The PhysSys Ontology for Physical Systems*

Pim Borst ‡ Hans Akkermans ‡§ Anita Pos ‡ Jan Top ¶

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

§Netherlands Energy Research Foundation ECN
P.O. Box 1, NL-1755 ZG Petten (NH), The Netherlands
E-mail: akkermans@ecn.nl

¶Agro-Technological Research Organization ATO-DLO
P.O. Box 17, NL-7500 AA Wageningen, The Netherlands
E-mail: J.L.Top@ato.agro.nl

Abstract: We give an outline of a formal ontology for physical systems. It is based upon system dynamics theory as practiced in engineering modelling, simulation and design. We introduce multiple engineering ontologies — system layout, physical processes underlying behaviour, descriptive mathematical relations — that express different conceptual viewpoints upon a physical system. These three views combine the QR device, process and constraint approaches. It is discussed how these viewpoints can be formally specified in terms of separate generic ontologies, which are relatively loosely coupled through ontology mapping rules. These ontologies provide a formal conceptual foundation for the structure of the OLMECO library of reusable models for engineering design in mechatronics currently under development.

1 Introduction

In this paper we discuss a formal ontology for physical systems. The background of our work is the construction of model libraries for engineering design. Here, a diversity of physical systems are being investigated, including heating systems, automotive components and machine tools [6, 20]. The present work is based upon experiences deriving from these applications. This paper shows how AI research into ontologies can be practically relevant to systems engineering science.

The notion of ontology has quite different meanings in various fields. In philosophy it stands for the general theory 'on what there is', and it dates back to Aristotle and medieval Scholastic philosophy [4]. AI introduces quite an important twist to this notion. Gruber [10] defines ontology as a (formal) 'specification of a conceptualization'. Thus, in computer science the focus is not on theoretical claims about what exists in the world, but on how the world is being conceptualized by various agents. Conceptual specifications that are found to be common across many agents provide a foundation for the sharability and reusability of their knowledge. It is interesting to mention here the position of the philosopher Quine. He sees ontological questions as being on a par with questions of natural science [14]. This 'naturalized epistemology' is summarized in the slogans 'what exists is what can be quantified over' and 'to be is to be the value of a variable'. What exists is what is presupposed in our scientific theories about the world; and an ontological commitment then is the collection of things that must be assumed to exist in order for a theory to be true. Such a philosophical view is quite compatible with the AI formal specification view, including the approach in this work.

In physics and engineering the notion of ontology as such is not (yet) being used. There are however activities that could well be classified under the banner of ontology. For example, the conceptual interpretation problems associated with quantum mechanics [12, 2] have a clear ontological flavour. In engineering science the rather more practical problem is how to give automated support to expensive and knowledge-intensive tasks such as engineering modelling and de-

* This work has been partially supported by the CEC Esprit-III project P6521 'OLMECO'. The partners in the OLMECO project are PSA Peugeot Citroen (France), BIM (Belgium), Fagor (Spain), Ikerlan (Spain), Imagine (France), University of Twente (The Netherlands) and the Netherlands Energy Research Foundation ECN (The Netherlands). Further support came from the Netherlands Computer Science Research Foundation SION with financial support from the Netherlands Organization for Scientific Research NWO.

sign. Here, the issue is to provide specifications for information systems and engineering databases that are both formally and computationally sound *and* adequately reflect the conceptual foundations of physics and engineering systems theory. Our work relates to these issues.

Work on engineering ontologies will be fruitful for various reasons:

- It helps to formally define how design engineers and other end users of Computer Aided Engineering systems look at their world domain. This can be employed to enhance CAE user support for tasks such as engineering modelling [18].
- It provides a foundation for the conceptual schema for data structuring in engineering databases, libraries and other CAE information systems (e.g., [8]).

One idea that is currently popular both in computer science and engineering disciplines is to supply libraries of reusable models and model fragments [7]. This paper shows that there are generic ontologies with respect to physical systems that can be employed to increase the structuredness and reusability of libraries and to constrain the modelling decisions [18]. The ontological organization of our model libraries is depicted in Figure 1. Three ontological viewpoints on physical systems are distinguished: system components, physical processes and mathematical relations. Of course, this is not new in itself, as it resembles the device, process and constraint approaches in QR, and these viewpoints are expressible in, for example, the Compositional Modelling Language [7]. What is different, however, in the present work is that the underlying ontological commitments can be formalized in a generic, domain-independent way, and that different ontological viewpoints can be kept well-separated, thus enhancing reusability. Each ontological level has its own relatively independent conceptualization and representation, and a complete physical model comprises all of these three levels, thus supporting the device, process and constraint approaches. This framework is used in practice as a basis for organizing the library under development in the European Esprit-III P6521 OLMECO project, which is concerned with reusable and sharable models for mechatronic component design.

In this paper we give a formal specification of the ontologies underlying such a library. As we will see, these ontologies derive from standard physics and engineering, especially systems theory and formal classical physics as incorporated into the so-called bond graph method [13]. In Sec. 2 we discuss the ontology for systems and system components covering topological (in space) and mereological (part-of) relationships. In Sec. 3 we present an ontology for dynamic physical

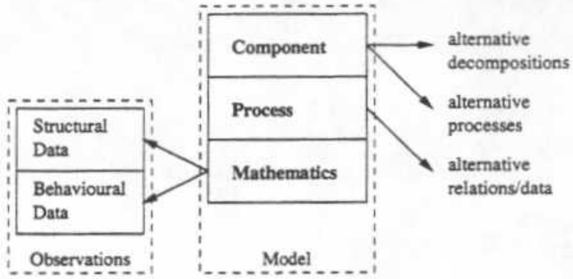

Figure 1: Ontological organization of engineering model library.

processes based on generic physical mechanisms and principles such as continuity and conservation of various kinds of physical 'stuff'. For the mathematical description level, it is shown how the EngMath ontology [11] can be reused, and connected to our PhysSys ontology through ontology mapping rules (Sec. 4). This paper is a shortened and revised version of [3], where also an Ontolingua [9] implementation of our physical systems ontology was given. The current PhysSys implementation is available from Pim Borst.

2 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 2.

This structural view on physical systems is based upon what we call a *component ontology*. Our component ontology has a hierarchical structure. In a sepa-

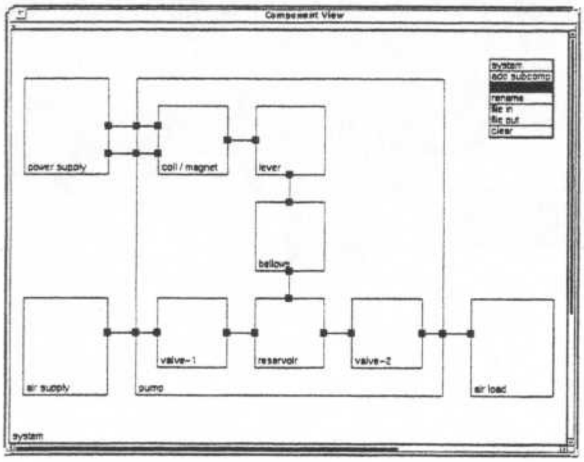

Figure 2: The component view on a physical system, showing a two-level part-of decomposition and the system topology for an air pump. Sub-components are drawn inside the area defined by their super-component.

rate 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* which can connect mereological individuals. This topological ontology provides a formal specification of what the intuitive notion of a system layout actually means and what its properties are.

The component ontology is the highest level of the hierarchy. The basic concepts within this ontology are the following. To start with, we have the concept (or class) of *systems*. The part-of relation is used to define a part-of decomposition which leads to *subsystems* and more elementary *components*. Each component has an interface to the outer world called *terminals*. The topology of a system arises by introducing *connections* which link different components via their terminals. Such a connection is simply a topological connection. Especially for physical systems it is possible to associate the terminals with specific *terminal types*. They define what type of interaction (e.g. electrical, thermal, etc.) is possible along the path indicated by the component connections. By considering a system as being laid out in terms of the constituting sets of components and connections, we can further easily define derived notions such as *system boundary*, *environment* and *open/closed systems*. Thus, the component ontology gives a formalization of the elementary notions well-known (but usually informally) from standard engineering systems theory.

Mereology. Our mereological ontology is simply an Ontolingua implementation of the Classical Extensional Mereology as described in [15]. We therefore only give a brief explanation of this ontology and refer to [15] for the details and more philosophical aspects. Two relations define part-of decompositions. The relation $equal(x, y)$ must hold for individuals which are to be considered mereologically equal. It is left open whether it only holds for $equal(x, x)$ or also for $equal(x, y)$ when x and y 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 mereological individual y, the relation $proper\text{–}part\text{–}of(x, y)$ holds. With these relations it is possible to write down a variety of axioms specifying desirable properties any system decomposition should have. One example is:

$$\begin{aligned} & instance\text{–}of(x, C) \rightarrow \\ & \quad \exists x : \forall y : overlap(y, x) \\ & \qquad \leftrightarrow \exists z : instance\text{–}of(z, C) \wedge overlap(y, z) \end{aligned} \tag{1}$$

This axiom is the so-called general sum principle which implies that for every class of mereological individuals there exists an individual which something overlaps iff it overlaps an individual of that class. Although this is a strong axiom it is not a problem in our application because the component ontology defines the *system*, which is the general sum of all mereological individuals which are components.

Topology. In the topological ontology a relation is defined which can be used 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 three things:

- Express that two individuals are connected.
- Multiple connections between components must be possible.
- It must be possible to say that a connection is of a certain kind.

A well known way of expressing topological information is described by B. L. Clarke [5]. 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 the last two requirements. These requirements can only be met when connections are introduced as entities which connect individuals. This has led to the relation $connects(c, x, y)$ which means that x and y are connected by c.

Five axioms are introduced to accomplish that only sound connections can be made. The associativity of the connects relation in the individuals is accomplished by the first axiom (2). Axiom (3) states that individuals may not be connected to their parts. This includes the case of an individual being connected to itself. The next axiom (4) covers what can be called the transitivity of the connection relation. It says that in a situation when a connection must cross a boundary of an individual, it must also connect this individual. The last two axioms state that connections have a locality property. Axiom (5) prohibits connections to fork (i.e. a connection is a line) and axiom (6) ensures that a connection is not used to connect pairs of individuals which are spatially separated (i.e. that a connection is *one* line).

$$connects(c,x,y) \rightarrow connects(c,y,x) \tag{2}$$

$$\begin{aligned} &connects(c,x,y) \\ &\quad \rightarrow \neg part\text{-}of(x,y) \wedge \neg part\text{-}of(y,x) \end{aligned} \tag{3}$$

$$\begin{aligned} &connects(c,x,y) \wedge part\text{-}of(x,z) \\ &\quad \rightarrow connects(c,z,y) \end{aligned} \tag{4}$$

$$\begin{aligned} &connects(c,x,y) \wedge connects(c,x,z) \\ &\quad \rightarrow \exists a : part\text{-}of(a,y) \wedge part\text{-}of(a,z) \\ &\qquad \wedge\ connects(c,x,z) \end{aligned} \tag{5}$$

$$\begin{aligned} &connects(c,x,a) \wedge connects(c,y,b) \\ &\quad \rightarrow part\text{-}of(x,y) \vee part\text{-}of(y,x) \\ &\qquad \vee\ part\text{-}of(x,b) \vee part\text{-}of(b,x) \end{aligned} \tag{6}$$

Systems theory. The component ontology now results from importing the above mereological and topological ontologies, and defining on top of them the standard system-theoretic notions such as system, subsystem, system boundary, environment, open/closedness etcetera. Lack of space prevents a further discussion here, but can be found in [3]. What we have achieved now is a precise formal specification of standard and widely shared engineering intuitions. It is further interesting to note that here it appears possible to take advantage of logico-philosophical work on mereology and topology carried out in a quite different context (satisfying one of the natural requirements for real knowledge sharing and reuse). Finally, it is noted that in this system-theoretic ontology a lot of useful things already can be said about a device, without making any commitments with respect to other physical system aspects such as dynamic behaviour. Thus, it is indeed possible to keep ontological commitments relatively independent and separate. This reduces the granularity of ontology specifications which is in our opinion important for enhancing sharability and reusability.

3 Physical Mechanism Ontology

In addition to the previously discussed structural view, we are also interested in a *behavioural* view on physical systems. This is a change of perspective which brings along a new vocabulary with concepts such as dynamics, process, mechanism or physical law. This is specified in what we call a *process or mechanism ontology.*

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, and we especially take advantage of its bond graph method [13]. An example of a physical process specification of a small mass-spring system is given in Figure 3 (right half).

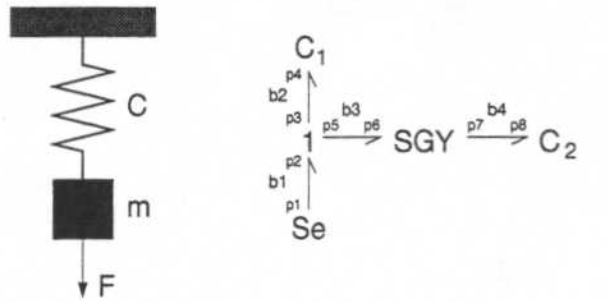

Figure 3: Mass-spring system (left) and its bond graph process description (right).

In the bond graph approach, each node of the graph represents a single physical mechanism. Conceptually a mechanism stands for an elementary physical principle or law (such as conservation of momentum, Newton's law of inertia, the Kirchhoff voltage and current laws etc.) and mathematically it is associated with a constraint equation (differential or algebraic) on the physical variables involved. The edges denote conceptually energy/power exchange links between the mechanisms and mathematically they indicate the variables or signals that are shared between equations. The totality of this represents the dynamic behaviour of the system. In practice, the method is used as a graphical front end to computer algebra and numerical simulation systems. An interesting feature of the approach is that it exploits in detail the analogies that exist between different physical domains. For example, the principle of conservation of momentum in mechanics is completely analogous to charge conservation 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.

Of course, this physics is well known to any physicist and engineer. However, the conceptual foundations are taught in an informal manner until it comes to the point of writing down mathematical equations. But it is possible also to build up the conceptual physical aspects *themselves* in a formal way. This is what we will do below. We note that the formalization itself is novel but that, again, it simply reflects standard physics and engineering knowledge.

To describe the process ontology we use a technique for representing conceptual schemata, which derive from Chen's entity-relationship (ER) diagrams.

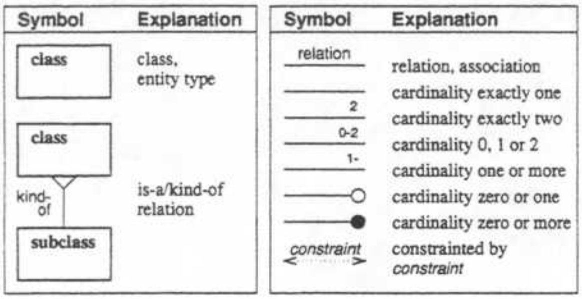

Figure 4: Notational conventions used for conceptual schema specifications.

physical–domain	*stuff*	*action*
transl–potential	*displacement*	*force*
transl–kinetic	*lin–momentum*	*velocity*
ang–potential	*angle*	*torque*
ang–kinetic	*ang–momentum*	*ang–velocity*
vol–potential	*volume*	*pressure*
vol–kinetic	*momentum–flux*	*volume–flow*
electric	*charge*	*voltage*
magnetic	*magetic–flux*	*current*
thermal	*entropy*	*temperature*

Table 1: Physical domains with stuff and action types.

The notational conventions used in this paper are presented in Figure 4. Rectangular boxes represent classes in the ontology. Solid lines represent relations between instances of the classes they connect. Numbers or balls are used to indicate the cardinality of the relation. The special is-a, or kind-of relation has a triangle symbol on the class-side of the relation. Important constraints between relations or entities are represented by dotted arrows.

Physical Domains. Figure 5 shows the conceptual scheme of the ontology of physical behaviour. First, the different categories of physical *stuff* are introduced. These are abstract physical entities that can be extensively defined (e.g. location, volume, charge, order). The amount of stuff of a certain type at a certain time is called a physical *quantity*. The practical usefulness of this is that a (temporal) conservation principle holds for each of these quantities in an isolated system. Dynamic exchanges of stuff between subsystems and in time can then be formulated in terms of the *flow* of this stuff; for this a (spatial) continuity principle generally holds. Bringing about such changes necessitates some action. Each stuff is therefore associated with a specific kind of *action*, the amount of which is called *effort*, which depends on the amount of stuff, or quantity, that is being handled. For example, 'pressure' is the type of action or effort that deals with the stuff type 'volume'; the amount of pressure is a function of the amount of volume. The combination of a type of stuff together with its matching type of action forms what is called a *physical domain*. In the ontology the relations *phys–dom.stuff* and *phys–dom.action* relate types of stuff and action to instances of the class *physical–domain*. Table 1 gives an overview of the physical domains defined in the ontology with the associated types of stuff and action.

Next, the notions of *energy* and *power* are introduced. Energy is a function of the quantities of *all* stuff types; its time derivative is called power and is defined as the product of effort and flow. The importance of the energy concept lies in the property that it couples the various domains and action types. It represents an overall action potential for which a separate conservation and continuity principle holds: the only one valid across all stuff and action types.

Having all this available we can introduce the set of physical *mechanisms* axiomatically.

Physical Mechanisms. A physical mechanism is a specification of a generic physical principle or law. For example, we have for each stuff type a storage mechanism that represents the accumulation of this stuff. The C-node within a bond graph is a generic notation for this, for which an instantiation exists in each domain (for example, Hooke's law in the mechanical potential energy domain, the workings of a capacitor in the electrical domain, and a fluid container in the hydraulic domain). In the conceptual scheme of the physical process ontology all such mechanisms defined are in the lower part of the figure on the right. The boxes contain beside the conceptual names the bond graph notation for these mechanisms. The physical mechanisms thus distinguished cover elementary source, storage, dissipation, conversion and distribution physical processes (see also [17]). Taking the Cartesian product with the physical domains or stuff types yields those concepts that are generally known as the standard laws of classical physics. Exchange of energy between physical processes is defined by connections of *multiports*, which are on the top of the taxonomy of physical processes. The *ports* of these multiports are connected through energy *bonds* and constitute the places through which physical stuff as well as energy can flow. Each port has the type of energy flow assigned to it through the relation *port.phys-dom*. The constraint *proccons1* stands for the necessary constraints to form a proper network of energy flows between the multiports.

Domain Coupling Laws. There is one additional group of physical laws that deserves special mention: *domain coupling laws*. Many physical domains have a so-called dual domain, which means that the effort related to one domain can be considered as the flow

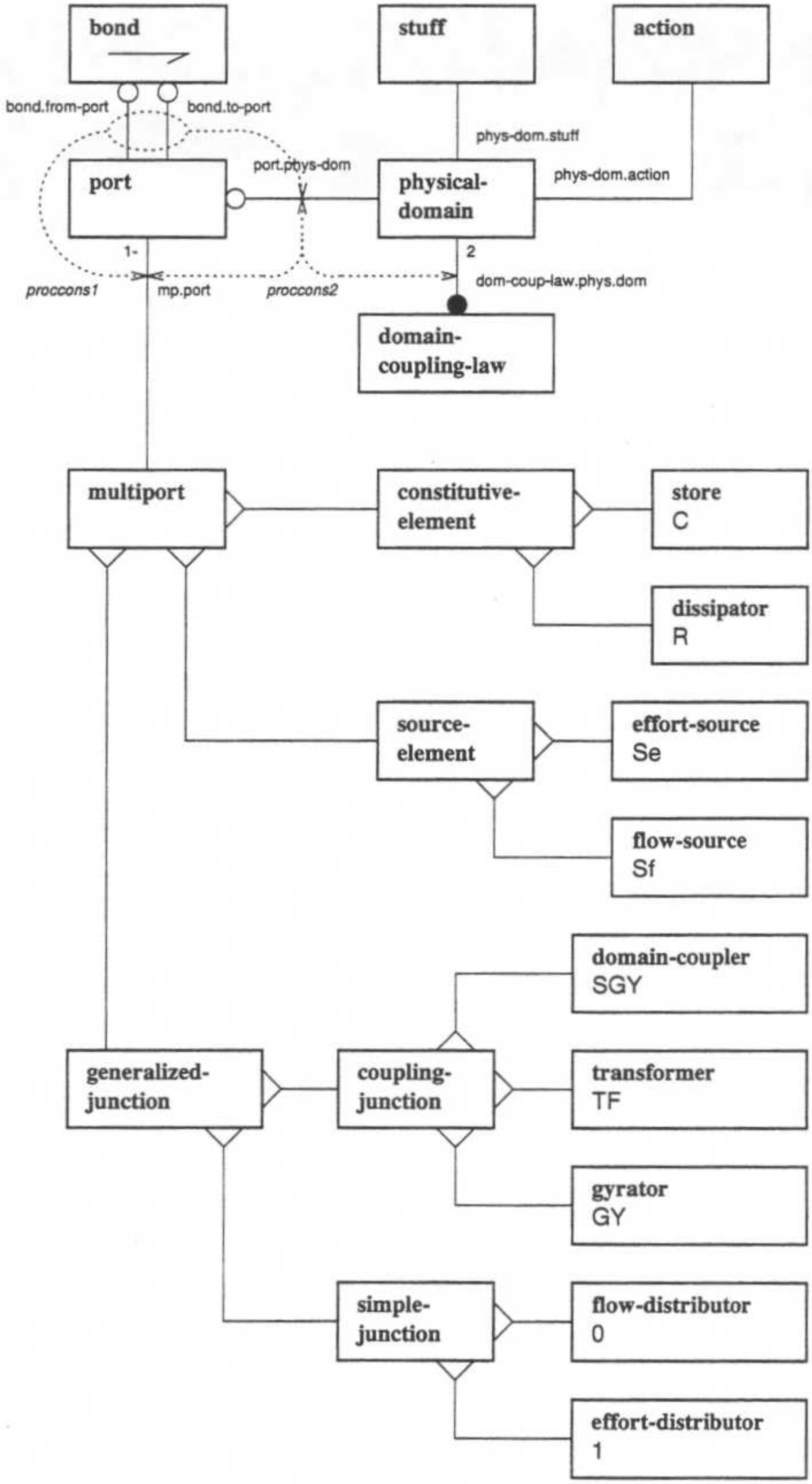

Figure 5: Conceptual scheme of the physical process ontology.

of the dual domain, and vice versa. Thus, certain action types can themselves be considered as types of stuff. Realizing the identity of two different concepts from different physical domains led to historically important discoveries. Faraday's law of induction is one specimen of this: it says that a change in magnetic flux (flow in the magnetic domain) corresponds to an electrical voltage (effort in the electrical domain). Another famous example is Newton's $F = ma$ law.

In figure 5 the class of domain coupling laws can be found. Each instance of this class, i.e. each law has the two domains it couples associated to it by the relation *dom–coup–law.phys–dom*. Table 2 shows the defined coupling laws with the coupled physical domains.

coupling–law	*domain₁*	*domain₂*
newton–trans	*transl–potential*	*transl–kinetic*
newton–ang	*ang–potential*	*ang–kinetic*
newton–vol	*vol–potential*	*vol–kinetic*
faraday	*electric*	*magnetic*

Table 2: Domain coupling laws.

The application of a domain coupling law in a physical process specification is represented by a *domain coupler* mechanism. This is a two-port mechanism which passes the energy it receives from one port to the second, only changing the physical domain. The requirement that the domains of the ports of a domain coupler are coupled by a domain coupling law is represented by the *procconsZ* constraint in Figure 5.

An Example. In Figure 3, a drawing of a mass-spring system was given on the left side and a bond graph process specification on the right. In this paragraph we will give a short explanation of the process description and show that the classes and relations in the physical process ontology are sufficient to describe the physical processes in the system.

The energy supplied by the force of gravity on the mass m accounts for the effort (force) source Se in the process description. The energy it supplies is distributed over the spring and the mass by the effort distributor *1*. The spring is represented by C_1, a store of displacement. The mass is also modelled as a store because it stores energy in the form of linear momentum. This is why the domain coupler SGY is used. It changes the domain of the energy from translation potential to translation kinetic, according Newton's $F = ma$ law.

In Figure 3, the ports and bonds are labeled to provide an easy link to the specification of the physical processes in terms of the ontology. This specification can be found below. First, the physical mechanisms are defined and after that the energy flows between them.

Se : *instance–of*$(Se, \textit{effort–source})$
mp.port(Se, p_1)
port.phys–dom$(p_1, \textit{transl–potential})$

1 : *instance–of*$(1, \textit{effort–distributor})$
mp.port$(1, p_2)$
port.phys–dom$(p_2, \textit{transl–potential})$
mp.port$(1, p_3)$
port.phys–dom$(p_3, \textit{transl–potential})$
mp.port$(1, p_5)$

C_1 : *instance–of*$(C_1, \textit{store})$
mp.port(C_1, p_4)
port.phys–dom$(p_4, \textit{transl–potential})$

SGY : *instance–of*$(SGY, \textit{domain–coupler})$
mp.port(SGY, p_6)
port.phys–dom$(p_6, \textit{transl–potential})$
mp.port(SGY, p_7)
port.phys–dom$(p_7, \textit{transl–kinetic})$

C_2 : *instance–of*$(c_2, \textit{store})$
mp.port(C_2, p_8)
port.phys–dom$(p_8, \textit{transl–kinetic})$

b_1 : *instance–of*$(b_1, \textit{bond})$
bond.from–port(b_1, p_1)
bond.to–port(b_1, p_2)

b_2 : *instance–of*$(b_2, \textit{bond})$
bond.from–port(b_2, p_3)
bond.to–port(b_2, p_4)

b_3 : *instance–of*$(b_3, \textit{bond})$
bond.from–port(b_3, p_5)
bond.to–port(b_3, p_6)

b_4 : *instance–of*$(b_4, \textit{bond})$
bond.from–port(b_4, p_7)
bond.to–port(b_4, p_8)

Summarizing. Summarizing, we have three main categories of knowledge within the process ontology:

- The physical things that vary in time: quantities of stuff, flow of stuff, amount of action (i.e. effort), energy, power.
- The ways through which this happens: physical processes, which can be laid out in terms of a finite set of primitive mechanisms (with corresponding mathematical laws).
- The substrate (space) on which all this takes place.

The last item actually links the component and process ontology. Subsystems and components abstract regions of space that are seen as carriers of physical processes. In switching to the process ontology, subsystems/components are generally called *multiport* *elements* and, in the behavioural view, become represented as assemblies of ideal-physical mechanisms. The *ports* of these elements are connected through energy *bonds* and constitute the places through which physical stuff as well as energy can flow.

4 Mathematical Ontology and Ontology Mappings

The mathematical ontology defines the mathematics required to describe physical processes. It is an ontology that can be used without the process ontology because the links to it are not integrated in the mathematical definitions, but are implemented by separate relations called *ontology mapping relations*.

The basis for the mathematical ontology is formed by the EngMath ontology [11], an ontology of the foundations of mathematical modelling in engineering. It introduces the concepts of physical quantities and mathematical relations between physical quantities. A physical quantity is a measure of some quantifiable aspect of the modelled world. The difference between a physical quantity and a number is that a physical quantity has a dimension indicating what kind of real world aspect is measured, for instance length, mass, time, electrical current etc. The EngMath ontology defines mathematical relations to specify dependencies between physical quantities.

Because the EngMath ontology is designed to be very generic, it imposes little structure on the physical quantities and mathematical relations defined. In our case however, because of this lack of structure, direct mapping of physical processes to concepts in the EngMath ontology would require a great number of complex mapping relations. Therefore we decided to do the mapping in two steps. First, PhysSys's mathematical relation ontology defines a number of classes of special mathematical relations in terms of the EngMath ontology. The ontology mapping relations then map physical processes to mathematical relations in these classes.

Figure 6 gives the conceptual scheme of PhysSys's mathematical ontology. In this scheme, light grey boxes are physical quantities from the EngMath ontology. The usual notation for these quantities in engineering is displayed inside these boxes underneath the class name. The white boxes are the classes of mathematical relations defined in PhysSys. What class of EngMath relations they define is printed underneath the class names.

An energy flow in a process description, represented by a bond, is mapped to a pair of physical quantities which values are the effort and flow of the energy flow. To make it easier to refer to such a pair of quantities, the concept *effort-flow-pair* was introduced. The physical dimension of the quantities can be derived from the physical domain of the energy flow they describe. This is shown in Table 3. Each PhysSys mathematical relation is a mathematical description of a physical mechanism which relates the efforts and flows of one or several effort-flow-pairs to each other. Because the effort and flow of an effort-flow-pair must be well defined, *mathcons1* establishes that exactly two mathematical relations use an effort flow pair.

physical–domain	*physical–dimension* of *effort* *physical–dimension* of *flow*
transl–potential	*force* *length* $time^{-1}$
transl–kinetic	*length* $time^{-1}$ *force*
ang–potential	*force length* $time^{-1}$
ang–kinetic	$time^{-1}$ *force length*
vol–potential	*force* $length^{-2}$ $length^{3}$ $time^{-1}$
vol–kinetic	$length^{3}$ $time^{-1}$ *force* $length^{-2}$
electric	*energy* $electrical\text{–}current^{-1}$ *electrical–current*
magnetic	*electrical–current* *energy* $electrical\text{–}current^{-1}$
thermal	*temperature* *energy* $temperature^{-1}$ $time^{-1}$

Table 3: Mapping of physical domains to physical dimensions of effort and flow.

To clarify the mapping relations, we will give an example of the way a physical process is mapped to a mathematical relation in the next paragraph.

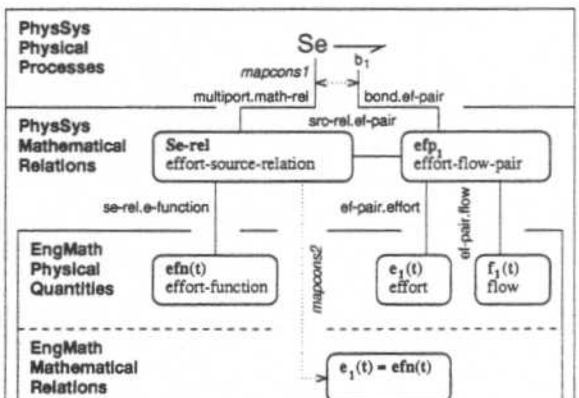

Figure 7: Mapping of a physical process to a mathematical relation.

An Example. Figure 7 shows the mapping of the effort source representing the gravitational force on the mass in Figure 3 to a mathematical relation. In the figure, rounded boxes are instances of classes in the mathematical ontology. The name of the instance

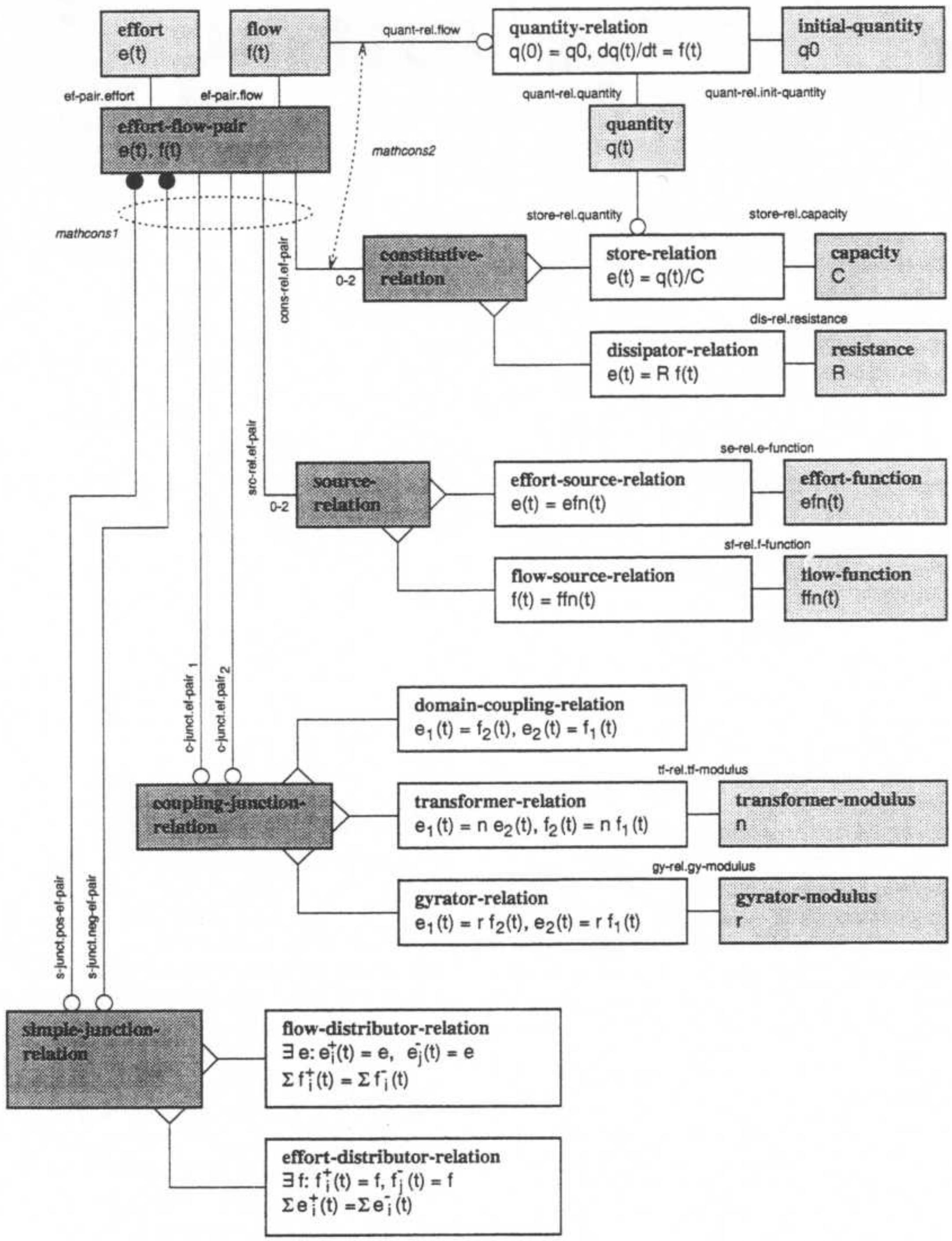

Figure 6: Conceptual scheme of the mathematical relation ontology.

is printed inside the box in bold face and below it the name of the class to which it belongs.

At the top of the figure, the physical process description of the gravitational force is shown. The ontology mapping relations *multiport.math-rel* and *bond.ef-pair* associate the multiport *Se* and the bond b_1 to the mathematical relation *Se–rel* and the effort-flow-pair efp_1.

Related to the effort source relation is an EngMath time dependent physical quantity that defines the amount of action the effort source supplies. The effort-flow-pair must off course be the same one the bond b_1 is mapped to (*mapcons1*). The effort and flow of this pair are time dependent physical quantities. The mapping relation between the physical domain and the dimension of these quantities is not shown in the figure. The existence of an effort source relation implies that there must be an EngMath mathematical relation stating that the effort of the effort-flow-pair is equal to the effort function (*mapcons2*). In the ontology, this is implemented as follows:

$$\begin{aligned} & \mathit{instance\text{–}of}(\mathit{Se\text{–}rel}, \mathit{effort\text{–}source\text{–}relation}) \\ & \quad \wedge\ \mathit{src\text{–}rel.ef\text{–}pair}(\mathit{Se\text{–}rel}, \mathit{efp}_1) \\ & \quad \wedge\ \mathit{ef\text{–}pair.effort}(\mathit{efp}_1, e_1(t)) \\ & \quad \wedge\ \mathit{src\text{–}rel.e\text{–}function}(\mathit{Se\text{–}rel}, \mathit{efn}(t)) \\ & \rightarrow e_1(t) = \mathit{efn}(t) \end{aligned} \tag{7}$$

Summarizing. PhysSys's mathematical relation ontology defines the part of mathematics required to describe physical behaviour of a system mathematically. The strict separation of mathematics and physical processes improves reusability of the ontology. A simulation tool for instance, has to translate a set of mathematical equations to a sequence of assignment statements in order to run a simulation, but does not need to know what physical processes are simulated. Because the mathematical relation ontology is free of references to the physical process ontology, it can be used for an ontology for simulation tools without changes. Only mapping relations to the assignment statements has to be added.

5 Conclusions

In this paper, a formalization of a part of systems theory, physical processes and the mathematics to describe physical behaviour was given. Important aspects of the ontology are its genericity and the reuse of other ontologies. Two properties have made this possible:

- The hierarchical structure.
- The separation of different views on one domain.

In the next paragraphs some examples of its genericity are given.

Partial Information. In the modelling tool QuBA [16, 18], the ontologies of the three views were used to build a system that supports evolutionary modelling. In evolutionary modelling, after an initial description of the component view on a model has been made, the model is further developed by incremental specification of decompositions of components, physical processes and mathematical relations. Besides the incremental aspects, the different perspectives also localize the effects of changes in the model. Parameter changes for instance, only affect the mathematical view and when a physical process is added, this only leads to changes in the process- and mathematical views. The QuBA system furthermore offers the user facilities to check the consistency of the model, to assign causality manually or automatically, and to detect causal conflicts, derivative causalities and algebraic loops. All these inferences are supported by the ontologies, or an extension of it which exists but has not been implemented in Ontolingua yet.

In the OLMECO project [19, 1], a database for reusable model fragments has been developed. The three views presented here served as a basis for the conceptual schemata of this database. Because the views are independent, the library is capable of storing models which only have one or two perspectives on a system. Models used by simulation tools, for instance, only have mathematical data. Also, a model fragment consisting of more than one view can be used partially by exporting only one or two views.

Reuse. In this paper we saw that there are two types of inclusion of ontologies: *hierarchical* inclusion and *mapped* inclusion of ontologies. With hierarchical inclusion, an included ontology is part of the domain formalized by the ontology one step higher in the hierarchy. Examples are mereology which is included in the topologogical ontology, topology which is included in the component view and the EngMath ontology which is included in the ontology of mathematical relations. Ontologies high in the hierarchy are the least domain dependent and are the most likeable to be reused.

In the component ontology, the hierarchy is *mereology-topology-components*. The mereological ontology is the most generic. It can not only be used for components, but for any individual that can be decomposed into parts. Examples are listed in Table 4 (above). The topological ontology is a bit less generic. It deals with individuals which can be connected in some way. Table 4 (below) lists some part-whole relations with possible connectivity relations. The least generic ontology is the component ontology: the connections between components are typed and components cannot share subcomponents.

The second form of inclusion combines different perspectives on the same domains. The concepts in the

Whole	Part
a (certain man)	his head
a head	the nose
an insect's life	its larval stage
a novel	its first chapter

Part-Whole relation	Possible Connectivity
components–subcomponents	exchanges energy with
countries–provinces–cities	trades goods with
countries–provinces–cities	is connected by railroad
WAN–LAN	exchanges information with

Table 4: Examples of part–whole relations (above) and connections (below).

different views are related to each other by mapping relations. Because the ontologies can be used individually, they can be included in other ontologies easily. Only the mapping relations will be different. An example of this is an ontology for simulation tools. Such an ontology can import the mathematical ontology described here, and map it to other concepts it needs.

In Ontolingua, only hierarchical dependency between ontologies can be made explicit. Because we think that mapped inclusion is a useful way of structuring ontologies, we would like to be able to specify this kind of inclusion in Ontolingua explicitly.

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 9th International Knowledge Acquisition Workshop KAW'95 (Banff, Alberta, 26 February - 3 March 1995), Volume 2*, pages 39.1–39.15, University of Calgary, 1995. SRDG Publications.

[2] J.S. Bell. *Speakable and Unspeakable in Quantum Mechanics.* Cambridge University Press, Cambridge, UK, 1987.

[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] H. Burkhardt and B. Smith, editors. *Handbook of Metaphysics and Ontology.* Philosophia Verlag GmbH, München, 1991. Two Volumes.

[5] B. L. Clarke. A calculus of individuals based on 'connection'. *Notre Dame Journal of Formal Logic*, 22(3):204–218, 1981.

[6] A. den Ouden, J.L. Top, and J.M. Akkermans. EBIB - Dynamic Model Components for a Heating Systems Library. Report ECN-C-94-027, Netherlands Energy Research Foundation ECN and University of Twente, Petten (NL), 1994.

[7] B. Falkenhainer and K. D. Forbus. Compositional modeling: Finding the right model for the job. *Artificial Intelligence*, 51:95–143, 1991.

[8] P.-J. Gailly, R. Fisset, and S. Konečni. OLMECO library software requirements document, initial version. Esprit P6521 OLMECO Report SRD 1.0, BIM, Brussels, April 1994.

[9] T.R. Gruber. Ontolingua: A mechanism to support portable ontologies (version 3.0). Knowledge systems laboratory ksl technical report, Stanford University, Palo Alto, CA, 1992.

[10] T.R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5:199–220, 1993.

[11] 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.

[12] M. Jammer. *The Philosophy of Quantum Mechanics.* John Wiley, New York, 1974.

[13] D.C. Karnopp, D.L. Margolis, and R.C. Rosenberg. *System Dynamics: A Unified Approach.* John Wiley & Sons, New York, 1990. Second Revised Edition.

[14] W. V. Quine. *From a Logical Point of View.* Harper & Row Publishers, New York, 1961. Second Revised Edition.

[15] Peter Simons. *Parts, A Study in Ontology*, chapter 1–3. Clarendon Press, Oxford, 1987.

[16] J.L. Top. *Conceptual Modelling of Physical Systems.* PhD thesis, University of Twente, Enschede, The Netherlands, September 1993. ISBN 90-375-0291-1.

[17] J.L. Top and J.M. Akkermans. Computational and physical causality. In *Proceedings IJCAI-91*, pages 1171–1176, San Mateo, CA, 1991. Morgan Kaufmann Publishers.

[18] J.L. Top and J.M. Akkermans. Tasks and ontologies in engineering modelling. *Int. J. Human-Computer Studies*, 41:585–617, 1994.

[19] J.L. Top, A. Breunese, J. van Dijk, J. Broenink, and H. Akkermans. Conceptual schema of the OLMECO library. OLMECO deliverable, ESPRIT project 6521 OLMECO/WP3.3/ECN/01/2.0, ECN and University of Twente, 1994.

[20] A. Yazman, P. Seigle, J. van Dijk, S. Stramigioli, P. Weustink, J. Top, C. Masson, FAGOR, and IKERLAN. Application elementary models; development of automotive application component models - power steering systems; development of machine tool component models. Esprit P6521 OLMECO Deliverables 2/1 and 2/2, OLMECO Consortium, April 1994.