Concepts and Object-Oriented

Knowledge Representation

Juha Petteri Pesonen

Contents

1. INTRODUCTION

> What, then, is time? I know well enough what it is, provided that nobody asks me; but if I am asked what it is and try to explain, I am baffled.
>
> -- Augustine; Confessions 11.14.17

The computational view of the mind rests on the assumption that psychological theories and laws can eventually be implemented in a computer - in a Universal Turing Machine. It is claimed that the mind consists of symbols with semantics and processes that manipulate those symbols; thought is the manipulation of symbols. By combining, moving and deleting symbols the mind can create new symbols with new meanings. (Pinker 1994/1997; Clark 1996.)

The symbol manipulation view of the mind arose in the fifties. It gave birth and justification to Artificial Intelligence (AI) and seemed to solve certain philosophical problems. Symbols have meaning, but since they are also something physical (ink on the paper, sound waves, neurons...) they can have causal powers at the same time, which is a possible explanation of how something mental, such as the mind, could have causal powers over physical entities. The idea that thought is computation also provided much more explanatory power than previous associative theories. So, if the mind was nothing but a symbol manipulator, then there should be no reason why a computer, a symbol cruncher par excellence, could not possess intelligence; hence the promise of AI.

The view that the mind is computational raises many questions. What kind of algorithms does it use? And how many? How is knowledge encoded and what kind of inference mechanisms are available? Humans, by all standards, possess a remarkable amount of conceptual knowledge touching a broad range of topics from mundane facts to profound theories: we have knowledge concerning our everyday life, how to get from one place to another, latest trends in fashion, bus time tables, what to do in case of illness; personal facts such as our names, kinship relations, phone numbers, education; we also know a vast

amount of information concerning topics such as history, biology, politics, literature, etc. How is this massive amount of conceptual knowledge represented? How is all the information organized? How can humans use it so efficiently and with such ease?

Knowledge representation plays a vital role in AI, because how information is encoded determines what kind of reasoning can be done with the knowledge base, how complex it is and how much storage capacity is needed for the information. But what kind of a knowledge representation system does a human cognizer use? In order to have psychological plausibility a knowledge representation tool should use the same representations as humans and enable the same inferences as humans are capable of conducting, but at the same time it should also disable all the inferences that we cannot make.

This study will examine current major theories of concepts put forward by cognitive psychologist and philosophers and see how naturally they could be simulated using the Object-Oriented (OO) knowledge representation. The goal of is to evaluate the psychological credibility of the OO approach to representing concepts. The reason for choosing the OO knowledge representation is that recently a new type of OO languages has emerged. These object-based languages offer possibilities lacking in previous languages. This study will show that these languages can overcome some problems associated with traditional OO languages and also with frames. This study will also point out two grave weaknesses in the entire OO paradigm.

How humans actually think and model the world is relevant to the design of programming languages and has always played a role in the development of new languages (Partridge 1996, 61). All programming languages, like natural languages, can be seen as models of the world. Languages in turn reflect our mental models of the world. Analogously a programming language creates a certain model of the world. Ideally, these models should converge. A language that would be more 'human-like' and conceptually intuitive would presumably be easier to learn, use and understand than, say, a machine language; programs created with this language would be similar to the concepts that we actually employ.

The following chapter will review the current major theories of concepts. The third chapter is devoted to knowledge representation and especially to the OO one and the two

paradigms in it. Some other tools are briefly considered also for the sake of comparison. In the fourth chapter I will provide implementations for the reviewed theories of concepts with both types of OO languages. The final chapter provides a short review and a summary of the findings.

On the terminology and the notations. No distinction is made between categories and concept, besides a brief note below. The terms "properties", "features" and "attributes" will also be used interchangeably. Names of concepts are written in uppercase. When referring to words double quotation marks are used. Thus, the word "dog" gets its meaning from the concept/category DOG that can have e.g. the properties/features/attributes barking, four-legged and animal; "dog" and DOG both refer to dogs. Italics is reserved for emphasis and markup.

2. THEORIES OF CONCEPTS

2.1 *What are concepts?*

"Concepts seem to be the very stuff of which cognitions are made" (Rey 1983, 237). Why are certain objects grouped together? What makes cats fall under the concept CAT and not under the concept DOG? What is it that unites members of a certain category? Categorization is one of the most basic cognitive skills in humans. It is so automatic and so deeply rooted that we might not even notice it until explicitly confronted with it. Every time we see, hear, feel or smell something as a *kind* of something we are categorizing and using concepts. Recognizing words as verbs or substantives, or seeing chairs or trees or clowns is categorizing; instead of seeing a particular or an individual we also see a category; it is the category that glues the individuals together into one unit. In performing any actions such as writing, tying shoes, trading stocks or giving a speech we use hundreds of categories. All of these actions differ in some respect - they are never identical - yet they are still actions of some kind and we know how to make actions of that kind. Categorization becomes more difficult in, for example, scientific theories where we don't always know to which category certain objects belong and when the boundary of two categories is fuzzy. Is a whale a kind of a fish or a mammal? What is the difference between an angry and a frustrated person?

A theory of concepts tries to explain what is it that unites and binds a group of individuals or events together. Sometimes concepts can appear to be very simple at least on the surface. Concepts such as CHAIR, DOG and CAT seem to be self-evident. Everybody knows what they are. On the other end are concepts, such as TIME, DEMOCRACY and LOVE, that everybody knows that they don't really know what they are. On top of this, there are categories that can be very exotic and perplexing (at least for the western mind). The ancient Chinese classification below divides the animal kingdom in thirteen categories:

> On those remote pages it is written that animals are divided into (a) those that belong to the Emperor, (b) embalmed ones, (c) those that are trained, (d) suckling pigs, (e) mermaids, (f) fabulous ones, (g) stray dogs, (h) those that are

included in this classification, (i) those that tremble as if they were mad, (j) innumerable ones, (k) those drawn with a very fine camel's hair brush, (l) others, (m) those that have just broken a flower vase, (n) those that resemble flies from a distance. (Lakoff 1987a, 92).

Australian aboriginal language Dyirbal has a category called "balan", which includes such things as women, fire and dangerous things. It also includes birds that are not dangerous and some exotic animals like platypus, bandicoot and echidna (Lakoff 1987a, 5). At first look, it seems that things such as children, jewelry, portable TVs, paintings, manuscripts and photograph albums have nothing in common. But if one thinks about THINGS TO TAKE OUT OF ONE'S HOME DURING A FIRE, there is, after all, something that connects the items (Murphy & Medin 1985, 303). All in all, it seems that categorization is not always such a simple matter. So, a theory of concepts must explain what *exactly* is it that language learners learn when they learn to use words such as "chair", "time" or "balan".

Categorization is a form of abstraction. It refers to the psychological capability to store sufficient amount of information - not every tiny little detail - from given objects or events in order to recognize similar future objects or events. Concepts do this work; they allow us to store useful and pertinent conceptual information that applies to a set of individuals and use this knowledge in the future. Without categories and the conceptual knowledge associated with them, we would have to learn all the information of all the objects we ever confront separately, but storing all the information of all the objects encountered is impossible for a finite thing such as a brain or a computer. Therefore, by grouping objects together into categories and storing information into them dramatically reduces the amount of storage space needed. Having learned that tigers are dangerous, we can do our best to escape a new one, because we know using our concept that it is dangerous. Without the concept we would have to learn for each and every tiger separately that it is dangerous.

> An intelligent being cannot treat every object it sees as a unique entity unlike anything else in the universe. It has to put objects in categories so that it may apply its hard-won knowledge about similar objects, encountered in the past, to the object at hand. (Pinker 1997, 12).

All natural languages reflect this cognitive ability. Languages are full of terms that refer to categories. The words "dog", "chair", "spoon", etc. do not refer to a certain individual, as

the words "Michael Jackson" and "Bill Clinton" do. Instead, they name categories that consist of possibly an infinite number of individuals. Without categories our languages would explode with words, because we would have to invent a different word for every single entity in the world. Concepts are what gives meaning to words. Words name concepts and refer to objects in the world *via* concepts (Fodor 1998, 9). Conceptual thinking is not tied to language only, but found also in prelinguistic thinking. Animals and children younger than twelve months have been shown to be able to perform categorization and patients who have lost their linguistic capacities due to aphasia have not lost their ability to think in concepts (Saariluoma 1990, 71).

Concepts are perhaps the most fundamental and central units of cognition. There are other types of representations, such as images and music, but the Representational Theory of the Mind (RTM) assumes that at least propositional mental states, such as beliefs, desires and knowing, have semantic content that relates them to things in the world and that they can be used in psychological explanations (Fodor 1998, 6-8). These mental states are represented using concepts.

> One of the most important properties of concepts is that they are semantically evaluable. A thought may be true or false, depending on how things are with that portion of the world which the thought is about. (Laurence & Margolis 1999, 13).

For example, the belief that CATS ARE ANIMALS is composed of the concepts/representations CATS, ARE and ANIMALS and inherits its meaning from only them; in other words, the semantic content of mental states derives *only* from the constituents of that particular state.

> RTM says that there is no believing-that-P episode without a corresponding tokening-of-a-mental-representation episode, and it contemplates no locus of original intentionality except the contents of mental representations. (Fodor 1998, 8).

Concepts are hence a type of mental representation with semantic properties. The mental state CATS ARE ANIMALS may be true or false, whereas other types of mental representations, such as images, do not convey this kind of semantic information. An image of a man walking up the hill may equally well be an image of a man walking down

the hill backwards. Images thus lack the sort of semantic values that concepts have (Fodor 1975, 180-181).

Concepts can be primitive or complex. Complex concepts contain other concepts as their parts. The parts can again be complex concepts, but in order to avoid infinite regress at some stage a concept can no longer have other concepts as its part. At this stage, the concept is a primitive concept. If a concept is composed of other concepts it is complex, otherwise primitive. Lexical concepts are concepts denoted by a single morpheme in a language, whereas phrasal concepts are denoted by several morphemes. That is, lexical concepts correspond to words and phrasal concepts to sentences. Lexical concepts can be either primitive or complex. It does not follow from the fact that the concept CAT has been lexicalized in most languages and is therefore a primitive part of the language, that the concept itself is primitive. Which concepts get lexicalized in a language depends greatly on what is important in the culture of a language community. It could be that the concept ALL BLACK CATS LIVING IN SOUTH FRANCE is lexicalized in some language with the term "flurg". In this case, "flurg" would be a primitive part of the language, a lexical concept, but FLURG would not be primitive. (Laurence & Margolis 1999, 4).

Concepts can have structure in two ways. The Containment Model claims that structured concepts literally have other concepts as their parts. If concept C has the concepts X and Y as its parts, then you cannot think of the concept C without invoking X and Y. The model has similarities to the way languages work. If you want to utter the sentence "Red is a color", then you must also utter the words "red", "is", "a" and "color" - there is no way around it. The Inferential Model says that concept's structure is the inferences that can be drawn from it. A concept can therefore be instantiated without instantiating the concepts that are part of its structure. Accordingly, one can entertain the concept RED without instantiating the concept COLOR; color can be inferred from RED, but it does not have to be instantiated when RED is. (Laurence & Margolis 1999, 5.)

2.2 What theories of concepts must explain?

Since concepts play such a vital role in cognition, a theory of concepts should explain quite a lot of psychological phenomena starting from categorization, reference determination and

concept learning. Here are Fodor's (1998, 23-34) five "non-negotiable" conditions that any theory of concepts must explain.

1. Concepts are mental particulars. (Fodor 1998, 23).

There is a philosophical debate concerning the ontological status of concepts. Are they located in the head or are they some external entities located in some mysterious platonic realm? From the point of view of this paper the ontology of concepts is quite irrelevant and will not be examined any further, except for remarking that the commitment to RTM assumes that concepts are mental particulars. This is also the standard view in psychology (Laurence & Margolis 1999, 8).

> Very roughly, concepts are constituents of mental states. Thus, for example, believing that *cats are animals* is a paradigmatic mental state, and the concept ANIMAL is a constituent of the belief that *cats are animals* [...]. So the natural home of a theory of concepts is as part of a theory of mental states. (Fodor 1998, 6).

> 2. Concepts are categories and are routinely employed as such. (Fodor 1998, 24).

Strictly speaking, a category and a concept are not exactly the same. A category is what bundles a group of objects together. A concept can, additionally, include all kinds of conceptual information about the category. A child can learn to categorize different pieces of metal objects as coins, but on top of this an adult has the concept MONEY with all kinds of conceptual knowledge built into it: money has value that depends on economics, it is illegal to counterfeit it, money is used as a tool of exchange and money would still be money even if the coins change their appearance. Concepts are, nevertheless, used as categories - they pick out entities in the world. The concept CAT picks out a group of entities in the world, namely cats. "To say that concepts are categories is to say that they apply to things in the world; things in the world 'fall under them'" (Fodor 1998, 24). As said before, concepts and categories are treated equally in this study.

> 3. Compositionality: concepts are the constituents of thoughts and, in indefinitely many cases, of one another. Mental representations inherit their contents from the contents of their constituents. (Fodor 1998, 25).

Productivity of thought means that there is no bound or upper limit to the number of distinct thoughts that we can potentially entertain. The same applies to languages: there is no theoretical limit to how long a sentence in a language can be. The following sentence can be extended forever:

John woke up.
John woke up this morning.
John woke up this morning and ate breakfast.
John woke up this morning and ate breakfast and brushed his teeth.
Etc.

All of the sentences above express different thoughts and since there are infinitely many of them, it follows that thought must be productive as well. Productivity is an idealization. It does not mean that we produce an infinite number of sentences practically. It only means that the mechanism responsible for generating sentences is only limited by performance limitation, such as memory and time.

> This is not, of course, to argue that the *practical* possibilities are *literally* infinite. Just as there is a longest-sentence-that-anyone-can-utter, so there must be a most-complex-situation-that-anyone-can-act-upon. The infinite capacity of the representational system is thus an idealization, but it is not an *arbitrary* idealization. (Fodor 1975, 31).

Thought is also said to be systematic. Once we can entertain a certain type of thought we can entertain an infinite set of similar thoughts. If we can think JOHN LOVES MARY, then we can also think MARY LOVES JOHN, ROMEO LOVES JULIA, JULIA LOVES ROMEO, etc. The systematicity of thought is not an a priori fact. It could well be that we would have to learn each of the above thoughts separately, but it just isn't so. When a child learns a certain expression, the child learns an infinite set of similar expressions at the same time. (Fodor 1998, 26.)

We can explain the productivity and systematicity of thought if we assume that thought is also compositional. Compositionality means that thoughts can be combined from smaller components by using a set of rules. If the rules allow the components to be used again and again, then the combining mechanism produces an infinite amount of different thoughts from the basic repertoire of components. Compositionality, therefore, explains the

systematicity and productivity of thought by postulating only a finite amount of objects in the mind (the primitive components and the rules).

As an example of compositionality we can imagine that the mind contains the following context-free grammar G = (V, Z, P, S):

V = { S, A, B, C, V, John, Mary, Romeo, Julia, loves, hates, likes, knows, who }
Z = { John, Mary, Romeo, Julia, loves, hates, likes, knows, who }
P = { S -> A V B,
    A -> John | Mary | Romeo | Julia,
    V -> loves | hates | likes | knows,
    B -> A | A C V B,
    C -> who }
S = { S }

This context-free grammar generates productively and systematically expressions using compositionality (all the expressions are composed from a finite base of symbols and recursive rules). The following sentences work as demonstrations:

John loves Mary
Mary loves Romeo
Romeo loves Romeo
Romeo hates Julia who likes Mary
Julia knows John who knows Mary who hates Romeo who likes Mary

    4. Quite a lot of concepts must turn out to be learned. (Fodor 1998, 27).

A theory of concepts must also explain how concepts are learned. Some concepts, e.g. primitives, may be innate, but most concepts must be learned; it is quite implausible that concepts like GRUNGE-ROCK, DONUT and INDUSTRIAL REVOLUTION are innate. What this means is that the ability to learn must be enabled in a knowledge representation system - if all concepts were just preprogrammed into it, it would not be a serious candidate as a psychological model.

    5. Concepts are *public*; they're the sorts of things that lots of people can, and
       do, *share*. (Fodor 1998, 28).

This is Fodor's argument against conceptual relativism, which shortly means that different people don't share the same concepts at all and even one individual might not have the

same concepts between different time slices. Concepts are type identical across people living in different times, cultures and with different backgrounds. That is, people share the same concepts. In order to communicate thoughts about dogs or to agree or disagree about dogs, we must have the concept DOG in common. Otherwise, communication breaks down and we would not have a real agreement or disagreement about anything - it would just be a verbal dispute. Different people surely have different knowledge about dogs and different experiences with them, but this is just to say that their concept of dog is related or associated to different concepts.

The following review of the current major theories of concepts follows closely the reviews done by Laurence & Margolis (1999) and Fodor (1998). I have added a separate chapter for the dual theories, because this is what some of the models will end up being. The two reviews exclude holistic theories of concepts and connectionist ones (see for example Clark (1993)). This paper will omit them likewise.

2.3 The classical theory

All theories of concepts can be seen as reactions to the classical, definitional view of concepts. The theory stems from the ontologies of Plato and Aristotle and has been among us for more than 2000 years: it is the standard view in science, but also in folk psychology and has not been questioned until this century. (Fodor & al. 1980, 263; Lakoff 1987a, xii; Laurence & Margolis 1999, 8.)

According to the classical view, categories are defined solely by the common properties shared by all members of the category. There is a set of necessary and sufficient properties to distinguish if an object belongs to a certain category. It either does or does not; there are no graded boundaries. It is a matter of two-valued logic: a statement of type "object x belongs to the category X" is either true or false. Classical concepts are then structured - their structure is the definition.

Katz gives an example of a definition. The concept CHAIR according to him has the following definition (1972, 40):

> (Object), (Physical), (Non-living), (Artifact), (Furniture), (Portable), (Something with legs), (Something with a back), (Something with a seat), (Seat for one)

CHAIR is thus composed of these concepts or features and if an object is structured of these and only these concepts it is a chair. Otherwise it is not. The components in the definition can themselves have a definition. The concept OBJECT used in the definition might be defined as "an organization of parts that are spatio-temporally contiguous which form a stable whole having an orientation in space" (Katz 1972, 40). But eventually one must reach primitive concepts that lack definitions. The primitives can either be viewed from the empiricist point of view as 'unproblematic' sensory percepts or from the rationalist point of view as innate concepts (Fodor & al. 1980, 266; Fodor 1981, 263; Laurence & Margolis 1999, 9).

2.3.1 The power of definitions

Not only is the classical theory simple, it provides good answers for several questions. First, because a concept is defined of necessary and sufficient features, the task of learning a concept can be seen as finding those features. Since a concept can be complex, it is required that the constituent concepts are learned first, but at the bottom of the stack there are concepts that the learner has innately or that are given as inputs from senses. If the primitive concepts are sensory features, then the learner can notice that in the environment a certain set of features are found together. These could be the features that encode the concept CAT and an another bundle of features might encode all the dogs found in the environment. What is required is a finite base of already known concepts; after that, new concepts, an infinite amount of them in fact, can be learned by assembling the already known ones into novel syntactical combinations. (Laurence & Margolis 1999, 10.)

Second, the explanation of categorization by the classical model is quite simple. Once a category is learned, it can be used to check if a new object falls under it by checking that it has all the features that are part of the category. If the object has these features, then it falls

under the category, otherwise it is outside the category. So, to verify if an encountered object is a bird, that is, if it is part of the extension of the concept BIRD, one must check that it has feathers, is an animal, can fly, has wings and whatever else is part of the definition. (Laurence & Margolis 1999, 11.)

Third, concepts also seem to have inferential properties. From the statement "there is a chair in the room", one can infer that there is a piece of furniture in the room, that there is something physical in the room, that there is something non-living in the room, etc. (Katz 1972, 41). But one cannot infer that there is something expensive in the room, that there is a carnivore in the room or that there is something drinkable in the room. The valid inferences seem to be obviously valid - they follow analytically from the meaning of CHAIR, whereas the invalid ones do not; it could be that there is something expensive in the room, but it is not an analytic truth - some chairs are cheap, some expensive. These inferences can be explained by appealing to the definition: if the concepts FURNITURE, PHYSICAL, NON-LIVING are part of CHAIR, then one can infer with certitude that if an object is a chair, then it is necessarily a piece of furniture as well. (Laurence & Margolis 1999, 12.)

Fourth, knowing a concept includes knowing which objects in the world it applies to. What comes to reference determination, the classical model once again explains this by appealing to the definition: a concept refers to all those objects that satisfy its definition. For complex concepts the reference is determined compositionally. It is the intersection of the extensions of the defining concepts. BACHELOR refers to all things that are unmarried *and* males. (Laurence & Margolis 1999, 13.)

2.3.2   Criticism of the classical theory

The classical view has been taken for granted until the 20th century when both philosophers and psychologist attacked it. As was seen above, the classical view explains in a quite simple and similar manner many questions.

> The classical mentalist view is that postulating concepts answers three questions; (a) what does a term in a natural language express; (b) what determines the extension of a term in a natural language; and (c) what does a

term in a natural language contribute to the semantics of the complex expressions in which it occurs. The beauty of the classical story is that it provides the *same* answer for all three of these questions: in particular, the very thing which determines the extension of a term *is also what the term contributes to the (syntactically) complex expressions that contain it*. (Fodor 1981, 295).

Regardless of the explanatory power of the theory its popularity has been diminishing. The gravest problem of the theory is simply the lack of definitions. Definitions are hard to come by. It is extremely difficult to define even simple things like apples or chairs. But in science also many basic concepts in any discipline are in the end left undefined. On top of this, definitions were supposed to decompose closer to the primitive level. But it can be doubted if any of the proposed definitions have managed to do this. Are the concepts UNMARRIED and MAN any closer to sensory data than the concept BACHELOR, which they were suppose to define? Recall Katz's monstrous definition for "object": "an organization of parts that are spatio-temporally contiguous which form a stable whole having an orientation in space". Or how about the definition for "ketch": "two masted sailing vessel such that the art (or 'mizzen') mast is stepped forward of the helm" (Fodor & al. 1980, 268). Have these definitions actually gotten us anywhere at all? The job that definitions were suppose to do was to get us eventually to the primitive level of sensory concepts, but at least these examples have not succeeded in it. Moreover, one can justifiably ask why try to explain a word with *other* words? Does it ever lead us closer to the *meaning* of words? (Fodor & al. 1980, 268; Laurence & Margolis 1999, 14.)

The problem that definitions are hard to come by doesn't mean that definitions haven't been proposed at all. On the contrary, any science is full of attempts to formulate good definitions that meet all conditions: bachelor = an unmarried man, mother = a woman who has given birth to a child and e.g. the above definition for chair. The problem is that one can always find counter examples to definitions. The classical theory stipulates that definitions are necessary and sufficient - all or nothing. It does not tolerate exceptions or unclear cases. But there are popes, homosexual males, Tarzans and Robinson Cruseos that are unmarried men, but who are not clearly bachelors. There are stepmothers, surrogate mothers, adoptive mothers, foster mothers, biological mothers and donor mothers. And there are rocking chairs, barber chairs, wheelchairs, beanbag chairs and electric chairs that don't really match the definition, but they are nonetheless chairs for all that[1]. No matter

how hard one tries it seems that definitions fail to capture the set of objects that fall under the concept. Either definitions are too loose or too strict.

It should be added that perhaps there might be few definitions out there somewhere. In the field of mathematics, for example, one might find some rock-solid definitions (even number = an integer divisible by two). Perhaps somewhere else too. But this is not a counter argument to the criticism: we have thousands, if not millions, of lexical concepts and if the classical model is to work it must find definitions for all them.

> - There are practically no defensible examples of definitions; for all the examples we've got, practically all words (/concepts) are undefinable. And, of course, if a word (/concept) doesn't have a definition, then its definition can't be its meaning. (Oh well, maybe there's one definition. Maybe BACHELOR has the content *unmarried man*. Maybe there are even six or seven definitions; why should I quibble? If there are six or seven definitions, or sixty or seventy, that still leaves a lot of words/concepts undefined, hence a lot of words/concepts of which the definitional theory of meaning is false. The OED lists half a million words, plus or minus a few.) (Fodor 1998, 45).

It can be remarked that maybe definitions haven't been found yet because they are just so hard to find. Perhaps, but, for example, philosophy can be said to be all about finding definitions. Philosophers ask questions of type What is knowledge, What is good, What is freedom, etc. The answers that philosophers have proposed are definitions: Knowledge is a true justified belief, Good is the maximization of pleasure, Freedom is the absence of coercion... It is an embarrassing fact that even after 2500 years of the efforts from the greatest minds in the world, philosophy has accomplished practically nothing. Couldn't this be interpreted so that perhaps there was something wrong with the enterprise in the first place? Maybe the assumption that there are answers to questions of type What is X is invalid. Just maybe definitions haven't been found, because there aren't any definitions to be found at all.

Putnam (1970) has put forward arguments that are even more devastating for the classical view. If he is correct, then it is useless to even have hope of finding definitions, because they cannot account for meaning of words even in theory. Putnam attacks what is called descriptivist views of semantics, which roughly means that word reference is determined via a description (a definition for example). Contrasting theories claim that descriptions are unnecessary to account for word meaning; meaning simply is reference. Classical view is a

descriptivist view, because a definition is needed to explain concept's reference. (Laurence & Margolis 1999, 21.)

The problem with any descriptivist view is that we can be deeply mistaken about our definitions and theories for concepts. Many concepts can undergo radical conceptual changes as has happened for an example to the concept DISEASE. The concept disease dates back to ancient Greek and Hippocrates, who held a humoral theory of the causes of diseases: diseases are caused by the imbalance of fluids in the body. This was already a radical conceptual change, because before Hippocrates and his disciples diseases were believed to be divine punishments inflected on people by gods. The first change to the humoral theory was made only in the 16th century in Italy by Fracastro. He suggested for the first time an idea that there are some small strange particles that transmit themselves from body to body causing a contagion that finally leads to a disease. In the 19th century, Louis Pasteur and others dropped the humoral theory for good and developed the modern germ theory of diseases. The transition from the humoral theory to the germ theory involved many radical conceptual changes. (Thagard 1999.)

The germ theory still prevails and from our point of view the earlier theorists were sadly mistaken about diseases. But to emphasize, they were mistaken about *diseases*; our theories are theories *about* diseases as are theirs. If all these people talk about the same thing, then it follows that their definitions cannot be essential of DISEASE; the concept is the same regardless of the definition surrounding it. If the definition was constitutive of the concept DISEASE, then we would not be disagreeing about diseases, instead, all examples above would be talking about some altogether different concepts. (Laurence & Margolis 1999, 21.)

All concepts can undergo same kinds of radical changes. Say, that some of the defining properties of lemons are yellow and sour, that cats are all four legged animals and that tigers are striped. Nothing guarantees that these properties are necessary. Maybe lemons actually are blue; it just happens that we have been in contact with some very unusual lemons that for some bizarre reason are yellow. Maybe cats turn out to be robots controlled by Martians. Maybe tigers have been painted with stripes only to deceive us. Since it seems that no definition is immune to revision it follows that they just cannot be the core of concepts. (Putnam 1970, 180.)

It seems that our definitions can be badly mistaken. But we can also be ignorant of them. If we imagine just for the sake of argument that there are perfect definitions and the modern definition of DISEASE is one them, then we find the classical theory in a situation where it is cognitively quite too demanding. Do people really, really know the definition of DISEASE? Most people are ignorant of the germ theory or have only a superficial understanding of it - one that they were forced to learn in school and have, most likely, happily forgotten long ago. If the definition of DISEASE is the current germ theory, we are forced to conclude that this is what people learn when they learn to use the word "disease". But isn't this a bit too much to ask? Even children understand something about diseases. Now, do we expect them to have mastered a modern scientific theory - a theory introduced only in high school and university courses? If the classical theory still wants to claim that there are definitions - although practically none have been found - it also has to explain why so many people are blissfully ignorant of them, but still know perfectly well how to use concepts. (Laurence & Margolis 1999, 22.)

Later Wittgenstein (1953, 1:65-78) also pointed out that there are many categories that do not fit the classical demand that all members of a category share the same properties. The category GAME does not possess common properties shared by all games. Some games are just for fun - there is no competition. Other games involve luck, like games played with dice, but in some games skill is everything, like in chess. Still others, like poker, involve both luck and skill. Skill is needed in tennis too, but it is very different kind of skill than in othello. Some games consist of thousands of competitors like big marathons in Olympic Games. In some games, there are only few players - yet patience and console games involve only one player. Category boundaries are also extendable: games come and go. New games are invented all the time and some games vanish for good. Think of role playing games for example or computer games played over the Internet, where the players may also participate in the game design.

Wittgenstein claimed that the different games are united by what he called "family resemblances". Games resemble each other, but there is no one single property manifest in all games. As we move from one game to an other similarities and relationships crop up and disappear. Games form a complex network of criss-crossing properties and are tied to

other games in a wide variety of ways, and this is what makes them a category - not one property or a set of properties shared by all games.

More formally, category C can consist of the following members: AB, BC, CD, AC and BD. Members are united in this case by family resemblance; no single property can be predicated to all members, instead, the members are connected to each other in many ways. Still they resemble each other enough to belong to the same category.

2.4   The prototype theory

Many people have an intuition that some members are better examples of a category than others, e.g. office chair is a more typical chair than a beanbag chair and similarly robin is birdier than flamingo. It is also common knowledge that there are no rules without exceptions. The classical theory cannot explain this, because if concepts encode necessary and sufficient features, then all members of the category share them and are therefore on equal footing; there is no reason why one member should more clearly fall under a concept than another. In a sense, it seems even a bit odd to ask in the classical framework how well an instance belongs to a category, because they are either in or out. The intuition that categories are graded and always contain exceptions is, in fact, backed up by a vast amount of psychological evidence. The prototype theory sets out to explain these and other empirical findings.

Rosch and her colleagues invented a battery of experiments to empirically test the assumptions of the classical theory (Rosch 1973a/1973b/1975/1978; Rosch & Mervis 1975; Rosch & al. 1976). First, some natural categories such as color and geometrical forms cannot be arbitrarily classified into any units as might be expected by the classical theory. Instead, some colors, called "focal colors", and geometrical forms, such as square, circle and triangle, are much easier to learn and memorize. This is not even dependent on language or culture. Speakers of Dani, a Stone Age culture in Indonesia with only two color words and no words at all for geometrical figures, learned focal colors and geometrical shapes easier than others, just as English speakers did. In fact, what colors are first lexicalized in languages is very systematic and universal: The first colors to enter a lexicon are black and white. They are followed by red, which is followed by either yellow,

blue or green (Lakoff 1987a, 25). Thus, contrary to what the classical theory would predict, the color space and geometrical planes cannot be divided in any way; some colors and geometrical forms are better than others. These members are called "natural prototypes" and categories form around them. In the case of color, there is even neurological evidence to support the cognitive primacy of prototypical colors. (Rosch 1973a.)

Members in a category are also in an asymmetrical relation to each other. The prototypes are cognitive reference points to which other members are compared. So, people tend to think that 97 is almost 100, but not that 100 is almost 97. Similarly, a line in an angle of 45 degrees is almost a horizontal line, but not vice versa. (Rosch 1975.)

When participants are asked to rank how good an item is of a category, they generally have no difficulty doing this and the results of these typicality effects (also called prototype effects) are systematic between individuals and quite consistent between different experiments. As an example, in one study participants were asked to judge how well items belong to a category, e.g. is a robin a better bird than a chicken. The following table shows results of rating the goodness of members in different categories in the scale 1-7, where 1.0 is the most typical (based on Rosch 1973b, 133):

Fruit		Vegetable		Bird	
Apple	1.3	Carrot	1.1	Robin	1.1
Plum	2.3	Asparagus	1.3	Eagle	1.2
Pineapple	2.3	Celery	1.7	Wren	1.4
Strawberry	2.3	Onion	2.7	Ostrich	3.3
Fig	4.7	Parsley	3.8	Chicken	3.8
Olive	6.2	Pickle	4.4	Bat	5.8
Vehicle		Disease		Crime	
Car	1.0	Cancer	1.2	Murder	1.0
Scooter	2.5	Malaria	1.4	Stealing	1.3
Boat	2.7	Muscular dystrophy	1.9	Assault	1.4
Tricycle	3.5	Measles	2.8	Blackmail	1.7
Skis	5.7	Rheumatism	3.5	Embezzling	1.8
Horse	5.9	Cold	4.7	Vagrancy	5.3

In another experiment participants were asked to list properties found in members of a category. The results suggest that properties are not shared by all members. Instead, some properties are statistically much more frequent in a category; some properties were listed only for few items in the category; only rarely was there a property shared by all members. The number of times each attribute is listed for items in a particular category corresponds to its importance or weight in a category, that is, if a property is found in ten members of a

category, then its weight is 10. Member's "family resemblance degree" or "cue validity" is the sum of all the weighted attributes listed for it. So, if three properties are listed for a member, each of them having a weight of 1, then the members family resemblance degree is 3 (1+1+1). (Rosch & Mervis 1975.)

The interesting thing to note was that the members with the highest family resemblance degree corresponded to the prototypes of that category. This can be seen in the table below (Rosch & Mervis 1975, 582):

Number of attributes in common to five most and five least prototypical members of six categories.		
Category	**Most typical members**	**Least typical members**
Furniture	13	2
Vehicle	36	2
Fruit	16	0
Weapon	9	0
Vegetable	3	0
Clothing	21	0

Prototypes, then, correspond to members "[…] with most properties in common with other members of the category and least attributes in common with other categories" (Rosch & Mervis 1975, 573). The study shows that attributes in a category have different weights. Some are judged to be more frequent and some less frequent. And if categories have family resemblance structure, then the formation of a prototype is a natural consequence. In Rosch's words, "[t]he present study is an empirical confirmation of Wittgenstein's (1953) argument that formal criteria are neither a logical nor psychological necessity; […]" (Rosch & Mervis 1975, 603).

Concepts form hierarchical taxonomies - they are not an unorganized mess. As we learn, we find that some categories have more specific subcategories and some are united by a higher, more abstract category. The concepts DOG, CAT and HORSE are connected to the higher level concept ANIMAL. Similarly DOG is above the concepts GOLDEN-RETRIEVER and BULLDOG. According to the classical view, no level should have a special status. Rosch & al. (1976) showed that this is not the case; psychologically the most important level is in the *middle* of the hierarchy. This level, called the "basic level", is the first level learned and it can be later specialized to a subordinate level or generalized to a superordinate level (Lakoff 1987a, 46):

```
SUPERORDINATE:      ANIMAL    FURNITURE
BASIC LEVEL:        DOG       CHAIR
SUBORDINATE:        RETRIEVER ROCKING CHAIR
```

When participants are asked to list attributes of a category, they usually list attributes found in the basic level, and very few attributes of the superordinate level. The basic level is hence the most inclusive level. It is the also the level at which children learn to use words first; at the age of three almost every word belongs to the basic level. Basic level objects have "interactional properties", meaning that people have ways to interact with them and to form mental images, whereas at the superordinate level objects cannot be interacted with and representing them needs a process of abstraction. This representation can no longer be an image. For example, we can form an image representation of a chair or a dog, but it is impossible to form an image of a furniture or an animal that would not be an object of the basic level. (Rosch & al. 1976.)

The basic level is (Lakoff 1987a, 46):

> - The highest level at which category members have similarly perceived overall shapes.
> - The highest level at which a single mental image can reflect the entire category.
> - The highest level at which a person uses similar motor actions for interacting with category members.
> - The level at which subjects are fastest at identifying category members.
> - The level with the most commonly used labels for category members.
> - The first level named and understood by children.
> - The first level to enter the lexicon of a language.
> - The level with the shortest primary lexems.
> - The level at which terms are used in neutral contexts. For example, *There's a dog on the porch* can be used in a neutral context, whereas special contexts are needed for *There's mammal on the porch* or *There's a wire-haired terrier on the porch*. (See Cruse 1977.)
> - The level at which most of our knowledge is organized.

The prototype theory explains this ensemble of findings - prototypes, cognitive reference points, typicality effects, family resemblance and basic level - by postulating that concepts are structured representations that consist of an open set of features. The features have different statistical frequencies within a category. Prototypes are learned simply by observing that a cluster of properties tends to correlate positively. Some features can be more frequent than others; some can even be missing, but as long as there is enough

similarity with the prototypical features the instance falls under a category. Instances don't, then, need to have all the features found in the category. There still remains features/properties as in the classical theory, but they are no longer considered to be necessary. Instead, the features have a strong statistical *tendency* to occur together.

It should be pointed out at this moment that the mere fact that categories are graded is not a decisive argument against the classical theory. Classical concepts can be fuzzy if any of the constituent concepts are fuzzy. If the concept C has the definition C = AB and the extension of A and/or B is graded, then C is graded also. But clearly C has a definition, namely AB (Fodor 1975, 62). What does undermine the classical view is that it has no natural way of explaining why typicality effects should rise at all. And why some levels and members are cognitively much more significant.

How then are typicality effects explained? Because the judgement, if an instance belongs to a category, is based on its similarity to the prototypical instances of the category, categorization is performed using some sort of a similarity metric. When the metric exceeds a threshold, the instance is positive, otherwise negative. So, if an instance clearly has many of the required salient features, then the threshold is reached rapidly, because only few features need to be checked. But when the instance barely has the features required, a high number, if not all of them need to be calculated before the threshold is reached. Calculating features is a time consuming process: the more features to check, the longer it takes. Hence, the closer the instance is to the prototype, the faster it is recognized. Exactly as participants in Rosch's experiment did. In the classical theory each instance should be recognized in the same time, because every feature must be found on the instance before it can be classified.

In sum, prototype categories differ from classical ones in several ways (adapted from Pinker & Prince 1999, 8):

- They lack *necessary and sufficient conditions* for membership.
- They have graded degrees of membership.
- The category can be summarized by an ideal member or *prototype*, sometimes but not always an actual exemplar of the category.
- There can be *unclear cases* - objects that may or may not be members of the category at all.
- They often display a family resemblance structure.

- Good members tend to have *characteristic nondefining features*.

2.4.1 Similarity metrics

An example of a similarity metric is Tversky's contrast rule used by Smith & al. (1988, 491). The similarity of an instance is calculated by checking how much it has in common with the prototype, how many features are distinct to the prototype and how many features are distinct to the instance. In terms of set theory this is equivalent of subtracting from the intersection of P and I, the difference of P and I and the difference of I and P, where P is the prototype and I is an instance. Formally it looks like this:

$$\text{Sim}(P, I) = af(P \ \& \ I) - bf(P - I) - cf(I - P)$$

where a, b and c are factors that weight the importance of each set.

This metric yields the highest number for an instance that is equal to the prototype (the prototype compared to itself). The result gets lower the more the instance differs from the prototype, that is, the more distinct features it possesses. The metric does not, however, say anything about the judgment if an instance falls under a category. For this an additional threshold is required. If we set this to zero for simplicity, then we have a computational model of categorization. If the value of the metric is above zero the object belongs to the category, otherwise it doesn't.

Other types of similarity metrics exist also. Artificial Neural Networks (ANN) have an inherent prototype extraction mechanism (Clark 1993, 20). In ANNs individual neurons are interpreted to be concepts[2]. If the neuron is activated by the input vectors from its presynaptic neurons, then the concept is instantiated. There are many types of activation functions, but a quite common one is the sigmoid function (Beale & Jackson 1990, 72) defined as

$$F(\text{net}) = 1/(1 + e^{k*\text{net}})$$

where k is a positive gain term. It has the range $0 < f(\text{net}) < 1$. The sigmoid function can be used as a typicality metric: the closer the output is to 1, the more typical the input is. A

threshold needs to be added to this function as well to determine if an input belongs to the category. If the threshold is set to 0.5, then a trained network will classify inputs into two classes according to the following algorithm:

```
if F(net) > 0.5
    output 1
else
    output 0
```

The output neuron can represent, for example, the concept BURGLAR. The network is given an input and if the burglar neuron outputs 1, the input belongs to the category BURGLAR. The goodness of the input can be measured from the activation function. An input that outputs 0.9 is a better burglar than an input with an output of only 0.6.

2.4.2   Criticism of the prototype theory

Prototype effects seem to be ubiquitous in concepts; there probably is no doubt about this, but what can actually be inferred from them? Armstrong & al. (1983) tested if prototype effects could be found in concepts that according to them are classical. The reasoning was that if prototype effects appeared also in concepts that have definitions, concepts that are all-or-none, then the conclusion that concepts are prototypes is invalid. And what they found was exactly that: some concepts do have definitions and also prototype effects. If this is the case, then the conclusion from prototype effects to prototype structure is unwarranted. The tests replicated those performed by Rosch and others, except that the concepts for which typicality effects were to be found were knife-edged, classical concepts: EVEN NUMBER, ODD NUMBER, FEMALE and PLANE GEOMETRY FIGURE. A concept such as ODD NUMBER has, according to Armstrong & al. (1983, 274) a definition: odd numbers are integers not divisible by two without remainder, whereas even numbers are divisible. For FEMALE and PLANE GEOMETRY Armstrong & al. don't give us a definition, but according to them they are well defined.

The results showed that even these concepts have typicality effects. People think the number 13 is odder than 23. FEMALE also showed prototype effects, housewife and ballerinas being better females than chairwomen or cowgirls. Participants also viewed squares and triangles as being more typical geometrical figures than ellipses. On top of

this, at the end of the study the participants were directly asked straight in the face if in their opinion these concepts are graded. That is, are the categories all-or-none or is it possible that some members belong to it more clearly than others? 100% of the participants felt that it doesn't make sense to rate the items degree of membership in the categories EVEN NUMBER, ODD NUMBER and PLANE GEOMETRY FIGURE (Armstrong & al. 1983, 287). Items are either in or out. For the category FEMALE the percentage was 86%. This result is surprising, because when asked to rank how well members belong to a category participants produced a clear and a systematic ranking of members, although they claimed that it doesn't make sense to say that some integers are odder or evener than others.

Armstrong & al. concluded from their study that the inference for prototype structure is invalid. Graded responses seem to be irrelevant for the structure of concepts, because both classical and prototype categories have them. The fact that there are prototype effects says nothing in the end about the structure of concepts - they are only effects and nothing else.

> These results do not suggest that categories such as *fruit* or *vehicle* are well-defined in the classical or any other sense - no more than they suggest that *odd number* is fuzzy. What they do suggest is that we are back at square one in discovering the structure of everyday categories *experimentally*. This is because our results indicate that certain techniques widely used to elicit and therefore elucidate the structure of such categories are flawed. This being so, the study of conceptual structure has not been put on an experimental footing, and the structure of those concepts studied by current techniques remains unknown. (Armstrong & al. 1983, 291).

Similar results were found in the English past tense system. Regular past tense verbs have a definition, which simply is the rule *stem+/ed/*, but the class of irregular verbs shows family resemblance structures and the verbs are organized around prototypical examples. These results seem to indicate that categories can be *both* classical and prototypical. (Pinker & Prince 1999, 47.)

Another common complaint against the prototype theory is that it is incapable of explaining the productivity of thought. By combining concepts into larger concepts, potentially an infinite number of complex concepts can be produced - this is the principle of compositionality as was explained earlier. It makes no sense to explain that a finite

organism, such as a brain or a computer, possesses an infinite amount of prototypes; there can't be a prototype for every thought that we can think of.

> There may, for example, be prototypical *cities* (London, Athens, Rome, New York); there may even be prototypical *American Cities* (New York, Chicago, Los Angeles); but there are surely no prototypical *American cities situated on the East Coast just a little south of Tennessee*. Similarly, there may be prototypical *grandmothers,* (Mary Worth) and there may be prototypical *properties of grandmothers* (*good, old* Mary Worth). But there are surely no prototypical properties of, say, *Chaucer's grandmothers*, and there are no prototypical properties of *grandmothers most of whose grandchildren are married to dentists.* (Fodor 1981, 296).

Consider also the following concepts (Laurence & Margolis 1999, 36):

U.S. MONARCH
31ST CENTURY INVENTION
GREAT-GREAT-GREAT GRANDCHILD OF CINDY CRAWFORD
PIECE OF PAPER I LEFT ON MY DESK LAST NIGHT
JUSTICE

These are perfectly fine concepts, but they lack a prototype. Some are too vague, some refer to nonexistent things: there is no U.S. monarch and the great-great-great grandchildren of Cindy Crawford aren't even born yet. How could anyone have a prototype, a best example, of them?

One might think that the examples are just cleverly thought exceptions. But they all reflect a deep and a pressing problem for the prototype theory: prototypes are not semantically compositional. Consider the concepts STRIPED APPLE and APPLE. A good instance (a) of a STRIPED APPLE is evidently not a very good example of an APPLE, since apples rarely have stripes. Now, because (a) is always less typical of APPLE than STRIPED APPLE, the following holds: C(STRIPED APPLE, (a)) > C(APPLE, (a)), where C is the typicality value of (a), which could be 0.9 and 0.2 respectively. The fuzzy set theory is usually used to model the set operations with prototypes. According to it, the intersection of two sets is defined as the minimum of the values of the sets. If this is the case, then (a) can *never* be higher than the minimum of STRIPED and APPLE. But this is not true, because (a) is more prototypical of STRIPED APPLE than APPLE. The min rule simply predicts this incorrectly: the typicality value of (a) would be 0.2, although it should be 0.9

in this case. Therefore, one cannot arrive at a prototypical instance of STRIPED APPLE from composing the concept STRIPED and APPLE - it will never be a *prototypical* striped apple. (Osherson & Smith 1981, 43.)

The underlying problem is that prototypes encode statistical features; what features happen to be prototypical is a contingent fact of the world. Consider the concept PET FISH. Typical pets are somewhat like cats and dogs and other furry animals with tails. Typical fish, such as trout or salmon, live in waters. If pet fish were composed from the prototypes of its constituents, then pet fish would probably have fur, wag a tail and live in waters. But they don't: typical pet fish, such as gold fish, are very small, live in bowls and are brightly colored. The concept PET FISH cannot be composed from the prototypes of PET and FISH, so "[i]t follows that if meanings were prototypes, then you could know what 'pet' means and know what 'fish' means and *still not know* what 'pet fish' means" (Fodor 1998, 102; emphasis mine). What happens to be a typical pet fish has more to do with fashion than with the underlying prototypes. One day it could become fashionable to have piranhas as pet fish. If the prototype of PET FISH is composed from its constituent prototypes, then they would have to change too. (Fodor 1998, 100-104.)

Another problem follows from the coding of statistical tendency: prototypes don't fix references properly. The prototype of tigers refers only to prototypical tigers. But a three legged, toothless, albino tiger is still a tiger, albeit not a very prototypical one, and should be included in the extension of TIGER. A prototypical grandmother is presumably old, has gray hair and glasses, bakes cookies and is kind to children. The problem is that this does not fix the extension of GRANDMOTHER properly. Somebody can have these features without being a grandmother and grandmothers are not always prototypical grandmothers. Whoopi Goldberg and Tina Turner are grandmothers also. (Laurence & Margolis 1999, 34).

The problems of compositionality and reference fixing have been noticed by the defenders of the prototype theory and some repairements have been done (see for example Smith & al. (1988)). What if the improvements to the theory or to the fuzzy set operations somehow overcome the problems? Will this help? No. Putnam's criticism against the classical theory applies to the prototype theory as well. In the end, the only difference is that where the classical theory demands that the features are necessary, the prototype theory relaxes this

demand to a statistical tendency. But it is nonetheless a descriptivist account and suffers from the same problems of ignorance and error. How can we ever be sure that our description is the correct one? A prototypical lemon is sour and yellow and typically tigers have stripes. But again, what if lemons really are blue or if future tigers turn out to be stripeless? Lemons would still be lemons and tigers would still be tigers, even though our prototypes of them would now be far from prototypical. (Laurence & Margolis 1999, 34.)

After all, it seems that we can perfectly well possess a concept without knowing its prototype - compositionality generates an endless number of such counter examples to the prototype theory. It then follows that the prototype theory cannot tell the whole story of concepts. There might be some concepts that are prototypes, but for a vast amount of them a different story is needed. The current status of the prototype theory is that it is not a theory of representations. This conclusion was even admitted by Rosch herself:

> What the work summarized does not tell us, however, is considerably more than it tells us. The pervasiveness of prototypes in real-world categories and of prototypicality as a variable indicates that prototypes must have some place in psychological theories of representation, processing, and learning. However, prototypes themselves do not constitute any particular model of processes, representations, or learning. (Rosch 1978, 39).

This does not mean that prototype effects aren't real. There probably is little doubt concerning the experimental data and whatever the final grand theory of concepts is, it must explain the prototype effects:

> Insofar as theses get established in cognitive psychology, I think we can take the reality of prototype structure as read. (Fodor 1981, 293).

> One enormous phenomenon stands firm: subjects do give graded responses when queried, in any number of ways, about concepts. So powerful is this phenomenon that it survives even confrontation with the very concepts (*odd number*) it could not possibly illuminate or even describe. (Armstrong & al. 1983, 291).

> The discovery of the massive presence of prototypicality effects in all sorts of mental processes is one of the success stories of cognitive science. (Fodor 1998, 93).

2.5   The dual theories

2.5.1   The core and the identification procedure

The classical model suffers from the inability to explain typicality effects whereas the prototype model has its own weaknesses. This has led some researchers to postulate dual theories where concepts are in the end classical (to explain compositionality), but where prototype effects rise due to performance restrictions (to explain prototype effects). For example, concepts can have a classical "core", but some "identification procedure" is responsible for the rise of typicality effects. Prototypes are called upon when classification needs to be done fast and with inadequate data. The classical core is called into work when there are no performance limitations and more considerate classification is needed. (Osherson & Smith 1981, 57; Armstrong & al. 1983, 292.)

This could well explain why Armstrong & al. paradoxically achieved prototype effects for concepts that are classical, concepts for which it didn't even make sense to rate their goodness. As in the category ODD NUMBER, participants produced typicality effects, but still claimed that all the members of the category are equally good examples. Maybe they used the identification procedure when asked to rank the members for their goodness, but when asked to think more carefully they consulted the classical core and realized that the category is after all all-or-none. (Armstrong & al. 1983, 293.)

2.5.2   The continuum version

Pinker & Prince (1999) propose a variant theory, which I will shall the continuum version. According to it, both categories are psychologically real. The two kinds of concepts can live side-by-side in the mind and exist for different purposes: the family-resemblance system is responsible for recording correlational clusters of properties in the world and the classical system is responsible for ignoring statistical correlations and finding idealized scientific laws and hidden essences behind the observable properties. (Pinker & Prince 1999, 32-39).

It looks like the family resemblance system is highly adapted for categorizing a world where evolution has a tendency to break classes into not so well-defined classes. Mutations and other natural forces cause variations in biological populations, but similar processes shape the evolution of languages too; languages are influenced by other languages, which cause irregularities and oddities in the language systems. The resulting dirty and fuzzy categories cannot be handled using a classical category. Nature is full of classes of this type. On the other hand, once we ignore the fuzziness and the ragged edges of the real world and abstract, we are able to find firm classical categories. Science is all about idealization. The laws of geometry and physics are idealizations: there are no perfect geometrical figures or frictionless planes that objects follow. Once we selectively ignore unimportant things we can rise above the perceptual world and find the underlying laws of nature. Without classical categories science would not be possible. Moreover, there are plenty of human-made categories, such as legal definitions, that really are not sharp, but we force them to be so. For example, legally "adult" and "region" are clearly defined, but in reality they are not; the difference between an adult and a child is of course graded and so is the border between two regions. With such different utilities it is possible to imagine that both types of conceptual systems are handy.

> Classical categories are defined by formal rules and allow us to make inferences within idealized law-governed systems. Family resemblance categories are defined by correlations among features in sets of similar memorized exemplars, and allow us to make inferences about the observable products of history. (Pinker & Prince 1999, 47).

The following four imagined worlds clarify the need to have both types of category systems. The worlds contain a pool of objects with just three properties A, B and C. First is a classical, law-governed world (Pinker & Prince 1999, 39-41; numbering changed and I added the fourth world):

(1)
ABC
ABC
ABC
ABC
DEF
DEF
DEF
DEF

In this beautiful world classical categories are extremely useful. An observer can build two groups X and Y and predict with absolute certainty to which group any object belongs to with just, say, finding the first property. Internalizing laws like A -> BC and D -> EF makes life a lot easier. All one has to do is to find one single property - the rest can be inferred.

This kind of world could evolve in an isolated and stable environment where the population is left to live for a long time in peace. Similar process could happen to a language that has no influence from other languages and the language speakers over generations slowly omit or forget all the annoying irregularities and exceptions of the language.

Second is an example of a world that has evolved from divergent processes. Mutations and influences have broken the two previously neat categories into family resemblance categories, where the individuals are only loosely connected:

```
(2)
ABC
ABR
PBC
AQC
DEF
DER
PEF
DQR
```

In this world, laws don't apply anymore. It still isn't a jungle yet, because an observer can build family resemblance categories and classify objects with at least some probability. It looks like As are still dominant in the first category and Ds in the second. Observing that an object has A, then it belongs to the first category and once it is in there one can predict with probability 0.75^3 that it has B. This is a lot better than 0.375 which is the rate of Bs in all objects. The probability is not 1.0 as in the world (1), but it is still useful enough to go through the trouble of building categories.

The world (3) is a random world, where divergent processes have caused a havoc:

```
(3)
ABC
ABF
AEC
AEF
DBC
DBF
DEC
DEF
```

Building categories in this world is not helpful; the probability of B, C, E or F occurring is always 0.5. Although there is nothing to gain from categorizing this world it can still be classified into two groups by the first property, which is either A or D. Pinker & Prince's list can be continued with one more world which is entirely chaotic:

```
(4)
DEC
QBA
BCE
BFA
QFE
AEF
PBC
EAD
```

This world cannot even be categorized; all the members have to be memorized in a rote learning fashion.

All in all, it looks like there is a continuum from random worlds to family resemblance worlds and finally to clean, law-like worlds. Processes can diverge a nicely evolved law-governed world into a family resemblance world. And finally, the family resemblance world can break into a chaotic random mess. On the other hand, convergent processes can converge family resemblances from a random world and then finally tidy the world into an ideal law-governed classical world again. Thus, a mind equipped with both classical and prototype categories plus a rote learning mechanism can take the best out of any world.

2.5.3 Criticism of the dual theories

The dual theories attempt to get the best of both worlds. Unfortunately, they also drag along all the problems from both worlds. The problem that practically no definitions have been found is still very much a problem. Even if some were found, we could not sit back with a peace of mind knowing that the definition is certainly the correct one; for all we know, the world might be entirely different tomorrow. Adding the prototype theory on top of the classical theory does not sweep away these original problems. (Laurence & Margolis 1999, 34.)

2.6   The neoclassical theory

2.6.1   Back to definitions

The neoclassical theory is an update to the classical theory. Instead of claiming that concepts are full-blown definitions the neoclassical theory postulates that concepts encode only partial definitions. The partial definitions don't do the entire job of explaining what concepts are, something else is also needed, but, nonetheless, neoclassicists assume that concepts do have some kind of definitions and this ties them to the classical theory. For example, the concept TIGER must at least contain the concept THING, and RED must at least contain COLOR as its constituent. It is evident that these necessary conditions are not sufficient to explain concepts: something more than THING is needed to distinguish tigers from bats and equally COLOR is not sufficient to distinguish red from green. Concepts, then, according to the neoclassical theory are structured representations that encode partial necessary conditions - plus something else on top of this. (Laurence & Margolis 1999, 52.)

The motivation for assuming partial definitions probably comes from observations of similarities in linguistic expressions. The following four expressions are all different, but at the same time they seem polysemic - there is something highly similar in them (Jackendoff 1989, 84; numbering changed):

> i. Harry kept the bird in the cage.
> ii. Susan kept the money.
> iii. Sam kept the crowd happy.
> iiii. Let's keep the trip on Saturday.

Now, according to Jackendoff there is a single rule behind all these keep sentences - a rule that unites their different meaning: "The *keep* sentences all denote the causation of a state that endures over a period of time" (Jackendoff 1989, 84). As was said, the rule is a partial definition; there has to be more to sentence meaning, but the meanings must in minimum encode the partial definition as a necessary condition for the application of KEEP. (Laurence & Margolis 1999, 53.)

2.6.2   Criticism of the neoclassical theory

The obvious criticism against any theory that postulates partial definitions is that unless the theory explains how concepts are individuated it is not a finished theory. Partial definitions are simply not enough. Even though the keep sentences might contain the partial definition mentioned (although not everybody agrees with it, see for example Fodor (1998, 50, footnote 7)) it does not fully explain the different meanings between the sentences. The partial definition needs to be turned into a full definition. Now, the problem is that this full definition will sadly be a classical definition and it is well known that it can't be made to work. Moreover, if the neoclassical theory doesn't come up with an alternative to the classical theory it has not provided anything new. So, in order to make the theory work partial definitions must be converted to full explanations of a concept's/sentence's identity, but they need to take a different route than the classical theory. What this route is, is left unanswered. (Laurence & Margolis 1999, 54.)

Second, problems arise if we focus on the partial definitions more carefully. The partial definition CAUSE A STATE THAT ENDURES OVER TIME contains at least the concepts CAUSE, STATE, TIME and ENDURE always. These concepts are invoked in every meaning of KEEP, so they are "univocal", that is their meaning remains the same, whereas the state that is kept the same or the agent that causes the state are variables that change in every sentence. How are CAUSE and the rest to be explained? If they encode partial definitions, then the same question pops up for their definitions and again in those definitions. In order stop at some point, we must reach concepts embedded in partial definitions that no longer encode partial definitions. These concepts are not neoclassical, so another story is needed to explain them. But couldn't this story be applied already to

CAUSE? If so, why couldn't the same story be applied to KEEP itself and simply forget about the partial definitions? (Fodor 1998, 52.)

> Why not just say that 'keep' is univocal because it always means *keep*; just as, in order to avoid the regress, Jackendoff is required to say that 'CAUSE' is univocal because it always means *cause*. Or, quite generally, why not just say that *all* words are univocal across semantic fields because semantic fields don't affect meaning. (Fodor 1998, 52).

2.7 The theory-theory

The classical and prototype theories try to explain concepts using their properties. The theory-theory asks us to conceive concepts in fundamentally different terms. According to it, concepts are not defined by a list of properties, but by the role they play in our theories and explanations about the world. Their structure is the relation to other concepts determined by some mental theories.

> Concepts are constituents of beliefs; that is, propositions are represented by structures of concepts. Theories are complex mental structures consisting of a mentally represented domain of phenomena and explanatory principles that account for them. (Carey 1991, 258).

The important difference to the previous theories is that concept's identity is no longer explained by the Containment Model, where concepts have other concepts as their structure. Instead, concept's identity is explained by the Inferential Model, where the structure is determined by the relation it plays to other concepts. When theories change, so do the identities of the concepts that are embedded in the theories. Theories, according to the theory-theorists, do not need to be mature scientific theories. What is meant by "theory" is loosely any kinds of theories or explanations used by adults and children.

> When we argue that concepts are organized by theories, we use *theory* to mean any of a host of mental "explanations," rather than a complete, organized, scientific account. For example, causal knowledge certainly embodies a theory of certain phenomena; scripts may contain an implicit theory of the entailment relations between rule constituents; and book-learned, scientific knowledge certainly contains theories. Although it may seem to be glorifying some of these cases to call them theories, the term connotes a complex set of relations between concepts, usually with a causal basis. (Murphy & Medin 1985, 290).

Some theory-theorist even go to the extreme relativist view of claiming that people with radically different theories do not share the same concepts at all and that it is impossible to describe the concepts of one theory to another; the two conceptual systems are claimed to be incommensurable and their respective languages untranslatable (Carey 1991). Carey cites as examples the 17th century physicist thermal theories. They fused together the concepts HEAT and TEMPERATURE, but for the scientist of today these two concepts "[...] are two entirely different types of physical magnitudes; heat is an extensive quantity, whereas temperature is an intensive quantity" (Carey 1991, 463). According to Carey, these differences in concepts must be explained by the underlying theory.

> The only way to distinguish these two alternative representational states of affairs (false belief in perfect correlation and absence of one or the other concept) from conceptual nondifferentation is to analyze the roles that the concepts played in the theories in which they were embedded. (Carey 1991, 464).

Another example cited by Carey is the incommensurable differences in child's and adult's concepts of weights and densities of objects. For the adult mind it is, we are told, impossible to understand how the child could think that a particular piece of Styrofoam is weightless or that the weight of an object changes when it is turned on its side. (Carey 1991, 468.)

> The child's language cannot be translated into the adult's without a gloss. One cannot simply state the child's beliefs in terms of adult concepts - the child believes that air is not material, but the "air" in that sentence as it expresses the child's belief is not our "air", and the "material" is not our material. (Carey 1991, 484).

The theory-theory has a great advantage in explaining what might have gone wrong in the feature-based theories, namely the classical and the prototype. These theories rely on features to explain concept's identity, but they have nothing to say *why* certain features are chosen and *why* they are more important than others. Concepts are grouped together by their similarity, but similarity itself is left undefined. A toy-tiger is very similar to a real tiger; it probably has many of the features that a real tiger has: four legs, stripes, tail, teeth, etc. In this regard, a toy-tiger is similar to a tiger, but a toy-tiger is not a tiger. By properly choosing features one can make almost any object similar to another. Plums and lawnmower are very similar if we note that both weight less than 10 000 kg., both did not

exist 10 000 000 years ago, both cannot hear well, both can be dropped, both take up space, etc. It seems that the notion of similarity needs to be constrained and this could be accomplished by a theory. (Murphy & Medin 1985, 291.)

> The problem with the abstract notion of similarity is that it ignores both the perceptual and theory-related constraints on concepts, when in fact they are doing most of the explanatory work. [...] Thus, we attempt to provide part of the answer to how people choose relevant attributes for concepts and how they weight those attributes in their conceptual processes. (Murphy & Medin 1985, 292).

What features are chosen as being important could be explained by an underlying theory. A bird has wings and is light weighted, *because* it is an animal that flies; birds nest in trees, *because* it protects them from predators. Our knowledge about birds can help us deduce what features are correlated with the concept (Murphy & Medin 1985, 300). Similarly, all kinds of behavior is related to being drunk, such as jumping into a swimming pool with clothes on, dancing on the tables and howling on the streets. It is hard to believe that such features would be part of the structure of the concept. Instead, we can explain these features if we know that drunken people tend to do all kinds of silly things (Murphy & Medin 1985, 295).

2.7.1 The principle of psychological essentialism

People's judgement about what defines a category seems to be quite independent of features. There is increasing evidence that what defines a category is not some bundle of observable features, but some hidden essence (or a hidden property) or an appropriate internal structure. This principle is called psychological essentialism. Even 3-year old children have been noted to rely on essentialist thinking. When asked which resembles a pig, a piggy bank or a cow, children choose the piggy bank, but when asked which has the same kind of insides, they choose the cow; they could reasonably have chosen the piggy bank - why wouldn't two objects that look alike have the same insides? Children also think that it's what's inside that counts. If a dog is removed from its outsides, the fur, the tail and the rest, it still remains dog. But if the insides are tempered with, changing the blood and the bones and so on, then, according to 3-year old children, it seizes to be a dog. Children also tend to think that the essence determines how the object will potentially look and

behave in the future. A rabbit that is raised among monkeys will still have long ears as an adult and eat carrots, not bananas. (Gelman & Wellman 1991.)

It looks like some kind of a psychological essence determines an object's identity - not an n-tuple of features. Objects can go through drastic changes in their appearance, but still retain their identity - think of a seed developing into an apple or a caterpillar evolving into a butterfly. The theory-theory can provide an account of why people are essentialist: it is the theory that is the essence and it explains why certain features are considered to be characteristic for a particular concept at a particular moment and how those features are connected with each other.

2.7.2 Criticism of the theory-theory

The theory-theory seems to get away from the problems that we have encountered before: we might be mistaken about the features associated with a concept or might not know at all what they are. A theory can possibly explain what features are important for a concept. But, if we think more carefully, don't we have the same problems of ignorance and error for the theories? How do we know that our *theory* is the correct one? Theories can and do change over time as did the theories of disease. It seems that the criticism that the theory-theory was supposed to solve can equally well be applied against it. Our definition or a prototype of a concept can be mistaken, but so can a theory. One can explain the existence of humans by a theory of evolution or by a divine creation or by whatever, but if all these theories talk about the *same* thing, then the theory seems to be irrelevant for that thing. (Laurence & Margolis 1999, 47.)

In fact, in many, if not most cases we might not have the slightest clue about the theories behind our concepts. Most people do not know enough about the state-of-the-art theories of chemistry, physics or economics in order to talk about water, gravity or inflation. The point is that they don't need to.

> The suggestion seems to be that the difference between prototype theories and theory theories is that the latter entail that having a concept involves knowing the explanation of such correlations [...]. But, if so, it seems that theory theories set the conditions for concept possession impossibly high. I'm pretty confident

that being liquid and transparent at room temperature are correlated properties of water. But I have no idea *why* they are correlated. Notice, in particular, that learning that *being water* is *being H$_2$O* didn't advance my epistemic situation in this respect since *I don't know why being liquid and transparent at room temperature are correlated properties of H$_2$O*. Do you? (Fodor 1998, 118, footnote 26).

The problems of ignorance and error haunted definitions and prototypes, but, as can be seen, they haunt equally well the theory-theory; it is a descriptivist view and hence a target for Putnam's criticism.

Theory-theory relies on the principle of psychological essentialism, but psychological essentialism itself does not entail that the essence must be a theory. All that is needed is that there is some sort of an essence that people know and rely on when categorizing and explaining the behavior of objects. Therefore, this essence could equally well be, for example, a classical definition.

The notion of an essence i[s] the hypothesis that objects possess constitutive natures that make them what they are, that such an underlying nature is *distinct from* but *responsible for* more obvious external features. Having stripes is not essential to being a tiger, but it may be the direct consequence of having a tiger essence. Positing that there are essences does not mean identifying what those essences are; in fact, essences are typically hidden and unknown. (Gelman & Wellman 1991, 242).

Theory-theory should then provide solid arguments for why the essence of a concept is a theory and not something else. But the problem is that theory-theorists have not come forward with a precise specification of what a theory even is. Murphy and Medin "[...] agree that theories are made up of concepts (to a great extent) and urge that this fact be employed in our theories of concepts" (1985, 313). But what does the expression "to a great extent" mean and, by the way, why is it bracketed? If theories are nothing else than concepts, then it is hard to see how in the end the theory-theory differs from other theories. If, however, theories are something else than concepts, then we are not told what they really are. Although at first glance appealing, the theory-theory needs to come up with a precise account of what is meant by "theory". (Fodor 1998, 117.)

2.8   The atomistic theory

The final theory of concepts we will consider is the atomistic theory. The motivation behind the atomistic theory is the failure of the descriptive or structural theories of concepts. We have seen again and again the downfall of descriptions: concepts aren't definitions, nor prototypes, nor theories. This isn't to say that they have nothing to do with concepts. There certainly are prototype effects, maybe some definitions, and all kinds of idiosyncratic and cultural information as well attached to concepts. Someone's concept MONEY can include information that money makes the world go round or that it is the root of all evil. Money can be a symbol of success and power. Someone, say an expert, might even invent a bullet-proof definition for MONEY. Someone can have a theory of why money makes the world go round or why it doesn't. All types of descriptive information is associated to concepts. The point is that *all* this information seems to be *collateral - it is not constitutive* of concepts' identity. (Laurence & Margolis 1999, 64.)

The atomistic theory takes an entirely different approach to concepts. Since any type of description attached to concepts is a target of Putnam's criticism of ignorance and error, the atomistic theory claims that concepts are completely unstructured. Concepts, especially most lexical concepts, according to it are atomistic.

> [...] 'What is the structure of the mental representation DOG?'
>   And my answer will be that, on the evidence available, it's reasonable to suppose that such mental representations *have no structure*; it's reasonable to suppose that they are atoms. (Fodor 1998, 22).

The view that concepts are atomistic applies uniquely to lexical concepts. Phrasal concepts patently have structure, which simply is the constituents of the concept, that is, the concept BLACK CAT has as its structure the concepts BLACK and CAT. The important claim is that lexical concepts do not have any structure at all - they are primitive.

In the end, the atomistic theory is very close to the classical one. The only difference is that according to the classical theory the primitive level of concepts is supposed to be very small and simple, whereas the atomistic theory lifts up this level - it lifts it up all the way to the lexical level of language. This is the primitive level after which concepts do not

decompose any further. Therefore, the primitive level, according to the atomistic view, contains a massive amount of concepts and is far from simple.

The advantage of the atomistic theory is that it is safe from Putnam's criticism, because it is free to postulate any amount of structure/description to concepts; this structure is irrelevant, so we can be mistaken or ignorant about it, but still be able to possess a concept. But, how is concept's identity determined if it isn't by its structure? In other words, if the structure does not matter, then how can atomistic concepts, such as CAT and DOG, differ? The answer is that concepts work only for reference determination. Concepts are said to be triggered or activated by the appropriate things in the world. DOORKNOB is triggered by doorknobs, CAT is triggered by cats and so on. The only thing that matters is that the concept is in the right mind-world relationship. This relationship is causal: CAT expresses cats, if it is activated only by cats and not by carburetors. This is what determines concept's identity - not some structure, but a correct relationship to things out in the world. (Laurence & Margolis 1999, 60.)

2.8.1 Radical nativism

Primitive concepts, according to the atomistic viewpoint, are also innately known. So, in order to know which concepts belong to the innate, primitive base, we should check which lexical concepts decompose, that is, have definitions. Concepts that have definitions are not innately known, because they can be built from the primitive base. But, because there are practically no definitions for lexical concepts, it follows that most, if not all, lexical concepts are innately known. Hence, the atomistic theory leads to radical nativism. (Fodor 1975, 124-156; Fodor & al. 1980; Fodor 1998, 124.)

The conclusion that primitive concepts are known innately follows from a theory of how concepts are learned. The only learning method that can explain concept learning is inductive learning (Fodor 1998, 123). This is the process of constructing hypotheses to explain a given data and choosing the best hypothesis among competing ones. In order to learn a concept, say BACHELOR, the learner needs to find the correct properties/concepts that apply for things that fall under BACHELOR. After several trials the learner will choose the best hypothesis, which in this case might be that the concept BACHELOR

applies to UNMARRIED MALE. Inductive learning hence presupposes the following mechanical apparatus for learning (based on Fodor 1975, 42):

1. A format for representing the experiential data
2. A source of hypotheses for predicting future data
3. A metric which determines the level of confirmation that a given body of data bestows upon a given hypotheses

So, now that the learner has learned the concept BACHELOR from UNMARRIED and MALE, we can ask how those concepts are learned? The same way. But this process cannot go on forever. Therefore at some level the learner must already possess concepts that haven't been learned inductively, but since inductive learning is the only known way to learn concepts, it follows that these concepts are not learned at all. Hence, inductive learning presupposes an innate language in which the hypotheses are couched - this language of thought, in which the representations and hypotheses are carried out and evaluated, must be semantically at least as powerful as any natural language that we can ever learn.

The upshot would appear to be that one can learn L [some natural language] only if one already knows some language rich enough to express the extension of any predicate of L. To put it tendentiously, one can learn what the semantic properties of a term are only if one already knows a language which contains a term having the same semantic properties. (Fodor 1975, 80).

2.8.2 Criticism of the atomistic theory

One undesirable and implausible upshot of the atomistic theory is the commitment to a massive amount of innate concepts. If a concept doesn't have a definition, then it must be primitive and hence innate. The search for definitions for lexical concepts has failed and so have all other efforts to find any empirical evidence that some lexical concepts might be more complex than others (see Fodor 1975, 124-156; Fodor & al. 1980). If this is the case, then the inevitable and bold conclusion is that most lexical concepts are innate, meaning that we are genetically endowed with concepts such as DOORKNOB, MAFIA, MUFFIN and WALKMAN. Somewhere something must have gone wrong in the argumentation.

Or maybe not. People do have a strong intuition that most concepts must be learned, including lexical concepts, but in the end this is nothing but an intuition. If one looks at the evidence, as I suppose scientist should do, then the conclusion does follow. Since some concepts must be primitive, assuming compositionality, and they cannot be learned, assuming inductive learning, then primitive concepts are innately known. Complex concepts are learned, but because there is no evidence that lexical concepts are complex, then they must be primitive and hence innate. After all, the argument against radical nativism is nothing more than a gut feeling and gut feelings can be false.

> The moral, then, is this: what is not definable must be innate. Most of us are inclined to assume that it just *can't* be the case that *all* concepts are innate; most of us have therefore thought that many - indeed, *very* many - concepts must be definable. This is a persuasive line of argument *if Empiricism is true*. But what if Empiricism *isn't true*? (Fodor & al. 1980, 282).

But the atomistic view is not without its problems. If concepts are individuated only by their reference, then the theory runs into old problems encountered long ago by Frege. In modal contexts the meaning of concepts cannot be solely their reference, because terms such as "Muhamed Ali" and "Cassius Clay" have the same reference, but not necessarily the same meaning. For example, a person might believe that Muhamed Ali is the greatest boxer of all times, but he might not believe that Cassius Clay is the greatest boxer of all times. The reason is that he might not know that, in fact, Muhamed Ali is Cassius Clay. If meaning is only reference, then the above beliefs should always have the same meanings, which they don't. Same problem applies to empty concepts, such as JEDI or SANTA CLAUS, both of which don't have references at all. If reference is all there is to meaning, then how do empty concepts get their meaning? The way out of this problem was to postulate descriptions/intentions for concepts. If this is the only way out, then the atomistic theory is in trouble. (Laurence & Margolis 1999, 69.)

Second problem is the problem of misrepresentation. It is natural to be sometimes mistaken about things we perceive. In a dark and a rainy night we might mistakenly believe that a cow on the field is a horse. So cows in this case trigger the concept HORSE, but in normal cases horses trigger it. If horses and cows (and possibly many other things) can trigger HORSE, then the concept does not make any difference between them; it stands in a causal relation to many other things than horses. But HORSE is not the same concept as COW. It

should be triggered only by horses and COW by cows and so on. (Laurence & Margolis 1999, 60.)

Atomistic concepts are claimed to be safe from the problems of ignorance and error, because they don't have structure. But I think this really is not the case. The previous problems already hinted that we can be mistaken and ignorant about the reference of a concept. Somebody might believe that Michael Jackson is the king of pop. This information and possibly much more is encoded in his concept of Michael Jackson, but it can be the case that he might still not know the reference of the concept. That is, he might not recognize Michael Jackson on the street. Moreover, he might mistakenly think that the pop artist Michael Jackson is the same person as the beer and wines expert called Michael Jackson. If this is the case, then the problems of ignorance and error are back again. One way out is to claim that not even the reference is constitutive of concepts, but if neither structure nor reference plays any role in determining the identity of concepts, *then what does*?

3. KNOWLEDGE REPRESENTATION

3.1 The role of knowledge representation

"There is a sense in which every computer program contains knowledge about the problem it is solving." (Torsun 1995, 113). Knowledge representation, a subtopic of AI, studies how knowledge is organized and processed, what kind of data structures an intelligent agent uses and what kind of reasoning can and cannot be done with the knowledge. Given the computational framework of the mind, it doesn't make a difference if we are studying the mind or a computer. In both cases we need representations and data structures to represent that knowledge. Any problem solving task presupposes some sort of a knowledge representation. The psychologically interesting question is to find out what type or types of knowledge representations the mind uses. (Partridge 1996, 55.)

For example, a mental lexicon can be organized in the memory as a list or as a set. These two different structures permit different operations. A list encodes the order of the words whereas a set, by definition, is ignorant of the order of the items. So, a list allows us to query the first word, the second word, the third word, etc. This is not possible using a set. Whether we use a set or a list depends on the psychological theory that is modeled. If, according to it, the order of words is important, then a list needs to be used, if not, a set is acceptable. If a set is enough, then we need further precision of how to implement it. Using a hashtable would guarantee a constant search time (Weiss 1998, 553). Questions like these need to be addressed by the knowledge engineer, but after a certain point of precision some programming details are irrelevant to the psychological theory that is being modeled; whether we use `for` or `while` loops is most likely only an issue of programming technique. (Partridge 1996, 55.)

Knowledge representation has a vital role to play in AI, because it determines to a very large extent what kind of reasoning can be done with the knowledge, how fast it is, how much memory is consumed and how optimal and complete the algorithms that utilize the knowledge are. There are many different types of knowledge representations all having

their ups and downs. Torsun lists some criteria for evaluating them (Torsun 1995, 113-114; some items omitted and numbering changed):

> (1) *Semantics*: representations are usually intended as a medium for conveying meanings about some world or environment. A representation must therefore have a semantic theory which provides an account in which a particular representation corresponds to the external world or environment.
> (2) *Expressive adequacy*: What knowledge can and cannot be represented?
> (3) *Naturalness*: in some sense all knowledge can be represented in binary form. Such knowledge representation, however, will be very difficult to understand, update or reason with.
> (4) *Reasoning*: how sound and complete is the reasoning or inferencing mechanism? How efficient is the deductive process?
> (5) *Primitives*: what are the primitives (if any) in knowledge representation?
> (6) *Incompleteness*: how complete is the knowledge? Can the system reason with such a knowledge base?
> (7) *Revisable reasoning*: a good deal of what we know about the world is "almost always" true. How could such revisable reasoning be formalized and computed?
> (8) *Flexibility*: the ability to add new pieces of knowledge easily and freely.

From the perspective of this study, it is mostly the third point that bears importance: basically anything can be done with anything, but the question is how *natural* is the OO knowledge representation for representing conceptual knowledge? The following chapters will be briefly look at other knowledge representation systems from a purely psychological perspective before turning into the OO one.

3.2 Knowledge representation systems

3.2.1 Logic

Logic is probably the oldest form of knowledge representation. It is a well-known system with well-known semantics. "It provides a mathematically precise account of how a formalism can describe and reason about entities outside the computer" (Torsun 1995, 116). Logic is also a productive system that allows an infinite amount of complex expressions to be composed from simpler ones. Once we know certain facts, premises we can deduce what other facts follow from them. These inferences are rigid and truth preserving. The standard logic has also been extended to predicate calculus, intentional logic and nonmonotonic logic. (Niemelä 1993, 116-124; Torsun 1995, 115-124.)

Logic is a well-suited general-purpose formalism for expressing knowledge about many domains. It certainly has its merits in AI applications, but cognitive science is more concerned about how much it says about human reasoning. Unfortunately, not much. It looks like people don't reason with such rigor and precision as logicians would wish. Instead, human reasoning uses probabilities and analogies. For example, humans can apply the logical rule *modus ponens* much more successfully than *modus tollens* and also think that the conclusion follows more likely in *modus ponens* than in *modus tollens*. (Anderson 1995, 307-321; Thagard 1996, 35-39.)

In the Wason's selection task participants are given four cards. All cards have two sides. In one side there is a letter and on the other there is a number. They are then given a rule *if a card has an A on one side, then it has 2 on the other side*. Participants are finally asked to decide which of the four cards need to be checked necessarily to see if the rule is true (Johnson-Laird & Byrne 1991, 75):

A B 2 3

Only few participants get the correct answer, which is A and 3. Most people understand that turning A is necessary. This can be interpreted that they are using *modus ponens*; the rule was *if A, then 2* and since the premise A is true, it should be verified that the conclusion is also true. But this is not enough. The card 3 needs to be checked too, because if there is an A on the other side, then it would falsify the rule. Only few choose to check the card 3. On the other hand, many elect to turn the cards 2 and B, both of which are uninformative choices and cannot falsify the rule. (Wason & Johnson-Laird 1972, 173; Johnson-Laird & Byrne 1991, 75.)

The Wason experiment indicates that people are not formal logicians. Participants with a formal schooling in logic performed no better (Johnson-Laird & Wason 1972, 175). However, once people are given concrete examples from everyday life they perform extremely well. For example, if the rule is *if a person is in the bar, then he or she is over 21* and the cards are labeled IN-BAR, NOT-IN-BAR, 23, 18, then most people perform well and understand to check the card 18 also, that is to apply *modus tollens*. It seems that concrete examples are easier for people. If people were formal logicians they should

perform equally well on both tasks. (Wason & Johnson-Laird 1972, 191; Johnson-Laird & Byrne 1991, 77.)

All in all, logic seems to be a prescriptive, not a descriptive, science. It says how people *should* think if they were to think logically, but it has not much to say about how people *actually* think.

3.2.2 Production systems

Production systems (Nuuttila 1993, 125-133; Torsun 1995, 124-133) are a form of knowledge representation where the knowledge is represented in a list of rules. Rules are encoded in a set of *if-then* clauses. The rules contain conditions and actions that fire if the conditions are satisfied. The rule base can contain any amount of rules and corresponds to long-term memory, because its state rarely changes, whereas the short-term memory holds the data or the facts that are manipulated by the rules and its state changes continuously.

The interpreter is responsible for searching the rule base and finding the rule(s) that match the current facts, that is, the interpreter checks the state of the facts in the working memory and then searches the rule base, the long term memory, to find clauses that match the facts. In case where there are several matching rules, the interpreter must use some strategy to resolve conflicts. These include e.g., adding priorities to the rules, deleting rules already executed or using the rule that has been used most recently (Nuuttila 1993, 129).

The interpreter continues this match-select-execute loop until it finds no more rules that satisfy a condition, it encounters a halt command or exceeds the number of loops allowed. At this stage the interpreter stops and hopefully the goal of the system is accomplished.

A rule base might, for example, contain following rules (Anderson 1995, 248):

> *If* the goal is to drive a standard transmission car
> and the car is in first gear
> and the car is going more than 10 miles an hour,
> *Then* shift the car into second gear

Production systems offer many advantages. They are flexible. Adding, deleting or modifying rules is relatively easy. They are also modular. Each rule can be seen as an independent piece of knowledge. Thus, they are easy to maintain, since the knowledge engineer can fix a particular rule without touching other ones. This also makes learning easy, because rules can be added and modified as the system learns. As a downside, once the rule base gets very large it becomes virtually impossible to understand what the system does and to debug it[4]. (Nuuttila 1993, 130; Torsun 1995, 132.)

Production systems play also an important role in psychology. They have been used successfully in AI applications and also in modeling many domains of human cognition such as planning, decision making and learning (Nuuttila 1993, 131; Torsun 1995, 124; Thagard 1996, 51). Rules are quite flexible in the sense that they can correspond to all kinds of things, such as things in the world (*if* x is retired, *then* x is old), or to actions (*if* you are overspeeding, *then* break), or to linguistic rules (*if* x is a verb, *then* add /ed/ to get past tense). Rules can have multiple conditions and complex action parts. Compared to logic, rules are less strict, which is a positive property, because it allows to use rules that apply as defaults - rules that apply usually, but which can also tolerate some exceptions. For example, a rule *if x is student, then x is overworked* can be used as a default, but in some cases more appropriate rules could match, e.g. *if x is a student and taking only easy courses, then x is not overworked*, without creating a contradiction in the system. (Thagard 1996, 43-57.)

Moreover, the current widely-held Chomskian theory of language processing presupposes an innate Universal Grammar (UG) that contains the rules that generate all possible natural languages. Language learning is a process where the children's task is to find which rules of the UG apply to their language environment. English, for example, contains rules such as *if regular verb, then add /ed/ to the stem to get the past tense form* or *if substantive, then add /s/ to get the plural form*. The important point is that language ability cannot be explained without rules of the UG. Not any type of rules will do; the rules need to have recursions to explain the constituent structure of natural language expressions. (Pinker 1994.)

3.2.3 Frames and scripts

Another approach to representing knowledge is frames and scripts (Minsky 1975; Schank & Abelson 1977, 36-68; Hayes 1979). A frame is simply a data structure that holds pieces of information together in its slots. The contents of the slots individuate a frame by giving it its characteristic values. An example of a frame can be the frame named J.W. Smithy (Torsun 1995, 139):

> Marital status: married
> Number of children: None
> Married-to: Silvia
> Skill: Teaching
> Age: 45

As can be seen, different slots can get different types of values; some can be numbers, some strings, some can even be other frames as in the Married-to slot, which can refer to a frame named Silvia. Slots can also have default values: the number of children for example could be zero by default. Slots can also be empty and certain restrictions can constrain them. The age slot, for example, could only accept values in the range [0,150]. Frames may also have an IsA link, which makes a frame a subframe of another frame.

Concepts also encode information about the sequence of events. The RESTAURANT concept might have slots such as ENTERING, ORDERING, EATING and EXITING. Schank and Abelson (1977, 36-68) call these kinds of concepts "scripts". Scripts are basically the same thing as frames, but the important difference is that they concentrate on the order of the attributes. A script is a structure that encodes knowledge about the appropriate sequence of events in a stereotypical situation or a story. The RESTAURANT script encodes information that one cannot eat food without having ordered the food first and one cannot exit a restaurant without first having entered it.

When hearing stories people generally have no difficulty in filling in the blanks that have been left out. Consider the following stories (Schank & Abelson 1977, 38-39; numbering changed):

> 1. John went to a restaurant. He asked the waitress for coq au vin. He paid the check and left.

2. John went to Bill's birthday party. Bill opened his presents. John ate the cake and left.

Both of these stories are understandable, because we have general knowledge about the world that fills the details left out. We know that John was sitting at a table and probably had a look at the menu before ordering. We also know that you are expected to pay the check before you leave a restaurant. Without the script we might be wondering why he paid a check and what on earth is a waitress. Similarly, we know that it is a custom (in some cultures) to bring presents and to eat cakes at birthday parties. Storytellers can leave out this information, because they rely on people knowing it. In fact, stories get very boring if they describe every minute detail.

Frames and scripts are capable of representing the kind of conceptual knowledge that intelligent agents possess. Logic and production systems are interested in reasoning and finding the correct solutions. Frames focus instead on the fact that we possess a vast amount of conceptual information about all sorts of things in the universe: cats, dogs, rooms, friends, relatives, enemies, cars, hospitals, politicians, laws, regulations, crime, revolutions, evolutions, birthday parties, restaurants, etc. Frame-based representation allows us to encode this knowledge in a manageable way and in a way that pays attention to the semantics of the information: how things are related and associated with other things. People know what to expect when entering a room or a car or a restaurant and they know how to act appropriately in particular situations, such as birthdays and funerals, because they have prestored knowledge about them. Frames and scripts try to explain what is the nature and form of this knowledge. How is it organized and accessed. In logic and production systems, all the conceptual information is encoded separately and is spread all over the system; they don't have much to say how the information is *represented* in the memory. (Hyvönen 1993, 134.)

There is psychological evidence of people possessing concepts with default values. Participants in a study were asked to wait in an office room while the researcher was finishing with the previous group. After 35 seconds the participants were taken to a seminar room and asked to write down what they remembered about the office room. Participants remembered well items that belong to a typical office room: chairs, desk, posters and shelves. But only few recalled that there was a bulletin board and a skull. Some

claimed to have seen books and pens, which there were not. The study is interpreted as showing that participants have a concept OFFICE ROOM with default values that effect recall. If people don't remember all the details of a new office room, then the default values of a corresponding frame stored in the long-term memory are used to fill in the blanks. (Brewer & Treyens 1981.)

Similar type of results was obtained for scripts. Participants read several stories after which they were asked to recall what kind of events took place in the stories. Participants recalled many of the typical events, but they also listed typical events that were not mentioned at all in the stories. In some of the stories actions were deliberately made to happen in an unusual order. For example, in a story that takes place in a restaurant, the bill might be paid before ordering the food and the menu read just before leaving the restaurant. When recalling these stories participants put many of the events back in their normal order. All these experiments point to a conclusion that people represent prototypical sequences of event concepts. These scripts effect recall to the effect that they fill in information that is otherwise missing and also to the effect they can even distort memory. (Bower & al. 1979.)

Frames can also account for a phenomenon called spreading activation. If one concept is activated, it spreads its activation to other concepts that are close to it semantically. A chair can activate other concepts close by, such as desk, living room and furniture. Computationally this can be done with frames. Once a frame is activated, it can spread the activation by activating other frames that are connected to it in the slots or in the IsA link. And when these associated frames activate other frames in turn, the activation will spread like a wave. (Anderson 1995, 183; Thagard 1996, 63.)

It is quite evident that frames and scripts are very close to the OO knowledge representation and also to other types of data structures such as records in Pascal and structs in C. In fact, OO languages can be seen as a computational implementation of frames (Hyvönen 1993, 139). Classes are equal to frame descriptions and objects are equal to frame instances after filling in the slots. As frames, objects can have default values. Both have a link to the superordinate level, that is, they both have an IsA relation.

In general, frames and objects are the same thing. To puzzle things even more the term "schema" has also been used by psychologists to refer to frames and "event schema" to

refer to scripts (Anderson 1995, 154, 161; Partridge 1996, 74). For the sake of simplicity this study will equate all these terms. The question of terminology, whether to talk of frames or objects, has more to do with a viewpoint - the level at which we look at things (Partridge 1996, 74). Any knowledge representation system can be implemented at a lower level with the same programming language. A frame-based knowledge representation can be implemented in an OO programming language, but it may as well be implemented in a procedural language. The procedural and OO languages might end up being implemented with the same lower level language. In the final stage all knowledge is just binary digits. What matters is how things are seen from a higher viewpoint. Here objects and frames are both seen as tools for representing knowledge; how they are implemented in the end is irrelevant. Hence, all the findings that suggest an existence of frame-like representations equally support the OO knowledge representation.

The following chapters are devoted to the OO knowledge representation, because it contains a new paradigm that offers features not found in frames. I will also talk more about scripts when implementing the theory-theory, because it will point out a weakness in the entire OO knowledge representation.

3.3   The object-oriented knowledge representation

"The *object-oriented* approach to programming is based on an intuitive correspondence between a software simulation of a physical system and the physical system itself" (Abadi & Cardelli 1996, 7). The paradigm rose in the seventies. In traditional paradigms programs consist of variables and functions that take those variables as arguments and return new, changed variables. In the OO paradigm programs consist of standalone objects, that *incorporate* variables and functions that manipulate them. As a result, the behavior of a certain object is built into it - it no longer needs to be passed as a variable to a function somewhere else in the code in order to change its state; this can now be done by directly manipulating the object. The advantages can be summarized in the following three features (Abadi & Cardelli 1996, 8):

- The analogy between software models and physical models.
- The resilience of the software models.
- The reusability of the components of the software models.

Reusability and resilience are issues of programming techniques and are not of interest here; it is the first feature, the analysis, that we will concentrate on.

A great difference between programs designed with objects, as compared with traditional paradigms, is that the programs are said to resemble conceptually the actual problem domain that is being analyzed: the components in the program are equal to the components in the real world (Abadi & Cardelli 1996, 7; Campione & al. 2001). This changes the way programs are being designed. In a procedural language the problem domain is first analyzed, then it is left to the programmer to decide how it can be simulated in a computer. In OO languages the actual design of the program is more isomorphic to the components and their behavior in the real world - more precisely, to those components and behaviors that result from a certain analysis, which of course varies from person to person. Another advantage is that the programmer and the people who need to use the program can communicate using the same concepts, because the components of the program and the real world are the same.

As an example, say a programmer needs to make a program that simulates the behavior of animals and plants in a forest. In a procedural language this would most likely be accomplished by creating variables of some sort for each kind of animal and plant and possibly grouping these variables with tables or linked lists, e.g. all owls could be in a table and all the flowers in some other table. The actual behavior of the inhabitants of this forest is accomplished with functions that do something particular, for example, moves its argument from one tree to an another or makes the passed argument bark. It is up to the programmer to make sure that an owl is not passed to the bark function or a dog to the fly function, resulting in weird behavior. A code like this can end up being very complicated and difficult to maintain - also known as 'spaghetti code'. If a change is needed, the programmer needs to find the place in the code where this particular change needs to be done and to make sure that the change does not damage other parts of the program. This model takes no advantage of the fact that many objects in the forest share some features: all animals move, have weight and color, etc.

The OO style takes advantage of common features in objects and no difference is made between variables and functions. All animals could be derived from a common superclass

Animal, which would incorporate common features shared by all animals. The same way all plants could have a common superclass. The individual animals would all inherit from the Animal class: for example, an owl would automatically inherit some behavior common to all animals, but on top of it, it would add its own behavior, flying, big eyes, etc. This guarantees also that an owl's behavior is incorporated in the object itself and therefore cannot be accidentally made to behave in a wrong manner. The programmer calls the move method of an owl object, which will result in the owl moving in its own specific manner, probably by flying. A similar call to a snake object will move the snake, probably by crawling. Making an owl crawl is now impossible, because the data and the functions are all wrapped in one package.

> The big difference introduced with the object-oriented style is that initial representational commitments are made, more or less equally, to both function and data. The two are identified, developed and combined on a more or less equal footing; neither has clear precedence. Furthermore, the computational model is based on representations of the "objects" identified in the theory. (Partridge 1996, 62).

Another way of seeing how programs model the world is noticing that originally programs were written in a machine language, which soon proved to be inhumane. Higher-level human readable languages with compilers were invented. This way more and more machine language instructions could be achieved with fewer and fewer higher-level language instructions. The higher-level instructions were just shortcuts that still pointed to the underlying machine architecture, but as the languages evolved, the language constructs no longer pointed to the machine, but instead to the world outside the computer. Objects are not only shortcuts of machine instructions - they are computer simulations of real objects.

3.4   Common concepts used in object-oriented languages

The OO language community is very diverse. There are many different languages and there is no agreed upon common standard constructs, notations or a formalism for representing them. The following is a very rough overview of common concepts used in the OO paradigm. There are many languages that have many different features, but the following concepts are found in most of them. The idea is just to set up some common

terminology and to see the difference between class-based and object-based languages. The review is based mostly on Abadi & Cardelli (1996), but also on Arnold & Gosling (1996), Campione & al. (2001), Hyvönen (1993), Lassila (1993), Niemeyer & Peck (1998), Partridge (1996), Taivalsaari (1997) and Torsun (1995, 147-160). The reality is much more diverse: there can be classes inside classes; classes that cannot be subclassed; abstract classes that cannot be instantiated; classes that only act like interfaces...

3.4.1 Classes

Classes are the fundamental building blocks of objects. A class itself doesn't do anything - it is a blueprint, a general description of objects created from it. A class defines fields to hold data and methods that manipulate the data. Fields and methods are collectively called attributes. Once a class is instantiated you have objects. Objects vary in what the values of their fields are, but what attributes they all share is defined by their class, which is also called their signature or type or interface.

As an example, the class Planet can define fields weight, temperature, distance from the Sun, number of moons, etc. All objects instantiated from this class have these attributes with some values. A programmer can instantiate nine planets from this class to model our solar system setting the values correctly for Venus, Mercury, Earth and the rest.

Here is a declaration of simple class called Point, which represents points on a two-dimensional plane (the pseudo-notation is taken from Abadi & Cardelli (1996)):

```
class Point is
    var x: Integer := 0;
    var y: Integer := 0;
    method getX(): Integer is
        return self.x;
    end;
    method getY(): Integer is
        return self.y;
    end;
    method setX(x: Integer) is
        self.x := x;
    end;
    method setY(y: Integer) is
        self.y := y;
    end;
end;
```

The Point class has two integer fields, x and y, that mark an object's current position. Both are initially set to zero. It has also two methods to retrieve those values and two methods to set the values. With the set methods objects can be moved on the plane. All point objects created from this class have these attributes. The x and y values can, of course, (and should) differ between objects.

3.4.2 Objects

OO languages, like the name suggests, are based on objects. An object can be made to represent anything: a file, a rock-band, a number, a court order, Socrates, a birthday party, earthquake, etc. Objects are behavioral units that a programmer can manipulate in order to achieve the intended behavior. Objects are construed of fields and methods to manipulate those fields. An object's state at a certain moment is the values of its fields.

From the Point class objects can be instantiated to represent points on a plane. The following instructions create two objects from the Point class:

```
var firstPoint: InstanceTypeOf(Point):= new Point;
var secondPoint: InstanceTypeOf(Point):= new Point;
```

Now that there are two objects, their methods can be called. For example, the firstPoint can be moved to the position 3,3:

```
firstPoint.setX(3);
firstPoint.setY(3);
```

The firstPoint occupies now the position 3,3 while the secondPoint still remains at its original default position 0,0.

3.4.3 Inheritance and subclassing

Classes don't always have to be built from a scratch. A class can inherit behavior from another class, which is called its superclass. By subclassing the new class automatically

inherits attributes from the superclass, but it can also add more attributes. It can also override the inherited attributes, if it so wishes. For example, the class ColorPoint can inherit attributes from the class Point and add new attributes in the new class. Both classes will share attributes, but ColorPoint will have more functionality than Point.

The following code declares a class ColorPoint that extends the Point class:

```
subclass ColorPoint of Point is
    var color: Integer := 0;
    method setColor(c: Integer) is
        self.color := c;
    end;
end;
```

The field color was added to represent the color of the point (encoded as an integer) and the method setColor for setting the color. Because ColorPoint subclassed the Point class, it has all the same attributes as Point has. An instance created from the ColorPoint has the fields x and y as any object of the class Point. Additionally, it also has the field color. ColorPoint can now be instantiated with an object called darkPoint:

```
var darkPoint: InstanceTypeOf(ColorPoint):= new ColorPoint;
```

It is now completely legal to make the following calls:

```
darkPoint.setX(5);
darkPoint.setY(5);
```

The darkPoint object behaves just like the previous point objects. It would now be in the position 5,5.

When subclassing the new class does not necessarily need to inherit any of the attributes of the superclass. A subclass can, if it so wishes, override all the attributes and inherit nothing but the type. Therefore, in OO languages one needs to distinguish subclassing from inheritance. (Abadi & Cardelli 1996, 15.)

3.4.4 Subsumption

Subsumption, also known as polymorphism, is the ability to use a subclass where an object of its superclass is expected. As all subclasses of a given class share the same attributes, i.e. the same type, similar behavior can safely be excepted. The opposite is not possible, since the superclass does not (necessarily) have the same attributes. If a class Dog defines a method bark, then all of its subclasses, e.g. GoldenRetriever, can bark and can therefore be used where an object of a class Dog is expected. A GoldenRetriever is thus subsumed to a Dog. And so are color points to points; a ColorPoint instance can be used where an instance of Point is expected.

3.4.5 Taxonomies - Class hierarchies

A class can have infinitely many subclasses, which can in turn be subclassed. This creates a hierarchy of classes, a taxonomy, starting from the top-most class or classes and expanding like a tree. Subclasses add attributes on top of all the inherited ones thereby specializing their behavior. A language can have one class from which all objects must initially be subclassed (e.g. in Java the Object class) resulting in a taxonomy where there is only one tree. Some languages (e.g. C++) allow several uppermost classes to be specified by the programmer resulting in many trees, a forest. Graphically, class hierarchies can look as in the figure below:
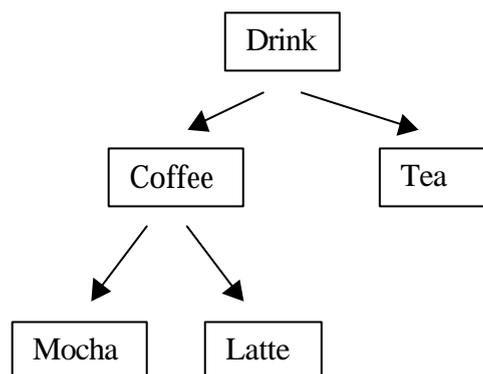
Subclassing creates a hierarchy of objects and classes. The top-most class is the most abstract, that is, it has the least number of features, whereas the bottom-most classes - the leaves - are most inclusive; they have inherited all the features on the way down. So, a Mocha class has whatever attributes are in Drink, plus the attributes in Coffee and finally the attributes that are unique to Mocha objects.

3.5 *Object-based and class-based languages*

The review has so far concentrated on class-based languages. Before any objects can be created, it is mandatory to first create a class description. Once a class exists, objects can be created from it. In object-based languages (Abadi & Cardelli 1996, 35-50; Taivalsaari 1997), also called prototype-based languages, a different route is taken: in these languages objects are created directly - the notion of classes is dropped. Instances are immediately fully operable, concrete objects. This makes the language much simpler and it also adds many nice features. The two points can now be created again using the object-based notation:

```
object firstPoint is
    var x: Integer := 0;
    var y: Integer := 0;
    method getX(): Integer is
        return self.x;
    end;
    method getY(): Integer is
        return self.y;
    end;
    method setX(x: Integer) is
        self.x := x;
    end;
    method setY(y: Integer) is
        self.y := y;
    end;
end;
```

The secondPoint object could be created exactly the same way, but this would be repetitious. In class-based languages objects are generated from the class with the **new** operation. The object-based languages have a different mechanism for instantiating objects. First of all, it is called cloning. New objects are created by making copies of objects that already exist - this is similar to what happens in real nature, hence the name:

```
var secondPoint := clone firstPoint;
```

This will create a new point named secondPoint. It shares the same attributes as the firstPoint, but with an independent state. Any amount of objects can be cloned from the firstPoint. SecondPoint is now also a fully blown object ready to do its work; there is no need for the **new** operation. Method calls for secondPoint don't effect the firstPoint and

vice versa - both objects have their own states. There are now two objects in the 'universe', but no classes. In the class-based version the universe consisted of two objects plus the additional platonic class. From an ontological point of view, the object universe is much simpler.

Cloning itself is not useful unless there is some way to mutate the new objects, that is, to modify their behavior as was done with subclasses. The prototype-based languages offer more choices for inheritance than the class-based languages. There are two ways to achieve inheritance: delegation and embedding. Delegation is a mechanism that delegates behavior to other objects, whereas embedding is a mechanism that copies behavior from other objects to the new object. The following creates the colorPoint object in two ways to illustrate the difference. First by delegation:

```
object colorPoint child of firstPoint is
    var color: Integer := 0;
    method setColor(c: Integer) is
        self.color := c;
    end;
end;
```

Then by embedding:

```
object colorPoint child extends firstPoint is
    var color: Integer := 0;
    method setColor(c: Integer) is
        self.color := c;
    end;
end;
```

The result is similar whichever mechanism is used: the universe now contains three objects[5]. Two points and one color point. The color point behaves as the two points, that is, it has the same attributes as the points. By subsumption it can be used anywhere the firstPoint can be used. The result looks the same as in the class-based version. But there is a subtle but an important difference. When the inheritance is done with delegation, the situation is analogous to the class-based version. Since the subclasses, or child objects, inherit their behavior from the superclass or parent object a change in that code will effect all the subclasses or child objects. If the getX method is modified, it changes the behavior of the colorPoint subclass as well and similarly for the colorPoint child object. So, the delegation inheritance mechanism is similar to the class-based one, but the embedding

mechanism behaves differently. When an object embeds the behavior of a parent, it is physically embedded in the object. The result is that changes in the donor object have no effect at all on its children. This possibility is not available for class-based languages.

The prototype languages have still more to offer. In fact, it is not mandatory to inherit all the features of the parent object. An object can selectively inherit useful features from its parent. Still more, objects can even delegate behavior selectively to other objects. For example, the following point delegates its x coordinate to the firstPoint and the y coordinate to the secondPoint:

```
object copycat is
    method getX() is
        return delegate firstPoint.getX();
    end;
    method getY() is
        return delegate secondPoint.getY();
    end;
end;
```

Embedding could have been used also, but this would not have created an object that 'copies' the two coordinates. The embedded object would have had the same values as the two points only at the time of the creation. Delegation allows us to create an object that can copy behavior from wherever it sees fit.

The final point of object-based languages is that when using delegation the parent can be defined as an attribute as any other. As a result, it can be *dynamically* changed. Evidently, this creates an extremely dynamic environment where objects can undergo dramatic changes in their attributes. Here is the code to change the parent of the copycat object to colorPoint:

```
reparent copycat to colorPoint;
```

The copycat would now be a copy of the colorPoint. It wouldn't know anything anymore about the firstPoint or the secondPoint.

In summary, the object-based approach concentrates on direct manipulation of concrete objects. The notion of classes is dropped as being redundant - same results can be achieved with cloning and inheritance. Object-based languages have two different inheritance

mechanisms available to them, embedding and delegation. The delegation mechanism also offers the possibility to create objects from assembling attributes selectively from other objects. And finally, the most dynamic feature is the possibility for on-the-fly reparenting - a feature not available in class-based languages nor in frames.

4. OBJECT-ORIENTED IMPLEMENTATIONS

4.1   The classical theory

The class-based languages have a close affinity to the classical theory of concepts. A concept has a set of necessary and sufficient attributes that defines it. This is exactly what a class does - it defines the attributes of all objects created from it. If dogs are defined as objects that have a name, fur, legs, bark and wag a tail, then the following class can be build to represent the concept DOG:

```
subclass Dog of Animal is
    var name: String := null;
    var fur: InstanceTypeOf(Fur) := new Fur;
    var legs: Integer := 4;
    method wagTail() is
        // ...
    end;
    method bark() is
        // ...
    end;
end;
```

All objects created from the class Dog share these same defining features as required by the classical theory. They also posses their individual differences: some dogs can be very heavy, some have unique styles of barking, some are faster, etc. The fact that attributes have values allows for little individual differences. Note that the class extended the Animal class. Due to this, all dogs are animals. ANIMAL is a superordinate category of dogs; whatever defines animals, defines dogs as well. If animals have weight, age, color and movement, then so do all dogs, because they inherit the attributes. In OO terms this means that by subsumption a dog is an animal - Dog instances have all the behavior that Animal instances do.

Furthermore, Taivalsaari (1997) notes that class-based languages correspond precisely to the ancient view of categories held by Plato and Aristotle, according to whom the essence of a category is defined by the defining properties and distinguishing properties as follows (Taivalsaari 1997, 2):

essence = genus + differentia

That is, a category (essence) is defined by its parent category (genus) and distinguishing properties (differentia). This is the classical theory: cats and dogs are both animals, which is their genus, but both must have some defining features that individuate them. "This corresponds precisely to the idea behind traditional class-based object-oriented programming in which a class is defined in terms of its superclass (genus) and a set of additional variables and methods (differentia)" (Taivalsaari 1997, 2).

Then how about the object-based languages? The difference of class-based and object-based languages was that the latter dropped the notion of classes; there are only concrete objects. But the whole idea of concepts is to store information pertinent to all members that fall under the concept. Once there are categories, knowledge can be added, deleted and modified in them and this knowledge then applies to *all* objects that belong to the category. If there are only objects, then we need to update each and every one of them, after learning something new about the category.

Given this fundament cognitive utility of concepts, it clearly follows that the prototype-based languages lack something. Namely the fact that there is no language construct at all that refers to categories, a set of objects. There are no classes, only individual objects. How can we refer to categories? A language without categories would be like a language where you would have the words "Plato", "Aristotle" and "Nietzsche", but not the word "philosopher".

This lack has been noticed in the prototype-based languages, and, consequently, concepts like family, traits and prototypes have been introduced to refer to groups of objects (Abadi & Cardelli 1996, 47; Taivalsaari 1997). A prototype, in this case, means a prototypical object that is not a usual object. There are now two types of objects: normal objects and prototypes. Prototypes are special objects that cannot be used where normal objects can; they are meant to be only blueprints from which normal objects are cloned. Thus, a special object, named a prototype, takes the role of classes. But the classical theory says nothing of this kind. Categories are not identified by one example of the category. Instead, a category is an abstract description of the items that fall under it.

Traits, on the other hand, are abstract descriptions of objects. They are safe from the above criticism and are good descriptions of categories like classes. But, how in the end are they different from classes? From a purely psychological perspective, speaking about classes or traits makes no difference - they both represent classical categories and do it equally well.

The problems encountered by the classical theory are immediately at hand for the class-based languages and also for frames. Definitions are never perfect. Classes encode defining attributes, therefore they are inherited by their subclasses. The problem is that there are always some unusual members that don't fit the tight definition. Chairs, by definition, have legs, but what to do with a beanbag chair without legs or an office chair with wheels? If they are instantiated from a subclass of Chair, then they *must* have legs - no excuses, no exceptions. (Partridge 1996, 71.)

One way around this problem is to add possibilities for canceling inheritance of some attributes, but this way we must abandon subsumption also. Another way is to have means for representing what is typically found in members. The problem, according to Partridge (1996, 71-72), is that there are no really good ways for deciding when canceling is allowed or how to measure typicality.

> A better interpretation of the relationship in question [inheritance] is that "birds typically have wings," or "birds usually have wings." But, then, what is the precise meaning of *typically* or of *usually*, […]?
> There is none. And once absolute definitional status is abandoned, we are catapulted into a limbo world of hazy meaning and multiple interpretations. (Partridge 1996, 72).

4.2   The prototype theory

The class-based paradigm does a good job in modeling the classical theory. It is probably because of this that it fails miserably in modeling the prototype theory. All instances of a class share the same attributes and belong equally well to the class. How could there be typicality effects? The classical theory cannot explain unusual members as has been seen. In addition to this, there simply just cannot be family resemblances when all objects share exactly the same attributes.

In order to achieve prototype effects, some kind of a mechanism must added to the language. The simplest possible option is to force the classes to include a query mechanism that returns the typicality of an instance, say, in the range of [0,1]. So, if every object has a method called something like **getTypicality()** we can query their goodness and, for example, sort the objects using their typicality values.

Another, albeit a much more complicated way, to achieve the same is to encode diagnosticity and salience values of each attribute as suggested by Smith & al. (1988). In their model, each attribute is given a diagnosticity value, which indicates the importance of that attribute; some attributes are more important than others - their family resemblance degree is higher - in determining the category of an object as demonstrated by Rosch & Mervis (1975). The values of attributes have different importances also; e.g. red is a more salient value for color than brown, because apples rarely are brown (Smith & al. 1988, 489). The figure below illustrates these values (based on Smith & al. 1988, 490):

Apple(A)		I_1	
1 color	red 25 green 5 brown	color	red 30 green brown
0.50 shape	round 15 square cylindrical 5	shape	round 20 square cylindrical
0.25 texture	smooth 25 rough 5 bumpy	texture	smooth 30 rough bumpy

The apple (A) on the left is the prototypical apple. The one on the right is an instance. To compare the instance with the prototype, Tversky's contrast rule is used for each attribute and then the results are added together. So, the similarity of I_1 to A is (Smith & al. 1988, 491):

$$\text{Sim}(A, I_1) = 1(25\text{-}5\text{-}5) + 0.50(15\text{-}5\text{-}5) + 0.25(25\text{-}5\text{-}5)$$
$$= 15 + 2.5 + 3.75$$
$$= 21.25$$

So, once we know the prototypical object and the diagnosticity and salience values we can count how prototypical an item is. This model assumes quite a bit though. First of all, we should know the prototype. Second, we should know all the diagnosticity and salience

values. But where would all these values come from? Well, there is no way to calculate them in the model. The same applies to the first, much simpler model. We have to assume that the programmer inserts the typicality values manually or that input devices somehow give them.

Objects in the model would still have a classical core. The typicality effects were just added on top. The result is a dual theory; categories are defined by their attributes, but can still be rated in terms of their goodness. For practical purposes this could be used, just as an example, in search results on an online furniture store. If customers search for chairs, it is probably advisable to present them a very typical chair first, instead of an electric chair or a wheel chair. Or if the search lists all the chairs available in the store, then it might be a good idea to sort them using their typicality: first in the list would be the most typical, normal chairs and the more exotic ones in the end.

How about the prototype languages then? Initially, it seems that they, as even the name suggests, should be more on a par with the prototype theory. The languages operate on concrete objects out of which one could be a prototype. As was seen above, this actually was one of the proposals to refer to a group of objects (Abadi & Cardelli 1996, 47); it is an interesting coincidence that such a similar solutions have been proposed in the context of programming languages and theories of concepts. But to talk of prototype-based languages as being psychologically more real (Taivalsaari 1997) just because of this, is based on an incorrect interpretation dubbed "*The Prototype = Representation Interpretation*" by Lakoff (1987a, 137; 1987b, 63). What the prototype theory says is, that there is an *open* set of features that are statistically more widespread in the category, not that there is literally one single object that represents the category. Although, initially the prototype theory was interpreted as meaning that the mind stores the prototypical example to represent the category, this is a misinterpretation demolished later by even Rosch herself:

> To speak of *a prototype* at all is simply a convenient grammatical fiction; what is really referred to are judgements of degree of prototypicality. Only in some artificial categories is there by definition a literal single prototype [...]. For natural-language categories, to speak of a single entity that is the prototype is either a gross misunderstanding of the empirical data or a covert theory of mental representation. (Rosch 1978, 40).

But there is a way in which the prototype languages could, in fact, model the prototype theory better. The fact that they are not tied to defining features allows them to create family resemblance categories - this is impossible for class-based languages and frames. And as Rosch showed, once you have family resemblances you have prototypes (Rosch & Mervis 1975). If we could only modify the languages, so that they would have a construct of a class/trait that would only function as placeholder for object. In other words, the class/trait would not dictate any attributes for objects - they would be *computed* by the runtime environment as explained below[6]. All the programmer would need to do is assign objects to certain categories. This way "knowledge about a correlation between properties is *computed* from an exemplar-based representation when needed, rather than *prestored* in the representation" (Smith & Medin 1981, 157).

Pinker & Prince's world (2) serves as an example of a family resemblance category:

    ABC
    ABR
    PBC
    AQC
    DEF
    DER
    PEF
    DQR

We could arrive at a world like this by first creating the ABC object. After that, a new object will decide to inherit behavior from it by delegating or embedding A and B. But the it will decide not to inherit the C property, instead it will replace it with the R property. The third object, PBC, does the same, but now for the A attribute; it replaces it with P. The last object will replace the B property with Q. Similar mechanism can produce the objects in the second category starting from DEF. Note that objects could even add more properties that are on the examples; an object could have the properties ABQCD for example.

There are now two family resemblance categories, each consisting of four objects and there isn't a single property held by all the members in either category. The family resemblance metric is used to compute the prototypes. The first category contains five attributes altogether with the following weights A=3, B=3, C=3, R=1, P=1 and Q=1. The second

category attributes contain the weights D=3, E=3, F=2, R=2, P=1 and Q=1. Each object's family resemblance degree is the sum of the weights of each attribute:

ABC	3+3+3	9	DEF	3+3+2	8
ABR	3+3+1	7	DER	3+3+2	8
PBC	1+3+3	7	PEF	1+3+2	6
AQC	3+1+3	7	DQR	3+1+2	6

From these values, a prototype for each category can be assigned. For the first one, it is ABC. For the second one, there are two prototypes, DEF and DER. The model thus allows us to create family resemblance categories and to compute the prototypes using a mechanism that is backed up by psychological theories. This is, then, a mathematically *precise* account for what is meant by "typically" and can by used to overcome the inheritance problems encountered above with class-based languages and frames without being catapulted into Partridge's "limbo world of hazy meaning and multiple interpretations" (1996, 72).

The model does not do the classification however; it is still left to the programmer to place objects in categories. In order to compute the cue validities of a category, the members need to be known already. But once the prototypes are known, the model could possibly be used for classification also, because if we encounter an unknown lonesome object, we could count its family resemblance degree in every category and assign it to the one where it scores highest. For example, PER would score 2 (1+0+1) in the first category and 6 (1+3+2) in the second. Surely, it feels more at home in the second category.

Another interesting thing with this model is that it can simulate Pinker & Prince's changing worlds. It could well happen that all the objects in the category would share the same attributes - that is, they would delegate exactly the same behavior. In this case, the distribution of each attribute would be the same. But we can go even to the other extreme - to the worlds (3) and (4); all objects would be completely different and hence their family resemblance degree the same.

The model can handle the continuum from chaotic worlds to family resemblance worlds and finally back to classical well-defined worlds, but it does have one weakness. If the objects are left to converge to a classical world, then all the attributes would have a similar

distribution and hence the same typicality values. But this is not psychologically adequate. As was seen in the experiments by Armstrong & al. (1983), participants could rate item's typicality even if the category was a digital, all-or-none, category. In this model, the typicality effects disappear once the category becomes classical. If this model has been at all honest to Pinker & Prince's theory, then this conclusion equally well undermines their version of the dual theory.

The prototype languages, then, provide a way for encoding family resemblance categories and a precise account of typicality (assuming that the model is technically feasible). Unfortunately, as appealing as it may first look, the model suffers from the same weakness as the theory being modeled: it does not explain the compositionality of thought. Prototypes encode the features typically found in members, but this is why they don't compose. Our model can encode the features typically found in pets and fish, but these features are not found in typical pet fish. Any family resemblance model suffers from this problem. When we need to create a new class from already existing ones, which attributes will it inherit? The new complex concept will inherit the attributes that just *happen* to be widely distributed at that time. What it should inherit is the essence of the concept - whatever it is...

These models have managed to provide models for the prototype theory with both types of languages. But probably the biggest weakness in the entire OO paradigm, is that it forces the design to start from the uppermost, abstract categories - the superordinate level - although psychological studies have clearly shown that this is not the case; categorization starts from the middle, the basic level, and later proceeds to the super- and subordinate levels.

Experts are people who know very high and low levels of categories. The process of becoming an expert is a matter of years. It takes time and lots of work to find higher and higher levels of abstractions. However, we all have to start at the intuitive basic level. But the OO paradigm does not allow this. Instead, it expects us to be natural born experts and to start immediately from the top level. But most likely we will get hierarchy wrong, because we can't be excepted to know in advance what it is. The result is that we will have to revise the class hierarchy several times:

If the implementation of a class hierarchy is started *a priori*, i.e., before a sufficient level of expertise has been reached, substantial iteration in the implementation of the library is inevitable, since later experience is bound to reveal generalizations and new abstractions that will necessitate changes in the superclasses. (Taivalsaari 1997, 8).

4.3   The dual theories

All the models for the prototype theory resulted in implementing the two types of dual theories. First, we saw two ways to add the identification procedure on top of the classical core. The second model modified the object-based languages in order to create family resemblance categories, which are in the middle of a continuum from classical to random categories, and offered also a procedure to extract the prototypes.

Both models suffer from the criticism directed against dual theories: there are not many people out there who still have hope of finding definitions and since prototypes encode the attributes that objects statistically tend to posses, they don't compose. Or they do, but the results are not semantically proper, like was seen in the pet fish example.

4.4   The neoclassical theory

According to the neoclassical theory, concepts are defined by partial definitions. The problem is that it was left unclear how to turn the partial definitions into full definitions without retreating to the classical theory. We have seen that OO approach copes perfectly well with the classical theory, so from that we can say that it copes equally with the neoclassical one, because the partial definitions are classical definitions. Since the neoclassical theory does not reveal what its difference to the classical one is, it is impossible to examine how the OO implementation might look like.

But just for curiosity we can try to see how Jackendoff rule "the causation of a state that endures over a period of time" could be programmed:

```
class Causative is
    method cause is (S: State, T: Time) is
        return S.endure(T);
```

         end;
     end;

The above loosely defined class is an example of what unites the causatives and also how they differ: the difference is in the values given to the cause method.

The interesting thing to note is that the example actually proves Fodor's counter arguments. If the differences in the causative expressions are the semantic markers, the values given to the cause method, then in all cases CAUSE remains the same. Fodor's argument was precisely that partial definitions use some concepts univocally and these concepts do not behave neoclassically. In the example above, CAUSE does not change its meaning and neither does ENDURE - they always remain the same. We could have built separate classes for ENDURE and CAUSE, but this would not help, because those classes would have some attributes that are univocal. Attributes in classes do not change - their values do. Hence, there are always some concepts that are not neoclassical.

4.5   The theory-theory

The theory-theory takes a different approach to concepts than the feature-based theories. The latter ones rely on attributes to explain what concepts are, whereas the theory-theory claims that concepts are embedded in theories. It is the theory that explains why objects have the attributes that they happen to have.

The problem was that we are not told what a theory is. Is it something else than concepts or is it just the way concepts are organized? If it is just the way concepts are organized, then perhaps class or object hierarchies could be a way to model the organization. The problem is that this would not actually be any different from the models for the classical or the prototype theory. All we would do is just *say* that the organization is the theory - there is no language construct that would represent the theory. It seems odd that the same model could implement both the classical and the theory-theory. If the theory is something else than concepts, then it is hard to see what it could be in terms of OO languages; if concepts are likened to classes and objects, then there is nothing else in the languages that could be the theory without being a concept. In OO languages, as the name implies, there are only

objects. So, it seems to me that the only way to model theories in the end is the class or object hierarchies.

I think the theory-theory is the best place to talk about scripts, because it is the only one of our theories that can explain why the features come in a certain order. Beneath scripts there can be a theory that explains why people typically order the food *before* eating it or why tickets are paid *before* seeing a movie. These attributes can easily be added in an object or a class as is done with other concepts, but neither language has anything to say about the order of the attributes. There is no way to prevent the method for exiting a restaurant from being called before the method for entering a restaurant or for making sure that food is not paid before it is eaten. Hence, OO languages provide no natural way to model scripts, event concepts.

The theory-theory relies on the principle of psychological essentialism - the observation that objects can undergo dramatic changes in their appearance, but still be treated as the same object. Take again the example of a butterfly. First, it starts its life as a caterpillar and later develops into an adult butterfly. At this stage, it probably has none of the visible attributes that it had as a caterpillar. An apple starts as a seed and later takes the form of a fruit. How can we model these kinds of metamorphoses?

The class-based languages have no natural model for this, because once an object is instantiated from a class it remains fixed throughout its life cycle. This is a problem for class-based languages, because not only can objects go through metamorphoses, categories can also be extended on the fly. If a man carrying heavy boxes decides to take a rest and sits on one of the boxes, that box becomes a chair. Or if a spoon is moved to an art gallery it becomes a piece of art. This problem has been noted in the context of frames:

> Similarly, there are always situations that do not fit any particular frame or schema very comfortably. If I sit on a log, it becomes, to some extent, a chair (therefore, a friend might say, "Can I share your chair?"). But this possibility is unlikely to be in any chair frame. As another example, is a large canopy on the lawn to be reasoned about using a room schema (rooms do not usually have grass on the floor, but they do have tables and chairs in and a ceiling overhead)? (Partridge 1996, 76).

With the notion of reparenting, the object-based languages can overcome these situations. The previous example of a box becoming a chair could be well modeled by changing the parent of the box object to a chair. In fact, with reparenting we can model any type and any amount of metamorphoses. An object can undergo drastic changes by reparenting itself, but it would still be the same object - as is required by psychological essentialism. Note that objects can still have some attributes that don't change, namely those that are not inherited. Prototype-based languages clearly win this round. Even if an on-the-fly change of the superclass would be possible for the class-based languages, this would not help, because using it will change the superclass of *all* objects created from that class, which is not what is wanted in these situations. All that is needed is to change the parent of the *one* particular box that ends up being a chair for a while - the other boxes must remain intact. Creating a new intermediate object from the class Chair does not help either, because then there would be *two* objects, one would be the chair and one would be the box. But this contradicts the principle of psychological essentialism. There is just *one* object that becomes a chair for a short while.

4.6   The atomistic theory

Among all the theories of concepts, the atomistic theory is a strange beast. According to it, concepts have no structure whatsoever. They are simply atoms. How can unstructured concepts be modeled with the OO approach? The answer is very short: they can't.

If concepts have no structure, then there is nothing to write into a class or an object. All our classes and objects would be empty. There is nothing in them and out of nothing comes nothing. Classes surely have different names, but there is nothing that individuates them. The situation becomes even clearer when inheriting is considered. Say, that there are the following primitive concepts: BLACK, BROWN, CAT and MUFFIN. The complex concept BLACK CAT could be subclassed from the classes Black and Cat as is done below.

```
subclass BlackCat of Black, Cat is
end;
```

How would this class look like? It would not look much like anything, because there is nothing in it. It would not have inherited any behavior, because there is, again, nothing to inherit. Even worse, it could have subclassed *any* classes. So, if BlackCat is created from Brown and Cat, it would be identical to the previous BlackCat. And, in fact, so would the BlackCat created from Brown and Muffin. All these classes would be the same. Surely something has gone badly wrong. A concept composed of BLACK and CAT is not the same concept as the one composed of BROWN and MUFFIN.

The atomistic theory is a devastating blow to the OO knowledge representation. But so it is for any knowledge representation tool that relies on structure, because as soon as we add something into a class or a frame or a script or whatever, there is structure and structure won't do in this case. The second problem is the sheer amount of primitive concepts. If all lexical concepts are primitive, then the number is somewhat close to the number of words in a language. Any model that tries to handle this will most likely run into severe problems of computational complexity. So, if the theory is correct, then it maybe cannot be handled computationally at all (Fodor 1975, 155-156; Salo 1999).

5. SUMMARY AND SUGGESTIONS

This study has reviewed six major theories of concepts. None of them has a final word to say about what concepts are; each has its own advantages and weaknesses. The study has also covered the OO knowledge representation and the difference between the traditional class-based languages and the new prototype languages. Finally, I suggested OO implementations for each theory of concepts with class-based and object-based languages and evaluated their naturalness as a knowledge representation tool. The following is a short review.

The classical theory. The traditional class-based languages provide a surprisingly plausible model for the classical theory. The same can be achieved with the object-based languages with the introduction of traits. But the models also run into all the problems encountered for the classical theory of concepts. There are always exceptions to definitions and unclear cases that don't fit into classes or traits.

The prototype theory. The class-based languages are far from being suitable models for the prototype theory. They lack typicality effects and family resemblance structures. The typicality effects could be achieved with some modifications to the language as was shown, but there is no way to model family resemblance categories if all members have exactly the same properties. Initially, it might seem that prototype languages resemble the prototype theory; they operate on concrete objects out of which one could be the prototype. But this conclusion is based on a misunderstanding of the theory. However, there is a way the prototype languages can model family resemblance categories (provided that traits work only as placeholders for objects) and from this prototype effects rise naturally. The proposed model created true family resemblance categories and showed how to compute the prototypes. It also worked as an example of how to define precisely what is typicality - a problem noted for frames when inheritance is used and unusual cases are encountered.

The dual theories. The two implementations for the prototype theory ended up actually being dual theories of concepts. The first version was the core and the identification procedure version and the second implemented Pinker & Prince's continuum version,

although in the second model prototype effects disappear once the objects converge back to a classical world. This, in my opinion, is not a problem of the model. Instead, it undermines Pinker & Prince's version of the dual theory. Being dual theories both of these models suffer from double psychological criticism: the criticism directed against the classical *and* the prototype theories of concepts.

The neoclassical theory. No implementation was found for the neoclassical theory. The partial definitions - being the same as classical definitions - are easy to model, but a final evaluation is impossible to give until the theory tells us how partial definitions are converted into full definitions. The proposed very loosely defined example did, however, support Fodor's counter arguments against the entire theory.

The theory-theory. There is no appealing way to model the theory-theory. This is due to the fact that until a precise account of what the theory-theory means by "theory", it is hard to say how well the OO paradigm could model it. The only proposed solution was to liken theories to class hierarchies, but I doubt that this satisfies the theory-theorists. Scripts, event concepts were discussed in the context of the theory-theory, but we saw that the OO languages provide no natural means for modeling scripts - the order of events. Methods can be called in any order - including the wrong one. The principle of psychological essentialism - the fact that objects can undergo metamorphosis and still retain their identity - was also discussed within the theory-theory. Object can also be viewed differently in different contexts: a box can become a chair, if someone decides to sit on it. The class-based languages and the frame-based knowledge representation have a hard time with this. Instead, this study showed that the object-based languages have a way out of this problem. With the notion of reparenting objects can change their attributes any way they want and still keep their identity - their psychological essence. The object-based languages are thus capable of overcoming this problem.

The atomistic theory. The inevitable conclusion is that there is no way to model atomistic concepts with the OO paradigm, because they are unstructured. Moreover, it is impossible to do it with any knowledge representation tool that relies on structure. If the atomistic theory turns out to be correct, then perhaps it cannot be modeled computationally at all. But this doesn't mean that knowledge representation engineers should throw in the towel. The only thing that the atomistic theory says is that the structure cannot be a *constitutive*

part of concepts. This in no way denies that descriptions don't exist at all; there surely is plenty of information plugged into concepts - it just plays no role in concept's identity.

The main problem is that cognitive science does not know what concepts are. We are left with just bits of scattered findings: there are typicality effects, some concepts might be classical, some form family resemblances, concepts are compositional, the basic level is cognitively more important, people are essentialists in their thinking and have knowledge about why some features are important and how they are ordered.

The following table summarizes the main findings of this study:

	Class-based	Object-based
Definitions	Yes.	Yes, with traits.
Compositionality	Yes.	Yes, with traits. The family resemblance model suffers from the pet fish -problem.
Psychological essentialism	No. Once an object is created it remains immutable.	Yes. Reparenting allows dramatic changes in objects.
Family resemblance	No. All objects have the same attributes.	Yes. Objects can selectively inherit features.
Prototype effects	No, but can be added.	Yes. Family resemblance structure generates prototypes.
Basic level	No.	No.
Scripts	No.	No.

This study set out to show that the object-based languages offer many advantages for modeling concepts and that the entire OO paradigm suffers from two weaknesses. This can be seen from the table. The object-based languages are better suited for more cases than the class-based languages. They can do all that the class-based languages can, but they can also simulate family resemblance categories and allow objects to undergo changes in their attributes, but still retain their essence; the class-based languages and frames cannot do this. The table also shows the two weaknesses in both languages: the lack of basic level and script concepts.

In sum, it can be said that the OO paradigm models well concepts. The idea of incorporating functions and data into objects is surely what concepts do also. Moreover, the class-based model is very close to the classical theory of concepts and the prototype languages to family resemblance concepts. The problem is that both of these theories have been more or less abandoned in cognitive science or at least serious modifications are needed for them; so one should keep this in mind before boasting that the OO approach is psychologically plausible or intuitive.

Before the final theory of concepts arrives, future research could investigate how to modify the languages to support what is already known about concepts. First, as was said, neither type of OO language supports the primacy of the basic level of concepts. Programming in OO languages is incremental and therefore it starts from the root of the hierarchy. But categorization starts from the middle. This point has also practical implications. Lots of money and time is wasted if class hierarchies have to be revised time after time. The idea of getting it correct at the first shot is justifiable in the case where programmers already know everything about the universe to be modeled, but this is rarely the case. In real life, programmers work under overly optimistic timetables, vague requirements and constant pressure for new features in the products (Brooks 1987; Taivalsaari 1997). If this is the case, then the modeling tool should permit the engineers to start intuitively from the basic level and let them find the super- and subordinate levels in due course.

Second, the idea of incorporating data and functions prevented the programmers from passing data to wrong functions. Since methods in the OO style are part of the objects, these types of mistakes cannot happen any more; the language secures that. But it does not prevent the methods from being called in the wrong order. When discussing scripts, we saw that people have sequential knowledge attached to concepts. The concept BIRTHDAY-PARTY contains information about the typical order of the attributes. This type of information is not available in objects. Adding this feature would make the languages more secure, because once a class is correctly designed, it could prevent the class users from accidentally calling the methods in a wrong order. This can have crucial importance in some situations: changing traffic lights or performing airplane checkup functions in the wrong order for example. I am not sure how these things could be added to the languages, or even if they can be, but the point is that there remains some beneficial features that would improve the intuitiveness of the OO style.

6. NOTES

1. The examples are taken from Lakoff (1987a/1987b). Both books provide many other similar examples.

2. I assume that the reader is familiar with the basic architectures of ANNs and training algorithms. For an introduction see Beale & Jackson (1990). Clark (1993/1996) explains the philosophical issues and how concepts are represented in ANNs.

3. Pinker & Prince obtain the number O.67. Someone is wrong and I am not sure whom, but it doesn't really matter. It is still useful enough to classify the objects, because the probability is much higher than 0.375.

4. I suppose this criticism can possibly be directed against any knowledge representation system; once things get bigger, they get more complicated and in the end nobody understands them...

5. An ontological point is in order. Out of the two inheritance mechanisms only the embedding is ontologically plausible. If real objects could inherit behavior by delegation, then one can imagine a science fiction world where an object could delegate, for example, its vision or digestion system to other objects. An object could also change its parents on the fly. Surely this is impossible - one cannot rechoose her parents or use someone else's vision system. Real objects inherit properties at the moment they are born, that is, by embedding. (Lecture notes, Petri Mäenpää, 1999.)

6. The language could, for example, have a construct category for assigning objects in them. As an example,

    object pluto, category Dog, is ...

    could create an object named pluto and assign it in the Dog category. It is up to the runtime environment to keep track of all the categories and the objects in them. Each time a new object is classified into a category the family resemblance degrees of the objects found in the category need to be computed again.

7. REFERENCES

Abadi, Martin and Cardelli, Luca (1996). *A Theory of Objects*. New York: Springer, 396.

Anderson, John R. (1995). *Cognitive Psychology and Its Implications.* New York: W.H. Freeman and Company, 519.

Armstrong, Susan L., Gleitman Lila R. and Gleitman, Henry (1983). What Some Concepts Might Not Be. *Cognition*, 13, 263-308.

Arnold, Ken and Gosling, James (1996). *The Java^{TM} Programming Language*. Addison-Wesley Publishing Company, 333.

Beale, Russell and Jackson Tom (1990). *Neural Computing: An Introduction*. Bristol: Institute of Physics Publishing, 240.

Boden, Margaret A. (Ed.) (1996). *Artificial Intelligence*. San Diego: Academic Press, 384.

Bower, Gordon H., Black, John B. and Turner, Terrence J. (1979). Scripts in Memory for Text. *Cognitive Psychology*, 11, 177-220.

Brachman, Ronald J. and Levesque, Hector J. (Eds.) (1985). *Readings in Knowledge Representation*. Los Altos, CA: Morgan Kaufmann, 571.

Brewer, William F. and Treyens James C. (1981). Role of Schemata in Memory for Places. *Cognitive Psychology*, 13, 207-230.

Brooks, Frederick P. Jr. (1987). No Silver Bullet: Essence and Accidents of Software Engineering. *IEEE Computer*, vol. 20, no. 4 (April), 10-19.

Campione, M., Huml, A. and Walrath, K. (2001). *The Java^{TM} Tutorial*. Sun Microsystems: http://java.sun.com/docs/books/tutorial/index.html (14.1.2002).

Carey, Susan (1991). Knowledge Acquisition: Enrichment or Conceptual Change? Reprinted in Laurence & Margolis (1999), 459-487. Originally in Carey, S. and Gelman R. (Eds.). *The Epigenesis of Mind: Essays on Biology and Cognition*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Clark, Andy (1993). *Associative Engines: Connectionism, Concepts, and Representational Change*. Cambridge, MA: MIT Press, 252.

Clark, Andy (1996). Philosophical Foundations. In Boden (1996), 1-22.

Fodor, Jerry A. (1975). *The Language of Thought*. Cambridge, MA: Harward University Press, 214.

Fodor, Jerry A. (1981). The Present Status of the Innateness Controversy. In Fodor, Jerry A. *RePresentations: Philosophical Essays on the Foundations of Cognitive Science*. Cambridge, MA: MIT Press, 257-316.

Fodor, Jerry A. (1998). *Concepts: Where Cognitive Science Went Wrong*. New York: Oxford University Press, 174.

Fodor, Jerry A., Garrett, Merill F., Walker, Edward C. T. and Parkes, Cornelia H. (1980). Against Definitions. *Cognition*, 8, 263-367.

Gelman, Susan A. and Wellman, Henry M. (1991). Insides and Essences: Early Understanding of the Non-Obvious. *Cognition*, 38, 213-244.

Hayes, P. (1979). The Logic of Frames. Reprinted in Brachman & Levesque (1985), 288-295. Originally in Metzing, D. (1979). *Frame Conceptions and Text Understanding*. Berlin: Walter de Gruyter and Co.

Hyvönen, Eero (1993). Tietämyksen Rakenteen Esittäminen. In Hyvönen & al. (1993), 134-144.

Hyvönen, Eero, Karanta, Ilkka and Syrjänen Markku (Eds.) (1993). *Tekoälyn Ensyklopedia*. Hämeenlinna: Gaudeamus, 356.

Jackendoff, Ray (1989). What Is a Concept, That a Person May Grasp It? *Mind and Language*, 4, 68-102.

Johnson-Laird, Philip Nicholas and Byrne, Ruth M. J. (1991). *Deduction*. London: Lawrence Erlbaum Associates, 243.

Katz, Jerrold J. (1972). *Semantic Theory*. New York: Harper & Row, 464.

Lakoff, George (1987a). *Women, Fire, and Dangerous Things: What Categories Reveal about the Mind*. Chicago: University of Chicago Press, 614.

Lakoff, George (1987b). Cognitive Models and Prototype Theory. In Neisser, U. (Ed.), *Concepts and Conceptual Development: Ecological and Intellectual Factors in Categorization*. New York: Cambridge University Press, 63-100.

Lassila, Ora (1993). Olio-ohjelmointi. In Hyvönen & al. (1993), 315-318.

Loocke, Philip Van (Ed.) (1999). *The Nature of Concepts: Evolution, Structure and Representation*. NY: Routledge, 254.

Laurence, Stephen and Margolis, Eric (Eds.) (1999). *Concepts: Core Readings.* Cambridge, MA: MIT Press, 652.

Minsky, Marvin (1975). A Framework for Representing Knowledge. Reprinted in Brachman & Levesque (1985), 246-262. Originally in Winston, P. (Ed.). *The Psychology of Computer Vision*. New York: McGraw-Hill. Also in Minsky, Marvin (1974). *Marvin Minsky's homepage*. MIT: http://www.media.mit.edu/people/minsky/papers/Frames/frames.html (23.10.2001).

Murphy, Gregory L. and Medin, Douglas L. (1985). The Role of Theories in Conceptual Coherence. *Psychological Review*, 92(3), 289-316.

Niemelä, Ilkka (1993). Logiikka tietämyskielenä. In Hyvönen & al. (1993), 116-124.

Niemeyer, Patrick and Peck, Joshua (1998). *Exploring Java^{TM}*. O'Reilly, 407.

Nuutila, Esko (1993). Sääntöjärjestelmät. In Hyvönen & al. (1993), 134-144.

Osherson, Daniel N. and Smith, Edward E. (1981). On the Adequacy of Prototype Theory as a Theory of Concepts. *Cognition*, 9, 35-58.

Partridge, Derek (1996). Representation of Knowledge. In Boden (1996), 55-87.

Pinker, Steven (1994). *The Language Instinct: The New Science of Language and Mind*. London: Penguin Books, 494.

Pinker, Steven (1997). *How the Mind Works.* New York: Norton, 660.

Pinker, Steven and Prince, Alan (1999). The Nature of Human Concepts: Evidence from an Unusual Source. In Loocke (1999), 8-51.

Putnam, Hilary (1970). Is Semantics Possible? Reprinted in Laurence & Margolis (1999), 177-187. Originally in Kiefer, H. and Munitz, M. (Eds.). *Languages, Belief and Metaphysics*, vol. 1. New York: State University of New York Press.

Rey, Georges (1983). Concepts and Stereotypes. *Cognition*, 15, 237-262.

Rosch, Eleanor H. (1973a). Natural Categories. *Cognitive Psychology*, 4, 328-350.

Rosch, Eleanor H. (1973b). On the Internal Structure of Perceptual and Semantic Categories. In Moore, T. E. (Ed.). *Cognitive Development and the Acquisition of Language*. New York: Academic Press, 111-144.

Rosch, Eleanor H. (1975). Cognitive Reference Points. *Cognitive Psychology*, 7, 532-547.

Rosch, Eleanor H. (1978). Principles of Categorization. In Rosch, Eleanor H. and Lloyd, B. B. (Eds.). *Cognition and Categorization*. Hillsdale, NJ: Lawrence Erlbaum Associates, 27-48

Rosch, Eleanor H. and Mervis, Carolyn B. (1975). Family Resemblances: Studies in the Internal Structure of Categories. *Cognitive Psychology*, 7, 573-605.

Rosch, Eleanor H., Mervis, Carolyn B., Gray, Wayne D., Johnson, David M. and Boyes-Braem, Penny (1976). Basic Objects in Natural Categories. *Cognitive Psychology*, 8, 382-439.

Saariluoma, Pertti (1990). *Taitavan Ajattelun Psykologia*. Keuruu: Otava, 221.

Salo, Pauli J. (1999). Ovatko Käsitteet Prototyyppejä? *Psykologia*, 3, 171-180.

Schank, Roger and Abelson, Robert (1977). *Scripts, Plans, Goals and Understanding: An Inquiry into Human Knowledge Structures*. Hillsdale, NJ: Lawrence Erlbaum Associates, 248.

Smith, Edward E. and Medin, Douglas L. (1981). *Categories and Concepts*. Cambridge, MA: Harward University Press, 203.

Smith, Edward E., Osherson Daniel N., Rips, Lance J. and Keane, Margaret (1988). Combining Prototypes: A Selective Modification Model. *Cognitive Science*, vol. 12, no. 4, 485-527.

Taivalsaari, Antero (1997). Classes Versus Prototypes: Some Philosophical and Historical Observations. *The Journal of Object Oriented Programming* (JOOP), vol. 10, no. 7, 44-50. Also in Taivalsaari, Antero (1996). ResearchIndex, The NECI Scientific Literature Digital Library: http://citeseer.nj.nec.com/taivalsaari96classes.html (15.1.2002).

Thagard, Paul (1996). *Mind: Introduction to Cognitive Science*. Cambridge, MA: MIT Press, 213.

Thagard, Paul (1999). The Concept of Disease: Structure and Change. In Loocke (1999), 215-242.

Torsun, I. S. (1995). *Foundations of Intelligent Knowledge-Based Systems*. San Diego: Academic Press, 507.

Wason, P. C. and Johnson-Laird, P. N. (1972). *Psychology of Reasoning: Structure and Content*. London: B.T. Batsford, 264.

Weiss, Mark Allen (1998). *Data Structures and Problem Solving Using Java^{TM}*. Addison Wesley Longman, 780.

Wittgenstein, Ludwig (1953). *Philosophical Investigations.* Anscombe, G.E.M. (Tr.). Oxford: Basil Blackwell, 516.