Data & Knowledge Engineering 20 (1996) 347-383 – North-Holland, Elsevier.

Part-Whole Relations in Object-Centered Systems: An Overview

Alessandro Artale [†‡] – artale@ladseb.pd.cnr.it,
Enrico Franconi [§] – franconi@irst.itc.it,
Nicola Guarino [‡] – guarino@ladseb.pd.cnr.it,
Luca Pazzi [¶] – pazzi@c220.unimo.it

Abstract

Knowledge bases, data bases and object-oriented systems (referred to in the paper as Object-Centered systems) all rely on attributes as the main construct used to associate properties to objects; among these, a fundamental role is played by the so-called *part-whole* relation. The representation of such a structural information usually requires a particular semantics together with specialized inference and update mechanisms, but rarely do current modeling formalisms and methodologies give it a specific, "first-class" dignity.

The main thesis of this paper is that the part-whole relation cannot simply be considered as an ordinary attribute: its specific ontological nature requires to be understood and integrated within data modeling formalisms and methodologies. On the basis of such an ontological perspective, we survey the conceptual modeling issues involving part-whole relations, and the various modeling frameworks provided by knowledge representation and object-oriented formalisms.

1 Introduction

Despite their different purposes and the tendency to mutual isolation, knowledge base, data base and object-oriented communities heavily rely on conceptual models which have a lot in common. They can be collectively classified as Object-Centered Systems, since they all organize the knowledge around the unifying abstract notion of *objects*. *Classes*, *attributes* and *individuals* could be considered as the main conceptual building blocks. A class is a description which gathers common features within a collection of *individuals*. From a logical point of view, it is a unary predicate which denotes a set of individuals,

[†]Dip. di Sistemi ed Informatica, Università di Firenze, Italy
[‡]Ladseb-CNR, I-35127 Padova PD, Italy
[§]Knowledge Representation and Reasoning group, IRST, I-38050 Povo TN, Italy
[¶]Department of Engineering Sciences, I-41100 Modena MO, Italy

called the instances of the class. Attributes are intended to describe the relevant properties of instances belonging to a particular class, and they usually have the semantics of generic relations – associating such instances with other individuals. Classes are described as clusters of attributes, and explicit or implicit subsumption is used to describe generalization/specialization relationships among classes. To be as general as possible, we shall use the term *class*, currently used in object-oriented systems, as synonymous of *frame* in frame-based languages, of *entity* in E-R data models, of *node* in Semantic Networks, of *concept* in Description Logics. The term *object* will be sometimes used as a synonymous of *individual*, but – when referring to the object-oriented approach, it will be preferably intended as denoting a data structure rather than an individual in the world. The term *attribute* will be synonymous of *slot*, *link*, and *role*.

Among attributes, a fundamental role is often played by various forms of *part-whole* relations, which contribute to describe the *composite structure* of the instances of a class. The representation of such a structural information usually requires a particular semantics together with specialized inference and update mechanisms, but rarely do current modeling formalisms and methodologies give it a specific, "first-class" dignity. A common way to interpret the role played by single attributes within a class description is by the "has-a" paraphrase: a house has an owner, a price, a location, as well as a door, a roof. As discussed by Wilensky [80], who dubbed it as the "belonging fallacy", such a view risks to hide the nature of the relationship between an attribute and the class it applies, flattening all attributes to be just *arbitrary* relations (see [31]). In this way, when modeling the characteristics of a whole like *house*, it is difficult to distinguish parts like *door* from other attributes like *price* or *color*, and ad-hoc mechanisms may be required in order to perform simple inferences, such as deriving that parts of parts are themselves parts, or computing the total weight of an object from the weight of its parts.

The main thesis of this paper is that part-whole relations cannot simply be modeled by ordinary attributes like *price* or *color*: their specific ontological nature requires to be studied, understood and integrated within knowledge and data modeling formalisms and methodologies, without leaving all the burden to the user. In other words, representation formalisms – or at least conceptual analysis methodologies – should take its peculiar meaning into due account. As discussed in [32], such a choice corresponds to an *ontological level* characterization of the domain structuring primitives.

We shall examine here in some detail the peculiarities of part-whole relations which motivate their specific introduction as "first-class citizens" within object-centered formalisms. Such peculiarities are well known to linguists and philosophers, but unfortunately they tend to be ignored by the data and knowledge engineering community. After a review of relevant philosophical and linguistic issues (section 2), we shall focus on the conceptual modeling process involving part-whole relations within real world domains (section 3), and on the various frameworks provided by current knowledge representation (section 4) and object-oriented systems (section 5).

2 Parts and Wholes

Theories of parts and wholes have been studied since early philosophers have started to investigate the essence of the physical world. The first formal accounts appeared only at the beginning of our century, dating back to the independent works on *Mereology* by the polish logician Stanisław Leśniewski and on the *Calculus of Individuals* by Leonard and Goodman. Nowadays, there is an accepted minimal unified view on the theory of parts and wholes: Classical or General Extensional Mereology (GEM).

Basically, GEM introduces a *proper-part-of* binary relation "$\prec$" as a strict ordering relation - i.e., transitive and asymmetric – on the domain, satisfying a set of additional principles. The most important of them may be informally summarized as follows:

`Extensionality:` Two individuals are identical if and only if they have the same parts.

`Principle of Sum:` There always exists the individual composed by any two individuals of the theory – i.e., the mereological sum.

`Supplementation:` If an individual x is a proper-part-of an individual y, then a different individual z exists which is the missing part from y.

Clearly, the *part-of* relation "$\preceq$" may be defined as the disjunction between the proper-part-of and the identity relations; it is therefore a partial ordering relation.

It turns out that the basic principles of GEM are inappropriate when considering real domains of application of the theory. Sometimes, the extensionality and sum principles are too strong. How to deal with individuals described not only by means of their parts, but also by means of their properties? On the other hand, how to deal with individuals which maintain their identity while loosing or acquiring some part at different times (as in the case of a person who loses a hair)? In both cases, GEM is not able to account for the subtle aspects related to identity principles: in the former case, two individuals may have different identifying properties while being made of the same parts; in the latter case, two individuals may have the same identifying properties while being made of different parts. Again, how to deal with real domains where general sums do not exist? In the real world, the individual denoting the object composed by Johns' left hand and the door of your house does not make any sense. See [71] for a comprehensive overview of the relevant philosophical literature in this area.

Moreover, GEM appears to be too weak to capture the notion of a *whole* as a one-piece entity, as opposed to a scattered entity made up of several disconnected parts. This is the reason why various forms of integrating mereology within non-extensional and/or topological frameworks are being studied. See [78] for a more recent systematization of various axiomatizations of mereology and their relationships with topology.

Despite the subtleties of the various mereological frameworks, object-centered formalisms have considered a simple view of the theory of parts and wholes, by taking its only apparently uncontroversial aspect: the part-of relation is at least a partial ordering relation. However, even this minimal aspect introduces problems, as explained in the next section.

2.1 Kinds of Part-Whole Relations

Among the algebraic properties peculiar to the notion of part, the most discussed one is *transitivity*. We can say, for example, that the finger is part of the hand, the hand is part of the human body, and then infer that the finger is part of the human body, too. However, various authors [60, 52, 18, 81, 39, 29] have noticed that transitivity does not always hold, trying to characterize where transitive inferences are valid or not. As an example, consider the following case: an arm is part of a musician, the musician is part of an orchestra, but it would sound a bit strange to state that the arm is part of the orchestra.

On the basis of linguistic and cognitive studies, Winston, Chaffin and Herrmann (WCH) in [81] proposed a distinction among various kinds of specialized part-whole relations, with the aim of overcoming such apparent transitivity paradoxes, ascribed to the mixing of different kinds of part-whole relations. The main idea was to capture the different ways in which parts contribute to the structure of the whole, by introducing six different types of meronymic relations, distinguishable on the basis of the three criteria of *functionality*, *homeomericity* and *separability*:

> "[..] *Functional* parts are restricted, by their function, in their spatial or temporal location. For example, the handle of a cup can only be placed in a limited number of positions if it is to function as a handle. *Homeomerous* parts are of the same kind of thing as their wholes, for example 'slice-pie', while non-homeomerous parts are different from their wholes, for example, 'tree-forest'. *Separable* parts can, in principle, be separated from the whole, for example, 'handle-cup', while inseparable parts cannot, for example 'steel-bike'." [81]

The following is a summary of the part-whole relationships proposed by WCH:

- `Component/Integral-Object`: Integral objects are characterized by having a structure, while their components are separable and have a specific functionality. For example, "Wheels are parts of cars" or "Phonology is part of linguistics".

- `Member/Collection`: Captures the notion of membership in a collection. Members do not play any functional role with respect to the whole they are part of, but they can be separated from it. For example, "A tree is part of a forest".

- `Portion/Mass`: The whole is considered as a homogeneous aggregate and its portions are similar to it (homeomerous) and separable, as in "This slice is part of a pie".

- `Stuff/Object`: It expresses constituency of things and can be paraphrased using *is partly* or *is made of*, as in, "The bike is partly steel". Essentially, in order to distinguish these part relations from the other ones, the argument is that the stuff of which a thing is made of cannot be separated from the object, it does not have any functional role nor is it homeomerous.

- `Feature/Activity`: Designates a phase of an activity. A phase, like a component, has a functional role but it is not separable. For example, we can say that "Grasping is part of stacking objects".

- `Place/Area`: It is a spatial relation among regions occupied by different objects. Like the portion/mass relation, the place/area is homeomerous since every part of a region is similar to the whole region, but they cannot be separated. For example, we can say that "An oasis is part of a desert".

The above distinctions among different kinds of part-whole relations allow the authors to offer a reason to the apparent lack of transitivity in examples like the one cited at the beginning of this section: "[..] So long as we are careful to keep to a single sense of *part* [..] it seems that the part-whole relation is always transitive. However, when we inadvertently mix different meronymic relations problems with transitivity arise" [81]. The strangeness of musician's arm example is due to a mixing of two different part-whole relations. Saying that the arm is part of a musician involves a component/object reading of the part relation, while stating that the musician is part of the orchestra entails a member/collection reading.

Despite its role as a first important contribution to the understanding of the cognitive nature of part-whole relations, a number of criticisms have been moved to the WCH approach. First of all, although the WCH approach seems to exclude the existence of a single, very general *part-of* relation assumed to be transitive, various authors have stressed the fact that the different *part-whole* relations introduced by WCH can be seen as specializations of a single, general part-of relation satisfying the basic axioms of mereology [71, 78][1]. The particular behaviour of the different *part-whole* relations may lie, among other things, in the ontological nature of both the whole – including notions like integrity – and the part. Important problems of composition (including transitivity as a special case) may arise for part-whole relations[2], but they do not affect the transitivity of the part-of relation.

A further criticism to the WCH approach comes from the fact that their theory is motivated by linguistic examples which are sometimes questionable in their interpretations, and therefore some of the distinctions proposed by the authors (which are not supported by a formal evidence) may be obscure or debatable. In [39], Iris, Litowitz and Evens offer a comprehensive overview of these linguistic issues, and propose to reduce the six cases introduced in the WCH approach to four basic schemes, namely *component-whole*, *segment-whole*, *member-collection*, *subset-set*. In [29, 28], Gerstl and Pribbenow move a further step towards the understanding of part-whole relations by isolating three basic kinds of wholes on the basis of their compositional structure, namely *masses*, *collections* and *complexes* (whose parts are respectively called *quantities*, *members* and *components*), and two further ways of isolating parts of them on the basis of intrinsic properties (*portions*) or external schemes (*segments*).

3 The Conceptual Modeling Perspective

Since the early stages of the conceptual modeling process, the peculiarities of the notions of part and whole pose a number of problems, which require careful choices. Most of these

[1]In the following, we shall use the terms *part-of* and *part-whole* consistently to this interpretation.

[2]We are not aware of any substantial work done on composition tables for different part-whole relations: some preliminary ideas appear in [8, 64].

problems are independent from the particular part-whole relation considered, since they regard the very general notion of part-of. In our opinion, the minimal requirements of a conceptual model able to capture the ontological nature of both parts and wholes can be summarized as follows:

- Explicit introduction of (complex) wholes in the model.
- Clear distinction between parts and other attributes of a whole.
- Built-in transitivity of parts.
- Possibility to refer to parts by generic names.
- Capability to express "vertical" relationships between the parts and the whole.
- Capability to express "horizontal" constraints among the parts of a whole.

Transitivity issues have already been discussed above, and their impact on knowledge representation formalisms will be analyzed in section 4. We will focus in this chapter on the remaining requirements.

3.1 Implicit versus Explicit Wholes

A preliminary issue that has to be addressed when modeling a complex world regards the identification of relevant domain elements. This task is not so trivial as it may appear at a first sight, since, when dealing with interrelated individuals, it is often not clear whether it is worth while either modeling them by an implicit web of references or rather considering such individuals as parts of a whole – introducing therefore the whole as an explicit object. Two different modeling philosophies result:

- *Implicit* modeling. In this case the connections between the interrelated individuals are modeled by means of local attributes, and the overall knowledge pertaining to the whole is distributed among different objects.
- *Explicit* modeling. In this case a new object, holding part-of links to the interrelated objects, is made explicit. The overall knowledge pertaining to the whole is held in the whole itself.

As an example of the *implicit approach*, consider the case of a family, where the interrelationships between the spouses *john* and *mary* are expressed as follows:

john : `AGE` : 35,	*mary* : `AGE` : 33,
`SEX` : `Male`,	`SEX` : `Female`,
`HEIGHT` : 175,	`HEIGHT` : 165,
<u>`WIFE`</u> : *mary*.	<u>`HUSBAND`</u> : *john*.

Observe that the relationship binding the two objects *john* and *mary* is realized by an attribute in the object *john* keeping a reference to *mary* (`WIFE`), and vice versa via the attribute `HUSBAND`.

According to the *explicit approach*, the example above would be modeled by introducing an additional object – denoting Mary's and John's family – which stands for the whole:

john : AGE : 35,	*mary* : AGE : 33,	*m-j-family* : <u>WIFE</u> : *mary*,
SEX : Male,	SEX : Female,	<u>HUSBAND</u> : *john*.
HEIGHT : 175.	HEIGHT : 165.	

Notice that the two attributes, WIFE and HUSBAND, appear now in the object which models the whole with a different meaning, since they model a relation which exists between a family and a particular person part of it[3].

The choice between the two approaches above is not just a matter of point of view, since the implicit choice may often have drawbacks in terms of *reusability*, *understandability* and *extendibility*, which are considered as crucial modeling quality factors [53].

In terms of *reusability*, it is evident that a particular modeling abstraction is more reusable if it contains only those elements which are specific to its nature, and independent from the context. For example, suppose we have to model a gear, consisting of a wheel a which moves a wheel b. If we stick to the implicit approach, using an attribute *moves* in a whose filler is b, we cannot reuse a in different contexts, say that of a wheel moving two or more wheels, unless by modifying the structure of a adding new attributes for the other wheels moved.

Regarding *understandability*, if an object description contains only the information specific to its nature, then it is likely to be easily understandable. On the other hand, if attribute links transcend the boundary of a single object, then understandability is jeopardized. If we have n object descriptions which are related in the implicit manner, we have to examine all of them in order to discover that there is a *whole* behind them. If we have instead a single object pointing to n different objects as components, then the interpretation is straightforward.

Finally, in terms of *extendibility*, we can observe that, in the example above, if the gear is modeled explicitly as an object relating wheels a and b, it can be easily extended by adding a third related wheel c without touching the description of a.

3.2 Names of Parts

As we have seen in the preceding section, while modeling composite individuals it seems natural to give specific names to attributes denoting parts: the attributes WIFE and HUSBAND denote two specific HAS-PART relations in the context of the composite individual *m-j-family*. They are used to refer to parts, and at the same time they carry extra-information about the role and the nature of such parts.

In the domain of artifacts, we can consider for example the individual *car1* of type Car. A first modeling choice amount to saying that it has a part which is a wheel:

$$\texttt{Car}(\mathit{car1}) \land \mathit{wheel1} \preceq \mathit{car1} \land \texttt{Wheel}(\mathit{wheel1}).$$

On the other hand, a second modeling choice can make use of the attribute HAS-WHEEL[4]:

[3] On the ways of distinguishing these part-denoting attributes from other possible attributes of "*m*ary and *j*ohn's-*family*" see the next section.

[4] We prefer here to use the name HAS-WHEEL instead of WHEEL in order to avoid a possible confusion

$$\texttt{Car}(\mathit{car1}) \wedge \texttt{HAS-WHEEL}(\mathit{car1}, \mathit{wheel1}) \wedge \texttt{Wheel}(\mathit{wheel1}).$$

In this latter case, we have the hidden assumption that HAS-WHEEL is a kind of part attribute:

$$\texttt{HAS-WHEEL}(\mathit{car1}, \mathit{wheel1}) \longrightarrow \mathit{wheel1} \preceq \mathit{car1}.$$

Under such assumption, the latter formulation implies the former. In this way, one can build a model by making an explicit use of part-names as attributes or alternatively by using generic part attributes:

Car : HAS-WHEEL : Wheel, HAS-ENGINE : Engine, HAS-COLOR : Red.	Car : HAS-PART : Wheel, HAS-PART : Engine, HAS-COLOR : Red.

It is clear that, in order to distinguish between meronymic (like HAS-WHEEL) and non-meronymic (like HAS-COLOR) attributes, it is necessary to mark the part-names used as attribute. It should be noted, however, that the formal properties typical of the general PART-OF relation – like its transitivity, for instance – do not necessarily hold for all of its sub-relations, and therefore they shouldn't be automatically associated to all specific meronymic attributes. Only expressive enough formalisms – like, e.g., description logics with hierarchies of attributes – can correctly handle the general HAS-PART relation, with its formal properties, together with its specialized sub-relations. In this way, it is possible both to have an easy access to the different types of parts of a composite individual – denoted by the named part attributes – and to specify other peculiar characteristics of such specific part attributes – like, e.g., cardinality or domain restrictions.

Both modeling choices presented above are followed in practice. However, there are cases where the choice of having an explicit part attribute hierarchy is definitely the most appropriate. Consider the part-names introduced by WCH (like, e.g., MEMBER), or those like CENTER, TOP, TIP, BOTTOM, EDGE. They can hardly be thought of as "stand-alone" concepts like Wheel or Engine: the reason is that they denote *dependent* names, in the sense that, in order to such names denote particular objects, these objects must already be part of a whole. For instance, if X is a edge, something else of which X is a part must exist [72]. Due to this dependence, these part-names have an intrinsic relational meaning, and they should naturally correspond therefore to attributes. Therefore, to capture the dependence of these part relationships from the general part-of relation we need to organize them in a role hierarchy.

3.3 Vertical and Horizontal Relationships

In order to correctly model the notion of a whole, we cannot limit ourselves to describing its meronymic structure, but we should be able somehow to express *how* the whole is related to the parts, and how the parts are "glued together" to form a whole. Again, we point to [71] (chapter 9) for some general issues related to the notion of *integrity*,

with the unary predicate Wheel.

while we focus here on our conceptual modeling perspective. Among the various integrity relationships holding within a whole, the following distinctions can be made:

- "Vertical" relationships.
  - Dependence relationships between the *existence* of the whole and the existence of (a certain number of) parts (and vice versa).
  - Dependence relationships between the *properties* of the whole and the properties of the parts (and vice versa)
- "Horizontal" relationships.
  - Constraints among parts which characterize the integrity of the whole.

We do not discuss in detail the various cases of horizontal relationships, since, although extremely important for capturing the notion of a whole, they find a little space in current modeling formalisms (see however sections 4.6, 4.7.5 and 5.5). It will suffice here to report the classical example of an arch which can be considered as a whole made out of inter-related parts. For an arch, its parts should satisfy the following constraints: the lintel is `supported-by` the uprights, and each upright is `on-the-side-of` and `not-connected-with` the other.

Vertical relationships deserve here more attention. The first class of vertical relationships mentioned above refers to what may be called *existential* dependence, discussed at length in [71]. According to Simons, we say that an individual is *rigidly dependent* on another individual if the former cannot exist unless the latter exists; further, an individual is *generically dependent* on a *class* if, in order the individual to exist, an instance of such class has to exist. As an example of rigid dependence, consider the relationship between a person and its brain: if we change the brain, we cannot assume that the person is the same any more. On the other hand, we can assume that a person is only generically dependent on his/her heart, since this can be replaced. For our modeling purposes, we can focus the attention on generic dependence, considering the following special cases:

- The whole is generically dependent on a particular class of parts: with a bit of philosophical freedom, we shall refer to these parts as *essential parts*[5].
- A part is generically dependent on the whole: in this case, the part will be called a *dependent part*.
- There exists at most one whole containing a particular part: such part is in this case *exclusive* for that whole.

The second class of vertical relationships refers to dependence relationships between properties cases:

[5]A more accurate definition of essential parts would require modal concepts, and should be based on rigid dependence rather than on generic dependence.

- Properties which the whole inherits from its parts: in many cases, for instance, the whole is defective if one of its parts is defective.

- Properties which the parts inherit from the whole: for instance, within a certain granularity, certain locative properties (e.g., "being on the table") of the whole hold also for its parts.

- Properties of the parts which are systematically related to properties of the whole: for instance, the region occupied by a single part is always inside the region occupied by the whole, and the weight of a single part is always less than the weight of the whole.

4 The Knowledge Representation Perspective

After the general analysis outlined above, we shall discuss in details in this chapter various modeling proposals related to parts and wholes in the area of Object-Centered Knowledge Representation Systems. Part-whole relationships has been used for describing the structure of objects since the beginning of knowledge representation systems. The early solutions, briefly discussed in the section 4.2, made use of primitive attributes without considering their logical properties [35, 69, 9]. They are still very popular, since they do not require any special addition to the existing systems.

Within this chapter, however, we shall give a special emphasis to those systems where the introduction of part-whole relations is accompanied by an appropriate semantics and by specific inferential mechanisms. We shall adopt here the common formal framework of description logics, briefly introduced in the next section. One of the main goals here is to make justice of the misunderstanding which sees partonomy and taxonomy as orthogonal each other: we will show the strong – semantically motivated – interrelationships between them.

4.1 Description Logics

Description Logics[6] are formal languages designed for a logical reconstruction of representation tools such as *frames*, *object-oriented* and *semantic* data models, *semantic networks*, KL-ONE-like languages [83], *type systems*, and *feature logics*. Nowadays, description logics are being considered the most important unifying formalism for the many object-centered representation languages used also in areas different from Knowledge Representation. In particular, [4, 13, 15, 14] propose a formal mapping between description logics and Object-Oriented formalisms. Important characteristics of description logics are high expressivity together with decidability and completeness, which guarantee that reasoning algorithms always terminate with the correct answers. Differently from object-oriented systems, description logics do not stress the representation of the behavioral aspect of information, for which they are still considered inadequate.

[6]Description Logics have been also called *Frame-Based Description Languages*, *Term Subsumption Languages*, *Terminological Logics*, *Taxonomic Logics*, *Concept Languages* or KL-ONE-*like languages*.

$$
\begin{array}{llrcll}
C, D & \to & A \mid & A^{\mathcal{I}} & \subseteq & \Delta & \text{(atomic concept)} \\
& & \top \mid & \top^{\mathcal{I}} & = & \Delta & \text{(top)} \\
& & \bot \mid & \bot^{\mathcal{I}} & = & \emptyset & \text{(bottom)} \\
& & \neg C \mid & (\neg C)^{\mathcal{I}} & = & \Delta \setminus C^{\mathcal{I}} & \text{(complement)} \\
& & C \sqcap D \mid & (C \sqcap D)^{\mathcal{I}} & = & C^{\mathcal{I}} \cap D^{\mathcal{I}} & \text{(conjunction)} \\
& & C \sqcup D \mid & (C \sqcup D)^{\mathcal{I}} & = & C^{\mathcal{I}} \cup D^{\mathcal{I}} & \text{(disjunction)} \\
& & \forall R.C \mid & (\forall R.C)^{\mathcal{I}} & = & \{i \in \Delta \mid \forall j.\, R^{\mathcal{I}}(i,j) \Rightarrow C^{\mathcal{I}}(j)\} & \text{(universal quantifier)} \\
& & \exists R.C \mid & (\exists R.C)^{\mathcal{I}} & = & \{i \in \Delta \mid \exists j.\, R^{\mathcal{I}}(i,j) \wedge C^{\mathcal{I}}(j)\} & \text{(existential quantifier)}
\end{array}
$$

Figure 1: Syntax and semantics for $\mathcal{ALC}$ concepts.

In this section we give a very brief introduction to description logics. With respect to the formal apparatus, we stick to the concept language $\mathcal{ALC}$ introduced in [67] and further elaborated for example in [23, 12, 24, 19]. In this perspective, description logics are considered as a *structured* fragment of predicate logic; $\mathcal{ALC}$ is the minimal description language including full negation and disjunction – i.e., propositional calculus. As it is expected, basic types of the language are *concepts*, *roles* and *individuals*. According to the syntax rules of column 1 in figure 1, $\mathcal{ALC}$ *concepts* (denoted by the letters C and D) are built out of *atomic concepts* (denoted by the letter A) and *roles* (denoted by the letter R).

Let us consider, as an example, the following generic representation of a student and its possible formalization in description logic:

$$
\begin{array}{ll}
\texttt{Student} : \texttt{Person}, & \\
\qquad \texttt{NAME: String}, & \\
\qquad \texttt{ADDRESS: \{String\}}, & \\
\qquad \texttt{ENROLLED: Course}. &
\end{array}
\qquad
\begin{array}{rl}
\texttt{Student} \doteq & \texttt{Person} \sqcap \\
& \exists\texttt{NAME.String} \sqcap \\
& \forall\texttt{ADDRESS.String} \sqcap \\
& \exists\texttt{ENROLLED.Course}
\end{array}
$$

a student is a person, it has a name[7] and possibly many addresses – which are strings – and it is enrolled in at least a course. The attributes of the description can be quantified either universally or existentially: any address of a student is always expressed like a string in the representation, whereas it is possible that a student is enrolled in something which is not a course – e.g., in the army – being necessary just his/her enrollment in at least one course in order to be a student.

Moreover, as we have said before, it may be possible to have disjunction and negation, like in the following example:

$$(\exists\texttt{TEACHES.Course}) \;\sqsubseteq\; (\neg\texttt{Undergrad} \sqcap (\texttt{Researcher} \sqcup \texttt{Professor}))$$

meaning that anyone teaching some course should be graduated and either a researcher or a professor. According to the definition above, not every graduated researcher or professor necessarily teaches some course, since the "$\sqsubseteq$" relation is meant as an implication.

The individual *john* is an *instance* of the student frame, whose name is John, living in Abbey Road and attending the Computer Science course 415:

[7] The `NAME` role should be defined as *functional*, since people have only one name.

john: `Student`,
`NAME`: "John",
`ADDRESS`: "Abbey Road...",
`ENROLLED`: *cs415*.

Its representation in description logics is a collection of statements, where unary and binary predicates stand for concepts and roles:

`Student(john)`,
`NAME(john, "John")`, `String("John")`,
`ADDRESS(john, "Abbey Road...")`, `String("Abbey Road...")`,
`ENROLLED(john, cs415)`, `Course(cs415)`.

It is clear that a simple deduction from the above knowledge is that John is a person, i.e., that `Person(john)`.

Let us consider now the formal semantics of a description logic. We define the *meaning* of concept expressions as sets of individuals – as for unary predicates – and the meaning of roles as sets of pairs of individuals – as for binary predicates. Formally, an *interpretation* $\mathcal{I}$ is a pair $(\Delta, \cdot^{\mathcal{I}})$ consisting of a set Δ of individuals (the *domain* of $\mathcal{I}$) and a function $\cdot^{\mathcal{I}}$ (the *interpretation function* of $\mathcal{I}$) which maps every concept to a subset of the domain, every role to a subset of pairs of the domain and every individual into a different element of the domain (satisfying the Unique Name Assumption) such that the equations of column 2 in figure 1 are satisfied.

An interpretation $\mathcal{I}$ satisfies $C \sqsubseteq D$ if and only if the interpretation of C is included in the interpretation of D, i.e., $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. A concept C is *subsumed* by a concept D (written $C \sqsubseteq D$) if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ in every model. For example, the concept `Person` subsumes the concept `Student` – `Student`$\sqsubseteq$`Person`, since every student is also a person.

A set of TBox axioms and ABox statements is called a *knowledge base*. Terminological axioms are of the form $C \sqsubseteq D$, with C and D are generic concept expressions. Note that concept definitions of the type $A \doteq C$ – where A is an atomic concept – can be reduced to $(A \sqsubseteq C) \wedge (C \sqsubseteq A)$. A *terminology* (TBox) is a finite set of terminological axioms. ABox statements are referred to as *assertions*; an assertion is a statement of the form $C(a)$ or $R(a, b)$.

4.2 Preliminary Works

The first attempts to introduce part-whole relations in knowledge representation date back to the late '70s. The semantic network proposed by Philip Hayes in [35] – called CSAW, for "Choosing the Senses of Ambiguous Words" – makes use of the meronymic relation as one of its most important structural components. A `PART-OF` link is used to describe the structure of objects. For example, to represent the class `Human`, Hayes first individuates its anatomical parts – an `Arm`, a `Torso`, a `Head`; then, he connects each part to the node denoting the concept `Human` by means of the `PART-OF` role. Hayes' work gives a contribution to the broader investigation, made between the late '70s and the first '80s, aimed to clarify the meaning of nodes and links in semantic networks. Hayes proposal goes further on introducing a form of *dependence* between parts and wholes (see section 3.3) by relating them in a so called SQN structure (for Sine Qua Non structure). Thus, the fact that an arm exists only if a human possessing it exists, is expressed saying that `Human` is a SQN concept of `Arm`. A SQN structure then forms a partonomy distinct from taxonomic graphs (called generalizer structures): "[..] SQN and generalizer structures are orthogonal in the sense that SQN structure is independent and complementary to generalizer structure" [35]. For a critics to this position see section 4.5.

Schubert in [69] introduces a representation called "parts graphs" for reasoning about parts. The `PART-OF` relation is characterized as a partial order plus the extensionality axiom and the principle of sum (see section 2). Furthermore, parts are pairwise disjoint and jointly exhaustive; thus, they express a form of *partitioning*. The work uses part graphs for representing and reasoning about parts of given individuals. Although the problem of deducing part-of relations – like checking whether the individual a is part of the individual b – is showed to be co-NP-complete, the author suggests to put additional constraints on part graphs in order to permit efficient information extraction. The notion of *closed graph* is defined, allowing for linear space-time algorithms for question-answering – even if the size of the closed graph may be exponential in the size of the equivalent open part graph. The most serious limitation of this work, observed by the author too, is that parts are not considered at the abstract level of concepts: there is no way to express such a generic knowledge, like saying that an `Arm` is part of any `Human`, and consequently reason on this representation.

Within the application domains which need an explicit representation of medical knowledge, Haimowitz, Patil and Szolovits in [33] describe a system for diagnosis of acid-base electrolyte disorders with the aims of formulating hypothetical diagnosis of a patient by using the terminological language NIKL [68]. The authors point out that representing anatomical relationships in NIKL is particularly troubling, since part-whole relations could not be represented in such a way that both the requirement of making natural associated inferences – mainly, transitivity dependent inferences – and the requirement of having a resulting knowledge base with a structure similar to the domain being modeled. As a matter of fact, one of the explicit requests is to extend terminological logics with transitive closure of roles: "[..] Being able to define one relation as the transitive closure of another would be particularly useful [..]" [33].

The NIKL builders, Schmolze and Mark, in [68] attempt to represent transitivity of part-whole relations by resorting to the transitivity of subsumption. The idea is to introduce a new concept that represents *all* the possible parts of a given class of objects, while connecting it to the related whole by means of a primitive `HAS-PART` role. For example, if you want to represent a bicycle, you may create a concept that stands for all parts of bicycles, say `Bicycle-Part`, which has `Wheel` as one of its subconcepts. Since you need to represent parts of wheels, too, you have to build another concept for these parts, says `Wheel-Part`, with, for instance, `Tire` as one of its subconcepts. As the authors say, this is not "[..] an ideal method for representing part/whole" [68]. Obvious deductions are lost: for instance, it is no more true that tires are parts of bicycles, unless you *explicitly* declare that the concept `Wheel-Part` is a subconcept of `Bicycle-Part`. This approach leaves to the designer the burden of asserting all the consequences of transitivity, which would be instead automatically inferred if part roles were modeled as transitive. Another drawback is the proliferation of concepts, due to the introduction of concepts collecting parts. For example, a concept like `Bicycle-Part` is only needed to simulate transitivity, but it does not add further structure to the domain description.

Jang and Patil recognize in [41] that, in order to deal with the part-of relation, a representation system requires transitive roles plus some notion of vertical relationships between the parts and the whole. They introduce the language KOLA with *indirect-transitive* as well as transitive roles. Specifying that a role R is *indirectly transitive via the role* S allows the system to infer R from the composition of R itself with the transitive closure of S – in symbols, $R \circ S^* \rightarrow R$. This notion permits, for example, to deduce diseases of an organ from diseases of its parts (see for more details sections 4.4 and 4.5). Despite this work suffers from the lack of both a clear semantics and a proof of correctness of the algorithms, it has however the merit to recognize the importance of explicitly dealing with the properties of the part-of relation within a knowledge representation formalism.

4.3 $\mathcal{P}$: A Language for an Engineering Application

The concept language presented in [64] by Sattler deals with composite objects in an application domain like the modeling of huge chemical plants. The demands individuated in such a technical domain call for a knowledge representation system able to handle:

- Different kinds of part-whole relations, (see section 2.1).
- Inverse of these relations.
- A composition table of different part relations.
- Specific interactions between parts and whole – for example, parts belonging *exclusively* to one whole.

The language $\mathcal{P}$ is a first attempt to formalize these demands in a description logic. The *transitivity* of the part role is obtained by means of a specific role forming operator. The transitive closure operator, first introduced in [6], applies to a given role (say R) and builds up its transitive closure (in symbols, R^*)[8]. In this way, it is possible to capture the transitivity with a proper subsumption algorithm and there is neither ambiguity nor separation between taxonomy and partonomy. For example, if the has-part role (in symbols, $\succeq$) is modeled as the transitive closure of a primitive part relation, we can describe a car by saying that it has wheels which in turn have tires as their parts:

$\succeq \doteq (\texttt{PRIMITIVE-PART})^*$
$\texttt{Car} \doteq \exists\succeq.(\texttt{Wheel} \sqcap \exists\succeq.\texttt{Tire})$

then, the fact that tires are also parts of cars: $\texttt{Car} \sqsubseteq \exists\succeq.\texttt{Tire}$ follows. In order to differentiate among various part relations, the author introduces six primitive roles: `IS-D-COMPONENT`, `IS-D-MEMBER`, `IS-D-SEGMENT`, `IS-D-QUANTITY`, `IS-D-STUFF`, `IS-D-INGREDIENT` where `D` stands for *direct* – e.g., `IS-D-COMPONENT` can be read as "*is a direct*

[8] For the semantics of the transitive and reflexive closure operator we have: $(R^*)^{\mathcal{I}} = \bigcup_{n\geq 0}(R^{\mathcal{I}})^n$, while with R^+ we denote the composition $R \circ R^*$.

component of". A composition table takes into account the interactions between the mixed relations so that, for example, the composition of Member/Collection with Component/Object results in the Component/Object relation[9]. In such a way, the generic part-roles are described as the transitive closure of the respective direct roles and the role chain obtained from the composition table. For example, the Component/Object relation is defined as:

$$\texttt{IS-COMPONENT} \doteq \big(\texttt{IS-D-COMPONENT} \sqcup (\texttt{IS-MEMBER} \circ \texttt{IS-D-COMPONENT})\big)^{+}$$

The fact that *has-part* roles (e.g., HAS-COMPONENT) are the inverse of the *is-part-of* roles (e.g., IS-COMPONENT) is expressed by means of inverse roles[10] (e.g., $\texttt{HAS-COMPONENT} \doteq \texttt{IS-COMPONENT}^{-}$).

We want make a point here on the introduction of the *direct primitive roles*. Although it is correct to define the generic part starting from the primitive notion of direct part, that relation has to be modeled more carefully with a proper semantics. Such direct-parts have to verify the definition of *immediate inferior*, taken from lattice theory: given a partially ordered set P, we say that a is an *immediate inferior* of b if $a < b$, but $a < x < b$ for no $x \in P$. In the $\mathcal{P}$ language, no formal characterization is given to the direct-part roles in order to agree with the above definition. As a matter of fact, concept expressions in $\mathcal{P}$ may have non intended models with respect to the direct-part role – e.g., $\{c \prec_d a,\ c \prec_d b,\ b \prec_d a\}$ is a valid interpretation for $\prec_d$, the generic is-direct-part-of relation. Thus, direct-part roles should affect the ABox reasoning process in order to discard non-intended models like the above one. With respect to pure TBox inferences, it is likely for the $\mathcal{P}$ language that if a $\mathcal{P}$ TBox has a model, then it has a model that conforms with the intended meaning of direct-parts. Thus, we can simply take the primitive notion of direct-part – without any specific semantics – when we compute logical consequences[11]. However, this has still to be shown formally.

Features that characterize the interdependence between the parts and wholes are expressible in $\mathcal{P}$ with qualified number restrictions on roles[12] [37]. Saying that a part is an *Exclusive part* of a whole implies that one of the direct is-part-of roles has at most one filler. For example, $\mathcal{P}$ allows us to express that a motor is an exclusive part of a car:

$$\texttt{Motor} \doteq (\leq\ 1\ \texttt{IS-D-COMPONENT}\ \texttt{Car}) \sqcap \ldots$$

With *Multi-Possessed parts* the author refers to the case of an object whose parts have a common sub-part – suppose a "System made of some Reactors that all use the same Tank"[13]:

[9] As stated by the author, such composition table is obtained from a case study on the particular domain of interest, but its validity on general domains seems to us hard to defend.

[10] Given a role R, with R^{-} we denote its inverse.

[11] The same considerations are valid when speaking of antisymmetric models, which capture the acyclicity of the part-whole relation.

[12] Qualified Number Restrictions have the form: $(\leq\ n\ R\ C), (\geq\ n\ R\ C)$ with the following semantics: $(\leq\ n\ R\ C)^{\mathcal{I}} = \{i \in \Delta \mid\ \parallel \{j \mid R^{\mathcal{I}}(i,j) \wedge C^{\mathcal{I}}(j)\} \parallel \leq n\}$, where $\parallel \cdot \parallel$ denotes the cardinality of sets.

[13] Given a role R, with R^n ($\texttt{IS-D-COMPONENT}^2$) we denote the composition $R \circ R \circ \cdots \circ R$ ($\texttt{IS-D-COMPONENT} \circ \texttt{IS-D-COMPONENT}$).

$$\texttt{System} \doteq (= \ 1 \ (\texttt{IS-D-COMPONENT}^{-})^2 \ \texttt{Tank}) \sqcap$$
$$\forall \texttt{IS-D-COMPONENT}^{-}.(\neg\texttt{Reactor} \sqcup \exists \texttt{IS-D-COMPONENT}^{-}.\texttt{Tank}) \sqcap \ldots$$

The notion of *Owner-Restricted parts* is similar, but it assumes the point of view of the part objects; i.e., we want to model that a given tank is used exclusively by the reactors of the system:

$$\texttt{Tank} \doteq (= \ 1 \ \texttt{IS-D-COMPONENT}^2 \ \texttt{System}) \sqcap$$
$$\forall \texttt{IS-D-COMPONENT}.(\neg\texttt{Reactor} \sqcup \exists \texttt{IS-D-COMPONENT}.\texttt{System}) \sqcap \ldots$$

Sometimes the existence of an object is constrained to the existence of some *essential parts* (see section 3.3). This can be expressed by using the existential quantifier on role parts – an essential part of a human being could be its brain:

$$\texttt{Human} \doteq \exists \texttt{IS-D-COMPONENT}^{-}.\texttt{Brain} \sqcap \ldots$$

On the other side, parts can be in turn *Dependent* on the existence of the whole. Also dependent parts can be represented by using the existential quantifier:

$$\texttt{Ceiling} \doteq \exists \texttt{IS-D-COMPONENT}.\texttt{Room} \sqcap \ldots$$

4.3.1 Complexity issues

The author ends up with some interesting considerations on the complexity of subsumption computation in $\mathcal{P}$. To adequately represent part-whole relationships, the language $\mathcal{P}$ needs a high expressive power on role constructs. This is obtained by extending $\mathcal{ALC}$ with the following role-forming operators: inverse roles (R^{-}), role disjunction ($R \sqcup S$), role composition ($R \circ S$), transitive closure of roles (R^{*}). This is essentially the same expressive power requested by the language studied in [65], called $\mathcal{FSL}$, where subsumption can be computed in double exponential time. Moreover, $\mathcal{P}$ needs also to express qualified number restrictions on complex roles (Exclusive, Multi-Possessed and Owner-restricted parts) and conjunction of roles (to express disjointness between different part-whole relations[14]). Such language is at least as expressive as $\mathcal{FSLR}$, i.e., $\mathcal{FSL}$ plus role conjunctions, shown to be undecidable [65].

4.4 $\mathcal{ALCS}$: Quantifying over Collections and Parts

The $\mathcal{ALCS}$ language ($\mathcal{ALC}$ with $\mathcal{S}$ets) proposed by Franconi in [26] is an extension of a description logic in order to represent *collections*. Its motivation is to give a semantics for plurals and plurals quantifiers in natural language, but it can be naturally extended to handling parts and wholes.

Collections are contingent aggregates of individuals called *members* or *elements* of the collection, selected by means of a primitive *membership* binary relation (denoted by $\ni$). The $\ni$ relationship can be interpreted as the meronymic relation of `Member/Collection`.

[14] We think that disjointness is not generally verified. Suppose for example: *Herbert is part of this orchestra* and *Herbert is the director of this orchestra*, then you have that both Member-Collection and Component-Object relationships hold between *Herbert* and *this orchestra*.

$$
\begin{array}{lll}
\lhd R(a,b) & \text{iff} & \forall x.\ \ni(a,x) \rightarrow R(x,b) \\
\rhd R(a,b) & \text{iff} & \forall x.\ \ni(b,x) \rightarrow R(a,x) \\
\unlhd R(a,b) & \text{iff} & \forall x.\ \ni(a,x) \rightarrow \big(R(x,b) \vee (\exists s.\ \ni(s,x) \wedge R(s,b))\big) \\
\unrhd R(a,b) & \text{iff} & \forall x.\ \ni(b,x) \rightarrow \big(R(a,x) \vee (\exists s.\ \ni(s,x) \wedge R(a,s))\big)
\end{array}
$$

Figure 2: Semantics for the Plural Quantifiers.

For example, the statements "The Beatles are John, Paul, George and Ringo" and "John is the leader of the Beatles" can be expressed as follows:

$\ni(beatles,\ paul),\ \ni(beatles,\ john), \ni(beatles,\ ringo),\ \ni(beatles,\ george),$

$\texttt{LED-BY}(beatles, john).$

where the plural entity *beatles* is an individual interpreted as a collection, composed by the entities *john*, *paul*, *ringo* and *george*.

Predications on collections apply in various ways on the elements composing the actual collection. As an example, take the possible readings involving a sentence like "The Beatles sing 'Yesterday' ". In the *collective reading*, the four singers together sing the song; in the *distributive reading*, each singer sings separately; finally, with the *cumulative reading* we describe a mixed situation where, say, one sings alone and the others sing together, with the proviso that all of them are involved in some action of singing. Thus, a relation like (SING) not only can hold between the objects of predication (*Beatles, Yesterday*), but it can also distribute to (groupings of) elements of such objects, if they are collections. In order to capture the more structured semantics of predications on plural entities, the $\lhd$ and $\unlhd$ (resp. $\rhd$ and $\unrhd$) operators – called *plural quantifiers* – introduce the left (resp. right) distributive and cumulative readings for generic binary relations having a collection as left (resp. right) argument.

In the following, the meaning of the new operators is explained by means of some examples, while in figure 2 the plural quantifiers are formally defined. In the sentence "John is the leader of the Beatles", the role LED-BY has a *collective* reading:

$\texttt{LED-BY}(beatles, john)$

i.e., *john* is the leader of the whole collection *beatles*. The relation BORN-IN, in "The Beatles were born in Liverpool", is *left distributive* over the components of the *beatles*:

$\lhd\texttt{BORN-IN}(beatles,\ liverpool)$

that is, each member was born in Liverpool:

$\texttt{BORN-IN}(john,\ liverpool),\ \texttt{BORN-IN}(paul,\ liverpool),$
$\texttt{BORN-IN}(george,\ liverpool),\ \texttt{BORN-IN}(ringo,\ liverpool).$

Finally, the sentence "The Beatles sing 'Yesterday' " can be expressed according to a *left cumulative* reading for the relation SING:

$\unlhd\texttt{SING}(beatles,\ yesterday)$

In this case, it is possible that any collection containing components of *beatles* sings 'Yesterday', with the proviso that the union of such collections should include at least all the *beatles* members. Then, a possible interpretation for the cumulative reading could be:

$\ni(C_1,\ paul),\ \ni(C_1,\ john),\ \ni(C_1,\ elvis),$
$\ni(C_2,\ paul),\ \ni(C_2,\ george),\ \ni(C_2,\ ringo),$
$\mathtt{SING}(C_1,\ yesterday),\ \mathtt{SING}(C_2,\ yesterday).$

i.e., the relation SING holds between two collective entities C_1, C_2 and *yesterday*, where $C1$ and $C2$ together contain every single member of the Beatles, plus Elvis. Observe that, given the semantics in figure 2, the cumulative reading includes the collective and distributive readings as particular cases.

The language $\mathcal{ALCS}$ extends $\mathcal{ALC}$ by enriching the expressivity for roles – i.e., it adds the $\ni$ role together with the role-forming operators $\lhd$, $\rhd$, $\unlhd$ and $\unrhd$ that introduce the distributive and cumulative readings for generic roles. As an example of the expressive power, let us see how a concept representing any pop group can be defined using the $\mathcal{ALCS}$ language:

$$\begin{aligned}\mathtt{Pop\text{-}Group} \doteq\ & \forall\ni.\,\mathtt{Person} \sqcap \forall\mathtt{LED\text{-}BY}.\,\mathtt{Person} \sqcap \\ & \forall\lhd\mathtt{BORN\text{-}IN}.\,\mathtt{City} \sqcap \forall\unlhd\mathtt{SING}.\,\mathtt{Pop\text{-}Song}\end{aligned}$$

The definition states that a pop group is composed by persons, that the relation LED-BY is inherently collective, that the relation BORN-IN inherently distributes to the single persons composing the group, and that the relation SING has a cumulative reading for the group.

4.4.1 Quantifying over Parts

In the second part of [26], $\mathcal{ALCS}$ is adapted to represent part-whole relations by just turning the membership relation $\ni$ into the partial-ordering "$\succeq$": as usual, $\succeq(x, y)$ means "x has part y". The notion of plural quantifiers, transposed in the mereological setting, formalizes the notion of *distribution* of properties along a partonomy, giving a clear semantics to correctly manage this phenomenon in description logics.

The $\lhd$ and $\rhd$ operators express the left and right distributive readings. Moreover, they can be *qualified* by a qualification predicate C, which specifies that the relation necessarily holds for all the parts of a certain type C[15]. In analogy with the previously defined plural quantifiers, the $\unlhd$ and $\unrhd$ operators introduce the cumulative plural quantifiers: the relation holds for some parts covering all the qualified parts. The semantics of the operators is defined in figure 3.

Let us show the expressive power of these operators with some examples. If a role is left-distributive, then the relation which holds for the whole is also true for the parts, i.e., more formally, the following theorem holds:

$$\exists(\lhd R).\, C \sqsubseteq \forall\succeq.\,(\exists R.\, C) \tag{1}$$

Then, to capture, for example, that the "The location of a car is the same as that of its

[15] As a convention, when we omit the qualification we intend that the qualification predicate C is the generic universal concept – i.e, $\lhd R \equiv \lhd\top.\, R$.

$$\lhd C.\, R(a,b) \;\text{ iff }\; \forall x.\, (\succeq(a,x) \wedge C(x)) \rightarrow R(x,b)$$
$$\rhd C.\, R(a,b) \;\text{ iff }\; \forall x.\, (\succeq(b,x) \wedge C(x)) \rightarrow R(a,x)$$
$$\unlhd C.\, R(a,b) \;\text{ iff }\; \forall x.\, (\succeq(a,x) \wedge C(x)) \rightarrow (\exists s.\, \succeq(s,x) \wedge R(s,b))$$
$$\unrhd C.\, R(a,b) \;\text{ iff }\; \forall x.\, (\succeq(b,x) \wedge C(x)) \rightarrow (\exists s.\, \succeq(s,x) \wedge R(a,s))$$

Figure 3: Semantics for the Plural Quantifiers in the mereological setting.

engine" we can state:

$$\texttt{Car} \doteq \exists\succeq.\, \texttt{Engine} \sqcap \exists(\lhd\texttt{LOC}).\, \texttt{City} \sqcap \ldots$$

i.e., a car has, among other things, an engine and it is located in a city. Now, thanks to (1), the engine is also in the city:

$$\texttt{Car} \sqsubseteq \exists\succeq.\, (\texttt{Engine} \sqcap \exists\texttt{LOC}.\, \texttt{City})$$

The above example does not make use of the qualification. This feature becomes relevant if we want distributivity to apply only to parts of a given kind. For example, for an artifact with connected and unconnected parts, the location distributes only to the connected ones – i.e., "The location of the remote control of a stereo system is (usually!) not the same as the location of the stereo":

$$\texttt{Stereo-System} \doteq \exists(\lhd\texttt{Connected}.\, \texttt{LOC}).\, \texttt{Room} \sqcap \exists\succeq.\, \texttt{Remote-Control} \sqcap \ldots$$
$$\texttt{Remote-Control} \doteq \neg\texttt{Connected} \sqcap \ldots$$

meaning that every connected part of a `Stereo-System` shares the same location, while nothing can be said about the location of the remote control, an unconnected part of the stereo.

The right distributivity covers the case where a relation holding between two objects also holds between the first object and the parts of the second object; more formally:

$$\exists(\rhd R).\, (C \sqcap \exists\succeq.\, D) \sqsubseteq \exists R.\, D \tag{2}$$

In the medical domain, it is useful to deduce diseases of organs from diseases of parts of organs [25]. The example 3 in section 4.5 shows how the right distributivity in $\mathcal{ALCS}$ can capture that "The fracture of the condyle of the femur" is also "The fracture of the femur".

Cumulative readings are useful for expressing the distribution of a property along unidentified groupings of parts of a whole covering all its (possibly qualified) parts. Let us take, for example, a powered-on `Stereo-System`. It is clear that, in order for a `Stereo-System` to be on, all its *main components* should be on for themselves. However, it may be not known how actually the `Stereo-System` is composed, i.e., whether it has just one main component – a *mini*-system – or more than one – a combination of CD-player, amplifier, cassette deck and so on. In fact, the expression "*The stereo system is on*" implies just that all of its main components should be on, without any commitment about which and how many these components should be. The expression

$$\texttt{Powered-Stereo-System} \doteq \texttt{Stereo-System} \sqcap \exists(\trianglelefteq \texttt{Elec-Obj.POWER-STATE}).\texttt{On}$$

defines a stereo system having a (unknown) number of powered-on *main components*, obtained by grouping all the electronic parts – i.e., the qualified parts – of the system. Thus, the property of being powered-on of these wholes – the main components – including all the relevant qualified parts – all the electronic parts of the stereo system – is "upward inherited" to the composite object – the stereo system – in the sense that the stereo system may be said to be *cumulatively* powered-on. Actually, the whole stereo system is not powered-on *per se*, but because all its main components are on. Note that, as a particular case, a stereo system may be on even if it is the only powered-on component – the case of one-component *mini*-system:

$$\texttt{Stereo-System} \sqcap \exists \texttt{POWER-STATE.On} \sqsubseteq \texttt{Powered-Stereo-System}$$

4.4.2 Complexity issues

Since the $\mathcal{ALCS}$ language – as it is presented in [26] – does not fully account for transitivity in its semantics, the proposed subsumption algorithm is not able to completely capture the semantics of collections. As a matter of fact, the soundness and completeness of the subsumption algorithm is proven with respect to $\mathcal{ALCS}^-$, which is a weaker variant of $\mathcal{ALCS}$ – the one where the relations among collections and the plural quantifiers are defined with just necessary conditions. It is interesting to note that full $\mathcal{ALCS}$ is able to express terminological axioms known to have EXPTIME-hard satisfiability [12]. Recently, in [20] De Giacomo and Lenzerini have studied the computational properties of a description logic called $\mathcal{CATS}$ which allows for the representation of sets. $\mathcal{CATS}$ has many similarities with $\mathcal{ALCS}$; however, unlike collections, sets do have the extensionality property and plural quantifiers are not considered.

The decidability and the complexity of $\mathcal{ALCS}$ with the PART-OF "$\preceq$" relation is unknown, mainly because of the presence of plural quantifiers – recalling role-value-maps [66]. However, if just $\mathcal{ALC}$ plus PART-OF relation is taken into consideration, then results in modal logics show that satisfiability is still a PSPACE-complete problem.

4.5 Bernward: A Medical Application

The representation language Bernward (Building Essential concept Representations in Well-Arranged Restricted Domains) [7] has been developed by Bernauer in the context of the European GALEN project [27]. It considers the part-whole relation for the purpose of modeling concepts in the medical domain, and it allows for the specification of anatomical objects together with their parts.

The claim is for a logic-based representation of medical terminologies with formal syntax and semantics, that supports complex concept descriptions and classification of concepts in the tradition of terminological languages of the KL-One family. The model, based on Sowa's conceptual graphs [73], emphasizes part-of relationships, trying to integrate features for dealing with parts within the classifier. Besides classical subsumption, the need to cope with the effect of part-whole relation on subsumption forces the author to define a new kind of subsumption, called *part-sensitive subsumption*, based on two different hierarchical structures: taxonomy and partonomy. In the following section we shall show how such a solution weakens the functionality of the overall system.

The language allows for a limited expressive power: a concept in normal form is composed by a single primitive concept (called by the author the *base concept*), plus a set of role restrictions (called *criteria*). The `PART-OF` role, assumed to be transitive, is the "partitive criterion" of the concept. Notice that the model allows for at most one of such roles among the criteria set associated to a given concept. As a consequence of this requirement, a concept cannot be defined to be part of different conceptual wholes, as for example in the case where the `Pancreas` is part of both the `Digestive-System` and the `EndocrineSystem`. As an example of a concept description we show the concept of *articular fracture of the condyle* (let us call `Art-Cond-Frac` such a concept):

$$\begin{aligned}\texttt{Art-Cond-Frac} \doteq (\texttt{Fracture}\ & (:\texttt{LOC}\ \texttt{Condyle})\\ & (:\texttt{COMPL}\ \texttt{Articular}))\end{aligned}$$

Its base concept is `Fracture`, while the criteria (:`LOC Condyle`) and (:`COMPL Articular`) say that it is located in the condyle and it has an articular complexity.

A simple structural algorithm computes subsumption between concepts. While the algorithm is sound, it is not complete with respect to the "dedicated semantics" claimed by the author for the `PART-OF` role. For example, the transitive property of the part-of relation is not taken into account by the subsumption algorithm. In fact, only apparently the transitivity of the `PART-OF` role is achieved by organizing the concepts in a *partonomy*: with the separation between taxonomy and partonomy we lose those subsumption and meronymic relations which hold due to their cross-dependence. Let us consider the following example, adapted from Bernauer's paper, and depicted in figure 4:

$$\begin{aligned}&\texttt{Femur} \sqsubseteq (\texttt{Bone}\ (:\texttt{PART-OF}\ \texttt{Leg}))\\ &\texttt{Leg-Junction} \doteq (\texttt{Junction}\ (:\texttt{PART-OF}\ \texttt{Leg}))\\ &\texttt{Femur-Condyle} \sqsubseteq (\texttt{Junction}\ (:\texttt{PART-OF}\ \texttt{Femur}))\\ &\texttt{Lateral-Femur-Condyle} \doteq (\texttt{Anatomic-Thing}\ (:\texttt{PART-OF}\ \texttt{Femur-Condyle}))\\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (:\texttt{ORIENT}\ \texttt{Lateral}))\end{aligned}$$

Whereas the partonomy captures the part-whole relation between `Femur-Condyle` and `Leg`, the obvious deduction that every `Femur-Condyle` is also a `Leg-Junction` is lost, since the subsumption algorithm does not consider transitive roles. As a consequence, BERNWARD also misses the meronymic relation between `Lateral-Femur-Condyle` and `Leg-Junction`. These failed inferences – showed as dashed lines in figure 4 – would be recovered if transitive roles were taken into account in the subsumption algorithm.

Another important point stressed by Bernauer is the *distribution* of roles along the partonomy. Stating, for example, that the location of a part is the same as the location of the whole, amounts to expressing the distribution of the `LOC` role over the `PART-OF` role. This would validate the common-sense inference that a *fracture of the condyle of the femur* is a *fracture of the femur*, too. Bernauer points out however that such principle of distribution is neither true for all roles nor globally true for the same role (for example, a *scoliosis of the thoracic spine* is not a *scoliosis of the spine*, and therefore the `LOC` role does not distribute). To address this problem, the author proposes to separate classical subsumption from *part-sensitive* subsumption, where every role is supposed to distribute

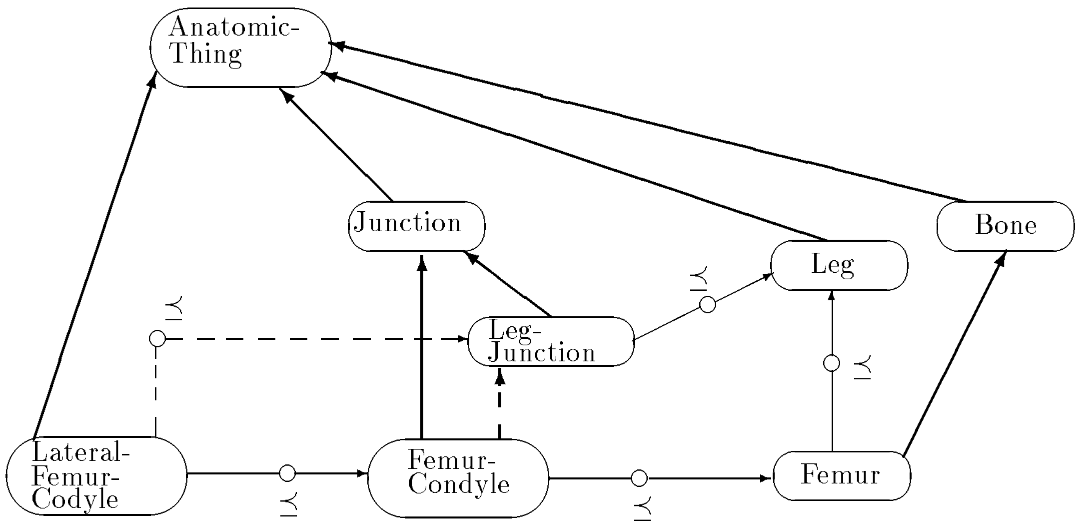

Figure 4: Meronymic and subsumption relations. Dashed lines represent missed inferences.

over the PART-OF role. In our opinion, while we think that a right modeling of the part-of relation has to take care of the distributivity problem, this solution appears too weak, since, from one side, the classical subsumption misses some desired inferences, while, on the other hand, the new criterion (part-sensitive subsumption) does not discriminate those particular situations where the principle of distribution does not hold. Instead of different subsumption relations, we suggest to adopt the operators for roles introduced in section 4.4, in order to capture the complex behavior of a relation when applied to composite objects. Notice that these operators can be applied *locally* to a role, so that the same role can have a distributive reading in a concept description and a classical one in another. For instance, from the following statements[16]:

$$\begin{aligned}
&\texttt{Femur-Fracture} \doteq \texttt{Fracture} \sqcap \exists \texttt{LOC}.\texttt{Femur} \\
&\texttt{Femur-Condyle-Fracture} \doteq \texttt{Fracture} \sqcap \exists (\rhd^{-}\texttt{LOC}).(\texttt{Condyle} \sqcap \exists \preceq .\texttt{Femur}) \\
&\texttt{Spine-Scoliosis} \doteq \texttt{Scoliosis} \sqcap \exists \texttt{LOC}.\texttt{Spine} \\
&\texttt{Thoracic-Scoliosis} \doteq \texttt{Scoliosis} \sqcap \exists \texttt{LOC}.(\texttt{Thoracic-Spine} \sqcap \exists \preceq .\texttt{Spine})
\end{aligned}$$

then, due to (2) in 4.4.1, we can derive that:

$$\texttt{Femur-Condyle-Fracture} \sqsubseteq \texttt{Femur-Fracture} \tag{3}$$

but not:

$$\texttt{Thoracic-Scoliosis} \sqsubseteq \texttt{Spine-Scoliosis}$$

[16]We indicate with $\lhd^{-}$ and $\rhd^{-}$ the distributivity operators with respect to the role $\preceq$ (*is part of*).

4.6 Part-Whole Relation as Compositional Inclusion

The problem of representing and managing the structure of composite objects like *documents* has been studied in [58, 50] by Lambrix and Padgham. In this approach, key issues in both expressivity and inferential power have been identified. The system supports the definition of different kinds of documents, their classification, the recognition of specific documents and the reconstruction of their parts. Thus, besides deducing whether an object is part of another object, the system should support inferences about the existence of composite objects based on the existence of their parts. Moreover, the system is intended to be able to reason about relationships between classes, i.e., to determine whether a particular class is a possible *building block* for another class. In this context, the possibility of specifying both vertical and horizontal constraints between parts and wholes is considered important.

The authors propose a simple extension of description logics, where part-whole relations are distinguished from standard roles just by marking them with the label **part** – thus, they are not transitive. Value and cardinality restrictions can be stated on these basic part roles, and horizontal constraints may appear between parts. For example, the expression:

(**part** `title-p string`) $\sqcap$
(**atleastp** `1 title-p`) $\sqcap$ (**atmostp** `1 title-p`) $\sqcap$
(**part** `abstract-p abstract`) $\sqcap$
(**atleastp** `1 abstract-p`) $\sqcap$ (**atmostp** `1 abstract-p`) $\sqcap$
(**pp-constraint** `larger-font title-p abstract-p`)

denotes any document having exactly one part called `title-p` – which is a string – having a font larger than the one used in the other part of the document called `abstract-p`. Moreover, two operators are introduced in order to propagate values of properties from wholes to parts and vice versa. For example, the expression:

(**same-filler** `title page-p.running-title`)

indicates that the title of the composite object – e.g., a document – is the same as the running title in the headlines of each page composing the document itself. The expression:

(**aggregate** `document-p author responsible`)

denotes a composite object – e.g., a folder – having documents as parts, such that each document's author is also responsible of the folder itself[17].

An object *builds* another one if the former is either a part or a grouping of parts – called *module* – of the latter. For example, the *body* of a document which includes all the parts of the document with the exception of, say, the title page, is a *module* – and then a *building block*, too – of the document itself. *Compositional inclusion* is defined as the inverse of the transitive closure of *builds*; e.g., a document compositionally includes both its parts – like the title and the sections – and the body of the document itself. Thus, compositional inclusion plays the role of the *general* part-whole relation, grounded on the basic part-of relation and on the notion of module. It turns out from the definition

[17] Both **same-filler** and **aggregate** are restricted forms of role-value-map, where full composition of roles is not allowed [66].

of module that two individuals have the same parts if and only if they have the same modules. It is worth noting that a module is *not* part of the composite it is a module of; it is, of course, compositionally included in the composite object. Nonetheless, the extensionality property does not hold.

A strong emphasis is given to reasoning within this representational framework. It can be said that reasoning involving parts is *external* to the standard description logic reasoning services. So, terminological subsumption and instance recognition are not affected by the particular meaning that the part-whole roles should convey. On the contrary, Padgham and Lambrix propose ad-hoc reasoning services, together with an analysis of their algorithms.

The simplest service is checking whether an object is a general part of another object, and it relies on the computation of transitivity of compositional inclusion. Then, they propose a service deducing whether a concept is a general part – with the respect to the meronymic hierarchy based on compositional inclusion – of another concept. The attempt to introduce concepts as general parts of classes denoting composite objects is a weak aspect of the theory. As an example, the concept of `string` is considered a general part of a document since the `title-p` is a part name of a document – i.e., (**part** title-p string) – whereas we believe that there is no ontological relation between strings "as such" and documents. A better logical foundation of the notion of general part (or of *building block*) at the intensional level is needed.

Important reasoning tasks are *assembly* – also called *compositional extension* – and *completion*. Assembly is the task of building a new composite individual out of its composing parts. It is a bottom-up procedure: compositional inferencing assembles new composite objects on the basis of the existence of available parts. The algorithm collects the parts necessary to build a new composite object by choosing among potential parts and modules, according to the definition of the composite type. Of course, there may be more than one possible choice – i.e., extension; thus, credulous and skeptical compositional extensions are defined. For example, if we have a title, an abstract, and three sections, it is possible to introduce a new composite individual, i.e., a document having exactly those parts. However, other composite individuals may be possibly introduced, by considering different combinations of parts: e.g., a document with the title, the abstract and a section, and a document body with the remaining sections. The procedure makes use of the meronymic hierarchy based on compositional inclusion in order to look for specific types for the objects necessary to build new composite individuals. For the reasons explained above, the weakness of the intensional partonomy may induce an over-generation of hypotheses.

Completion infers missing parts from composite objects; it is a top-down procedure. A candidate completion should be chosen among the smallest possible extensions, according to some reasonable preferential ordering. This assures that *unnecessary* parts are not introduced in the model, if they are already available. We believe that experimentation in particular domains and in particular applications is needed in order to understand whether the introduced preferential ordering – motivated just by economy reasons – is effective.

4.7 Further Approaches

In this section we shall briefly describe some further approaches to the representation of part-whole relations which, being developed with specific needs in mind, lack in our opinion to fully capture the semantics of these relations and a proper inference mechanisms.

4.7.1 ElkLogic

ElkLogic [77, 61] – by Uschold – is based on typed lambda calculus with an object-oriented style of representation. The target domain is the ecological modeling, with the main goal of assisting naive users to clarifying the nature of part-whole relationships required in this environment. Special attention is given to the representation of *sets*: a set type constructor is used to build nested set types, e.g., `set(sheep)`, `set(set(sheep))` and so on. The author distinguishes three kinds of part-whole relationships, namely member-collection, sub-collection-collection and part-composite. The exact semantics of such relations is not discussed, but the emphasis of the paper is rather on the opportunity to present to the user just one very general, abstract relation called "component-whole", relying on the system to isolate its sub-relations on the basis of the types of their arguments: for instance, if the user states that *hoof1* is a component of *sheep1*, and *sheep1* is in turn a component of *flock1*, the system is able to understand that the former is a part-composite relationship since both arguments are not collections, while the latter is a member-collection relationship since *sheep1* is an individual of type `sheep` while *flock1* is a collection of type `set(sheep)`.

This approach however is complicated by the fact that, in order to better constrain the "possible components" of a whole, the author introduces a further primitive relation called *possible part*, intended to be specified directly by the user. In this way the original purpose of avoiding for the users the burden of specifying specific kinds of part-whole relations seems to be seriously jeopardized. Moreover, the lack of a formal semantics for the primitives introduced makes unclear the kinds of inference services offered and their possible formal properties like soundness, completeness or computational complexity.

4.7.2 Part-of as co-subsumption

Napoli [56] considers both the inheritance mechanism and the part-of relation as two aspects of a sort of subsumption relation within object-based representations. According to this framework, taxonomies and partonomies are managed as orthogonal hierarchies, both based on a so-called *classification-based reasoning*. Classification-based reasoning could be useful in order to organize objects in hierarchies and to solve synthesis problems. In particular, this idea has been applied in a system that assists a chemist in planning the synthesis of new organic molecules.

Since both subsumption and part-of relations are partial orders, the author proposes to apply classification not only to subsumption but also to the part-whole relation, called here *co-subsumption*. Moreover, it is claimed that the structural interrelations between subsuming concepts are in analogy with the structural interrelations between a whole and its parts. Thus, the same structural algorithm is applied for computing both subsumption and whole-part relationships.

This approach has some major weaknesses. First, we have already seen in section 4.5 that partonomies and taxonomies are not independent. Second, classification with respect to the part-of relation – i.e., the computation of the *immediate* predecessors and successors of an object – makes sense only if there is a clear and explicit notion of *direct* part in the application domain (see the discussion on this point in section 4.3).

4.7.3 Parts as meta-attributes

Boldrin [8] suggests to explicitly model the various part-whole relations at a meta-level, where meaning postulates, in the form of procedural inference rules, are introduced. Specific schemata, modeling particular domains, should be considered as instances of the proposed meta-model; thus, according to the author, re-usability for knowledge sharing should be easily achieved. In fact, the meta-model is claimed to be topic independent and somehow ontologically well founded.

In this architecture, a strong emphasis is given to the inference mechanism associated to the meta-model, which should be inherited by the particular domain models. The proposed set of meaning postulates covers:

- The possibility to constrain the *assertional* nature of an attribute depending on the concept the attribute is attached to. For instance, it is possible to declare constraints like $(_{\texttt{Encyclopedia}}|\texttt{HAS-VOLUME}) \sqsubseteq \texttt{HAS-COMPONENT}$ and $(_{\texttt{Library}}|\texttt{HAS-VOLUME}) \sqsubseteq \texttt{HAS-MEMBER}$ for the `HAS-VOLUME` role[18]. This is expressed by means of meta-level statements which involve both the attribute and its domain.

- The inter-relation between specialized part-whole relations like `HAS-COMPONENT` and the concept that denotes their range (`COMPONENT`). In fact, accordingly to [31], attributes are introduced as *relational interpretations* of concepts.

- A detailed primitive taxonomy of part-whole relations that further extends and specializes the one introduced by WCH – trying to account for the distinction between dependent and independent part names discussed in section 3.3 – together with an analysis of composition rules for such different part-whole relations.

- The inheritance of properties along the part-of hierarchy – i.e., along compositions of part-whole relations; different rules are identified for the terminological and instance levels.

While the general idea of defining the properties of some ontologically relevant attributes – like the part-of attribute – at an abstract meta-level is an interesting one, the approach lacks of formal grounds, since there is no formal semantics associated to the newly introduced meronymic relations. In this way, it is not possible to check the validity (and the completeness) of the proposed inference rules.

[18] Read $(_C|R)$ as the role R whose domain is restricted to the concept C.

4.7.4 Modeling composition

Artale, Cesarini, Grazzini, Pippolini and Soda [3] propose a new concept-forming operator to model the part-of relation in the setting of description logics. The compos operator describes the various component of a whole by *listing* them, associating at the same time cardinality constraints to each component. The following declaration

$$\mathtt{C} \doteq \mathsf{compos}((n_1, \mathtt{C}_1) + (n_2, \mathtt{C}_2) + \ldots + (n_k, \mathtt{C}_k))$$

defines a concept C as made up of at least n_1 components of type $\mathtt{C}_1$ *plus* at least n_2 components of type $\mathtt{C}_2$ *plus* at least n_k components of type $\mathtt{C}_k$. The key of the intended semantics of the compos operator is in the meaning given to the term "plus" in the above statement.

For example, a Working-Group made up of at least one manager plus at least one secretary plus at least two other employees could be described as:

$$\mathtt{Working\text{-}Group} \doteq \mathsf{compos}((1, \mathtt{Manager}) + (1, \mathtt{Secretary}) + (2, \mathtt{Employee}))$$

To explain the interpretation given to the concept above, let John be a manager, Mary be a secretary, Bob and Fred be employees. Let us assume that managers and secretaries are employees, too. The individual collection composed by John, Mary and Bob is not of type Working-Group, since in this case either John would be both a manager and an employee or Mary would be both a secretary and an employee while the purpose of the compos operator is to distinguish at least four different individuals: one belonging to the class Manager, one to the class Secretary, and the other two to the class Employee. A correct instance of Working-Group would be an individual composed by John, Mary, Bob and Fred.

The semantics sketched above implies that we cannot simulate the compos operator by a simple use of the qualified number restriction operator (see footnote 12), as we show in the following – where COMPOS is a primitive role:

$$\begin{aligned}\mathtt{Working\text{-}Group} \doteq\ & (\geq\ 1\ \mathtt{COMPOS}\ \mathtt{Manager}) \sqcap (\geq\ 1\ \mathtt{COMPOS}\ \mathtt{Secretary}) \sqcap \\ & (\geq\ 2\ \mathtt{COMPOS}\ \mathtt{Employee})\end{aligned}$$

Instead, a right translation needs to consider also number restrictions on any combination of component concepts:

$$\begin{aligned}\mathtt{Working\text{-}Group} \doteq\ & (\geq\ 1\ \mathtt{COMPOS}\ \mathtt{Manager}) \sqcap (\geq\ 1\ \mathtt{COMPOS}\ \mathtt{Secretary}) \sqcap \\ & (\geq\ 2\ \mathtt{COMPOS}\ \mathtt{Employee}) \sqcap \\ & (\geq\ 1+1\ \mathtt{COMPOS}\ \mathtt{Manager} \sqcup \mathtt{Secretary}) \sqcap \\ & (\geq\ 1+2\ \mathtt{COMPOS}\ \mathtt{Manager} \sqcup \mathtt{Employee}) \sqcap \\ & (\geq\ 1+2\ \mathtt{COMPOS}\ \mathtt{Secretary} \sqcup \mathtt{Employee}) \sqcap \\ & (\geq\ 1+1+2\ \mathtt{COMPOS}\ \mathtt{Manager} \sqcup \mathtt{Secretary} \sqcup \mathtt{Employee})\end{aligned}$$

In other words, the proposed compos operator can be remapped onto an ordinary description logic by means of qualified number restrictions, but the resulting description would be exponential in the number of component concepts.

The authors give an algorithm to compute subsumption between two composite concepts that is showed to be sound and complete and that requires a time exponential on the number of component concepts. Since nothing is said about transitivity and other characteristic properties of such an operator, subsumption misses some intuitive inferences.

4.7.5 Extending Classic with parts

Speel and Patel-Schneider [74, 75] study the introduction of what they call *physical whole-part relations* – i.e., the component/integral-object relations, see section 2.1 – in the description logic based system Classic [10]. This study has been carried on within the Plinius project, which aims at developing a knowledge-based system for semi-automatic extraction of knowledge from scientific publications in the field of ceramic materials. In the knowledge specifications of the Plinius project, the *material, phase, chemical substance, group* and *chemical elements* concepts are related each other by a physical whole-part relation. In particular, in Plinius there is the need of representing *substructures* between *collections*; this specific relation has been identified with a generic *decomposition-of* relation, whose meaning has been tentatively associated to the physical whole-part relation. Another application where the authors claim the usefulness of a whole-part relation is within the configuration domain; in fact, they present examples for stereo systems configurations.

Distinct hierarchies of roles separate the physical composition relations from standard roles. Moreover, using the inverse role construct it is possible to describe both the *has-part* and the *is-part-of* relation between concepts. For example, the concept `Two-Housed-Stereo-System` is defined as having four different kinds of physical parts: at most two `SH`elves-`COMP`onents, exactly one `AMPL`ifier, at most two `CASS`ette decks and at most one `CD` player:

$$
\begin{aligned}
&\texttt{Two-Housed-Stereo-System} \doteq \\
&\quad \forall \texttt{SH-COMP.Shelf-Component} \sqcap \leq 2\, \texttt{SH-COMP} \sqcap \\
&\quad \forall \texttt{AMPL.Amplifier} \sqcap \geq 1\, \texttt{AMP} \sqcap \leq 1\, \texttt{AMP} \sqcap \\
&\quad \forall \texttt{CASS.Cassette-Deck} \sqcap \leq 2\, \texttt{CASS} \sqcap \\
&\quad \forall \texttt{CD.Cd-Player} \sqcap \leq 1\, \texttt{CD}
\end{aligned}
$$

A primitive notion of *connection* – a type of horizontal relation, section 3.3 – is then added, with a meaning similar to the one of the **pp-constraint** construction proposed by Lambrix and Padgham – see section 4.6. As such, the connection relation looses all its semantic peculiarities, and it is in fact nothing more that a statement of some primitive relation among parts. For example, specifying that the amplifier and cd player, in the `Two-Housed-Stereo-System` concept, are connected via the `PLAY` connection means that all parts corresponding to `AMPL` are related to all possible `CD` parts via a connection of type `PLAY`. The authors explicitly say that they do not have an algorithm to compute subsumption and recognition for the language extended with *connection*.

Even if transitivity should be the basic property of any part-whole relations, it is considered only for querying ABox individuals – and not by subsumption inferences for the whole-part roles in the TBox. Thus, this is a weak approach: it is not possible to get positive subsumptions like $\exists \texttt{W}.\exists \texttt{W}.\texttt{C} \sqsubseteq \exists \texttt{W}.\texttt{C}$, i.e., wholes having sub-parts which have sub-parts themselves of a certain kind are not wholes having *directly* sub-parts of that specific kind.

5 The Object-Oriented Perspective

In this final chapter we complement the knowledge representation perspective discussed above with an overview of the various approaches to the modeling of parts and wholes within object-oriented data models. Such kinds of models have been motivated by the advent of new generation applications, like CAD/CAM, multi-media and knowledge based systems, whose domains exhibit a complex internal structure (for example a VLSI chip, an airplane wing, a structured territorial entity); for this reason, the issues addressed in this paper are particularly relevant to the object-oriented community.

Traditional data models, such as the Codd's Relational Model of Data [17], decompose complex entities in terms of flat normalized relations, and are therefore appropriate for the representation of poorly structured domains, typical of business applications. As the structural complexity increases, the semantics of decomposed data structure can be obscure, since the information about the global aggregation resides in the applications interacting with the database – in the form of relational joins between the relations.

In order to fill the semantic gap between real world complex entities and the existing relational data modeling tools, *Semantic Data Models* (SDM) [38] and *Object-Oriented Data Models* (O-ODM) [45] are emerging, with the aim of providing increasingly expressive data abstraction constructs. SDM introduce a higher level of abstraction by means of a set of constructs for representing the structural aspects of the domain, while object-oriented programming add a *behavioral layer* on the top of such structural abstractions, realized through the method and message-passing mechanism in a Smalltalk-like style [30]. In this way, O-ODM can be seen as a dynamic extension of SDM.

O-ODM introduce the notion of an "object" – intended to denote classes or individuals – as a complex data structure. Within objects, attributes may have other objects as values: thus, the information about the structure of represented entities may reside directly in the logical schema of data. This mechanism of reference is known in the literature as *link*, or more eloquently as *implicit join* [44]. A directed acyclic graph of objects connected by links is termed *complex object*. O-ODM embody complex-values constructs (such as tuples and sets), features borrowed from object-oriented programming languages *plus* original features like the concept of object identity [5]. They realize the "one entity/one object" principle [22], which can be considered, together with inheritance, the essence of the object-oriented paradigm. The capability to represent interrelated objects in a straightforward way by means of object-valued attributes yields a conceptual economy, which represents the evolutive advantage over relational database systems. This is the reason why the full exploitation and evolution of the O-O paradigm as a modeling methodology depends crucially on the notion of whole and of part. Nonetheless, O-ODM still lack an ontologically well founded notion of whole – most notably the capability to deal with wholes at the conceptual level.

The most representative and organic attempt of modeling `PART-OF` relationships in O-ODM is given by Kim's approach of Composite Objects, which assigns a clean-cut `PART-OF` semantics to references joining parts and wholes. Composite Objects can therefore be seen as an answer to the problem of modeling *vertical* part-whole relationships. Current applications require however to deal with complex objects in a more comprehensive manner, and therefore appropriate tools to deal with *horizontal* constraints among the parts of a whole seem to be needed. Only in this way we can capture the "[..] special semantics (which) pertains to those objects where operation and properties apply to a collection of objects as a whole" [51]. A modeling construct which embodies both horizontal and vertical relationships will be referred to in the following as an *aggregate*. Unfortunately, the term "aggregation" has also been used in conceptual and data models for the cartesian type constructor used to make structured types (e.g., *address*) from simple attributes (e.g., *street, city* and *zip*). In the context of this paper (and in most of the object-oriented literature) the term *tuple type constructor* is preferred for the purpose of grouping attributes, while we reserve the term *aggregate* for the purpose of making structured objects from component ones.

This comprehensive view of aggregates is however not fully supported, at the conceptual level, by O-ODM: as observed by Roger King [48] "[..] in an object-oriented model, one must generally simulate an aggregation with a set of relationships". Conceptual modeling of aggregates has been in fact realized by either adopting the *implicit modeling* approach mentioned in section 3.1 – which hides aggregates in favor of mutual references between objects – or adopting an *explicit modeling* approach.

After a detailed discussion of the Composite Objects proposal in the next section, we present in section 5.2 a proposal for modeling sets within an object-oriented context. In section 5.3 we discuss how the various O-O formalisms can be related to the implicit/explicit choice in the modeling. We discuss then in section 5.4 the relevance of behavioral aspects in the modeling of wholes and aggregates. Finally, in section 5.5 open and advanced issues – together with flaws and criticisms to the current paradigm – are tentatively arranged as a path towards a new perspective in object-oriented modeling of parts and wholes.

5.1 Composite Objects

As stated by Kim and colleagues in [42], a traditional database model "[...] requires the ability to define and manipulate a set of objects as a single logical entity for purposes of semantic integrity, and efficient storage and retrieval". They introduce the notion of *composite object*, essentially to isolate an object together with its parts and to express constraints of *exclusiveness* and *dependence* between the whole and its parts. A composite object has to be intended as that part of a conventional complex object which is described in terms of `PART-OF` relationships. In other words, a composite object induces a meronymic hierarchy of objects, where the root object is called *compound* object, and a non-root object is termed *component* object [42, 43, 45].

Composite objects augment the semantic integrity of an object-oriented data model through the notion of dependent and exclusive objects [46]. A dependent object is one whose existence depends on the existence of another object of which it is a component. A dependent object cannot be created if its owner does not already exist. Further, when an object is deleted, all its dependent objects must also be deleted. An exclusive object is one which is part of exactly one object. Consider for example the part-whole relation between a car and its body. The fact that a body cannot exist without the respective car, to which it belongs to, is modeled by describing the component object body as dependent on the compound object car. Every time we delete a car from the database, we must also delete the body, a dependent part of the car. Furthermore, we cannot insert a body in the database unless the car of which the body is part (its owner) does not already exists.

Kim [43, 45] distinguishes between *weak* and *composite references*. Weak references are used in standard O-O systems, and carry no special semantics. Composite references are weak references with a super-imposed part-of meaning. In order to distinguish weak from composite references, a `composite` keyword is used; this approach makes use of marked part-names as noted in section 3.2. As discussed in section 3.3, an exclusive reference denotes a part which is not shared by any other whole, and the existence of a composite reference may depend on the existence of the part. Kim gives a classification of composite objects based on four kinds of composite references:

1. Exclusive dependent composite reference;
2. Exclusive independent composite reference;
3. Shared dependent composite reference;
4. Shared independent composite reference.

In Kim's terminology, *physical part hierarchies* are composite objects in which all the references are exclusive (1 and 2 from the list above), while *logical part hierarchies* are composite objects in which some references may be shared – the structure of the composite object is no more a tree, but a directed acyclic graph. A physical part hierarchy is used to model physical objects, in which a component object is supposed to be part of at most one compound object. The exclusiveness constraint is relaxed in logical part hierarchies, which are used to model non physical objects, as, for example, electronic documents which may share chapters.

In this setting, composite objects are used to improve the performance of the database system ORION described in [46]. Two physical level operations, like *clustering* and *locking*, profit by the introduction of composite objects. As a matter of fact, it is advantageous to store all constituents close one to each other, since an access to the root object is likely to be followed by an access to its dependent parts. Clustering composite objects makes more efficient to retrieve a large collection of part-related objects. Further, in a concurrent access to the database, a composite object may be locked as a unit rather than requiring a lock for each component. This policy minimizes the number of locks that must be set in retrieving a composite object.

5.2 Collections in Object-Oriented Formalisms

Motschnig-Pitrik and Storey propose in [55] a series of requirements in order to support the modeling of collections within an object-oriented context. Although the title of the paper refers to the notion of set membership, the authors focus their analysis on what they call *groupings*, which differ from sets in the fact that they are characterized by properties other than set membership, and the extensionality axiom does not hold anymore.

Groupings amount therefore to what we have called "collections" in section 4.4. The proposal is grounded on research in cognitive psychology and linguistics [81], where the member-collection relation is classified among the various meronymic relations despite its intransitivity [39].

The authors start from the notion of *association*, introduced by Brodie [11] to model a *set class* as association of a *member class* – e.g., a `Committee` is an association of instances of `Person`. The authors argue that, in order to capture the inherent semantics of collections, we need to model the notion of membership along with a certain number of constraints which characterize the mutual dependencies between a collection and its members. This is realized through the relation `MEMBER-OF`, plus a bunch of specialized subrelations in order to distinguish the different roles played by different members, and suitable cardinality constraints in order to capture the existential dependence constraints discussed in section 3.3.

The domain of the `MEMBER-OF` relation is called the *member class* while the range is the *grouping class*. A grouping class is in fact a collection, identified by having at least one *set-determining* attribute, i.e., a relation that ranges over a member class. The possibility to attach various set-determining attributes to a grouping class allows us to model a collection by distinguishing the different roles played by its members. Furthermore, it makes possible to represent *heterogeneous* collections, i.e., collections made of different member classes – e.g., a `Project-Team` has both `Engineer` and `Programmer` as member classes. The mark "`-MBS`" is used to distinguish set-determining attributes from other attributes[19]. For example, to describe the grouping class `Tennis-Club` with president, clients and instructors as members the authors propose to use the set-determining attributes `PRESIDENT-MBS`, `CLIENT-MBS` and `INSTRUCTOR-MBS`. A set-determining attribute is intended as a subrelation of the inverse of the member-of relationship (for example, $\texttt{PRESIDENT-MBS} \sqsubseteq \texttt{MEMBER-OF}^-$). Thus, every value of a set-determining attribute of a given grouping class is also a member of such class.

A deep investigation is devoted to what we have called existential dependence constraints, realized by means of cardinality constraints over set-determining attributes[20]. We have a *member covering* constraint in the case where a grouping class is a complete covering of the member classes. For example, imposing that the grouping class `Employees` is a covering of the member class `Employee` implies that every single employee is a member of an individual grouping of type `Employees`. This is obtained by stating that the relation `MEMBER-OF` attached to the class `Employee` has minimum cardinality of "1" (i.e., $\texttt{Employee} \sqsubseteq (\geq\ 1\ \texttt{MEMBER-OF Employees})$). On the other hand, imposing a minimum cardinality of "0" does not force the covering constraint. As an example of this latter case, we could not require that every `Tennis-Player` is a member of a `Tennis-Club` grouping class. Cardinality constraints can be also used to express *mutual exclusion* and *partitioning*. In the former case, to specify that each person may be a member of at most one political party a maximum cardinality of "1" is used:

[19] Generic attributes can be used to describe properties of a grouping class as a whole.

[20] We adopt here a description logic language with qualified number restrictions (see footnote 12) to capture the examples proposed by the authors in a graphical language, "emphasizing" in this way the equivalences between different modeling paradigms.

`Person` $\sqsubseteq$ ($\leq$ 1 `MEMBER-OF Political-Party`)

In the latter case, an example of partitioning is the situation where every employee must be a member of exactly one departement:

`Employee` $\sqsubseteq$ ($=$ 1 `MEMBER-OF Department`)

Cardinality restrictions can also be expressed over subroles of `MEMBER-OF` and its inverses. For example, in order to specify that the set-determining attribute `PRESIDENT-MBS` is a 1-1 mapping from the group class `Tennis-Club` to the member class `Tennis-Player` amounts to imposing a the number restriction of exactly "1" to the `PRESIDENT-MBS` attribute and the cardinality $(0, 1)$ to its inverse:

`Tennis-Club` $\sqsubseteq$ ($=$ 1 `PRESIDENT-MBS Tennis-Player`)
`Tennis-Player` $\sqsubseteq$ ($\leq$ 1 `PRESIDENT-MBS`$^{-}$ `Tennis-Club`)

The authors end with an evaluation of the diverse approaches for modeling collections within the object-oriented paradigmd. They offer an extensive analysis of the requirements needed for an adeguate modeling of set entities, used to justify their own highly expressive set of modeling constructs. No attention is payed however to the computational problems bound to such high expressivity, and in particular to the need to guarantee the consistency of the conceptual model.

5.3 Modeling Wholes in Object-Oriented Data Models

Let us briefly review now different formalisms proposed in the literature – still lacking a specific semantics for the part-of relation – which can be related to the modeling approaches discussed in section 3.1. As noticed in [38]: "[..] various models place different emphasis on the various constructs for interrelating object classes. One approach stresses the use of attributes to interrelate objects; the other places an emphasis on explicit type constructors".

Representative examples of *attribute oriented* formalisms are the Functional Data Model (FDM) [70] and the Semantic Database Model [34]. The attributes provided by this kind of models can be either simple-valued or object-valued.

The Entity Relationship (ER) model proposed by Chen in 1976 [16] makes a strong commitment towards the explicit representation of aggregation through the *Relationship* type construct. The ER model makes a clear distinction between Entities and Relationships, and in particular it forbids the use of *object-valued* attributes for structuring Entities. In the *family* example (see section 3.1), since `WIFE` and `HUSBAND` are object-based attributes the ER model forces us to explicitly introduce the relationship `Family`. Other data models which provide an explicit support for modeling aggregation are IRIS [21], IFO [1] and SAM* [76] and Nijssen's Information Analysis Methodology (NIAM) [79, 57]. Type constructors allowing an explicit representation of relationships are also provided in data models aimed at theoretical investigation, like the Logical Data Model (LDM) [49].

The object-oriented model allows for object-valued attributes, and therefore does not commit itself towards any of the two approaches. Rumbaugh [62] argued for an extended object-oriented model called *object-relationship* model, where the explicit modeling of relationships is aimed at preserving conceptual information concerning aggregation that may get lost in the implementation mechanism. In that work it is observed that explicit relationships "[..] abstract interactions among objects in a natural way [..] (and) [..] affect the partitioning of a system into its parts". In [63] it is observed that "[..] during conceptual modeling, you should not bury pointers or other object references inside objects as attributes". Rumbaugh contrasts some authors' opinion that the explicit modeling of aggregation is a violation of the encapsulation paradigm. He affirms that some information inherently transcends a single class, and the failure to treat this fact in a proper manner leads to a modeling which contains "[..] hidden assumptions and dependencies". Albano [2] follows Rumbaugh's proposal, observing that an object-oriented data model should support abstraction mechanisms to model directly both structural and behavioral aspects of complex database applications. Albano observes also that modeling aggregations as attributes has several limitations which make the description of them unnatural. Jagadish and Qian [40] address the problem of where to record inter-object constraints: although it is recognized an unitary aspect of the semantics at the specification level, their proposal is about splitting aggregates by compiling them in the participating object classes.

5.4 Behavioral Issues

O-O Data Models are aimed at the joint modeling of structural and behavioral aspects of real world entities. A criticism to complex and composite objects is that they realize structural but not behavioral wholeness. As we praised the object model for its economy in representing complex entities by a single data structure, the same is desirable for their behavior. As an example of the need for behavioral modeling, consider a CAD operation requiring to change the position of a particular mechanical assembly, which needs to be propagated to all its components. If the behavior of the assembly is modeled as a whole, a single `move` operation has to be applied just once; otherwise, it has to be decomposed by the recursive application of single operations to the components of the main entity.

The distinction between structural and behavioral modeling capabilities is clearly present in Dittrich's classification of object-oriented systems [22]. A *structural* data model is defined as one which includes facilities to represent complex structured entities, while an *operational* data model provides the support for encapsulating the behavior of simple (i.e., non-complex) objects. A higher level of abstraction, named *behavioral object orientation*, is obtained by combining the two features.

As noticed in [59], behavioral aspects may play an important role in the identification of a whole: for instance, process-based formalisms like Hoare's CSP [36] and Milner's CCS [54] allow for a straightforward definition of aggregation, by implicitly committing to an ontology where an individual is represented as a process, i.e., by a sequence of events. Given two processes representing simple entities, they aggregate to a third, more complex process, if and only if they can be synchronized on some common event or sub-process. For example, a bolt is made of a screw and a nut. Each of these entities has its own vocabulary of events (like move up, down, left or right) and processes (like rotate left, rotate right, etc.). The process representing the behavior of the bolt simply amounts (in both CCS and CSP event algebra terms) to the process obtained from the synchronization of the processes representing the screw and the nut, i.e., its components. The striking simplicity of dynamic aggregation emphasizes the importance of dynamic aspects to recognize and understand conceptual aggregation: any time we observe two or more synchronized entities, then we should consider the introduction of a whole having them as components.

5.5 New Perspectives in O-ODM

Vertical and horizontal relationships, as we have called them, correspond to separate and independent pieces of domain knowledge in the current practice of object-oriented conceptual modeling. For instance, within classical object-oriented analysis methodologies [82, 63] part-of aspects are modeled independently from the dynamic and relational knowledge among the entities of the domain. As observed at the beginning of this chapter, however, vertical relationships are not sufficient for the needs of new applications, which require support for a more adequate notion of *whole*. Therefore, a correct explicit modeling of horizontal inter-object relationships is crucial for achieving an effective modeling of complex entities.

In conclusion, we would like to suggest that an ontologically well founded approach may offer a new perspective to object-oriented data modeling. In fact, in the spirit of [47], it can be observed that, due to its programming origin, the current object-oriented paradigm is driven by implementative considerations, rather than conceptual aspects. Object references appear to be one of these aspects. Therefore, shifting the paradigm to a conceptual level means both assigning a definite semantics to references joining parts and wholes – as in Kim's composite objects – and committing the paradigm towards the explicit modeling of aggregation relationships. These two tasks require to be carefully balanced, since explicitly modeling horizontal relationships means introducing new objects, which, on their turn, have objects as parts.

By suggesting ontologically well founded directions to the modeler, and by providing the appropriate conceptual tools, there is the hope that object-oriented conceptual modeling can easily evolve towards the production of good quality abstractions.

6 Conclusions

Part-whole relations have been extensively used in order to convey structural information. We have seen how their semantic peculiarities pose a number of modeling and reasoning problems, which require careful choices. Most of these problems are related to logical properties and inference schemes which regard the notion of part-whole, like the transitivity property, the interrelations among parts and whole, the possibility to manage specialized part-whole relations organizing them in an appropriate hierarchy. In this paper we have surveyed a variety of Object-Centered systems which offer different capabilities in this respect. We must observe, however, that no one of these systems is able to comprehensively cover the requirements we have discussed. We gave a special emphasis to those systems where the introduction of part-whole relations is accompanied by an appropriate semantics and by inferential mechanisms adequate for common sense reasoning.

We would like to conclude this paper mentioning that many problems have still to be faced regarding the ontological nature of meronymic relations. An open issue remains the adequate characterization of the different part-whole relations, in order to clarify the apparent dichotomy between generic part-of and specialized part-whole relations, and to define the behavior of composition in the latter case.

Acknowledgements

This work has been partially supported by the Italian National Research Council (CNR) project "Ontological and Linguistic Tools for Conceptual Modeling". The authors wish to thank Francesca Cesarini and Patrick Lambrix for their enlightening comments on a previous version of this paper. We would also like to thank Ulrike Sattler for the helpful discussions we had with her about the usefulness of the different part-whole relations in a real application domain.

References

[1] S. Abiteboul and R. Hull. IFO: A formal semantic database model. *ACM Transactions On Database Systems*, 12(4):525–565, 1987.

[2] A. Albano, G. Ghelli, and R. Orsini. A relationship mechanism for a strongly typed object-oriented database programming language. In *Proc. of the 17 th Int. Conference on Very Large Data Bases, VLDB-91*, pages 565–575, Barcelona, September 1991.

[3] Alessandro Artale, Francesca Cesarini, Elisabetta Grazzini, Fabio Pippolini, and Giovanni Soda. Modelling composition in a terminological language environment. In N. Guarino, S. Pribbenow, and L. Vieu, editors, *Workshop Notes of the ECAI-94 Workshop on Parts and Wholes*, pages 93–101, Amsterdam, August 1994.

[4] Alessandro Artale, Francesca Cesarini, and Giovanni Soda. Describing database objects in a concept language environment. *IEEE Transaction on Knowledge and Data Engineering*, to appear.

[5] M. Atkinson. *The Object-Oriented Database System Manifesto.* Altair, 1989.

[6] Franz Baader. Augmenting concept languages by transitive closure of roles: An alternative to terminological cycles. In *Proc. of the 12 th IJCAI*, pages 446–451, Sidney, Australia, 1991.

[7] Jochen Bernauer. Analysis of part-whole relation and subsumption in the medical domain. *Data & Knowledge Engineering*, 1995. This special issue.

[8] Luca Boldrin. Una rappresentazione terminologica della relazione parte-tutto. *AI*IA notizie, the Italian Association for Artificial Intelligence Journal*, 5(4):15–32, December 1992. (in italian).

[9] Ronald J. Brachman. On the epistemological status of semantic networks. In N.V. Findler, editor, *Associative Networks: Representation and use of knowledge by computers*, pages 3–50. Academic Press, London, 1979.

[10] Ronald J. Brachman, Deborah L. McGuiness, Peter F. Patel-Schneider, Lori Alperin Resnick, and Alexander Borgida. Living with CLASSIC: When and how to use a KL-ONE-like language. In J. Sowa, editor, *Principles of Semantic Networks*. Morgan Kaufmann, 1991.

[11] M.L. Brodie. Association: A database abstraction. In P.P. Chen, editor, *Entity-Relationship Approach to Information Modeling and Analysis*, pages 583–608. North-Holland, 1981.

[12] Martin Buchheit, Francesco M. Donini, and Andrea Schaerf. Decidable reasoning in terminological knowledge representation systems. In *Proc. of the 13 th IJCAI*, pages 704–709, Chambery, France, August 1993.

[13] Martin Buchheit, Manfred A. Jeusfeld, Werner Nutt, and Martin Staudt. Subsumption between queries to object-oriented databases. *Information Systems*, 19(1):33–54, 1994.

[14] Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. Structured objects: Modeling and reasoning. In *In Proc. of the* 4^{th} *International Conference on Deductive and Object-Oriented Databases, DOOD'95*, Singapore, December 1995.

[15] Diego Calvanese, Maurizio Lenzerini, and Daniele Nardi. A unified framework for class-based representation formalisms. In *Proc. of the 4 th International Conference on Principles of Knowledge Representation and Reasoning*, Bonn, Germany, May 1994.

[16] P.P. Chen. The entity-relationship model: toward a unified view of data. *ACM Transactions On Database Systems*, 1(1):9–36, 1976.

[17] E.F. Codd. A relational model for large shared data banks. *Communications of the ACM*, 13(6), 1970.

[18] D. A. Cruse. *Lexical Semantics*. Cambridge University Press, Cambridge, 1986.

[19] Giuseppe De Giacomo and Maurizio Lenzerini. Boosting the correspondence between description logics and propositional dynamic logics. In *Proc. of AAAI-94*, Seattle, WA, 1994.

[20] Giuseppe De Giacomo and Maurizio Lenzerini. What's in an aggregate: Foundations for description logics with tuples and sets. In *Proc. of the 13 th IJCAI*, Montreal, Canada, 1995.

[21] N. Derret, W. Kent, and P. Lyngbaek. Some aspects of operations in an object-oriented database. *IEEE Database Engineering Bulletin*, 8(4), 1985.

[22] K.R. Dittrich. Object-oriented database systems: The notion and the issues. In *Proc. of the Int. Workshop On Object-Oriented Database Systems*, Los Alamitos, CA, 1987.

[23] F. M. Donini, B. Hollunder, M. Lenzerini, A. Marchetti Spaccamela, D. Nardi, and W. Nutt. The complexity of existential quantification in concept languages. *Artificial Intelligence*, 53:309–327, 1992.

[24] F. M. Donini, M. Lenzerini, D. Nardi, and A. Schaerf. Deduction in concept languages: from subsumption to instance checking. *Journal of Logic and Computation*, 4(4):423–452, 1994.

[25] J. Doyle and R.S. Patil. Two theses of knowledge representation: language restrictions, taxonomic classification, and the utility of representation services. *Artificial Intelligence*, 48:261–297, 1991.

[26] Enrico Franconi. A treatment of plurals and plural quantifications based on a theory of collections. *Minds and Machines*, 3(4):453–474, November 1993.

[27] GALEN. Generalized architecture for language encyclopedias and nomenclatures in medicine, 1993. Project No:A2012 – The GRAIL Kernel: GALEN Representation and Integration Language, version 1.

[28] Peter Gerstl and Simone Pribbenow. A conceptual theory of part-whole relations and its applications. *Data & Knowledge Engineering*, 1995. This special issue.

[29] Peter Gerstl and Simone Pribbenow. Midwinters, end games, and bodyparts. A classification of part-whole relations. *Int. J. of Human-Computer Studies*, 43, 1995. To appear.

[30] A. Goldberg and D. Robson. *Smaltalk-80: The Language and its Implementation.* Addison-Wesley, Reading, MA, 1983.

[31] Nicola Guarino. Concepts attributes and arbitrary relations: Some linguistic and ontological criteria for structuring knowledge bases. *Data & Knowledge Engineering*, 8:249–261, 1992.

[32] Nicola Guarino. The ontological level. In R.Casati, B.Smith, and G.White, editors, *Philosophy and the Cognitive Sciences*, pages 443–456. Hölder-Pichler-Tempsky, 1994.

[33] Ira J. Haimowitz, Ramesh S. Patil, and Peter Szolovits. Representing medical knowledge in a terminological language is difficult. In R.A. Greenes, editor, *Proc. of the 12^{th} Symposium on Computer Applications in Medical Care*, pages 101–105, Washington DC, 1988. IEEE Computer Society Press.

[34] M. Hammer and D. McLeod. Database description with SDM: A semantic database model. *ACM Transactions On Database Systems*, 6(3):351–386, 1981.

[35] Philip Hayes. On semantic nets, frames and associations. In *Proc. of the 5^{th} IJCAI*, pages 99–107, Cambridge, MA, 1977.

[36] C.A.R. Hoare. *Communicating Sequential Processes.* Prentice-Hall, 1985.

[37] B. Hollunder and F. Baader. Qualifying number restriction in concept languages. In *Proc. of the 2^{nd} International Conference on Principles of Knowledge Representation and Reasoning*, pages 335–346, Cambridge, MA, 1991.

[38] R. Hull and R. King. Semantic database modeling: Survey, applications, and research issues. *ACM Computing Surveys*, 19(3):201–260, 1987.

[39] M. Iris, B. Lutowitz, and M. Evens. Problems of the part-whole relation. In M. Evens, editor, *Relational models of the lexicon*, pages 261–288. Cambridge University Press, Cambridge, 1988.

[40] H.V. Jagadish and X. Qian. Integrity maintenance in an object-oriented database. In *Proc. of the 18^{th} Int. Conference on Very Large Data Bases, VLDB-92*, Vancouver, British Columbia, Canada, 1992.

[41] Yeona Jang and Ramesh S. Patil. KOLA: A knowledge organization language. In *Proc. of the 13^{th} Symposium on Computer Applications in Medical Care*, pages 71–75. IEEE Computer Society Press, 1989.

[42] W. Kim, J. Banerjee, H.T. Chou, Jorge F. Garza, and Darrel Woelk. Composite objects support in an object-oriented database system. In *Proceedings of OOPSLA-87*, pages 118–125, Orlando, FL, October 1987. ACM Press.

[43] W. Kim, Elisa Bertino, and Jorge F. Garza. Composite objects revisited. In *Proc. of the ACM SIGMOD Int. Conf. on the Management of Data*, pages 337–347, Portland, OR, June 1989. ACM Press.

[44] Won Kim. A model of queries for object-oriented databases. In *Proc. of the 15^{th} Int. Conference on Very Large Data Bases, VLDB-89*, Amsterdam, 1989.

[45] Won Kim. *Introduction to object-oriented databases.* MIT Press, Cambridge, 1990.

[46] Won Kim, Nat Ballou, Hong-Tai Chou, Jorge F. Garza, and Darrel Woelk. Features of the ORION object-oriented database system. In Won Kim and Frederick H. Lochovsky, editors, *Object-Oriented Concepts, Databases, and Applications*, pages 251–282. Addison Wesley, Reading, MA, 1989.

[47] Won Kim, Nat Ballou, Hong-Tai Chou, Jorge F. Garza, and Darrel Woelk. A proposal for a formal model of objects. In Won Kim and Frederick H. Lochovsky, editors, *Object-Oriented Concepts, Databases, and Applications*, pages 537–559. Addison Wesley, Reading, MA, 1989.

[48] Roger King. My cat is object-oriented. In Won Kim and Frederick H. Lochovsky, editors, *Object-Oriented Concepts, Databases, and Applications*, pages 23–30. Addison Wesley, Reading, MA, 1989.

[49] G. Kuper and M. Vardi. The logical data model. *ACM Transactions On Database Systems*, 18(3), 1993.

[50] Patrick Lambrix and Lin Padgham. Part-of reasoning in description logics: A document management application. In A. Borgida, M.Lenzerini, D. Nardi, and B. Nebel, editors, *Proceedings of the International Workshop on Description Logics*, pages 106–108, Rome, Italy, June 1995. University of Rome "La Sapienza".

[51] M.E.S. Loomis. Object database semantics. *Journal of Object Oriented Programming*, 6(4):26–33, 1993.

[52] J. Lyons. *Semantics*. Cambridge University Press, Cambridge, 1977.

[53] B. Meyer. *Object-oriented Software Construction*. Prentice Hall International Series in Computer Science. Prentice Hall, New York, 1988.

[54] R. Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer-Verlag, 1979.

[55] Renate Motschnig-Pitrik and Veda C.Storey. Modelling of set membership: The notion and the issues. *Data & Knowledge Engineering*, 16(2):147–185, 1995.

[56] Amedeo Napoli. Subsumption and classification-based reasoning in object-based representations. In *Proc. of the 10 th ECAI*, Vienna, Austria, 1992.

[57] G. Nijssen and T.A. Halpin. *Conceptual Schema and Relational Database Design: A Fact Oriented Approach*. Prentice Hall of Australia, 1989.

[58] Lin Padgham and Patrick Lambrix. A framework for part-of hierarchies in terminological logics. In *Proc. of the 4 th International Conference on Principles of Knowledge Representation and Reasoning*, pages 485–496, Bonn, Germany, May 1994.

[59] Luca Pazzi. Dynamically-based complex objects: From process synchronization to entity aggregation. In N. Guarino, S. Pribbenow, and L. Vieu, editors, *Workshop Notes of the ECAI-94 Workshop on Parts and Wholes*, pages 57–67, Amsterdam, August 1994.

[60] N. Rescher. Axioms for the part relation. *Philosophical Studies*, 6:8–11, 1955.

[61] D. Robertson, A. Bundy, R. Muetzelfeldt, M. Uschold, and M. Haggith. *Eco-Logic: Logic-based approaches to ecological modelling*. MIT Press, 1991.

[62] J. Rumbaugh. Relations as semantic constructs in an object-oriented language. In *Proceedings of OOPSLA-87*, pages 466–481, Orlando, FL, October 1987. ACM Press.

[63] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen. *Object-Oriented Modeling and Design*. Prentice-Hall, Englewood Cliffs, NJ, 1990.

[64] Ulrike Sattler. A concept language for an engineering application with part-whole relations. In A. Borgida, M.Lenzerini, D. Nardi, and B. Nebel, editors, *Proceedings of the International Workshop on Description Logics*, pages 119–123, Rome, Italy, June 1995. University of Rome "La Sapienza".

[65] Klaus D. Schild. A correspondence theory for terminological logics: Preliminary report. In *Proc. of the 12 th IJCAI*, pages 466–471, Sidney, Australia, 1991.

[66] M. Schmidt-Schauß. Subsumption in KL-ONE is undecidable. In R. J. Brachman, H. J. Levesque, and R. Reiter, editors, *Proc. of the 1 st International Conference on Principles of Knowledge Representation and Reasoning*, pages 421–431, Toronto, Ontario, Canada, 1989. Morgan Kaufmann.

[67] M. Schmidt-Schauß and G. Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48(1):1–26, 1991.

[68] J. G. Schmolze and W. S. Mark. The NIKL experience. *Computational Intelligence*, 6:48–69, 1991.

[69] Lenhart Schubert. Problems with parts. In *Proc. of the 6 th IJCAI*, pages 778–784, Tokio, 1979.

[70] D. Shipman. The functional data model and the data language DAPLEX. *ACM Transactions On Database Systems*, 6(1), 1981.

[71] Peter Simons. *Parts: A Study in Ontology.* Clarendon Press, Oxford, 1987.

[72] Barry Smith. Mereotopology: A theory of parts and boundaries. *Data & Knowledge Engineering*, 1995. This special issue.

[73] J. F. Sowa. *Conceptual Structures: Information Processing in Mind and Machine.* Addison-Wesley, 1984.

[74] Piet-Hein Speel and Peter F. Patel-Schneider. CLASSIC extended with physical whole-part relations. In *Proceedings of the International Workshop on Description Logics*, pages 45–50. DFKI, Saarbrücken, Bonn, Germany, May 1994.

[75] Piet-Hein Speel and Peter F. Patel-Schneider. A whole-part extension for description logics. In N. Guarino, S. Pribbenow, and L. Vieu, editors, *Workshop Notes of the ECAI-94 Workshop on Parts and Wholes*, pages 111–121, Amsterdam, August 1994.

[76] S. Su. SAM*: A semantic association model for corporate and scientific statistical database. *Information Science*, 29, 1983.

[77] Mike Uschold. The use of the typed lambda calculus to guide naive users in the representation and acquisition of part-whole knowledge. *Data & Knowledge Engineering*, 1995. This special issue.

[78] Achille C. Varzi. Parts, wholes, and part-whole relations: The prospects of mereotopology. *Data & Knowledge Engineering*, 1995. This special issue.

[79] G. Verheijen and J. Van Bekkum. NIAM: An information analysis method. In *Information System Methodologies*, pages 537–589. North Holland, Amsterdam, NL, 1982.

[80] Robert Wilensky. Some problems and proposals for knowledge representation. UCB/CSD Report 87/351, University of California, Berkeley, CA, May 1987.

[81] Morton E. Winston, Roger Chaffin, and Douglas Herrmann. A taxonomy of part-whole relations. *Cognitive Science*, 11:417–444, 1987.

[82] R. Wirfs-Brock, B. Wilkerson, and L. Wiener. *Designing object-oriented software.* Prentice-Hall, 1990.

[83] William A. Woods and James G. Schmolze. The KL-ONE family. *Computer and Mathematics with Applications, special issue: Semantic Networks in Artificial Intelligence*, 23(2-5):133–177, March-May 1992.