The Intentional Relationship of Representation between the Constructs of a Language and Reality

José M. Cañete-Valdeón *, Francisco J. Galán, Miguel Toro

Dept. of Computer Languages and Systems, University of Sevilla, Spain

Abstract

Specifications of conceptualisations (ontologies) are often employed for representing reality, both in knowledge representation and software engineering. While languages offer sophisticated constructs and rigorous semantics for building conceptual entities, no attention is paid to the relationship between such entities and the world they intend to represent. This paper studies such a relationship and provides empirical evidences in favour of two main hypotheses: (1) conceptualisations are insufficient to fully represent the specifics of reality; (2) languages (both representation and design-oriented) are general representations of (classes of) systems in the world, and they can be characterised as scientific theories. The first hypothesis establishes a problem for which we propose a solution based on the explicit elaboration of statements claiming the similarity (in some respects and to certain degrees of accuracy) between conceptual entities and real-world systems of interest. The second hypothesis constitutes a new perspective for understanding languages, whose advantages to representation and design are discussed in detail.

Key words: Language definition (19.1), Ontologies (26), Conceptual modelling (7), Knowledge representation techniques.

1 Introduction

A commonly accepted use of ontologies is the representation of human knowledge [6]. However several controversies remain open in this respect. One of

* Corresponding author. Tel.: (+34) 954 553 873. *Postal address:* Departamento de Lenguajes y Sistemas Informáticos. Escuela Técnica Superior de Ingeniería Informática. Avenida de la Reina Mercedes, S/N. 41012. Sevilla. Spain.

Email address: jmcv@us.es (José M. Cañete-Valdeón).

them concerns the kinds of knowledge that ontologies can suitably represent. Another difficulty is related with the achievement of a precise definition: while Gruber's proposal [1] still holds, it has some rough edges, principally being that there are no agreed-upon borders concerning what is in a specification [35].

In this paper we study the role of ontologies as mechanisms for representing the *real world* instead of representing *knowledge*, thus avoiding complex questions from Epistemology such as whether beliefs can or cannot be regarded as knowledge. Due to the fact that ontologies are mostly intended to represent real-world elements, we feel justified in making this decision.

Our focus is on ontologies behind languages employed in software development. We are considering languages intended for *representing* what exists as well as those intended for *designing* new entities. Please note that we employ the term "design" in a very general sense (as in [47]), not wishing to confine it to the concrete activity following analysis in typical software life cycles.

Our first main hypothesis claims that an important gap exists between a conceptualisation and a representation of the world. This can be stated as:

Hypothesis 1 *Conceptualisations are insufficient to fully represent the specifics of reality.*

Let us introduce an example to illustrate and motivate our point.

Example 1 Consider the following fragment of an ontology intended to represent the circulation in the Spanish railway transport. The specification language is OWL [46]. The names of classes and properties have been underlined for clarity.

```
<Route rdf:ID="AVE09615-Sevilla-Madrid">
  <train rdf:resource="#AVE09615"/>
  <departureTime rdf:datatype="&xsd;time"> 07:00:00 </departureTime>
  <arrivalTime rdf:datatype="&xsd;time"> 09:30:00 </arrivalTime>
  <departureStation rdf:resource="#Sevilla-SantaJusta"/>
  <arrivalStation rdf:resource="#Madrid-PuertaDeAtocha"/>
  <circulationPeriod rdf:datatype="&xsd;string"> Current year
    </circulationPeriod>
</Route>
```

Assuming that today is not December 31st, the following proposition can be deduced from the ontology:

> Tomorrow, train 09615 will depart "Sevilla-Santa Justa" station at 07:00:00 and will arrive at "Madrid-Puerta de Atocha" station at 09:30:00.

[1] An ontology is an explicit specification of a conceptualisation [20].

This statement is intended to represent what will happen tomorrow. However it is extremely likely that the prediction turns out to be false. Assuming we have agreed on a reference clock, the proposition would be true if the train left Sevilla station when the clock showed *exactly* 07:00:00, and arrived at Madrid station *exactly* when the clock indicated 09:30:00. Only one second earlier or later would make the proposition false.

In practice, passengers add a certain margin of uncertainty on predicates such as this one. If tomorrow the train arrived at 09:32:00, most passengers would probably still regard the prediction as "true" but certainly not if the train arrived at 09:50:00. It seems that people *complement* the predicate above with a margin of confidence to obtain a representation of what will happen tomorrow. Now consider the point of view of the Spanish railway company, Renfe. The company also complements the predicate with a margin of confidence. Indeed Renfe relies on such a margin: since the beginning of high-speed trains in Spain, the company committed itself to refund tickets of passengers whose trains arrived 5 or more minutes late. The representation of the world (on which Renfe relies) is not the simple assertion of the specification (ontology), but the statement of a certain confidence between the conceptualisation and what happens in the real world. But the underlying problem (and its consequences) is deeper than the mere specification of error margins as we will discuss throughout this paper.

Hypothesis 1 introduces an important problem: how can ontologies be employed for representing the world? In this paper we propose an answer from a knowledge field which has traditionally faced the problems of representation: Philosophy of Science and, particularly, Constructive Realism [16]. We regard ontologies, such as the one in Example 1 (at both the class and instance levels), as linguistic entities which *define* conceptualisations. These are not linguistic entities and therefore they do not make any claims about the real world, i.e. they are not representations *per se*. Representing the world requires an agent (e.g. a human being) with an intention, which may be materialised in the construction of an explicit predicate (a linguistic entity) linking a chosen conceptualisation to some identified part of the world. Later we will give some arguments in favour of characterising the relationship between a conceptualisation and the world as one of *similarity*. Finally, a representation of the world is a conceptualisation together with a predicate (or *hypothesis*) with the following structure:

> *This conceptualisation is similar to this identified real-world system (or class of systems) in such-and-such respects and to such-and-such degrees of accuracy.*

Thus, the ontology of which Example 1 shows a fragment *defines* a conceptualisation C; in particular, the conceptual train 09615 leaves conceptual Sevilla

exactly at 07:00:00 and arrives at conceptual Madrid exactly at 09:30:00, every day of the current year. Representing the world requires building a hypothesis, i.e. a predicate which relates a conceptualisation (e.g. C) with a real-world entity (e.g. the railway system in Spain) in some respect (e.g. tomorrow's departing time of train 09615) and within a certain degree of accuracy. The particular formulation of degrees will depend on each case. In this example, a positive value (resp., a negative value) denotes a fixed delay (resp., a fixed time early) and therefore it points out a fixed distance between the conceptualisation and the world in the claimed similarity. A zero value denotes no delay at all and therefore it is equivalent to claim an "exactly equal" similarity. Finally, an interval of values expresses uncertainty in the departing time and hence it introduces uncertainty in the hypothesis. For example:

> *Conceptualisation C is similar to the railway system in Spain with respect to "tomorrow's departing time of train* 09615*" within a degree of accuracy of* $[+1, +3]$ *minutes.*

With this hypothesis, we are claiming that tomorrow's train 09615 will leave Sevilla at some time between 07:01:00 and 07:03:00.

Our proposal highlights the *intentional* nature of representation. A conceptualisation is not a representation of anything merely by itself: one must explicitly claim so, in some respects and to certain degrees, for some purpose.

We validate Hypothesis 1 with three additional examples. Further, we seek the causes of such a hypothesis through the analysis of a sample of languages commonly employed in software development activities and some of them also in Artificial Intelligence: KAOS, i^*, Problem Frames, Statemate, Actors, and C++. These case studies provide evidence for our other main hypothesis:

Hypothesis 2 *Languages are general representations of certain classes of systems in the world, and they can be characterised as scientific theories.*

The hypothesis that languages can be regarded as representations of the world is of paramount importance. As with scientific theories, languages could be (empirically) validated, they could be chosen according to our current goals, and they could compete among themselves for becoming the most adequate representation for some goal.

The rest of this paper is organised as follows. The next section provides some background on scientific representation from a constructive realist perspective. Section 3 provides additional examples to validate Hypothesis 1 and proposes a rigorous formulation of similarity hypotheses. Section 4 identifies constructed genres in languages. Sections 5 and 6 provide evidences for Hypothesis 2 through the analysis of representation-oriented and design-oriented languages, respectively. Finally we discuss the benefits of our approach in

Section 7. We close in Section 8 with a survey of related works.

2 Background on representation in Philosophy of Science

This section presents an introduction to the Constructive Realist account on the structure of scientific theories. Two features make this account particularly appropriate for the purposes of this paper: on the one hand, its basis on the "model" concept and, on the other, the proposed relationship between models and the real world. Constructive Realism is better understood by first studying the view of scientific theories that was dominant in the Anglo-American philosophy of science until about 1960: the logical empiricist account.

2.1 Logical Empiricism

Logical Empiricism understands a scientific theory as *a linguistic entity* (i.e. a set of statements) *about the real world*. The account proposes a canonical formulation to which any "genuine theory" can be rewritten — otherwise it is not a theory. The canonical form consists of a system of axioms and a set of correspondence rules. The axioms are statements formulated in a first-order mathematical logic with identity, and they establish properties about the so-called "theoretical terms". Such non-logical terms are defined by employing "observational terms," which are directly interpreted as physical objects in turn. The correspondence rules are the statements that contain such definitions. For example, a rule may define the theoretical term "mass" as the result of performing certain measurements M on some object under circumstances S, where M and S are specified with observational terms [43].

We see that Logical Empiricism grants much importance to linguistic concerns; for this reason it is also known as the "Syntactic View" of scientific theories. And this is its main weakness: for example, as correspondence rules are part of a theory, any simple changes to the definition of the theoretical terms would constitute a new theory. We will explain a different argument against this doctrine below. For a comprehensive criticism the reader may refer to [43].

2.2 Rejecting Logical Empiricism: the model-based view of scientific theories

Ronald Giere [17, p. 122] introduces the term "model-based view" to generically refer to the "semantic" accounts on scientific theories. This term aims to reflect the common agreement among philosophers that predicates present

in texts about theories (i.e. linguistic entities such as "pendulum" in classical mechanics) do *not* refer to anything in the real world. Note that this is radically opposite to Logical Empiricism. Instead, predicates refer to conceptual, idealised entities called "models". For example, the predicate "simple pendulum" refers to a mass swinging from a massless string attached to a frictionless pivot, subject to a uniform gravitational force, and in an environment with no resistance. This is clearly an *ideal object*: no real pendulum exactly satisfies any of these conditions. So no real pendulum is a simple pendulum as characterised in classical mechanics. The same is true for more complex types: damped pendulums, driven pendulums, and so on.

There exist different philosophical accounts, some of them opposite, that share this idea of models as a common point. Next we present an introduction to a major account of this category: Constructive Realism.

2.3 Constructive Realism

Constructive Realism is a doctrine on the structure of scientific theories developed by Giere [15–18]. The author provides evidence from diverse fields such as classic mechanics, geology, and nuclear physics.

Giere proposes to regard the simple pendulum, the simple harmonic oscillator, and the other kinds of classic mechanics systems as *abstract entities* having all and only the properties ascribed to them in textbooks. For example, the simple harmonic oscillator is viewed as an ideal system that perfectly satisfies the statement $F = -kx$. According to the author all the objects referred to in scientific texts are *constructed entities*, indeed *socially* constructed entities, in the sense that they have no reality beyond that given to them by the community of scientists. He calls such systems "theoretical models," a term commonly used by scientists themselves.

Equations and sentences are not the only linguistic resources employed in science to define theoretical models: diagrams also play an important role [17, ch. 7]. But the particular resources used to characterise models are of at most secondary interest.

Models are employed by scientists to represent the world. A "theoretical hypothesis" is a linguistic entity, namely a statement or proposition, asserting some kind of relationship between a theoretical model and a designated real-world system or class of systems. This gives us the complete picture: a scientific theory is not a set of statements as in Logical Empiricism but an heterogeneous entity consisting on: (1) a family of models, and (2) a collection of statements (hypotheses) claiming some link between the models and the world.

Giere also analyses the relationship between theoretical models and systems in the world. He proposes to characterise such a relationship in terms of *similarity*, which avoids the epistemological problems associated with the concept of "truth". The author defends the notion by appealing to evidence from cognitive sciences, which suggest that human cognition and perception operate on the basis of some sort of similarity metric.

It follows that a theoretical hypothesis claims a certain similarity between a model and some part of the world. But since anything is similar to anything else in some respects and to some degree of accuracy, claims of similarity are vacuous without specifying the relevant *respects* and *degrees*. Therefore the general form of a theoretical hypothesis is [16, p. 81]:

> *Such-and-such identifiable real system is similar to a designated theoretical model in the following indicated respects and degrees of accuracy.*

For example: "the positions and velocities of the Earth and Moon in the Earth-Moon system are very close to those of a two-particle Newtonian model with an inverse square central force". Here the respects are "position" and "velocity," while the degree is claimed to be "very close". The identified part of reality is the system constituted by the Earth and the Moon.

3 Theoretical models and similarity hypotheses

This section begins with additional empirical evidence for Hypothesis 1 established at the Introduction. Next we propose a solution to the problem of representation with ontologies, based on Constructive Realism. In this sense we first establish a number of definitions, then we formalise some concepts, and finally we apply our technique to the previous examples.

3.1 Validating Hypothesis 1 with empirical evidence

In Section 1 we introduced an example to illustrate our first main hypothesis, which we reproduce for convenience:

Hypothesis 1 *Conceptualisations are insufficient to fully represent the specifics of reality.*

Here we present three additional examples to further validate our claim.

Example 2 The following is a fragment of an OWL specification intended to represent the customers of a bank:

```
<Customer rdf:ID="c05501">
  <firstName rdf:datatype="&xsd;string"> José </firstName>
  <lastName rdf:datatype="&xsd;string"> Pérez </lastName>
  <address rdf:datatype="&xsd;string"> 3 Diamante St, Sevilla 41009,
   Spain </address>
</Customer>
```

The bank interprets this specification as the following predicate:

Mr José Pérez lives at 3 Diamante St, Sevilla, Spain.

Can we state that this predicate constitutes a representation of the world? The statement might easily be false. Perhaps the address corresponds to Mr Pérez parents' home, where he lived until he moved to his own home some months ago but he has not communicated this event to the bank yet. Or perhaps Mr Pérez has passed away. Or perhaps he lied when he indicated such information to the bank.

Two questions arise from this example. On the one hand, what degree of confidence has the bank about the ontology? The specification contains no information about this. On the other hand, is this confidence of any interest to the bank? It is reasonable to think so. If the customer lives at another address, then the bank correspondence should be redirected. If the customer has passed away, the bank must legally communicate the death to the National Insurance under some circumstances. As in Example 1, the confidence in this information is not specified, so one might assume anything. Therefore this is not a representation of the world yet.

Example 3 Consider a software system which monitors the altitude of an aircraft. The aircraft is equipped with two altimeters which continuously communicate the current altitude to the system. A Java object `alt` of class `Altitude` gathers the current data from the altimeters. The following is a fragment of the class definition:

```
public class Altitude {
    private String aircraftId;
    private int altitude1;
    private int altitude2;
    // Rest of the class definition.
}
```

Object `alt` is intended as a *representation* of the aircraft's current altitude in feet (Jackson [28] adequately calls such kind of object a *model*). However no physical altimeter is perfect. Therefore the following predicate is false:

The aircraft altitude is exactly either `alt.altitude1` or `alt.altitude2`.

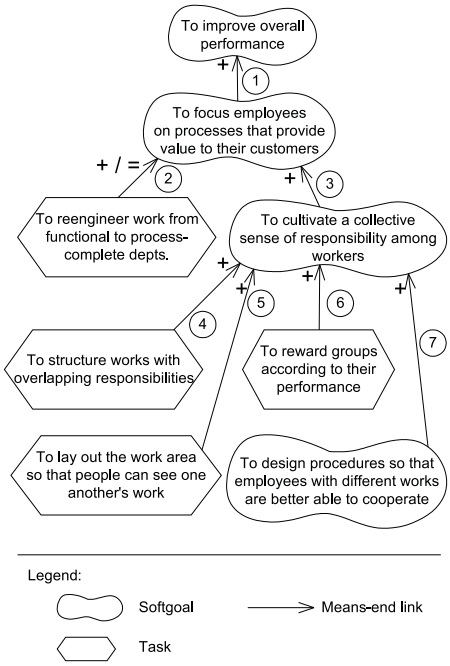

Fig. 1. Theoretical model on business performance.

Does this matter? Absolutely. The operations in Reduced Vertical Separation Minimum (RVSM) Airspace require the current altitude of the aircraft *to be known* within a ± 80 feet margin [12]. Therefore what the representation asserts must be a *true* hypothesis about the real aircraft.

Example 4 Majchrzak and Wang [31] present an analysis of several U.S. electronics manufacturers. They study the strategies adopted by such companies in order to focus their employees on processes that provide value to their customers. The authors first propose a particular theory about the companies they have considered as a sample. At the end of the paper Majchrzak and Wang extrapolate their results to general businesses, thus formulating a general theory. This example will be lengthier than the preceding ones as our intent will be to reconstruct the two representations of the world (the particular and the general theories) by employing the i^* language [48,49].

Regarding the particular sample, the authors observe that a certain percentage of the companies followed the strategy of reengineering work in order to focus employees on processes that clearly provide value to customers. The overall objective is to achieve better performance, understood as lower costs, shorter cycle times, and greater customer satisfaction. Reengineering work consists of transforming functional departments into process-complete ones. While the former focus on a certain function, the latter are able to perform all the cross-functional tasks required to meet customers' needs. However, the authors claim, this strategy did not always succeed in the companies of the

sample. In fact the only process-complete departments that were observed to have faster cycle times than the functional ones resulted to be those whose managers had implemented methods to cultivate a collective sense of responsibility among workers. Four of such methods were observed in the sample: to structure jobs with overlapping responsibilities, to base rewards on group performance, to lay out the work area so that people can see one another's work, and to design procedures so that employees with different jobs are better able to collaborate.

Majchrzak and Wang's representation is focused on strategies taken to achieve enterprise objectives and on the actual effect achieved by each strategy. For this reason we will centre on the portion of the i^* language known as "the Strategic Rationale". The Strategic Rationale requires the designer to classify the objectives of an organisation according to whether it is possible or not to sharply define their criteria of success at the outset, i.e. before the objectives are being pursued. In this sense two concepts are offered: "goal" and "softgoal," respectively. We note that this respect is not one of the topics of Majchrzak and Wang's theory, so we would have preferred not to specify it. Yet the family of i^* concepts is provided as-is, so we will have to make such a decision every time we find an objective. [2]

The global objective is to improve the overall performance of a company. In our opinion, it is not possible to precisely define at the outset how far this can be achieved, i.e. managers cannot construct this objective with accuracy. Therefore we would classify it as a softgoal. We have reasoned in a similar way for the remaining objectives, obtaining the subgoals depicted in Fig. 1 with peanut-like shapes.

As we understand the paper, the strategy of reengineering work is an activity (sequence of steps) to be done in order to achieve an objective. For this reason we have employed the "task" concept to build an entity that has the same meaning as the strategy. It is symbolised as an hexagon in Fig. 1. In contrast, the strategy of cultivating a collective sense of responsibility among workers is not a sequence of steps because we do not even know how to precisely define "collective sense of responsibility". In our opinion such a strategy is better understood as a softgoal.

We have employed the i^* concept of "means-end" to express Majchrzak and Wang's statements about the contributions of the two strategies to the objectives. According to the authors, reengineering work contributed positively in some cases to focus employees on processes that provide value to their customers, but in other cases the strategy resulted in no contribution at all. In contrast, the strategy of cultivating a collective sense of responsibility among

[2] Later we will see how to avoid making claims about such a respect. In consequence, the concrete category in which we classify each objective will be irrelevant.

workers is claimed to contribute positively to the objective. The "means-end" claims are symbolised with arrows in Fig. 1.

Previously we have indicated that the authors report four methods employed by the companies of the sample in order to favour the collective responsibility among workers. Three of them, namely to structure jobs with overlapping responsibilities, to reward groups, and to redesign the work area, are activities to be carried out. For this reason we will employ the "task" concept in the construction of the corresponding parts of the i^* model. In contrast, the fourth method (to design procedures so that employees with different jobs are better able to collaborate) has not a clear criterion of success: what does it mean to "collaborate better"? For this reason we will classify it as a softgoal. The authors state that the four methods were successful ways to favour the collective responsibility in the studied cases. We will express this statement with the concept of "means-end" as four positive contributions (see Fig. 1).

Interestingly, Majchrzak and Wang's representation does not end here. The authors make a number of hypotheses about both the particular sample of companies and the general case. We will label some of such statements for future reference:

- (*Particular sample*) h_1: All the departments which implemented the cited strategies achieved the corresponding contributions.
- (*General case*) h_2: Cultivating a collective sense of responsibility among workers will always produce a positive contribution to the objective of focusing employees on processes that provide value to their customers [31, p. 95].
- (*General case*) h_3: Regarding the four strategies to achieve a collective sense of responsibility among workers, the authors claim that "no one approach is appropriate for all process-complete departments" [31, p. 99].

Without hypotheses, Majchrzak and Wang's paper would not be claiming anything about the real world: there would be no representation at all.

We can identify the following representation problems from this example:

(1) The i^* conceptualisation requires to represent a fixed number of respects, independently of whether we are interested in them or not (in this example we were not interested in the respect of whether it is possible to sharply define the criteria of success for an objective at the outset).
(2) Majchrzak and Wang employ their model to represent a particular sample first and the general case next, asserting different margins of confidence. This separation between the model on the one hand and the real-world subject of interest on the other does not exist in i^*.
(3) According to i^* semantics, our model of Fig. 1 is interpreted as a direct claim about the world. Yet Majchrzak and Wang are not asserting that

their model is *exactly equal* to reality: they state a number of degrees of confidence which cannot be expressed using the i^* conceptualisation.

3.2 Adopting a constructive realist view

The problem with Examples 1 to 4 has its roots in our interpretation of the specifications, as propositions about reality. This is exactly the same view as the one adopted by logical empiricists about scientific theories.

In order to avoid the mistakes of Logical Empiricism, we propose the adoption of a constructive realist perspective. We regard the preceding specifications (ontologies) as linguistic resources *defining* conceptual entities. Therefore such entities satisfy their specifications in a trivial way (e.g. Example 2 defines a conceptual customer who lives at a conceptual address). Next, an *intention* is necessary to represent reality: this can be materialised as a statement claiming the *similarity* between the conceptualisation and some identified real-world system. As, according to Giere, anything can be similar to anything else in countless respects and degrees of accuracy, a claim of similarity must include the relevant respects and degrees. Next we provide precise definitions for these concepts.

3.3 Definitions

Definition 1 (*linguistic resource*): a linguistic resource is an entity employed for the purpose of description or definition.

Examples of linguistic resources are: equations, diagrams, ontologies, and predicates.

Definition 2 (*theoretical model*): a theoretical model is a non-linguistic, conceptual entity which is to be employed for representation.

Examples of theoretical models are: the conceptualisation about trains of Example 1 and the conceptual customer of Example 2.

Definition 3 (*similarity hypothesis*): a similarity hypothesis is a predicate which asserts the similarity between a theoretical model and an identified real-world system (or class of real-world systems) in some respects and to certain degrees of accuracy.

Examples of similarity hypotheses will be provided at the end of this section. In some cases we will be interested in representing not only physical individuals

but also existing conceptual entities such as a computer program or a legal contract.[3] Therefore, by "real-world system" we should understand anything that exists, whether physical or conceptual.

Mathematical functions and equations can be used as linguistic resources for expressing hypotheses in a formal way. Richter [36] surveys several alternative ways to represent similarity in Case-Based Reasoning (CBR) including a function $sim(x, y) : U^2 \rightarrow [0, 1]$ interpreted as the measure of the degree of similarity between objects x and y in a universe U, and a binary predicate $SIM(x, y) \subseteq U^2$ interpreted as "x and y are similar". We adopt some of the measures reported by Richter and extend the CBR notion of similarity with respects.

Let m denote some theoretical model in a conceptual universe (denoted as U), let s denote an identified system (or class of systems) in the real world (denoted as RW), let $r \in RESP$ denote a respect of both m and s, and let DEG denote a (possibly ordered) set of degrees of accuracy. The following functions are alternatives for expressing similarities:

$sim(m, s, r) : U \times RW \times RESP \rightarrow DEG$

$sim(m, s, r) : U \times RW \times RESP \rightarrow Interval(DEG)$

$sim(m, s, r) : U \times RW \times RESP \rightarrow Set(DEG)$

A similarity hypothesis can now be expressed in a more rigorous way as an equation:

Equation 1 $sim(m, s, r) = d$

where d is a degree in DEG, $Interval(DEG)$, $Set(DEG)$, or some other structure on DEG. Equation 1 is interpreted as the predicate: "theoretical model m is similar to the identified real-world system s in respect r to degree of accuracy d". A hypothesis can express uncertainty by employing intervals and sets of degrees. Claims about several respects can be combined in a single hypothesis; for example:

$sim(m, s, r_1) = d_1 \wedge sim(m, s, r_2) = \{d_2, d_4\}$

In cases where units of measurement are interesting for the expression of degrees, they can be made part of the formulation of DEG. Thus, the interval of [+1, +3] minutes employed in Example 1 (page 4) can be expressed by taking DEG to be $\mathbb{Z} \times \{min\}$, where min is a constant denoting minutes. Then, an

[3] Both a program and a legal contract may have linguistic descriptions but their nature is conceptual.

element in $Interval(DEG)$ is $[(+1, min), (+3, min)]$. A shorthand notation is $[+1, +3]$ min.

In hypotheses about a class of real-world systems it may be interesting to include *how many* of such systems are similar to the theoretical model (in some respects and to certain degrees of accuracy). This can be expressed in a quantitative way (e.g. a range or a percentage) or in a qualitative way (e.g. labels such as all, most, almost none, and so on). Then the similarity hypothesis can be formulated as the equation:

Equation 2 $sim(m, s, r) = (f, d)$

where f is a frequency in F. Examples of F are an interval of percentages $[0, 100]$ and a set of labels $\{ALL,\ ALMOST\_ALL,\ MOST,\ ...\}$.

3.4 Establishing similarity hypotheses on the examples

Equipped with the preceding set of definitions, we revisit the four examples introduced so far to explicitly state their associated hypotheses, thus completing the representations.

Example 1 Let m be the conceptualisation denoted by the OWL ontology. In particular, m contains a conceptual train 09615 which leaves conceptual Sevilla everyday exactly at 07:00:00 and arrives at conceptual Madrid exactly at 09:30:00. Let s be the railway system in Spain and let r be the respect "tomorrow's departing time of train 09615". Next let us take DEG to be $\mathbb{Z} \times \{min\}$ (see above). Then a possible representation of the part of the real world s in respect r can be expressed as the following equation:

$sim(m, s, r) = [(+1, min), (+3, min)] = [+1, +3]\ min$

which is interpreted as the hypothesis: the conceptualisation described by the OWL ontology is similar to the Spanish railway system in the respect "tomorrow's departing time of train 09615" to a degree of accuracy equal to the interval between 1 and 3 minutes.

Example 2 We can distinguish several levels of confidence regarding the address of a bank customer, according to how the bank actually obtained that information. Typical possibilities include:

- A national identity card showed by the customer to the bank.
- A recent bill (of electricity, phone, etc) showed by the customer to the bank.
- A written statement signed by the customer to the bank.
- A verbal statement expressed by the customer to the bank.

If we associate a label to each possibility, we can obtain a simple set of degrees of accuracy for our hypotheses: $\{NIC, BIL, WST, VST\}$.

Now let m be the theoretical model described by the ontology of the example; in particular, the model contains a conceptual customer named José Pérez who lives at 3 Diamante St in Sevilla. As we are describing a conceptual customer, all we assert about him is trivially true. This conceptualisation is exactly what we obtain when we *interpret* the specification (ontology). Let s be the real-world bank customers, and let r be the respect "current José Pérez's address". Now the bank is able to make a representation of the world as the following hypothesis:

> *Model m is similar to the part of the world s in respect r to degree of accuracy NIC.*

Or, more formally:

$sim(m, s, r) = NIC$

Example 3 The `altitude1` and `altitude2` attributes of object `alt` are not a theoretical model. To be precise, it is *our interpretation* of such attributes the one which builds a conceptual entity: namely, a conceptual aircraft whose altitude is exactly either `alt.altitude1` or `alt.altitude2`. Let such a conceptual aircraft be our theoretical model, m, let s be the real-world aircraft, and let r be the respect "current altitude".

Next let us take DEG to be $\mathbb{Z} \times \{ft\}$, where ft is a constant denoting feet. Uncertainty in hypotheses about the altitude of the aircraft can be expressed as intervals of DEG. For example, assume that the aircraft altimeters have an error margin of ± 25 feet. Then we can make the following hypothesis:

> *The real-world aircraft, s, is similar to model m in respect r to degree of accuracy $[-25, +25]$ feet.*

More formally:

$sim(m, s, r) = [(-25, ft), (+25, ft)] = [-25, +25]\ ft$

Example 4 The i^* conceptualisation allows us to construct theoretical models but the language lacks resources for expressing similarity hypotheses about the relationship between the models and reality. In this case we propose to define degrees of accuracy in a relative way, by contrasting the value of a given respect r in a theoretical model and the value of r in a real-world system.

For example, let r be the respect "contribution to a softgoal". In i^* there are several types of contributions to softgoals including *makes*, *helps*, *positive*,

Table 1
Spectrum of relative similarity degrees in the respect "contribution to a softgoal"

Model	degree	Real world
	$\overset{P_{+2}}{\longleftrightarrow}$	makes
	$\overset{P_{+1}}{\longleftrightarrow}$	helps
positive	$\overset{P_{0}}{\longleftrightarrow}$	positive
	$\overset{P_{-1}}{\longleftrightarrow}$	equal
	$\overset{P_{-2}}{\longleftrightarrow}$	negative
	$\overset{P_{-3}}{\longleftrightarrow}$	hurts
	$\overset{P_{-4}}{\longleftrightarrow}$	breaks

equals, *negative*, *hurts*, and *breaks*. Assume a model where a certain softgoal receives a positive contribution and a real-world system where a softgoal receives a positive contribution too. This is an "exactly equal" similarity degree, which we may denote as P_0. Now assume that that the contribution to the softgoal is positive in the model and neutral in the real world. This defines another degree in the respect of interest. We may also agree that this degree denotes a lower similarity than the preceding one, so we may denote it as P_{-1}.

This way we introduce a new degree by contrasting a new pair of values (see Table 1). The first letters of the contribution may be employed for the label (e.g. N for "negative," HE for "helps," HU for "hurts"). As the effects of contributions to softgoals can be ordered from *makes* to *breaks*, we use signed subindexes to refer degrees in a relative manner.

Now we can rigorously state the hypothesis on the particular sample at page 11, h_1. Let m be the model denoted by the diagram of Fig. 1, let s_s be the real-world departments of the sample which implemented the strategies, and let r be the respect "contributions to softgoals". Consider a qualitative definition of $F = \{ALL,\ ALMOST\_ALL,\ MOST,\ ...\}$. Hypothesis h_1 can be stated as:

$$sim(m, s_s, r) = (ALL, \{P_0, N_0\})$$

which is interpreted as the predicate: all the departments of the sample share either a degree P_0 or N_0 with the model in the respect under consideration. This hypothesis claims that if the model indicates that strategy S produces a positive (respectively, neutral) contribution to softgoal G, then S produces a positive (respectively, neutral) contribution to G in the sample of real departments.

Next let us formulate the two general hypotheses of Majchrzak and Wang's work labelled as h_2 and h_3 (page 11). Both refer to real-world businesses whose departments are process-complete. We will denote such a class of real-world

systems as s_b. Regarding hypothesis h_2, let r_{m3} be the respect "contribution of cultivating a collective sense of responsibility among workers to the objective of focusing employees on processes that provide value to their customers". In the theoretical model such a respect corresponds to the contribution indicated by means-end 3 (see Fig. 1). Hypothesis h_2 can be stated as:

$sim(m, s_b, r_{m3}) = (ALL, P_0)$

Now consider the other general hypothesis, h_3. Regarding the four strategies to achieve a collective sense of responsibility among workers, the authors claim that "no one approach is appropriate for all process-complete departments". This means that none of the four strategies is universally P_0 for all businesses with process-complete departments, s_b. Consider four respects, from r_{m4} to r_{m7}, denoting the contribution of each strategy to the softgoal (see Fig. 1). Then hypothesis h_3 can be restated as four simpler hypotheses:

$\forall i \in [4, 7] \bullet sim(m, s_b, r_{mi}) \neq (ALL, P_0)$

4 Representation at the language level

So far we have investigated examples of representations employing instance-level concepts such as a particular `Route`, a particular `Customer`, a Java object, and a collection of softgoals and tasks. Our results have provided evidence in favour of the fact that something more is necessary in order that a conceptualisation can constitute a representation of reality. In this sense we have made a proposal based on Constructive Realism which fills the representational gap. However we are interested in going further and determine the cause of this situation. For this reason we will be *observing languages* in the rest of this paper.

In particular, this section analyses three case studies: KAOS, i^*, and Problem Frames; the obtained results allow us to reject a syntactic or literal interpretation of the terms and predicates included in texts defining languages, and to embrace a semantic view.

4.1 Rejecting the syntactic view of languages

Languages employed in software development activities are intended for representation, design, or both. Representation means to build an image of what exists; design refers to create a new (conceptual) entity. The term "modelling" is popularly employed for the activities of representing and designing.

In this section we focus on languages which include representation among their purposes. The material for our study consists of the books, papers, and specifications where such languages are described.

The language proposed as part of the KAOS project [10] is intended for the "acquisition" of both functional and non-functional requirements of composite systems. Requirements acquisition involves learning and negotiation [10, p. 6]. To this aim the language allows for the representation of requirements:

> It is aimed at being sufficiently rich to allow *both* functional and non-functional requirements for *any* kind of composite system to be captured in a precise and natural way.

The i^* framework is intended for "modelling and reasoning about organisational environments and their information systems" [48, p. 227]. It includes a language which allows the building of two kinds of entities: the Strategic Dependency model and the Strategic Rationale model. Both of them may be used for representation of the existing organisation [48, p. 227]:

> The Strategic Dependency (SD) model is used to describe the dependency relationships among various actors in an organizational context. The Strategic Rationale (SR) model is used to describe stakeholder interests and concerns, and how they might be addressed by various configurations of systems and environments.

Problem Frames [28] is an approach for analysing and structuring software development problems. It includes a language for building "context diagrams" and "problem diagrams," which allow to represent the physical elements of a composite system-to-be, their relationships, and the overall requirement that must be satisfied. According to Jackson [28, p. 48]:

> A context diagram shows the parts of the world where your problem and its solution machine are located, and the interfaces by which those parts are connected. But the problem itself — that is, the requirement — is not represented in the context diagram. [...] The requirement is always about the problem domains, so you need to show how it's related to these domains, and what roles the domains play in the problem. A diagram that shows these things is a problem diagram.

A common feature of all the texts which describe languages is that a number of terms are introduced. Some from our three case studies are:

- KAOS: goal, agent, operationalization, responsibility.
- i^*: intentional actor, goal, belief, ability.
- Problem Frames: domain, phenomenon, interface, event.

One may reasonably argue that these terms must be interpreted as referring to elements in the real world. Under this perspective, the interpretation of

syntactic terms in the world would constitute their semantics.

The problem with this view is that general propositions included in texts and involving such terms must also be interpreted as claims about the world; this makes such propositions empirically false. For example, our reference text about KAOS claims that the behaviour of agents consists of discrete states, and that agents are able to control their own state transitions [10, p. 18]. At the same page the text identifies humans as examples of agents. But we cannot find any human whose behaviour is discretized, so the statement is not true with respect to the world.

In the i^* approach, organizations are regarded as consisting of intentional actors. These are claimed to be semi-autonomous units, whose behaviour is regulated by social relationships within which they have freedom of action [49, p. 127]. This proposition cannot be regarded as a general truth applicable to the real world. The reason is that actors may represent humans [49], and people violating social constraints have been observed in real organisations.

Regarding our third example, events in Problem Frames are claimed to be instantaneous; this proposition is obviously false if we insist on regarding events as real-world elements.

This analysis reveals that the preceding interpretation of the constructs of a language has the same problems as the Syntactic View for understanding the structure of scientific theories (Section 2).

4.2 Adopting a semantic view

A solution to the preceding problem is suggested from Philosophy of Science: the "Semantic View". Instead of regarding the propositions that characterise the language terms in texts as claims about something that exists in the world, we will consider them as referring to conceptual, idealised entities which satisfy the propositions by definition. This way all the propositions are true, although in a trivial way. Under this approach the language terms do not stand for already-existing elements but for concepts, or *genres* of idealised entities, such as "agent" in KAOS, "intentional actor" in i^*, and "event" in Problem Frames.

This proposal seems to fit well the view that authors themselves have of their languages. For example, the text that describes KAOS refers to the language concepts as "abstractions" [10, p. 46], and not as syntactic terms interpreted as real-world things. "Abstraction" is also the word employed by Jackson for referring to "phenomena" of Problem Frames [28, p. 35].

Users of a language employ its genres to build particular, conceptual entities.

It is customary to call such entities "models" if the language genres have an associated diagrammatic notation. For example the term "goal model" [29] is used to refer not to a box-and-arrow diagram, but to a conceptual entity built with the KAOS genres "goal" and "refinement".

Most statements in texts about languages depict a characterisation of the introduced terms. Our semantic view regards such statements as *definitions* of new genres and not as claims about the world. The collection of all features of the introduced genres matches what Harel and Rumpe call a semantic domain [24]. In fact, the language genres are the subject of semantic formalisations. Yet texts contain a small percentage of statements that do not define any genres, they are never formalised, but they do represent an important contribution to the language. We will analyse them in the next section.

Summing up our conclusions so far, languages intended for representation introduce a family of concepts. We propose the term "*constructed genres*" to refer to them. Our reasons are, on the one hand, that such concepts are genres or categories and not specific entities like, for example, a concrete, imaginary house. On the other hand, the "constructed" adjective explicitly reminds us that genres do not represent categories of real-world objects but they are classes of idealised entities whose common features have been *designed* by the authors of the language. We hope this denomination will help to avoid thinking that genres of languages "really describe" how the world actually is. A simple definition of "constructed genre" is:

Definition 4 (*constructed genre*): a constructed genre is a meta-level concept in a conceptualisation.

5 Representation-oriented Languages as Scientific Theories

This section and the next one analyse several case studies including both representation and design-oriented languages. The obtained results will provide evidence for Hypothesis 2 stated at the Introduction, which we reproduce here for convenience:

Hypothesis 2 *Languages are general representations of certain classes of systems in the world, and they can be characterised as scientific theories.*

In the previous section we rejected the syntactic view of languages. Although promising, understanding languages with the semantic view raises an important difficulty: if constructed genres do not refer to anything in the real world, how can we explain the fact that languages such as KAOS can be employed for representing the world?

According to the semantic view of theories, scientists represent the real world by building idealised entities called models, which are employed as conceptual "images" of reality.

Giere argues that models appear in Science with varying degrees of specificity [16]. Thus, an example of an abstract model is a conceptual entity characterised only by three general statements called "Newton's laws of motion". A more concrete model is a conceptual pendulum, which satisfies the mentioned laws together with others. A still more concrete model is obtained by providing values to the generic parameters of the conceptual pendulum.

As argued in the previous section, texts about languages describe a family of genres which one can employ for building particular, conceptual entities. This suggests to look for analogies between the way scientific models represent reality and the way constructed genres do it. The latter can be matched with an abstract, general model, while entities built from genres by users can be regarded as specific models. However, can we defend that the family of constructed genres of a language is a *general representation* of the world, and, in consequence, that the particular entities one can build constitute *specific representations* of reality?

A possible answer comes from Constructive Realism. According to Giere [16] models are not enough but representation also requires *intentionality*: somebody (e.g. a scientist) must explicitly *claim* that some conceptual entity represents the world. Although in principle any entity can be used to represent anything else, Giere states that none represents any other simply *by itself*.

According to Constructive Realism (see Section 2.3), propositions claiming that a certain (conceptual) entity represents some identified object or class of objects in the world are *theoretical hypotheses*: they are fallible, which can be empirically validated. As with theoretical models, hypotheses can have a variety of levels of specificity.

Therefore, though it is tempting to regard the constructed genres of a language as a general, theoretical model of some part of the real world, we must first be sure that explicit claims exist asserting such a relationship, i.e. to find theoretical hypotheses. An empirical way of achieving such a validation is to look for statements in texts about languages. This strategy is also the one Giere followed for scientific theories.

We find the following quotation in the text that we are using as a reference for the language of the KAOS approach [10, p. 46]:

> Some experience with real requirements documents has convinced the authors that higher-level abstractions such as "goal," "operationalization," "ensuring action," "agent," "responsibility," or "alternative assignment" are found informally and explicitly in the requirements of non-toy systems.

This paragraph is a proposition about the world: on the one hand, it contains an *identification* of a bounded part of reality — the requirements documents of non-toy software systems; on the other hand, the statement is a *claim* that the proposed constructed genres ("abstractions") can be found "informally and explicitly" in such a reality, i.e., the genres are a general representation of real requirements documents. Therefore the paragraph is a *theoretical hypothesis* about the requirements of software problems.

Regarding i^*, the quotation reproduced at page 18 (Section 4.1) identifies a part of the world, real organisations, and it points out several aspects of such a reality: the dependencies among stakeholders, their particular interests and concerns, as well as how such interests might be addressed. The paragraph explicitly claims that such aspects can be "described" [4] with the proposed models (SD and SR). This statement of intention is a general hypothesis about the identified aspects of real organisations.

The aim of the "Problem Frames" approach is to analyse and to structure software development problems. The quotation reproduced at page 18 (Section 4.1) identifies the reality of interest: those parts of the world where the problem and the solution machine are located. And it claims that such a reality is "shown" in context and problem diagrams. Diagrams are linguistic entities which symbolise or denote the conceptual entities that one builds from constructed genres, as we explained in Section 4.2. So stating that context and problem diagrams show the parts of the world where the problem and the machine are located is equivalent to claim that the corresponding constructed genres represent such a reality: a theoretical hypothesis.

These evidences support our previous conjecture that the constructed genres of representation-oriented languages play the same role as (general) theoretical models in the constructive realist view of scientific theories. Each family of genres is explicitly claimed to represent some identified part of the world.

[4] We prefer the term "to represent" over "to describe" to refer to the purpose of constructed genres with respect to the world. The latter verb is misleading as it may make us to think that genres are syntactic entities: terms or sentences.

Therefore we conclude that languages intended for representation can be regarded as *scientific theories* about some identified part of the real world. In fact, the existence of an explicit claim on the family of genres allows that such genres can be employed by the language users to build particular representations of concrete parts of the world.

According to this reasoning, representation in languages is based on the same principles as scientific theories: somebody builds a conceptual entity using the constructed genres, and then she formulates the explicit claim that the entity represents some identified part of the world.

We explained in Section 2.3 that, according to Constructive Realism, claims between theoretical models and the world can be understood in terms of similarity. Giere argues that two entities can be similar in a number of respects and to certain degrees of accuracy.

Giere's arguments in favour of similarity are equally valid in the case of the constructed genres of a language. For example, KAOS agents exhibit discrete behaviour but this does not happen with real humans. However while perceiving other humans we build categories of typical behaviours: to move an arm, to read, to stand up, to talk, and so on. Therefore KAOS agents and humans can be claimed to be similar in the respect "behaviour," although to a limited degree. In the same respect, agents and computers can be claimed to share a higher degree of similarity because the latter are designed to perform discrete operations (e.g. to copy the contents of a register into memory, to add two numbers, etc).

Our experience on this analysis shows that theoretical hypotheses contained in texts about languages are stated with few details. Therefore it is generally difficult to find respects and degrees in the statements about representation. The most specific paragraph we have found in the references about KAOS is the following one, which refers to the family of constructed genres [10, p. 6]:

> It is aimed at being sufficiently rich to allow both functional and non-functional requirements for any kind of composite system to be captured in a precise and natural way.

The paragraph identifies a part of the world, composite systems, and one respect of such systems: their requirements. Then it *claims* that such a respect can be "*captured*" (i.e. represented) with *precision*. Under our characterisation, this is a statement of the similarity between the family of constructed genres and composite systems in the respect "requirements" to the degree of accuracy "precise". We have not been able to find any additional details about such a representation in the reference texts.

The case studies of representation-oriented languages analysed in this section are evidences in favour of Hypothesis 2: languages can be regarded as scientific theories. But the case studies have also shown that texts defining languages do not clearly describe the representation relationship between constructed genres and reality, nor they provide any resources for users to build their own theoretical hypotheses. This situation justifies the fact that users focus on building entities (models) employing the language genres but they are unaware of the need to make statements which relate their models with the modelled reality.

6 Design-oriented Languages as Scientific Theories

This section continues the provision of evidence to support Hypothesis 2 with a focus on languages whose primary purpose is that of designing. Two cases are considered:

- High-level design, with the study of the Statemate [23] and Actors [2] modelling languages.
- Low-level design, with the study of a programming language, C++ [11].

6.1 Case study: Statemate and Actors languages

Texts defining Statemate and Actors explicitly *identify* two classes of computer systems: reactive systems and open distributed systems, respectively. The latter is a subset of the former. Note that the existence of such systems is independent of the existence of the languages. Indeed, texts usually characterise the corresponding class at the outset, before presenting the language constructs (see for example [23, pp. 3–4] in the case of Statemate and [4, p. 2] and [5, p. 155] in the case of Actors).

The first question is whether the syntactic view is appropriate for understanding these languages. Texts about Statemate introduce a number of terms (e.g. "activity") and they include many predicates about such terms. For example: "the system is viewed as a collection of functional components or activities [...] organized into a hierarchy" [23, p. 20]. If the introduced terms referred to real-world entities, then the predicates would be true in the real world.

Currently there exist several repositories of source code that are open to public access, such as SourceForge.net and Google Code. Assume that we regard source code as a description of the functionality and behaviour respects of computer systems. Searching on the repositories for systems which can be classi-

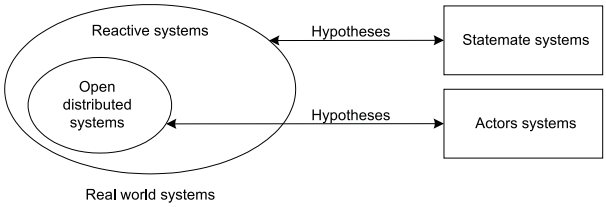

Fig. 2. Theories of reactive systems and open distributed systems.

fied as "reactive," we find that the functionality of such systems is spread over Java objects, C functions, Perl subroutines, etc. But none of these elements has exactly the same features as Statemate activities, nor the functionality of the software hosted in the repositories is exactly organised into a hierarchy. In the case of Actors, texts ascribe a number of ideal properties to terms [4] which obviously are not present in the real world. Therefore, the syntactic interpretation of terms is not correct.

We propose to understand terms as standing for constructed genres, as in the case of languages intended for representation. In consequence, entities that Statemate and Actors users build with such genres are conceptual too. Such entities are the ones software engineers call "models," and they are often symbolised with a notation that mixes diagrams and text. Summing up, the propositions contained in texts about reactive systems and open distributed systems do not literally refer to the world but they *construct* two classes of conceptual entities with their own special features.

This semantic view fits well the conception that software engineers themselves have of the "models" they design. The genres of a "modelling language" such as Statemate and Actors are constructed to be more expressive than those of a "programming language" and in consequence, designing an (expressive) entity with the former requires less effort and the entity will be easier to understand and to analyse. For these reasons designing models is a preceding task to designing programs in software development approaches.

While most part of the texts about Statemate and Actors are concerned with defining properties of the genres, we can also find some statements which relate the constructed classes of conceptual entities with the real-world ones. Therefore such statements can be understood as *hypotheses* about the classes of reactive systems and open distributed systems. Next we present several examples:

- Statemate activities are claimed to *represent* functional components such as objects, processes, and functions [23, p. 20]. Such entities are the ones of typical programming languages.
- Statemate defines "flow-lines" as communication channels between activities. Flow-lines are claimed to *represent* a variety of means for information

transfer, such as parameter passing to procedures or global variables in programs and messages transferred along transmission lines in distributed systems [23, p. 28].

- Statemate contains a sub-language named "Statecharts" for characterising the behaviour of reactive activities. The language defines two kinds of reactivity in relation to time, which are called "models of timing" or "time schemes": synchronous and asynchronous [22, p. 316]. The former is claimed to *fit* systems that are highly synchronous and the latter is claimed to *fit* most kinds of asynchronous systems [22, p. 317].
- The "actor model of computation" is claimed to be "a *natural* model to use as a *theory* of open distributed computation" [4, p. 2]. In another cite, the actor model is claimed to provide "an effective method for *representing* computation in real-world systems" [5, p. 157]. Agha et al. explicitly state that the "actor theory [...] *abstracts* some fundamental aspects of open systems" [5, p. 155]. They also qualify the representation relationship as a "*reasonable abstraction* for open distributed systems" and as a "*realistic model* for a number of practical implementations" [5, p. 155].
- The Actors language defines an "actor configuration". Such a concept is claimed to be a *model* of an open system component [4, p. 3]. An actor is claimed to "provide a natural *generalization* for objects" [5, p. 155].

According to these evidences and the constructive realist characterisation of scientific theories, we can regard Statemate and Actors as two *theories* of reactive systems and open distributed systems, respectively. The theoretical models are the conceptual classes defined by the languages, and the similarity hypotheses are (more or less explicitly) included in the texts. Figure 2 shows the representation relationship in a diagram. Each model and the associated hypotheses constitute a theory. According to their scope, the Statemate theory is more general than the Actors one, which is concerned with a specific kind of reactive systems.

6.2 Case study: C++

Texts which describe the C++ language introduce a number of terms such as "object," "member function," and "pointer". The properties associated to them are so "low level" that we might be tempted of thinking of such terms as fancy names for hardware elements in real computers. For example, the Reference Manual (RM) states that "an object is a region of storage" [11, p. 13]. Indeed, it might seem that this syntactic view fitted well for the common use of the language; for example, C++ programmers use pointers to access the "real" memory addresses of the computer. [5] However one can also find

[5] Strictly speaking, what pointers can address is not the real memory but the operating system's own view of memory, which is a model of the physical memory

properties which do not literally apply to real computers. For example, a C++ object must be associated with a class and this association determines the functions which can legally be applied to the region of memory. Obviously this statement is not true with respect to the real hardware.

As in previous cases, we adopt a constructive realist perspective. We regard C++'s terms such as "object," "member function," and "pointer" as constructed genres which satisfy their stated properties by definition. They can be employed as pieces for building conceptual entities called programs, which have the property of being executable. Programs are described in linguistic entities named "source code" using a (text-based) syntax.

If terms in C++ texts denoted constructed genres, there should be statements (hypotheses) claiming the similarity between the language genres and the real world. The RM contains only one statement of this kind: "C++ is a general purpose programming language" [11, p. 1]. This statement indirectly establishes a relationship between the class of entities which can be potentially constructed with C++ and the class of real-world computers: the respect is "to be Turing complete" and the similarity degree is "exact". The RM statement represents the most explicit claim that current texts about languages include about this respect because, as Mitchell indicates [33, p. 21]: "today it is unlikely that a team of programming language designers would advertise that their language is sufficient to define all partial recursive functions. Most computer scientists nowadays [...] assume that most languages intended for general programming are Turing complete".

Additional hypotheses are found in texts explaining the design of the language itself. For example, Stroustrup [42, p. 4] claims that the language operations "directly reflect machine instructions". Therefore they can be regarded as a *representation* of the instruction set of a microprocessor. The degree of accuracy of such a representation is claimed to be "direct". This theoretical hypothesis cannot be stated about every programming language (at least not if we intend to say something that is true). For example, Perl provides the $=\sim$ operator for checking whether a regular expression matches a given string. This operator does not *directly* represent any machine instruction of current microprocessors, so there is no similarity between Perl's genres and real microprocessors with such a degree in that respect. As another example, "message expressions" in Smalltalk [19, p. 25], which allow to send a message to an object, cannot be claimed to represent any machine instruction with degree "direct".

Technical texts about microprocessors (including datasheets and manuals) provide a closer view of the world than C++'s. Several models can be found.

in turn — it typically includes the so-called "virtual addresses," constituting a space wider than the actual RAM size.

For example, the *Intel 386SX Manual* [25] offers a detailed view of the microprocessor structure and behaviour. A higher-level model of the Intel 32-bit processors is described at the *Intel Architecture Software Developer's Manual* [26], the so-called "processor's system-level architecture," mainly intended for developers of operating systems.

6.3 Knowledge in design-oriented languages

There are many factors driving the design of a design-oriented language. For example, in his book about the design of C++, Stroustrup [41] surveys the catalogue of 26 rules he followed. Design factors condition the constructed genres and their properties, which in turn determine the kind of representation the language is of the class of systems it is intended for.

We can find a nice parallelism with the scientific practice. According to Giere [18, p. 747], scientists use models to represent aspects of the world for various purposes. What the designer of a language develops is a representation of a class of systems in the world for the purpose of obtaining a design tool for such a class of systems. The specific design factors count as specialised purposes.

A design-oriented language contains design knowledge: sound principles for the target class of systems are hopefully embedded in the language constructs (see [41, p. 114] for the case of C++). Knowledge can also be found when one regards the language as a characterisation of a class of systems: the necessary theoretical hypotheses detail the scope and distance of such a characterisation. For example, both Statemate and Actors assume unbounded buffers in communications: this is an embedded design decision intended to guarantee delivery in reactive systems. A hypothesis is then necessary to claim some distance with reality in the respect "buffering capabilities," i.e. to express *something* about real systems. Referring to Actors systems and real systems in this respect, Agha [3, p. 8] asserts that "the reality is that we only have asynchronous agents with no buffering capabilities". While not explicitly, the author is establishing a wide distance between Actors systems and real systems in the cited respect.

7 Conclusions and discussion

This paper has provided evidence in favour of the claim that languages can be regarded as scientific theories. This perspective offers a number of advantages:

(1) It makes explicit that languages are *knowledge* about a class of systems.

In particular, the constructive realist characterisation of scientific theories introduces an explicit indication of the accuracy of such knowledge.

(2) Scientific theories can be empirically *validated*. Anyone can take real systems of the corresponding class and check whether the hypotheses of a language hold. For example, we can take the requirements of any non-toy system and check whether we can find elements such as goals, operationalisations, and ensuring actions, as held by the KAOS hypothesis stated at page 22. As another example, we can take any open distributed system and check whether its topology can be dynamically reconfigured, as held by the Actors hypothesis stated at [5, p. 157].

(3) Scientific theories targeted at the same class of systems can be *compared*. A particular consequence of this fact is that the most adequate theory can be chosen in each case depending on our concrete needs.

Theory comparison requires a previous identification of the class of systems which a language is intended to represent and a rigorous formulation of the theoretical hypotheses. Regrettably, our case studies show that theoretical hypotheses in languages are usually either unstated or fairly loosely indicated. When hypotheses are clearly stated, one can reason about which theory fits better her or his needs. Consider the following two hypotheses:

- Regarding the respect "duration of a reaction to an external stimulus," Statemate systems are different from real-world reactive systems.
- Regarding the respect "duration of a reaction to an external stimulus," UML systems are equal to real-world reactive systems.

Studying the theoretical models, Statemate systems react instantaneously while UML systems take some time [47]. If we consider this respect only and our purpose is the ease of design, then the Statemate theory is a better choice because it is simpler. But the hypothesis tell us in advance that the obtained Statemate system will be far from the final system in the respect under consideration.

The view of languages as scientific theories uncovers an important lack in current representation-oriented languages: they do not provide any resources for building theoretical hypotheses. This is a major problem because building models or conceptualisations is not enough to represent anything: the person who receives a model or a conceptualisation *ignores* what respects of reality were intended to be represented as well as the degree of accuracy of the representation. It is the existence of some explicitly specified similarities that makes possible the use of the model to represent the real system [18]. This paper has established a rigorous formulation of theoretical hypotheses and degrees of accuracy. The general schema for a similarity hypothesis, $sim(m, s, r) = d$, must be particularised for the specifics of each real-world domain that is to be represented. On the one hand, respects must be selected in the domain

according to the modeller's interests and purposes; on the other, catalogues of degrees of accuracy must be defined — probably by consensus through a social process in much the same way as an agreement must be reached when an ontology is been defined. We are currently working on an extension to OWL that allows an individual to complement her ontologies with similarity hypotheses. In particular, the supplementary language will permit to identify real-world systems (or classes of systems) of interest, [6] list respects, define catalogues of degrees of accuracy, and link a conceptualisation with a real-world system in a similarity hypothesis.

Regarding languages as scientific theories has an important consequence for designing. Language hypotheses claim one possible *characterisation* of an identified class of systems in the world to some degree of accuracy. Language genres have been constructed according to a number of purposes, among which there are design principles about the target class of systems. If we employ the language for building a particular design, the theoretical hypotheses guarantee that our design will be similar to the final system, at least in some general respects and to certain degrees of accuracy. For example, there is a hypothesis which asserts the full similarity between Actors systems and real open distributed systems in the property "dynamically reconfigurable topology" [5, p. 157]. Therefore, any design we develop with the Actors genres will have the capability of being dynamically reconfigurable built into.

Finally, abstraction can be explained in terms of representation. What we read in the manual of a microprocessor is a description of a theoretical model of a physical device. Similarity has been shown to be an adequate characterisation of the representational relationship. The model of manuals has such a high degree of accuracy with the electronic devices that we take it as our ground representation. The theoretical model of C++ has a lower similarity, but it is enough for programmers to build efficient systems. The theoretical model of Statemate is even less similar than C++'s, but simplicity is exactly what developers need when they are designing at early stages.

8 Related works

Knowledge representation is an important topic in Artificial Intelligence and it has long been studied. One of the pioneer approaches was the use of conceptual graphs [38]: systems of concepts and relations intended to represent what is known, thought of, or believed. The relationship between a conceptual graph and reality was not regarded as important since it was simply assumed

[6] It is noteworthy that this will be merely a syntactic identification, such as `Realworld train09615`, since it is impossible to give a formal semantics to reality.

to be one of *truth*. For example, Sowa stated that schemata (a kind of conceptual graphs) "are commonly true, but they may sometimes be wrong. To handle those cases where a schema does not apply, a law must be asserted that blocks the default" [38, p. 140]. Therefore we can observe an underlying logical empiricist basis: when conceptual graphs are asserted (as propositions), they are interpreted as *direct claims* about the world. However the assumed relationship of truth soon revealed some problems. To solve them, conceptual graphs were required to be *asserted* into some *context*, where a context is a linguistic entity describing some physical or imaginary situation [39]. However the problems [43] of Logical Empiricism persist since the relationship between contexts and the world is regarded to be one of truth.

In spite of these difficulties, contexts have been revealed to be useful for other purposes such as information packaging (e.g. in modelling information systems [13]). In the field of corporate knowledge management, context modelling is regarded a value: the context in which a piece of information is located can be very useful to determine the *relevance* of the piece of information in a given, new application context [1, p. 266]. In our approach, the context is simply part of the reality under study, and the "relevance" is related to the expected degree of accuracy. If model m is proposed for a real-world system of interest s (e.g. a supplier) located at a real-world context c (e.g. a price negotiation), then some respect r is under consideration (e.g. the supplier's willingness to negotiate) and some similarity is asserted: $sim(m, s \cup c, r) = d$, where $s \cup c$ denotes the real-world composite system of s and c. If the context changes to c' then we are identifying another part of the world (namely $s \cup c'$) and hence the previous hypothesis may not hold for such a part of the world.

The notion of truth, which is assumed by the previously cited references to link descriptions with the world, has been studied in classical logic from a formal conception. In particular, Tarski's view [44,45] is based on the idea that the valuation of a formal sentence must be done via a "new" entity called "*model*" which is located outside the formal (purely syntactical) language. However, nothing more is said about the properties we can require for such new entities. In fact, a model can be regarded as a mental construction or as the result of describing a certain reality. In the second case, the relationship between the model and the physical world is assumed to be one of "truth," thus in accordance with Logical Empiricism. In Constructive Realism, by contrast, the relationship between the model and the world is a main concern. Such a relationship is not one of truth but one of similarity.

Interestingly, logical empiricist postulates seem to have spread to the ontology community as well. Burton-Jones et al. [7] develop a number of metrics for assessing the quality of ontologies. Regarding the so-called "pragmatic quality," one of the proposed metrics addresses the information *accuracy*: the relation between the number of *false* claims an ontology makes and the total number

of statements in the ontology. Therefore the authors regard the relationship between an ontology and the world as one of truth/falsity. This conception has the same problems as Suppe [43] identified for the logical empiricist view of scientific theories.

The need for theoretical hypotheses for representation in software development has not been admitted so far, and very few accounts characterise the link between models and reality. A major exception is [27]; here Jackson employs the term "model" to refer to the structures often built in software systems (usually as databases) that are interpreted as information about something in the real world (for example, about the employees of a company). In [28] Jackson studies the relationship between such software structures and the real world, identifying some "model concerns" [28, pp. 202–206]: model imperfection, incompleteness, and time lag. The first two of these can be embedded in the general similarity relationship introduced in this paper. The last one refers to the time lag existing between the occurrence of an event in the real world and the appearance of its counterpart in the software model.

Marcos and Marcos [32] analyse two types of entities commonly employed in the database field: data models and conceptual schemata. They conclude that they both serve two purposes: on the one hand, to be the basis for a design ("model-as-original"), and, on the other hand, to be a representation of the real world ("model-as-copy"). These two purposes take place at different levels: more generic (in the case of data models) and more specific (in the case of schemas).

Ludewig [30, p. 6] cites Stachowiak's three criteria for determining if an entity is a model [40]:

- Mapping criterion: there is an original object or phenomenon that is mapped to the model. In the sequel, this original object or phenomenon is referred to as "the original".
- Reduction criterion: not all the properties of the original are mapped on to the model, but the model is somehow reduced. On the other hand, the model must mirror at least some properties of the original.
- Pragmatic criterion: the model can replace the original for some purpose, i.e. the model is useful.

The "mapping criterion" is simply the identification phenomenon referred to by Constructive Realism. The "reduction criterion" is covered by the respects indicated in the similarity hypotheses. The "pragmatic criterion" agrees with the existence of a purpose, as pointed out by Giere [18] and by Morgan and Morrison [34].

Ludewig [30, p. 8] defends that theories *are* models. However we have argued that theories *contain* models — together with theoretical hypotheses.

Seidewitz [37] focuses on entities that one can build with the so-called "modelling languages". He defines a model as "a set of statements about some system under study" [37, p. 27] so he seems to obviate non-linguistic matters. He also defines "correctness" as: "we consider the model correct if all its statements are true for the system under study" [37, p. 27]. Therefore he does not distinguish the statements that define the model from the statements that claim the relationship with the real world.

Regarding studies of languages, Harel and Rumpe [24] review the traditional characterisation consisting of three elements: a syntax, a semantic domain, and a mapping between both. The authors analyse each element with examples of modelling and non-modelling languages. In particular they express the following about the semantic domain [24, p. 67]:

> The semantic domain is not to be taken lightly: It specifies the very concepts that exist in the universe of discourse. As such, it serves as an abstraction of reality, capturing decisions about the kinds of things the language should express.

Therefore the authors seem to admit that the concepts of the semantic domain are different from reality, thus agreeing with the model-based view of theories (Section 2.2) and with the notion of "constructed genres".

Guarino [21] elaborates on the formal notion of conceptualisation first introduced by Genesereth and Nilsson in 1987 [14, ch. 2]. For the author, a conceptualisation establishes a correspondence between possible situations in the world (called "states of affairs" or "possible worlds") and their characterisations in terms of relevant relations, which are given a formal semantics. Moreover, Guarino reflects on the linguistic nature of ontologies and the impossibility, in the general case, that an ontology can completely specify a conceptualisation. According to the author, such coarse-grained, approximate ontologies are not only unavoidable but they reveal very useful for many practical purposes (e.g. they may increase the quality of the analysis process in the development of an information system).

Brewster and O'Hara [6] head an interesting collection of papers from diverse authors who discuss several controversies related with knowledge representation with ontologies, focused on the actual range of knowledge an ontology can successfully represent.

There is an extensive bibliography about similarity in the fields of databases and Case-Based Reasoning (CBR), where many characterisations of the concept and many measures have been proposed across a wide variety of domains. Richter [36] offers a comprehensive overview of different characterisations of the similarity concept, some of which have been adopted in this paper and extended with respects.

The ideas presented in this paper have their roots in the seminal work presented in a PhD thesis [8] and in our earlier study on the role of ontologies in software engineering [9].

Acknowledgements

The authors want to explicitly express their gratitude to the three anonymous reviewers of this paper. Their thorough reading and useful comments have greatly contributed to improve the clarity and quality of the work.

This work has been partially funded by the Spanish Ministry of Education and Science through project HUM2007-66607-C04-04.

References

[1] A. Abecker, A. Bernardi, K. Hinkelmann, O. Kühn, M. Sintek, Context-aware, proactive delivery of task-specific information: The KnowMore project, Information Systems Frontiers 2 (3-4) (2000) 253–276.

[2] G. Agha, Actors: A Model of Concurrent Computation in Distributed Systems, MIT Press, Cambridge, Mass., 1986.

[3] G. Agha, Supporting multiparadigm programming on actor architectures, in: Proceedings of the Parallel Architectures and Languages Europe (PARLE'89), Volume II: Parallel Languages, Springer-Verlag, 1989.

[4] G. Agha, I. A. Mason, S. F. Smith, C. L. Talcott, A foundation for actor computation, Journal of Functional Programming 7 (1) (1997) 1–72.

[5] G. Agha, P. Thati, R. Ziaei, Actors: a model for reasoning about open distributed systems, in: H. Bowman, J. Derrick (eds.), Formal methods for distributed processing: a survey of object-oriented approaches, Cambridge University Press, 2001.

[6] C. Brewster, K. O'Hara, Knowledge representation with ontologies: The present and the future, IEEE Intelligent Systems 19 (1) (2004) 72–81.

[7] A. Burton-Jones, V. C. Storey, V. Sugumaran, P. Ahluwalia, A semiotic metrics suite for assessing the quality of ontologies, Data and Knowledge Engineering 55 (1) (2005) 84–102.

[8] J. M. Cañete-Valdeón, A theory of languages and design methods in software engineering, Ph.D. thesis, Universidad de Sevilla (2006).

[9] J. M. Cañete-Valdeón and F. J. Galán, Towards a theory on the role of ontologies in Software Engineering problem solving. Conclusions from a theoretical model of methodological works, in: M. Cerioli (ed.), Fundamental Aspects on Software Engineering (FASE/ETAPS 2005), vol. 3442 of Lecture Notes in Computer Science, Springer-Verlag, 2005.

[10] A. Dardenne, A. van Lamsweerde, S. Fickas, Goal-directed requirements acquisition, Science of Computer Programming 20 (1993) 3–50.

[11] M. A. Ellis, B. Stroustrup, The Annotated C++ Reference Manual, Addison-Wesley, 1991.

[12] Federal Aviation Regulation, Appendix G to Part 91 - Operations in Reduced Vertical Separation Minimum (RVSM) Airspace, Tech. rep., Federal Aviation Administration.

[13] B. Garner, R. Raban, Context management in modeling information systems (IS), Information and Software Technology 41 (1999) 957–961.

[14] M. R. Genesereth, N. J. Nilsson, Logical Foundations of Artificial Intelligence, Morgan Kaufmann Publishers Inc., 1987.

[15] R. N. Giere, Constructive realism, in: P. Churchland, C. Hooker (eds.), Images of Science, University of Chicago Press, 1985.

[16] R. N. Giere, Explaining Science: A Cognitive Approach, University of Chicago Press, 1988.

[17] R. N. Giere, Science without Laws, University of Chicago Press, 1999.

[18] R. N. Giere, How models are used to represent reality, Philosophy of Science 71 (2004) 742–752.

[19] A. Goldberg, D. Robson, Smalltalk-80 : the language, Addison-Wesley, 1989.

[20] T. R. Gruber, A translation approach to portable ontology specifications, Knowledge Acquisition 5 (2) (1993) 199–220.

[21] N. Guarino, Formal ontology and information systems, in: N. Guarino (ed.), Proceedings of the 1st International Conference on Formal Ontologies in Information Systems (FOIS'98), IOS Press, 1998.

[22] D. Harel, A. Naamad, The STATEMATE semantics of Statecharts, ACM Transactions on Software Engineering and Methodology 5 (1996) 293–333.

[23] D. Harel, M. Politi, Modeling Reactive Systems with Statecharts: the Statemate Approach, McGraw-Hill, 1998.

[24] D. Harel, B. Rumpe, Meaningful modeling: What's the semantics of "semantics"?, IEEE Computer 37 (10) (2004) 64–72.

[25] Intel 386 SX Microprocessor, Tech. rep., Intel (Jan. 1994).

[26] Intel architecture software developer's manual – Volume 3: System programming, Tech. rep., Intel (1999).

[27] M. Jackson, Software Requirements & Specifications: a Lexicon of Practice, Principles, and Prejudices, Addison-Wesley, 1995.

[28] M. Jackson, Problem Frames. Analyzing and Structuring Software Development Problems, Addison-Wesley, ACM Press, 2001.

[29] E. Letier, Reasoning about agents in goal-oriented requirements engineering, Ph.D. thesis, Faculté des Sciences Appliquées, Université catholique de Louvain (May 2001).

[30] J. Ludewig, Models in software engineering –an introduction, Software and Systems Modelling 2 (2003) 5–14.

[31] A. Majchrzak, Q. Wang, Breaking the functional mind-set in process organizations, Harvard Business Review (1996) 93–99.

[32] E. Marcos, A. Marcos, A philosophical approach to the concept of data model: Is a data model, in fact, a model?, Information Systems Frontiers 3 (2) (2001) 267–274.

[33] J. C. Mitchell, Concepts in Programming Languages, Cambridge University Press, 2001.

[34] M. Morgan, M. Morrison, Models as Mediators: Perspectives on Natural and Social Science, Ideas in Context, Cambridge University Press, 1999.

[35] M. A. Musen, Ontologies: Necessary – indeed essential – but not sufficient, IEEE Intelligent Systems 19 (1) (2004) 77–78.

[36] M. M. Richter, Similarity, in: P. Perner (ed.), Case-Based Reasoning on Images and Signals (Studies in Computational Intelligence), Springer-Verlag Berlin Heidelberg, 2008.

[37] E. Seidewitz, What models mean, IEEE Software 20 (5) (2003) 26–32.

[38] J. F. Sowa, Conceptual Structures, Addison-Wesley, 1984.

[39] J. F. Sowa, Peircean foundations for a theory of context, in: Conceptual Structures: Fulfilling Peirce's Dream, Springer Berlin / Heidelberg, 1997.

[40] H. Stachowiak, Allgemeine Modelltheorie, Springer-Verlag, 1973.

[41] B. Stroustrup, The Design and Evolution of C++, Addison-Wesley, 1994.

[42] B. Stroustrup, Abstraction and the C++ machine model, in: International Conference on Embedded Software and Systems (ICESS'04), vol. 3605 of Lecture Notes in Computer Science, Springer-Verlag, 2005.

[43] F. Suppe, The Structure of Scientific Theories, Urbana: University of Illinois Press, 1974.

[44] A. Tarski, The semantic conception of truth and the foundations of semantics, Philosophy and Phenomenological Research 4 (3) (1944) 341–376.

[45] A. Tarski, The concept of truth in formalized languages, in: Logic, Semantics, Metamathematics, Clarendon Press, 1956.

[46] OWL Web Ontology Language guide, Tech. rep., World Wide Web Consortium (W3C) (Feb. 2004).
URL `http://www.w3.org/TR/2004/REC-owl-guide-20040210`

[47] R. J. Wieringa, Design Methods for Reactive Systems: Yourdon, Statemate and the UML, Morgan Kaufmann, 2003.

[48] E. Yu, Towards modelling and reasoning support for early-phase requirements engineering, in: Proceedings of the 3rd IEEE Int. Symp. on Requirements Engineering (RE'97), 1997.

[49] E. Yu, Strategic modelling for enterprise integration, in: Proceedings of the 14th World Congress of International Federation of Automatic Control (IFAC'99), Permagon, Elsevier Science, 1999.