory. In a separate ontology of mereology a *part-of relation* is defined that formally specifies the intuitive engineering notion of system or device decomposition. This mereological ontology is then imported into a second separate ontology which introduces *topological connections* that connect mereological individuals. This topological ontology provides a formal specification of what the intuitive notion of a network layout actually means and what its properties are. The ontology of systems theory includes the topological ontology and defines concepts like (open or closed) systems, system boundary etc. on top of it.

Mereology Our mereological ontology is simply an Ontolingua implementation of the Classical Extensional Mereology as described in [10]. We therefore only give a brief explanation of this ontology and refer to [10] for the details and more philosophical aspects. Two relations define part-of decompositions. The relation `equal(x,y)` defines which individuals are to be considered mereologically equal. In the usual case, it only holds for `equal(x,x)` but in some situations it is convenient to say that two individuals are equal when they have the same parts. An individual `x` is a mereological individual when `equal(x,x)` holds. When a mereological individual `x` is a part of a mereological individual `y`, the relation `proper-part-of(x,y)` holds. With these relations it is possible to write down a variety of axioms specifying desirable properties any system decomposition should have. Examples are the asymmetry and transitivity of the `proper-part-of` relation.

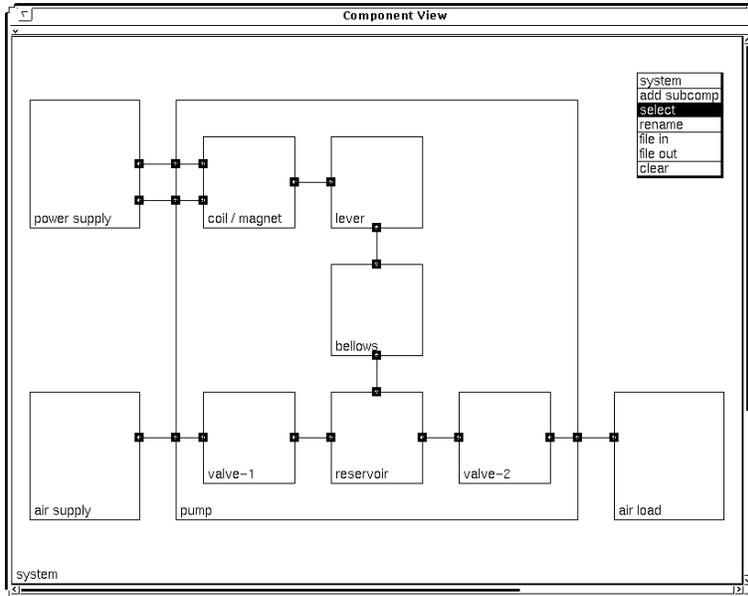


Figure 3: The component view on a physical system, showing a two-level part-of decomposition and the system topology for an air pump. Sub-components are drawn inside the area defined by their super-component.

```

1 define-theory mereology
2 define-class m-individual(x)
a   m-individual(x) <-> equal(x,x)
3 define-relation proper-part-of(x,y)
a   proper-part-of(x,y) -> not proper-part-of(y,x)
b   proper-part-of(x,y) and proper-part-of(y,z) -> proper-part-of(x,z)
4 define-relation direct-part-of(x,y)
a   direct-part-of(x,y) <-> proper-part-of(x,y) and not exists z: proper-part-of(z,y) and proper-part-of(x,z)
5 define-relation overlap(x,y)
a   overlap(x,y) <-> equal(x,y) or exists z: proper-part-of(z,x) and proper-part-of(z,y)
6 define-class simple-m-individual(x)
a   simple-m-individual(x) <-> m-individual(x) and not exists y: proper-part-of(y,x)

```

The listing above is an excerpt from the mereological ontology. Definition 2 defines the class of mereological individuals that was sketched above. The definition of the **proper-part-of** relation clearly shows the asymmetry (3a) and transitivity (3b) axioms. Note that these definitions only serve as an illustration and are not meant to be complete. The ontology furthermore defines the relation **overlap(x,y)** which holds for individuals that are equal or share a part and **simple-m-individual**, the class of individuals that have no decomposition.

Topology The topological ontology defines a relation to express the fact that mereological individuals are connected. We want to use this relation to define connections in the component view of a physical system, where being connected means *being able to exchange energy*. Because we have this application in mind, the topology must be capable of stating that: (i) two individuals are connected, (ii) there can be multiple connections between components and (iii) a connection is of a certain type.

A well known way of expressing topological information is described by B. L. Clarke [4]. He introduces a relation Cx, y to express that individuals x and y are connected. Unfortunately, his theory cannot be used here because it violates requirements (ii) and (iii). These requirements can only be met when connections are reified as entities. This has led to the relation **connects(c,x)** which means that individual x is connected by connection c (see the excerpt below).

```

1 define-theory topology
2 include-theory mereology
3 define-class connection(c)
a   connection(c) <-> exists x,y: x!=y and connects(c,x) and connects(c,y)
4 define-relation connects(c,x)
a   connects(c,x) and connects(c,y) -> not(part-of(x,y) or part-of(y,x)) 
b   connects(c,x) and connects(c,y) and part-of(x,z) and not overlap(z,y) -> connects(c,z) 
c   connects(c,x) and connects(c,y) and connects(c,z) -> overlap(x,y) or overlap(y,z) or overlap(z,x) 

```

The mapping performed in this ontology is of the type include and extend. Inclusion is done in line 2 and the extension takes shape by definition of new concepts and relations that use mereology in their axioms. This yields an ontology that has same level of abstraction as the included mereology. There are three axioms concerning connections. Axiom 4a prohibits that a part is connected to its whole. The second axiom (4b) ensures that when a part whose whole is disjoint with an individual connected to the part, the whole is also connected to the individual. The third axiom (4c) prohibits connections to fork.

Systems Theory On top of the topological ontology the standard system-theoretic notions such as system, subsystem, system boundary, environment, open/closedness etcetera can be defined. Some of these definitions can be found in the excerpt below. The ontology mapping is of the include and extend type, just like in the topological ontology.

```

1 define-theory system-theory
2 include-theory topology
3 define-class system(s)
a   system(s) -> m-individual(s)
4 define-relation in-system(x,s)
a   in-system(x,s) -> proper-part-of(x,s) and system(s) and not system(x)
5 define-relation in-boundary(c,s)
a   in-boundary(c,s) <-> connection(c) and system(s) and
    exists x,y: connects(c,x) and connects(c,y) and in-system(x,s) and not in-system(y,s)
6 define-relation subsystem-of(sub,sup)
a   subsystem-of(sub,sup) <-> system(sub) and system(sup) and proper-part-of(sub,sup)
7 define-class open-system(s)
a   open-system(s) <-> system(s) and exists c: in-boundary(c,s)

```

Definition 2 states that a system is a mereological individual. The `in-system(x,s)` holds for individuals that are proper part of the system but are not systems themselves. This is different from the `subsystem-of(sub,sup)`, where the part must be a system. A connection is in the boundary of a system when it connects an individual in the system to an individual outside the system. With this definition, the classes `open-system` and `closed-system` can be defined easily.

Components After the ontology inclusion and extension of the previous paragraphs, now a more complex mapping will be presented, i.e. the mapping of component ontology to the abstract system theory ontology: $system-theory \times component \rightarrow physical-system$. The component ontology defines the structural view on physical systems as depicted in Figure 3 i.e. components that can have subcomponents and terminals. The terminals are the interfaces of the components to the outer world. Therefore, connections hook on to terminals instead of on components. This interpretation of components and connections is a bit more complex than the networks of abstract individuals and connections in systems theory. The paragraph below describes the way this mapping takes place. Because this mapping makes abstract concepts more specific, this type of mapping is called *include and specialize*.

The listing below shows some definitions from the component view ontology. The important classes are the classes `component`, `terminal` and `physical-system`. The relations `comp.subcomp`, `comp.term` and `conn.term` relate components to their subcomponents, terminals to components and connections to terminals. Only the definitions contributing to the ontology mapping are shown in the figure. Ontology mapping consists of inclusion (line 2) and the definition of axioms that specify the abstraction of components to system theoretical concepts. Definition 3 shows how the ontological commitments for abstract mereological individuals are mapped onto components.

Definition 4 defines the meaning of the `comp.subcomp` relation in terms of mereology. The mapping of topological connections to component connections is performed by definition 5. Definition 6 defines the modelled device as a system of components. The fact that connections can be of a certain type has been left out due to lack of space.

```

1  define-theory component-view
2  include-theory system-theory
3  define-class component(c)
  a  component(c) -> m-individual(c)
4  define-relation comp.subcomp(c,s)
  a  comp.subcomp(c,s) <-> component(c) and component(s) and direct-part-of(s,c)
5  define-relation conn.term(conn,term)
  a  conn.term(conn,term) and comp.term(comp,term) -> connects(conn,comp)
  b  component(comp) and connects(conn,comp) -> exists term: conn.term(conn,term) and comp.term(comp,term)
6  define-class phys-system(s)
  a  phys-system(s) <-> system(s) and (in-system(c,s) -> component(c))

```

4.2 Process Ontology

Our physical process ontology specifies the behavioural view on physical systems. In the general case it is quite difficult to formalize what the notion of a dynamic process precisely entails. Fortunately, for a certain part of physics this has been done to a level where one can define really primitive process concepts. The approach we take here is known in engineering as system dynamics theory, which also forms the theoretical background of the bond graph method [9]. The basic idea behind this theory is that the dynamics of a system can always be captured by looking at the change of different kinds of *stuff*. In the theory, change of stuff is also called *flow*. For instance, in electrical systems, dynamic behaviour consists of the change of *electrical charge*, i.e. *electrical current*. Likewise, in the mechanical domain the stuff is called *location* and change of location is *velocity*. The thing required to bring about a flow is called *effort*.

The interesting thing about effort and flow is that the product of a flow with its related effort has the dimension *energy / time*, i.e. such a pair defines an energy flow. Physical behaviour can therefore be defined in terms of energy flows. The process ontology introduces physical mechanisms which are applications of physical laws or principles to one or more energy flows. An important feature of these mechanisms is that they exploit in detail the analogies that exist between different physical domains. For example, the principle of conservation of momentum in mechanics is completely analogous to induction in the electrical domain. Many more of these analogies exist. This approach is valid for standard classical, deterministic physics, covering such diverse fields as mechanics, electricity and magnetism, hydraulics, acoustics, and thermodynamics.

Complex process descriptions can be formed by making a network of mechanisms linked by energy flows. This abstraction is used to construct the process ontology. Just like the component ontology, the process ontology defines relatively simple concepts and relations onto which the system ontology is mapped. This can be seen in the listing below. Mechanisms are defined as simple mereological individuals. Energy flows, which have a certain direction, flow from one mechanism to another. The mapping is performed by stating that an energy flow is a topological connection that connects the two mechanisms. A process description can now simply be defined as a system of mechanisms. The definition of physical domains is not shown in the excerpt.

```

1  define-theory process-view
2  include-theory system-theory
3  define-class mechanism(m)
  a  mechanism(m) -> simple-m-individual(m)
4  define-class energy-flow(ef)
  a  energy-flow(ef) -> connection(ef)
5  define-relation ef.from-to(ef,f,t)
  a  ef.from-to(ef,x,y) -> connects(ef,x) and connects(ef,y)
  b  energy-flow(ef) and connects(ef,x) and connects(ef,y) -> ef.from-to(ef,x,y) or ef.from-to(ef,y,x)
6  define-class process(p)
  a  process(p) <-> system(p) and (in-system(m,p) -> mechanism(m))

```

Figure 4 shows the taxonomy of mechanisms as defined in the process ontology. All classes in the figure are present as classes in the ontology (except for ones on the right) and class-subclass relations are defined for every line in the figure. The discriminating properties used to

construct this taxonomy are the number of energy flows linked by a mechanism (connectivity), the mechanism type, whether effort or flow plays the most important role in respect to the mechanism type, conversion type and physical domain. Note that some discriminating properties are not useful for some types of mechanisms.

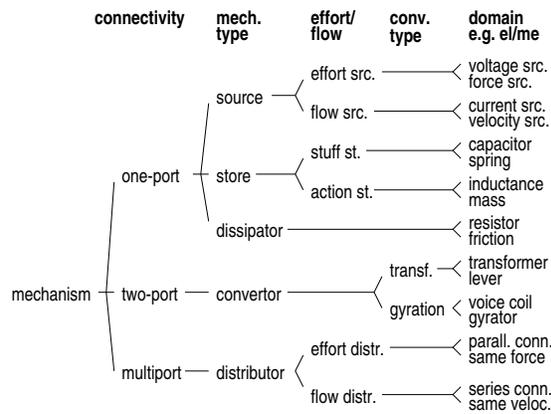


Figure 4: The taxonomy of physical mechanisms.

The order in which the discriminating properties are applied here is the opposite of the order used in the typical engineering education. There, the distinction between physical domains is made first: there are separate courses in mechanics, electrical engineering and thermodynamics. Only when students have mastered all courses, they are able to see the analogies between the domains that makes the process ontology as compact and elegant as it is.

4.3 Mathematical Ontology

The mathematical ontology defines the mathematics required to describe physical processes. The EngMath ontology [7], available in the Ontolingua ontology library, is perfectly suited for this job and has therefore been (re)used for this. In this section, only a very short description is given that should be enough to understand the mapping of the process ontology to mathematics. For detailed information on EngMath see [7].

The EngMath ontology formalizes mathematical modelling in engineering. The ontology includes conceptual foundations for scalar, vector, and tensor quantities, physical dimensions, units of measure, functions of quantities, and dimensionless quantities. A physical quantity is a measure of some quantifiable aspect of the modelled world characterized by a physical domain such as length, mass or time. Quantities in the EngMath ontology can be expressed in various units of measure e.g. meter, inch, kilogram, pound. etc. The ontology defines all relations between quantities, units of measure and dimensions. Important for the PHYSYS ontology are time-dependent-quantities and the meta-level relationship of Ontolingua (KIF) expressions and physical quantities. Time-dependent-quantities are continuous functions from a quantity with the time domain to another scalar quantity. Effort and flow quantities quantifying an energy flow are defined as time dependent quantities. The explicit relationship between Ontolingua relations and relations between physical quantities is used to specify the relationships between effort and flow quantities of energy flows imposed by the physical processes.

4.4 Mapping Ontologies

The PHYSYS ontology is a *mapping ontology*. It includes ontologies of different views on the domain and implements the relations between the concepts in different views. PHYSYS consists of two parts, one that includes the component and process ontologies and relate them to each other and an additional part that includes EngMath and maps processes to it. Figure 5 shows the

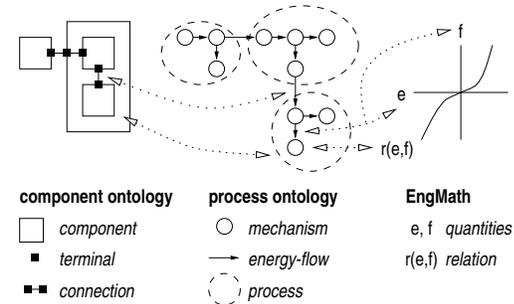


Figure 5: Mappings between the component, process and mathematical ontologies.

mapping between instantiated views on a physical system. Basically, it defines that components are the carriers of physical processes that can be mathematically described with physical quantities and mathematical relations.

Components to Processes The listing below shows the first part of the PHYSYS ontology that includes the three viewpoints (lines 2, 3 and 4) and maps the component and process views (definitions 5 and 6). The relation `comp.proc` (definition 5) implements the mapping of simple components to process descriptions. Axiom 5a states that every component must have a process description and axiom 5b that each mechanism must be part of the process description of a component. Axiom 5c ensures that a mechanism can only be part of one the process description of one component. The fact that energy flows between process descriptions of two components must go through a connection is expressed by definition 6. For each connection between components, the process descriptions of these components must interact via an energy flow (axiom 6a). Vice versa, axiom 6b defines that an energy flow between the process descriptions of two components goes through a connection. Note that the relationship between the type of a connection and the number and domain of the energy flows of this connection has not been included in the excerpt.

```

1  define-theory PhysSys
2  include-theory component
3  include-theory process
4  include-theory EngMath
5  define-relation comp.proc(c,p)
   a  component(c) and simple-m-individual(c) -> exists p: process(p) and comp.proc(c,p)
   b  mechanism(m) -> exists c,p: process(p) and in-system(m,p) and comp.proc(c,p)
   c  comp.proc(c1,p1) and comp.proc(c2,p2) and c1 != c2 -> not overlap(p1,p2)
6  define-relation conn.ef(c,ef)
   a  conn.term(c,t1) and conn.term(c,t2) and comp.term(c1,t1) and comp.term(c2,t2) and comp.proc(c1,p1) and
      comp.proc(c2,p2) -> exists ef: conn.ef(c,ef) and in-boundary(ef,p1) and in-boundary(ef,p2)
   b  energy-flow(ef) and process(p1) and in-boundary(ef,p1) and comp.proc(c1,p1)
      and process(p2) and in-boundary(ef,p2) and comp.proc(c2,p2)
      -> exists c: comp.term(c1,t1) and conn.term(c,t1) and comp.term(c2,t2) and conn.term(c,t2)

```

Processes to EngMath Mapping of the process ontology to EngMath is depicted in the right side of Figure 5. Due to lack of space, it is only described here in an informal way.

Informally, the mapping states that for each energy flow there are two time-dependent physical quantities, one for the effort and one for the flow. The domain of the energy flow determines the dimension of the quantities. For instance, an electrical effort quantity has the dimension `energy/electrical-current` (voltage) and the flow quantity the `electrical-current` dimension. For each mechanism there is a mathematical relation that relates the values of the physical quantities of the energy flows connecting the mechanism to each other. The mapping also imposes constraints on the relation. These constraints only depend on the mechanism type and they are independent of the domains of the energy flows. The mathematical relation in Figure 5 belongs to a dissipator mechanism. The constraints on such a relation are that it is a continuous function $r : e \mapsto f$ that lies in the first and third quadrant and that $r(0) = 0$. For an electrical energy flow, this can be an instantiation of Ohm's law $V = I \times R$ whereas in the mechanical domain it can model some kind of friction with $F = k \times v$.

5 Conclusions

In this paper we have pointed to issues in knowledge sharing that go beyond making available libraries of reusable ontologies. Retrieval and assembly from large libraries is itself a knowledge-intensive task. So, a relevant question becomes how we can guide the user in gradually building up the needed knowledge.

This papers discussed an approach to solving this problem by regarding the generative aspects of sharable ontologies. Our main conclusions are:

- Any application ontology seems to be built out of multiple but, on the other hand, rather general and separate ontologies. By considering an ontology of physical systems, we have shown

how such an application domain can be described in terms of a *lattice* of generic ontologies regarding, e.g., mereology, topology, systems-theory, physical processes and mathematics. These generic ontologies represent different aspects of or viewpoints taken on the domain at hand.

- Concerning the construction process, we have described how this can be facilitated by gradually adding new 'dimensions' of conceptual distinctions.
- Ontology mappings appear to be important construction operators in generating an application ontology from small generic ontologies. Import and renaming alone are insufficient to achieve this.

We feel that our work helped in finding better answers to the considered question of ontology construction. We hope that this can widen the circle of end users that can really profit from available libraries for knowledge sharing and reuse.

References

- [1] J. M. Akkermans, P. Borst, A. Pos, and J.L. Top. Experiences in conceptual modelling for a mechatronic design library. In B.R. Gaines and M. Musen, editors, *Proceedings of the ninth international knowledge acquisition workshop KAW'95*, pages volume 2, pages 39.1–39.15. University of Calgary, SRDG Publications, 1995.
- [2] P. Borst, J. M. Akkermans, A. Pos, and J. L. Top. The PhysSys ontology for physical systems. In R. Bredeweg, editor, *Working Papers Ninth International Workshop on Qualitative Reasoning QR'95 (Amsterdam, NL, May 16-19, 1995)*, pages 11–21, Amsterdam, the Netherlands, May 1995. Department of Social Science Informatics, University of Amsterdam.
- [3] P. Borst, A. Pos, J.L. Top, and J.M. Akkermans. Physical systems ontology. In N.J.I. Mars, editor, *Working Papers European Conference on Artificial Intelligence ECAI'94 Workshop on Implemented Ontologies*, pages 47–80, Amsterdam, 8–12 August 1994. ECCAI.
- [4] B. L. Clarke. A calculus of individuals based on 'connection'. *Notre Dame Journal of Formal Logic*, 22(3):204–218, 1981.
- [5] R. P. Grimaldi. *Discrete and combinatorial mathematics: an applied introduction*. Addison-Wesley, Reading, Massachusetts, 1989.
- [6] T.R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5:199–220, 1993.
- [7] T.R. Gruber and G.R. Olsen. An ontology for engineering mathematics. In J. Doyle, P. Torasso, and E. Sandewall, editors, *Proceedings Fourth International Conference on Principles of Knowledge Representation and Reasoning*, pages 258–269, San Mateo, CA, 1994. Morgan Kaufmann.
- [8] N. Guarino, M. Carrara, and P. Giaretta. An ontology of meta-level categories. In J. Doyle, E. Sandewall, and P. Torasso, editors, *Principles of knowledge representation and reasoning: proceedings of the fourth international conference (KR94)*, San Mateo, California, 1994. Morgan Kaufmann.
- [9] D.C. Karnopp, D.L. Margolis, and R.C. Rosenberg. *System Dynamics: A Unified Approach*. John Wiley & Sons, New York, 1990. Second Revised Edition.
- [10] Peter Simons. *Parts, A Study in Ontology*, chapter 1–3. Clarendon Press, Oxford, 1987.
- [11] J.F. Sowa. Top-level ontological categories. *Int. J. of Human Computer Studies*, 43:669–685, 1995.
- [12] G. van Heijst. *The role of ontologies in knowledge engineering*. PhD thesis, University of Amsterdam, SWI, 1995.
- [13] G. van Heijst, S. Falasconi, A. Abu-Hanna, A. Th. Schreiber, and M. Stefanelli. Principles for modularizing medical domain knowledge. Technical Report GAMES-II/T3.1/UvA/PP/018/1.0, University of Amsterdam and University of Pavia, 1994. Submitted for publication. Available from: University of Amsterdam, Social Science Informatics, Roetersstraat 15, NL-1018 WB, Amsterdam, The Netherlands.
- [14] B. Wielinga, G. Schreiber, W. Jansweijer, A. Anjewierden, and F. van Harmelen. Framework and formalism for expressing ontologies. ESPRIT Project 8145 KACTUS, deliverable DO1b.1, University of Amsterdam, 1994.
- [15] B. J. Wielinga, J. M. Akkermans, H. A. Hassan, O. Olsson, K. Orsvärn, A. Th. Schreiber, P. Terpstra, W. Van de Velde, and S. Wells. Expertise model definition document. deliverable ESPRIT Project P-5248 /KADS-II/M2/UvA/026/5.0, University of Amsterdam, Free University of Brussels and Netherlands Energy Research Centre ECN, June 1994.