SETS, TYPES AND CATEGORIES

PER MARTIN-LÖF

Note. This lecture was delivered by Per Martin-Löf at Leiden University on 6 February 2004. It was part of a symposium, *Philosophy of Constructive Type Theory*, held in order to mark the award of an honorary doctorate to its author. Göran Sundholm taped the lecture and arranged funds to have it transcribed by Aziz Abdalli, and also roughly edited the first transcript. The present version was prepared by Ansten Klev.

Types were introduced into logic by Russell in his attempt at coming to grips with the Russell Paradox. Originally that was in 1903, in the second appendix to the *Principles of Mathematics*, Appendix B, which bears the title "The doctrine of types". There he introduced a definition of type, which reads as follows:

> Every propositional function $\phi(x)$—so it is contended—has, in addition to its range of truth, a range of significance, *i.e.* a range within which x must lie if $\phi(x)$ is to be a proposition at all, whether true or false. This is the first point of the theory of types ; the second point is that ranges of significance form *types*...

Here we have, for the first time, Russell's attempt at a definition of what a type is, namely, that a type is the range of significance of a propositional function, and significance means of course, in this connection, meaningfulness. A type is thus the totality of arguments for which a propositional function is significant, or meaningful. The range of truth of a propositional function, $\phi(x)$, could be denoted

$$\hat{x}\phi(x)$$

This is the set of arguments for which $\phi(x)$ is true. Russell saw that this is not enough—as if x could range over anything whatsoever. We have to be careful with what x ranges over, because $\phi(x)$ need not be meaningful for any argument whatsoever. So we need to introduce something more here, namely the domain of the propositional function. This then is what Russell called a type. His formulation was that a type is the range of significance of a propositional function.

What particular types did Russell have in mind? First of all, there was the type of individuals, or terms as he said, and then, above that ground type of individuals, he had classes of individuals, classes of classes of individuals, and so on up to any finite level. So that was the first hierarchy of types—well, I will qualify that in a moment. Then Russell said that, outside this hierarchy, there is the type of propositions—if propositions can form a type, which he had grave doubts about. That he had doubts about that is clear from the fact that, if a type is defined in this way, as the range of significance of a propositional function, why do not all propositions then form a type? You certainly have the function which takes any proposition into itself, the identity function on propositions. That is clearly a propositional function in the sense that it has propositions as values, and what is its domain? Well, it clearly consists of all propositions. So if the range of significance of this particular propositional function consists of all propositions, they ought to form a type according to his definition of what a type is. On the other hand, Russell had grave doubts whether propositions form a type. He ends in fact this Appendix B by saying,

> The totality of all logical objects, or of all propositions, involves, it would seem, a fundamental logical difficulty.

He says that after having derived his second paradox—much less known than the first—from the idea that propositions form a type.

It was the quantification over propositions that Russell was rightly worried about: whether the impredicativity involved in quantification over propositions really makes good sense. Applying universal quantification to the particular choice of $\phi(x)$ that I mentioned, you get the proposition "for all x, x" where x ranges over propositions, and Russell thought that that hardly makes good sense. This suggests that what Russell really had in mind when he said that a type is the range of significance of a propositional function was that a type is what the quantifiers range over. Indeed, if we say instead that a type is an individual domain, or a quantificational domain, then Russell's doubts about forming a type of propositions make perfectly good sense.

When I said that it was Russell who introduced types into logic, maybe I ought to have said that he introduced the term type into logic, because, surely there was something in the way of types already before Russell. I am thinking then in particular of the universe of discourse: the universe of discourse of the Boolean tradition is very much like what we now call an individual domain, or a quantificational domain. That is clear enough from how we nowadays represent the Aristotelian forms of judgment. For instance, when we represent "all A are B" as "for all x in the universe of discourse, x is A implies x is B", then it is clear that the A and the B have to be properties, or classes, of individuals in the universe discourse that we quantify over. So the universe of discourse is the quantificational domain, and is to be identified with what we call types nowadays.

Secondly, before Russell there was Frege's hierarchy of Stufen, of levels of functions. Frege had this idea, which goes against type theory, that we have an all-embracing domain of all objects at the bottom, and then over that domain of all objects we have functions, and functions of functions, etc., up to any arbitrarily high finite level. This, to my mind, is really the first place where you have something like what we nowadays call a hierarchy of types. This was already in the *Grundgesetze*, published in 1893, so it was ten years before Russell had his hierarchy of classes of individuals and classes of classes of individuals.

These, then, seem to me to be the two basic prototypes for types before Russell introduced the term in 1903.

Russell introduced types in order to deal with the paradox, but for some reason unknown to me, he seems to have abandoned type theory as the solution and instead started to work on various other theories that he called the zigzag theory and the theory of limitation of size, and there was also the no-classes theory. In 1906 he is still writing about those three possible solutions, but soon after that he must have reverted to his first attempt, to the theory of types, because already in 1908, his type-theory paper, which sets out all his basic ideas about types, was published under the title "Mathematical logic as based on the theory of types".

The essential novelty in 1908 as compared with the Appendix in 1903 is that now the ramified types make their appearance. Ramified types are Russell's answer to his doubt as to whether all propositions form a type that you can meaningfully quantify over. He made up his mind in the negative and said: that does not make sense, we have to divide propositions into orders and similarly divide classes into classes of different orders. The propositions or classes or relations of a fixed order—that is the totality over which it does make sense to quantify.

We thus get the ramified theory of types, which then became the basic system of *Principia*. And we know what the problem was: Russell could not do his mathematics in the ramified theory of types, so he introduced the infamous axiom of reducibility, and it was he himself who was totally dissatisfied with it, as he says in the preface to the second edition of *Principia* from 1925. It is not the sort of axiom with which one can rest content. It has a purely pragmatic justification, namely that using this axiom, which had no justification, one can derive the mathematics that one needs to derive and that cannot be derived without it.

This is essentially how type theory was left by Russell, and this is the first phase, I would say, in the development of type theory: the phase of the ramified theory of types. Then came the second phase, which is the transition from the ramified theory of types to the simple theory of types, a transition that takes place gradually in the 1920s. The proposal of Chwistek in the early 1920s and Ramsey in 1926 was simply to do away with the ramification, do away with the orders and just retain the types. That means that now the type of propositions was accepted. Propositions were accepted as forming a type, and propositional functions over a given domain also form a type, and so on—we know, of course, the simple type hierarchy.

Something then had to be said about the interpretation. After all, Russell's worries were not fancies. They were real worries. If the hierarchy of simple types, or the very notion of simple types, was to be accepted, then something had to be said about their interpretation. That is Ramsey's contribution here. I know next to nothing exactly about what Chwistek did, and I would very much like to know from some expert, but at least I do not think that he contributed anything to the interpretation of the simple theory of types. It is Ramsey who for the first time proposed what we nowadays would say, namely that there is no problem with the type of propositions, because we just interpret propositions as truth values. The type of propositions has two elements and clearly forms a totality over which we can quantify.

That is the current interpretation, and that shifts the problem that Russell had to another place—or maybe I should not say that, but rather say that the problem about the axiom of reducibility is then shifted to the problem of the validity of the comprehension axiom in the simple theory of types: what right do we have to say that an arbitrarily complicated formula—containing free variables, quantification over all propositions or properties or relations and so on—denotes, or defines, a truth function of its arguments? That is the fundamental problem, and it is the analogue of the problem of the axiom of reducibility for the simple theory of types.

In any event, the simple theory of types was accepted by the logical community, and the next step after Ramsey was the step of introducing an explicit notation. In *Principia* you certainly had types, but there was no notation for types. Each variable had a type, so to say implicitly: an implicit type that was never shown explicitly. This is why the passages in the *Tractatus* about types are written the way they are: the signs for types are somehow the variables, because the variables implicitly carry types, but there is no explicit notation for types. It was therefore a very important step in the development of type theory to introduce an explicit notation for types. That comes, as far as I know, for the first time in Carnap's *Abriss der Logistik* from 1929, where he introduces one of the two simple type hierarchies that we are familiar with, namely the one that starts with a ground type 0 of individuals and then, given previously introduced types

$$\alpha_1, \ldots, \alpha_n,$$

one forms the type

$$(\alpha_1, \ldots, \alpha_n)$$

which is the type of n-ary relations over $\alpha_1, \ldots, \alpha_n$. This was the type hierarchy for which Carnap for the first time introduced an explicit notation.

You can also follow the transition from the ramified theory of types to the simple theory of types very clearly by looking at the various editions of Hilbert–Ackermann. The first edition is from 1928, so it is before Carnap, which means that these notations for types are not yet there. Instead there is a description of the ramified theory of types of *Principia.* In the second edition, however, which is ten years later, from 1938, the ramified theory has been toned down—is more or less absent—and has been replaced by the simple theory of types with this new explicit notation for the higher types that was introduced by Carnap.

After Carnap, a new impetus comes from Ajdukiewicz in his groundbreaking paper on categorial grammar from 1935. He also introduced an explicit notation for types, different from Carnap's. The differences are not only notational—not so great perhaps, but worth noticing. First of all, instead of having a single ground type, Ajdukiewicz has two, namely n and s, where n stands for name and s stands for sentence. That immediately shows something. When we say that 0 is the type of individuals and that $(\alpha_1, \ldots, \alpha_n)$ is the type of ordinary relations of the arguments of the appropriate types, we are using the usual objective way of speaking that is characteristic of mathematics: individuals and relations are mathematical objects. Ajdukiewicz, on the other hand, did not use the term type. He rather used the term meaning category, Bedeutungskategorie, that had been introduced by Husserl and transmitted by Leśniewski. Moreover, n was the category of names, and s was the category of sentences, which means that they were thought of as linguistic categories, meaning categories, or semantic categories—to use the slight change in the term contributed by Leśniewski as compared with Husserl.

The final piece in the development of the simple theory of types from the ramified theory is Church's paper from 1940, where he gives his formulation of the simple theory of types using the lambda notation for functional abstraction. Church's types were exactly like Ajdukiewicz's types, with two ground types that Church called ι and o, of individuals and propositions respectively. Again you notice the difference, because Church certainly thought about individuals and propositions and functions in an objective manner, contrary to Ajdukiewicz, who spoke in a linguistic manner.

This will have to end the description of the second phase in the development of type theory, which is when the simple theory of types crystallized. Then I come to the third phase, in which I have been involved, beginning in 1970. There were three papers of mine, in 1971, '72 and '73. The first was submitted to *Acta Mathematica*, but withdrawn after Girard's discovery of the paradox that now bears his name. The '72 version would eventually be published in 1998, but the '73 paper was published in due time in the proceedings of the Bristol symposium at which the paper was first presented. If I look back on what I then wrote—which I have not done for a long time—I am immediately struck by certain things which have been an improvement in understanding in these 30 years. In 1971, I wrote as my attempted definition of what a type is, or the fixing of the meaning of the term type,

> Every mathematical object is of a certain kind or type which is immediately associated with the object in question.

That was the first sentence. Then comes a second sentence,

> A type is defined by prescribing how we are allowed to construct objects of that type.

What I had in mind with the second sentence is clear enough. I mean, if you take the type of truth values, you define it by saying that it has the two elements true and false, and if you take the type of natural numbers, you define it by the first two Peano axioms. If you take the product of two types, you define it by saying that it is the type of pairs of elements coming from the two types, and so on. That is clearly what I had in mind with the second sentence.

If you look at the first sentence, however, it is clear that I had something else in mind, namely what in philosophical terminology you would call an essentialist view, or an essentialist doctrine, as far as mathematical objects are concerned. A mathematical object is not just an object: it is always a something. It is a natural number or an integer or a rational number or a real number or a complex number or a real-valued function or whatnot. Likewise, a mathematical structure—which is not just one mathematical object, but several mathematical objects taken collectively—is not just a structure. It is a group or a ring or a field or a Banach space or a category or whatever structure you are interested in. Structure by itself does not make sense: a structure is always a structure of a certain kind, "structure d'une espèce" in Bourbaki's terminology.

Both for objects and for structures it is thus clear that they are always a something, and it is this something which traditionally is the essence of the object. Every mathematical object is of a certain essence, or has a certain essence, namely that property of the object—the essential property—that makes it into what it is. That is simply the essentialist doctrine as applied to mathematics. That is quite clearly what I had in mind in the first sentence. If you look at professor De Bruijn's writings on AUTOMATH, he writes beautifully about this in several places, and it is clear that, for him also, this is what we must keep track of all the time: what kind of object each mathematical object is. He took that as his basic doctrine, so the first sentence here would fit his notion of type much better than the second one.

It is therefore perhaps not so strange that the initial formulation of type theory was inconsistent, because it was based on a conflation of two notions: on the one hand, that a type is defined by prescribing how its elements are formed, or what possible forms its elements can take, and, on the other hand, that a type is the whatness of a mathematical object, the essence of a mathematical object. Indeed, in the further development, which involved getting rid of the paradox, eventually these two notions were separated.

Two concepts have thus been conflated here, and we have a terminological problem. If we retain the term type, we have to choose whether we should use it for the first or for the second concept. This is a terminological problem with which we are still somehow struggling and which it is impossible to solve in any other way than by making a decision. I can make a decision on what terminology to use, but that will not decide what is going to be used generally. It is not quite clear yet what is going to be the final solution, but at least the terminology that I am using nowadays is that I call types in the first sense—like the type of truth values or the type of natural numbers and so on—I call them sets instead. When I just lay down what an object of the type is, then I call it a type.

This distinction is connected with what I began with, namely the earlier development of type theory, the ramified theory of *Principia* as opposed to the simple theory of types after *Principia*. The ramified types were quantificational domains, as I said, and those were specified by giving the rules for forming them, which meant the rules for forming formulas. The ramified types of propositions involved a notion of order, and what a formula of a certain order is is laid down by a definition where you say: these and these are the formulas—which is like defining a set in my terminology.

The ramified types are thus indeed sets in my terminology, and you can quantify over them, whereas when you go to the simple type structure, the simple types are explained simply by saying what an object of the type in question is. The type of propositions has to have an explanation of what a proposition is, and the function type will have a definition of what a function taking objects of type α to objects of type β is. That definition of function does not take the form of giving rules for how they are formed. It is essentially the definition that a function is something which, when applied to something of type α, yields something of type β—which is not to tell how functions are formed. It is an operative definition, telling how they operate, rather.

The distinction in my present terminology between sets and types is thus the crucial distinction that one had earlier between the ramified types and the simple types, the ramified types being sets and the simple types being types.

Type theory, like any other logical system, has to be specified by first displaying the forms of judgment that are used in the system and, after that, displaying the rules of inference. In the original form of type theory, the forms of judgment, as I would now write them, were the following:

$$\begin{aligned} A : \text{set} &\quad (x_1 : A_1, \ldots, x_n : A_n) \\ A = B : \text{set} &\quad (x_1 : A_1, \ldots, x_n : A_n) \\ a : A &\quad (x_1 : A_1, \ldots, x_n : A_n) \\ a = b : A &\quad (x_1 : A_1, \ldots, x_n : A_n) \end{aligned}$$

The first says that A is a set depending on the variables $x_1, \ldots, x_n$ ranging over previously introduced sets, $A_1, \ldots, A_n$, the second that two sets, A and B, are definitionally equal, where both may depend on variables in this way, the third that a is an element of the set A, again in such a context, and the fourth that a and b are definitionally equal elements of A, depending on the displayed variables. These are the forms of judgment in the original type theory, which we now call the lower-level type theory—the explanation for that terminology will come later.

Saying that these are the forms of judgment can be said in a different way by introducing the notion of category. The step is this, that if we start from the forms of judgment, we can of course make an ordinary grammatical analysis of sentences of these forms, just as we can with any other indicative, or declarative, sentence. If you ask in the first case, for instance, What is the subject and what is the predicate? The judgement says that A is a set depending on the variables $x_1, \ldots, x_n$, and what is the subject? The subject is that about which something is said in the sentence, so A is the subject here, and the rest,

$$\text{set} \quad (x_1 : A_1, \ldots, x_n : A_n)$$

is the predicate. In the second case we say of A and B that they are equal, and here again we have the subject to the left of the colon and all of what is to the right of the colon is the predicate.

In terms of the subject-predicate forms $S : P$ and $S = T : P$—here I use the convenient colon notation that De Bruijn has introduced for the copula—you see that above we specialize the predicate P to one of two possibilities: to the predicate that you have in the first two cases, namely

$$\text{set} \quad (x_1 : A_1, \ldots, x_n : A_n)$$

or to the predicate that you have in the last two cases,

$$A \ \ (x_1 : A_1, \ldots, x_n : A_n)$$

This is the predicate 'element of A depending on these variables', but I just write A for element of A. If we now call these *categories*, then you see what the correspondence between the categories and the forms of judgment is.

This thus explains how you go from the forms of judgment to the categories. Conversely, if these are the categories, and they are what you fill in for the predicate P, then you get the above forms of judgment.

Displaying the forms of judgment and displaying—in a table, if you want, or in a list, as Aristotle did—what the categories are is thus to do one and the same thing. Whatever way you do it, displaying the forms, as I have done here, is only half of what you have to do: you also have to give the other half, the semantical half, which means—in terms of the forms of judgment—explaining what a judgment of each one of these four forms means. I shall not undertake that now, but simply assume that it has been correctly done. So there are meaning explanations for these four forms of judgment, and they lay down, for each of these two forms of category, what an object of the category is and what it means for two objects of the category to be equal. The first explanation is what Dummett has introduced the convenient term criterion of application for, whereas the second explanation is what Frege introduced the term criterion of identity for. A category is thus defined by its criterion of application and its criterion of identity.

These are then the categories of the lower-level type theory. Observe that you have to proceed in this way, by simply displaying them. I ought to have made it clear also that category is the Aristotelian term and displaying them in this way, in terms of the forms of judgment, is in the Kantian spirit. The idea that you get to the categories by looking at your forms of judgment, that is precisely what Kant called "der transzendentale Leitfaden der Entdeckung aller reinen Verstandesbegriffe", so the pure concepts of understanding, which are the categories. The clue to their discovery was to look at the forms of judgment that you are using in your logic and from these extract the categories.

This was lower-level type theory. Then we have made probably the most important step after the early 1970s, in 1986, when the higher-level type theory was introduced. That was an extension of the lower-level type theory, and the extension meant that more forms of judgement were introduced, which is to say more categories. The difference is this, that the general structure here is retained, but sets are replaced by types. Remember the quotations that I have shown, where there was the conflation of two notions of set and type. In the lower-level type theory there is only talk about sets and their elements, but now in the higher-level type theory we have, not only sets, but also types. That means that these forms of judgment are, so to say, raised one level up, so that now we have type instead of set in the first two, and in the second two we have types rather than sets,

$$\begin{aligned} \alpha : \text{type} &\quad (x_1 : \alpha_1, \ldots, x_n : \alpha_n) \\ \alpha = \beta : \text{type} &\quad (x_1 : \alpha_1, \ldots, x_n : \alpha_n) \\ a : \alpha &\quad (x_1 : \alpha_1, \ldots, x_n : \alpha_n) \\ a = b : \alpha &\quad (x_1 : \alpha_1, \ldots, x_n : \alpha_n) \end{aligned}$$

As you can see, I use small Greek letters for types.

These are now the forms of judgment of the higher-level type theory, and if we use the same procedure here, that means that the categories of the lower-level type theory have now been changed to

$$\text{type} \;\; (x_1 : \alpha_1, \ldots, x_n : \alpha_n)$$

and

$$\alpha \;\; (x_1 : \alpha_1, \ldots, x_n : \alpha_n)$$

This is an extension of what you have in the lower-level type theory for the following reason. The two forms of category of the lower-level type theory are absorbed in the second form of category here because of the first two rules of type formation,

$$\text{set} : \text{type} \qquad\qquad \frac{A : \text{set}}{\text{elem}(A) : \text{type}}$$

The first rule says that set is a type, and the second rule that if A is a set, then A is also a type, namely the type of elements of A, which you may write as elem(A). We need only specialize α and $\alpha_1, \ldots, \alpha_n$ to these forms to get the categories of the lower-level type theory. The first form of category, however, namely the category

$$\text{type} \;\; (x_1 : \alpha_1, \ldots, x_n : \alpha_n)$$

was not on our previous list, so we have extended our system of categories.

The last rule of type formation says that if α is a type and β is a type depending on a variable that is ranging over α, then we may form the dependent function type, $(x : \alpha)\beta$,

$$\frac{\alpha : \text{type} \qquad \begin{matrix} (x : \alpha) \\ \beta : \text{type} \end{matrix}}{(x : \alpha)\beta : \text{type}}$$

These three rules of type formation clearly generalize Church's rules of type formation. The type set corresponds to Church's o, and if you have a fixed set A, then that gives rise to a type, elem(A), corresponding to Church's ι, although Church had only one fixed domain, whereas here the domains are generated in the theory itself. Finally, the dependent function type, $(x : \alpha)\beta$, includes as a special case the non-dependent function type, $(\alpha)\beta$, in the case when β does not depend on the variable x of type α. So this type structure is a generalization of Church's.

Now you see we are faced with the problem of meaning and the explanation of what a type is as well as what it means for two types to be equal. From this explanation it should become clear how types are related to categories, that is, in what sense types are similar to categories and in what way, nevertheless, there is a distinction between them.

The explanation of types is this, that the types—or more generally, families of types, if you have arguments—are generated by these rules. From the purely formal point of view, they are just generated by these rules. These rules give you the type structure. Simultaneously with the generation of the types, however, it has to be explained what an object of the type is, as well as what it means for two objects of the type to be the same. That means that you have to explain what a set is and what it means for two sets to be equal, which has been done already in the lower-level type theory. Then you have to explain, for a set A, what an element of A is and what it means for two elements of A to be equal, and that has also already been done in the lower-level type theory. Finally, you have to provide an explanation, given a type α and a family of types β, of what an object of the type $(x : \alpha)\beta$ is and of what it means for two objects of this type to be equal. The explanation is that an object of this type is a function, f say, and a function means by definition that when you apply it to an object of type α, you get as a result an object of type β with the appropriate substitution, and moreover, when you apply f to equal objects of type α, then you get equal objects of the corresponding type β as a result,

$$\frac{f : (x : \alpha)\beta \qquad a : \alpha}{f(a) : \beta(a/x)} \qquad\qquad \frac{f : (x : \alpha)\beta \qquad a = b : \alpha}{f(a) = f(b) : \beta(a/x)}$$

This is the explanation of what a function is, and it is the explanation of the dependent function type.

The notion of type is thus fixed by giving these rules for forming types, or generating the types, and simultaneously with that, explaining what an object of each type is and what it is for two objects of the type in question to be equal.

Now you see, first of all, the similarity with categories, because also a category is defined by explaining what an object of the category is and what it means for two objects of the category to be equal. For categories, that is all, hence the system of categories is left open to the possibility of introducing new categories. We have an example of that when we moved from the lower-level to the higher-level type theory, since then we introduced new categories and got a bigger list, or a bigger table. In my definition of type, however, I am saying something more stringent, namely that it is essential to the notion of type to be something which is generated, or formed, by means of these rules. There are no other types than those that can be formed by means of these rules. Of course, they are not formed purely formally by these rules, because with each rule there is associated the semantical explanations that I just gave, which took the form of explaining what an object of the type is and what it means for two objects of the type to be equal. There is thus a difference between types and categories, namely that there is a more stringent requirement on types.

The conclusion that I have reached here about how types are related to categories can perhaps be expressed as follows, that types are objectified categories. In any system of types—already when you have the usual rules for forming the simple type hierarchy—we are saying that something is a type,

$$\alpha : \text{type}$$

That means that you have a form of judgment here, where you say that something is a type, and that form of judgment is eventually the same as a category, the category of types in this case. So as soon as we give rules for forming types, we are forced to introduce this form of judgment, because without it we cannot formulate our rules. That means that we are introducing a new category, namely the category of types and making the types themselves into objects of that category. We can therefore say that types are objectified categories, or in the idiom preferred by De Bruijn, one can say that types are categories that are pushed from the metalanguage into the object language, as they are made into objects. If you want to put it in the other direction, you may say that categories are metalinguistic types rather than object-linguistic types. They are the metalinguistic analogues of types. [End of first side of tape.]

...objects of types and objects of categories, but there is a slight difference then in the concept of object, and there is an expression in computer science for this difference. Computer scientists talk about first-class objects, which are objects that can be assigned as values to variables, for instance—that is their most important property—whereas if you take $\sqrt{x}$ or something like that, it is a real-valued function, but it is not a first-class object, because you have a variable, x, in it. You may phrase this by saying that objects of types are first-class objects, whereas objects of categories are only second-class objects.

With respect to variables, there is a very clear difference between types and categories, namely that types are what the object-linguistic variables, $x, y, \ldots$ range over. That is because of the rule for introducing variables, namely that, whenever you have a type α, you may introduce an object-linguistic variable, x say, ranging over that type. On the other hand, the metalinguistic variables, or the schematic variables, $a, b, \ldots, \alpha, \beta, \ldots$ that I have been using here all the time, are always tied to a category. So metalinguistic variables are always tied to categories, whereas object-linguistic variables are tied to types.

Yet another way of explaining, or clarifying, the difference between types and categories is available for those who are familiar with universes in type theory—I am just taking for granted now that you know what a universe is in type theory. We may say that types are to categories as elements of a universe are to sets. This is because of how a universe is defined. A universe is defined by stipulating how certain codes for sets are generated, or formed, and simultaneously with that generation you have to explain what set it encodes. This is completely analogous to what I said about types, namely that we give the rules of type formation, we specify how types are formed, but simultaneously with that we have to explain what an object of the type is and what it means for two objects of the type to be equal, which is the same as saying what category the type in question determines. Just as types are objectified categories, in the same way, you may think of elements of a universe as elementized sets: sets made into elements of a universe.

Seeing the hierarchy that you have in type theory—with elements of sets, and sets themselves forming a type, and the types forming a category—and being in Leiden, it is impossible not to remember the Linnean classification of the animal kingdom and the vegetable kingdom and the mineral kingdom, namely, the classification into species, genera, orders and classes. For instance, the animal kingdom was divided into the classes of mammals, birds and so on. The correspondence here with the concepts of type theory is quite clear, so one could use the Linnean terminology if one wanted to. It would have been quite perfect to call sets *genera*, elements of sets *species* of the genus in question, and then for types you could use *orders*, and for the categories you could use *classes*, although of course we ought to pay respect to Aristotle and retain his term category.

Just as the animal, vegetable and mineral kingdoms are divided into classes, orders, genera and species, you may ask what realm, or kingdom, it is that is divided into categories and types and sets and elements? That is clear, of course: it is the logical realm, or one could say the ontological realm, namely the realm of logical objects. The objects of logic and pure mathematics are classified into this hierarchy. This realm came to the forefront in the most clear way for the first time with Bolzano, with his Sätze an sich and Vorstellungen an sich. They are logical objects in Frege's terminology, which was taken over by Wittgenstein and Russell. The second example of this kind are the objects of Cantorian set theory, which means first and foremost sets themselves: sets are objects belonging to this logical realm. Then, around the turn of the century, this conception had a kind of breakthrough within the Brentano school and in particular with Husserl, although he preferred the term categorial objects rather than logical objects.

My last remark is about the relation between categories thought of as linguistic categories, meaning categories—Bedeutungskategorien is Husserl's term—on the one hand, and objectual categories, on the other hand—in Husserl's terminology, the formal-ontological categories. Are categories categories of linguistic entities or are they ontological categories, categories of objects? To my mind, the answer to this old question, which arose immediately after Aristotle introduced the categories, because they appear both in the *Organon* and in the *Metaphysics*—so there was a discussion by the commentators on Aristotle in late Antiquity: are they primarily the one or the other? And how are the two conceptions related to each other? I think that the answer to this question is that the categories are as much logical as they are ontological categories, simply because the objects of ontology are meaning objects, that is, they are meanings in the objective sense. It is meanings, of course, that are the objects of semantical categories, the Bedeutungskategorien, but if the objects of the ontological categories are meaning objects, then the categories are after all the same, and the duplication of categories into meaning categories and ontological categories is actually a mere coincidence.