Semantic Paradoxes and Transparent Intensional Logic

Jiří Raclavský

Abstract

The paper describes the solution to semantic paradoxes pioneered by Pavel Tichý and further developed by the present author. Its main feature is an examination (and then refutation) of the hidden premise of paradoxes that the paradox-producing expression really means what it seems to mean. Semantic concepts are explicated as relative to language, thus also language is explicated. The so-called "explicit approach" easily treats paradoxes in which language is explicitly referred to. The residual paradoxes are solved by the "implicit approach" which employs ideas made explicit by the former one.

Introduction

Transparent Intensional Logic (*TIL*) is a rich and powerful logical system capable to treat, *inter alia*, a great amount of natural language phenomena. The aim of this rather short paper is to describe the TIL-based approach to semantic paradoxes. Pavel Tichý, the originator of TIL, developed its core ideas when he investigated and solved four versions of the Liar paradox (Tichý 1988, section 44). The present author has elaborated his ideas into an extensive theory in a number of writings (references are suppressed).

Semantic paradoxes (*SP*s), e.g. the Liar and Grelling's heterological paradox, are paradoxes concerning semantic concepts (e.g. truth, denotation, reference). Among their premises (since paradoxes are arguments), it always occurs a *paradox-producing term* (e.g. "This sentence is not true") which includes some *semantic term* expressing a *semantic concept*. A valuable solution to SPs should revise a. our *uncritical* (naïve) *theory* of semantic concepts or b. our ordinary, uncritical derivation rules, suggesting thus a *critical theory* of semantic concepts or derivation rules.

In recent decades, the classical 'hierarchical' approaches by Russell and Tarski, and even three-(many-)valued approaches by Łukasiewicz, Kripke and others, have been repudiated in favour of rather unclassical ones: Priest's paraconsistent logic (dialetheias), Gupta's and Belnap's revision theory (circular concepts and definitions), Field's paracompleteness, contextualism. In

contrast to these recent approaches, the TIL-approach is rather classical, offering an a.-style of explanation.

Here are the corner-stones of the TIL-approach:

a. critical examination, and then refutation, of the *hidden premise* of SPs that the paradox-producing expression meanswhat it seems to mean (generalized from Tichý 1988, 228);

b. since it is a truism that an expression may mean (denote, refer to) something only relative to a particular language, semantic concepts are explicated as inescapably *relative to language* (especially in Raclavský 2009), thus also the concept of *language* is explicated (*ibid.*).

The paper is divided into three parts. The section "1. TIL basics" explains the notion of construction, explication of meanings (semantic scheme), and the TIL type theory. The section "2. Explicit approach" provides an explication of language, explication of semantic concepts as *explicitly* relative to language, and a principle of solution to (many) SPs. The last section "3. Implicit approach" starts with an objection, the admission of which seems to lead to the revenge problem; then, semantic concepts *implicitly* relative to language are investigated and a solution to residual SPs is explained.

1 TIL basics

1.1 Constructions

One can distinguish two notions of function: function as a mere mapping (hereafter *function*), i.e. function in 'extensional' sense, and function as ašstructured recipe, procedure, i.e. function in 'intensional' sense. Recall that, for instance, Russell of the no-class theory repudiated functions in the first sense while espousing functions in the letter sense (viz. his propositional functions). Tichý treats functions in both senses, the latter ones explicated as certain *constructions*. For an extensive defence of the notion of construction see (Tichý 1988).

Constructions are structured abstract extra-language procedures (roughly: algorithms). Any object O is constructible by infinitely many *equivalent* (more precisely *v-congruent*, where v is a valuation), yet *not identical*, constructions. Two features specify each construction C: i. which object O (if any) is (v-)constructed by C; ii. how C (v-)constructs O (by means of which subconstructions).

For exact specification of constructions see (Tichý 1988, ch. 5). Four basic kinds of constructions are specified there; having thus (where X is any object or construction and $C_{(i)}$ is any construction):

i. variables x (not as letters!)

ii. trivializations ${}^0X$ ('constants')

iii. compositions $[C\ C_1\ ...\ C_n]$ ('applications')[1]

iv. closures λxC ('λ-abstractions').

(Of course, definitions of subconstructions, free / bound variables, open / closed constructions should be added here.) Recall that constructions are not λ-terms (which are expressions), λ-terms are only used to denote constructions. *Concepts* can be aptly explicated as certain constructions (e.g. Materna 2004).

1.2 Simple theory of types

Early development of TIL was framed within the simple theory of types (STT; *cf.* Tichý 1976).[2] Let B (base) be a set of pair-wise disjoint collections of (primitive) objects:

a. Every member of B is a type over B.

b. If ξ, ξ_1, ..., ξ_n are (any) types over B, then $(\xi\xi_1...\xi_n)$, i.e. collection of total and partial functions from ξ_1, ..., $\xi_n$to ξ, is a type over B.

For the analysis of natural discourse Tichý utilized $B_{TIL} = \{\iota, \mathrm{o}, \omega, \tau\}$, where ι collects *individuals*, o collects *truth-values* (just T and F), ω collects *possible worlds* (serving as a modal index), and τ collects *real numbers* (serving, *inter alia*, as a temporal index). Functions from possible world – moment of time couples are called *intensions*; intensions include *propositions*, *properties*, *relations-in-intension*, *individual offices*, etc. (Among non-intensions, the best known are classical unary or binary truth-functions, identity relation between ξ-objects, quantifiers as subclasses of classes of ξ-objects.)

1.3 Deduction

Tichý developed a deduction system with constructions (see papers in Tichý 2004). Because of partiality, classical derivation rules are a bit modified, yet they are still rather classical. Derivation rules exhibit properties of (and relations between) objects and even certain properties of their constructions. I view definitions as certain $\Leftrightarrow$-rules.

1.4 Explication of meaning

In order to explicate meanings of (natural) language, Tichý employed a *semantic scheme* prècised as follows:

[1] Some compositions v-construct nothing.

[2] Of course, any STT is immune to Russell's paradox.

an expression E
  | E *expresses* (*means*) in L:
a construction C = the *meaning* of E in L
  | C *constructs*:
an intension / non-intension = the *denotatum* of E in L

Empirical expressions ('the Pope', 'tiger', 'It rains in Nice', ...) denote intensions; non-empirical expressions ('not', '3', ...) denote non-intensions. The value of an intension in a possible world W at a time-moment T is the *referent*, in L, W and T, of an empirical expression. The denotatum (in L) and referent (in L, W, T) of a non-empirical expression are construed as identical.

To provide an example, the expression 'The Pope is popular' expresses the construction $\lambda w \lambda t$ [^{0}Popular$_{wt}$ ^{0}Pope$_{wt}$]. The construction constructs a proposition which maps world-time couples to T or F or nothing (truth-value gap). The proposition is the denotatum of the sentence in L. A particular value (if any) of the proposition in W, T is the referent of the sentence in L, W, T.

Well-known arguments show that intensional or 'sentencialistic' analyses of belief sentences (and other hyperintensional phenomena) are wrong. Tichý thus suggested to construe belief attitudes as attitudes towards constructions of propositions (not towards mere propositions or expressions): an agent only believes the construction expressed by the embedded sentence (and no other, though equivalent, construction). For instance, the sentence

"X believes that the Pope is popular"

expresses the (2nd-order) construction

$\lambda w \lambda t$ [^{0}Believe$_{wt}$ ^{0}X $^0\lambda w \lambda t$ [^{0}Popular$_{wt}$ ^{0}Pope$_{wt}$]].

Note that $^0\lambda w \lambda t$ [^{0}Popular$_{wt}$ ^{0}Pope$_{wt}$] constructs just $\lambda w \lambda t$ [^{0}Popular$_{wt}$ ^{0}Pope$_{wt}$]. Analogously, "X calculates $3 \div 0$" expresses $\lambda w \lambda t$ [^{0}Calculate$_{wt}$ ^{0}X ^{0}[^{0}3 $^0\div$ ^{0}0]]; the agent is reported to have an attitude towards the procedure [^{0}3 $^0\div$ ^{0}0], not to its (non-existing) numerical result. Such explicit 'mentioning' of constructions by trivialization and other ways of constructing of constructions (e.g. *via*quantification over them) leads to the ramification of STT.

1.5 Tichý's type theory

For precise definition of Tichý's (ramified) type theory (TTT) see (Tichý 1988, ch. 5). TTT has three layers:

1. STT (given above) which classifies first-order objects;

2. *1st-(2nd-, ..., n−)order constructions* (i.e. members of types $*_1$, $*_2$, ..., $*_n$, respectively) are constructions of 1st-(2nd-, ..., n−1-)order objects (or constructions);

3. functions from or to constructions (they belong, e.g., to the type $(*_1\tau)$).

The second level resembles to a Russellian ramified TT (*RTT*). Several kinds of cumulativity are inherent in TTT (e.g., every k-order construction is also a k+1-order construction). Known objections raised against Russell's RTT can be easily dismissed but one has to utilize a bit richer TTT than TTT over B_{TIL}.

I understand TTT as implementing four *Vicious Circle Principles* (hereafter *VCP*s).[3] Each of them is in fact a consequence of the *Principle of Specification*: one cannot precisely specify an item by means of the item itself (already Russell stated such claim). The *Functional VCP*: no function can contain itself among its own arguments or values (*cf.* the layer 1.). The *Constructional VCP*: no construction can (v-)construct itself (*cf.* 2.; this VCP resembles to that of Russell); to illustrate, a variable c for constructions cannot be in its own range, it cannot v-construct itself – otherwise it would not be specifiable. The *Functional-Constructional* VCP: no function F can contain a construction of F among its own arguments or values (*cf.* 3.). The *Constructional-Functional VCP*: no construction C can (v-)construct a function having C among its own arguments or values (*cf.* 2. and 3.).

Concluding the section 1.: unlike logical systems of rivalling solutions to SPs, it is explicitly stated what meanings are; the semantical theory is hyperintensional (not intensional or extensional), i.e. its underlying TT is ramified; the system is rather classical – bivalency and other classical logical laws are accepted, yet partiality is treated (thus logical laws are adapted).

2 Explicit approach

2.1 Language as hierarchy of codes

Language can be viewed as a normative system, such that people who conform to it are capable to exchange, communicate pieces of information. For our purposes it is sufficient to model language (in a synchronic sense) simply as a function from (Gödelized) expressions to meanings. Within TIL, a *k-order code* L^k is a function from real numbers to k-order constructions, it is

[3] E.g. (Raclavský 2009).

an $(*_k\tau)$-object (Tichý 1988, 228); there are various 1st-, 2nd-, ..., n-order codes (*ibid.*).[4]

However, it is not sufficient to model (say) English by a single, say a 1st-order, code. Rather, a whole *hierarchy of codes* (called 'family' in Raclavský 2009) should be invoked as a model of English. The key reason consists in that English as a natural language is capable *to code*, to express by some of its expression, constructions of higher orders.

It has a connection with an important fact about codes. No construction of L^1, most notably ${}^0L^1$, is among constructions expressible-codable in L^1. Recalling the Functional-Constructional VCP, if ${}^0L^1$ would be a value of L^1, L^1 were not be specifiable at all. Unfortunately, ${}^0L^1$ is naturally understood as the meaning of "L^1", the name of L^1. Thus when explicating "... in English ..." as expressive of [... ${}^0L^1$ ...], we need to take into account a higher-order code in which [... ${}^0L^1$ ...] is expressible.

From the just stated fact that *no construction of a k-order code L^k is codable in L^k* (only in a higher-order code) it follows that no expression referring to L^k is endowed with meaning in L^k (only in a higher-order code). By the compositionality-of-meaning principle, *no expression E, the subexpression of which refers to L^k, is endowed with meaning in L^k.*

Not any class of codes (of distinct orders) counts as a hierarchy of codes by which a particular language can be explicated. Some conditions should be imposed. A particular hierarchy of codes involves n codes L^1, ..., L^n such that:

a. they are of n mutually distinct orders;

b. each expression having a meaning in L^k has the same meaning in L^{k+1};

c. an expression lacking meaning in L^k can be meaningful in L^{k+1}.

Of course, most of everyday communication takes place in the 1st-order code L^1 of a hierarchy. Higher-order coding means (e.g. L^2) of a hierarchy are invoked rarely – only when one comments parts of (say) English by means of the other parts of English (in this way I implement the universality-of-language principle).

Some remarks. Every code of the same hierarchy shares the same expressions (no predicates are forbidden); quantification over all of them is unrestricted. Due to the order-cumulativity of objects, every k-order code is also a k+1-order code, thus the type $(*_n\tau)$ includes (practically) all codes of the hierarchy; we can quantify over them. A hierarchy of codes is a certain class (it is an $(o(*_n\tau))$-object); thus one can quantify even over families.

[4]Let me add that any grammatically correct composition of atomic expressions is included in a *rich code*.

Finally, a hierarchy of codes is a 'system' of coding vehicles, not a particular vehicle ('language'); thus we investigate meanings of expressions in the members of a hierarchy, e.g. in L^n, not in the hierarchy as a whole.

2.2 Explication of semantic concepts

According to the 'explicit approach', semantic concepts (concepts of semantic properties and relations) are explicated as explicitly relative to language-code. Here are some sample definitions[5] (e.g. Raclavský 2009):

$$[{}^0\mathrm{TheMeaningOfIn}^n\ n\ l^n] \Leftrightarrow^{*n} [l^n\ n]$$

$$[{}^0\mathrm{TheDenotatumOfIn}^{\xi}\ n\ l^n] \Leftrightarrow^{\xi} [{}^0\Gamma^{(\xi * n)}\ [l^n\ n]]$$

$$[{}^0\mathrm{TheReferentOfIn}^{I\zeta}{}_{wt}\ n\ l^n] \Leftrightarrow^{\zeta} [{}^0\Gamma^{(\xi * n)}\ [l^n\ n]]_{wt}$$

The construction $[l^n\ n]]$ v-constructs the value (if any) of an n-order code L^n for the expression E, i.e. E's meaning in L^n. The function $\Gamma^{(\xi * n)}$ maps any n-order construction C^n to the ξ-object (if any) v-constructed by C^n.

Truth can be construed as a property of propositions, constructions, and expressions (all defined in Raclavský 2008). Truth as a *property of propositions* can be defined as follows, having thus 2 kinds of such properties (p ranges over propositions):

$$[{}^0\mathrm{True}^{\pi\mathrm{P}}{}_{wt}\ p] \Leftrightarrow^{\mathrm{o}} p_{wt}$$

$$[{}^0\mathrm{True}^{\pi\mathrm{T}}{}_{wt}\ p] \Leftrightarrow^{\mathrm{o}} [{}^0\exists\ \lambda o\ [[o\ {}^0{=}\ p_{wt}]\ {}^0\wedge\ [o\ {}^0{=}\ {}^0\mathrm{T}]]].$$

The first defined concept is a concept in the *partial* ('^{P}') sense: a proposition P can be neither true$^{\pi P}$ or false$^{\pi P}$; the latter is a concept in the *total* ('^{T}') sense: a proposition P is true$^{\pi T}$ or not true$^{\pi T}$. Truth as a *property of constructions* have 4 kinds (each having n instances); a construction C^n is true^{*n} in W, T iff it v-constructs a proposition which is true$^{\pi}$ in W, T.

On the other hand, truth as a *property of expressions* is relative to a particular language-code (6 principal kinds):

$$[{}^0\mathrm{TrueIn}^{\mathrm{P}}{}_{wt}\ e\ l^n] \Leftrightarrow^{\mathrm{o}} [{}^0\mathrm{True}^{\pi\mathrm{P}}{}_{wt}\ [{}^0\Gamma^{(\pi * n)}\ [l^n\ e]]]$$

[5] Definitions can be aptly viewed as explications of the respective intuitive concepts.

$[^0\text{TrueIn}^{\text{T}}{}_{wt}\ e\ l^n] \Leftrightarrow^{\text{o}} [^0\exists\ \lambda o\ [[o\ {}^0{=}\ [^0\Gamma^{(\pi * n)}\ [l^n\ e]]\ {}^0\wedge\ [o\ {}^0{=}\ {}^0\text{T}]]]$.

Note the interrelation of truth and the other basic semantic concepts: an expression E is true in L^n, W, T iff E expresses-means in L^n a construction of a proposition which is true$^\pi$ in W, T, i.e. E refers (in L^n, W, T) to T.

2.3 Solution to SPs

Let me illustrate the solution to particular SPs on the example of the (Belnap) Paradox of Adder. Its paradox-producing expression is this:

D: '1 + the denotatum of D'.

(The paradox: D denotes N; N=1+the denotatum of D, i.e. $N = 1 + N$; but this is impossible because the adding-one function has no fixed point.)

Here is my critical examination of the paradox:

a. If we do properly understand D, we have to bring out in which language-code the denotation of D proceeds.

b. One thus *disambiguates* D to (say) '1 + the denotatum of D in L^1' (hereafter simply D).

c. Thus our understanding of D takes place in the (say) 2nd-order code L^2 of English.

d. In L^2, D means the 2nd-order construction $[^0 1\ {}^0{+}\ [^0\text{TheDenotatumOfIn}^{\tau}{}_{wt}\ {}^0\ulcorner\text{D}\urcorner\ {}^0L^1]]$ (where ${}^0\ulcorner\text{D}\urcorner$ constructs the Gödelian number of D).

e. Being a 2nd-order construction, it cannot be expressed by D already in the 1st-order code L^1, thus D is without a meaning in L^1.

f. Lacking meaning in L^1, D has no denotatum in L^1.

g. The construction $[^0 1\ {}^0{+}\ [^0\text{TheDenotatumOfIn}^{\tau}{}_{wt}\ {}^0\ulcorner\text{D}\urcorner\ {}^0L^1]]$ constructs nothing at all because the addition function obtains no suitable argument, since $[^0\text{TheDenotatumOfIn}^{\tau}{}_{wt}\ {}^0\ulcorner\text{D}\urcorner\ {}^0L^1]$ constructs nothing.

h. The premise of the paradox, that D denotes a number N, is refuted.

Quite analogously for various Liars, e.g. S: "S is not true". The 2nd-order construction $\lambda w \lambda t\ [^0\neg\ [^0\text{TrueIn}^{\text{T}}{}_{wt}\ {}^0\ulcorner\ \text{S}\urcorner\ {}^0L^1]]$ is not expressible in L^1, but in L^2. In L^2, S denotes a false$^\pi$ proposition because there is no true$^\pi$ proposition denoted by S already in L^1. Hence I reject the premise of the respective paradox that the proposition denoted by S can be true$^\pi$.

Contingent or strengthened versions of SPs make no counter-examples for this kind of solution. All known principal paradoxes of denotation and

reference are solved in (Raclavský 2009, 2011). All kinds of the Liar are solved in (Tichý 1988, section 44), (Raclavský 2009a).

Such solution seems to be a certain mix of 'golden' ideas of Russell (VCP, hierarchy of propositional functions), Tarski (language/metalanguage) and perhaps also Kripke (partiality of a truth-predicate). Yet there are also significant dissimilarities. Unlike Russellian RTT, TTT treats both 'extensional' and 'intensional' functions; the latter ones, viz. constructions, are carefully individuated. Unlike in Tarski, language is explicated as a system of expressions coding meanings-constructions (which conform to the respective VCPs); moreover, semantic concepts are explicated as explicitly language-relative. Unlike in Kripke, semantic concepts in the total sense are explicated as well.

The important conclusion of the explicit approach: *semantic concepts-constructions involving a construction of a k-order code L^k are not expressible-codable in (sufficiently rich) L^k*. Thus *every code (of a hierarchy) is limited in its expressive power.*

3 Implicit approach

One may raise the following objection to the explicit approach. As a solution to SPs the explicit approach rightly applies only to those paradox-producing expressions in which language is explicitly referred to; however, typical paradox-producing expressions need no disambiguation to the form in which language is explicitly referred to; hence, a number of SPs remains in fact unresolved.

I can admit such objection. Nevertheless, I still claim that there is always at least *implicit* relativity to language of such semantic terms (and the terms are *ambiguous* after all).

In order to admit the objection, the following principle has to be adopted:

> For every k+1-order construction of a property (relation) of expressions which involves a construction of a code L^k (that is l^k, ${}^0L^k$, etc.), there is an equivalent (v-congruent) k-order construction of the very same property (relation) involving no such construction of a code L^k.

To illustrate the principle, the 2nd-order construction:

$\lambda w \lambda t \lambda n\ [{}^0\neg\ [{}^0\mathrm{TrueIn}^{\mathrm{T}}{}_{wt}\ n\ {}^0L^1]]$

is equivalent to the 1st-order construction:

$\lambda w \lambda t \lambda n\ [{}^0\neg\ [{}^0\mathrm{True}^{\mathrm{T}L1}{}_{wt}\ n\ {}^0L^1]]$.

Realize that $\lambda w\lambda t\lambda n\ [^0\neg\ [^0\mathrm{True}^{\mathrm{T}L1}{}_{wt}\ n\ {}^0L^1]]$ *is definable* by means of $\lambda w\lambda t\lambda n$ $[^0\neg\ [^0\mathrm{TrueIn}^{\mathrm{T}}{}_{wt}\ n\ {}^0L^1]]$. Note also that "$L1$" in "$^0\mathrm{True}^{\mathrm{T}L1}$" indicates that the respective concept is related just to L^1, not to any other code (it is the definiens which shows that, i.e. removes the ambiguity of the respective intuitive concept). There is a number of such *implicitly language-relative semantic concepts*; my way of their explication is obvious.

Now, the expression "not true" (without "in") expresses in some code of the hierarchy the construction $\lambda w\lambda t\lambda n\ [^0\neg[^0\mathrm{True}^{\mathrm{T}L1}{}_{wt}\ n]]$ (i.e. "not true" is not disambiguated, e.g., to "not true in L^1").

However, there is a danger of *revenge* of a paradox if one assumes that "not true" expresses this construction already in the 1st-order code L^1.

It is readily seen that the Functional-Constructional VCP and related principles are incapable to preclude the revenge (as they do in explicit cases). Thus I can appeal here to nothing but the *proof* – easily generalizable from Tichý's *Corollaries 44.1-4* (Tichý 1988, 292-293) – that a k-order code cannot code constructions like $\lambda w\lambda t\lambda n\ [^0\neg\ [^0\mathrm{True}^{\mathrm{T}Lk}{}_{wt}\ n]]$.[6]

Here is the crucial idea of the proof. Assume, for *reductio*, that S expresses in L^1 a construction of a (total) proposition P, thus S denotes (in L^1) P; however, the construction $\lambda w\lambda t\ [^0\neg\ [^0\mathrm{True}^{\mathrm{T}Lk}{}_{wt}\ {}^0\ulcorner\mathrm{S}\urcorner]]$ constructs a (total) proposition Q which is true$^\pi$ if the proposition denoted by S in L^1 is not true$^\pi$ (Q is false$^\pi$ if the proposition denoted by S in L^1 is true$^\pi$); thus P cannot be identical with Q, hence S cannot express in L^1 a construction identical (or, more broadly, equivalent) with $\lambda w\lambda t[^0\neg\ [^0\mathrm{True}^{\mathrm{T}Lk}{}_{wt}\ {}^0\ulcorner\mathrm{S}\urcorner]]$.

How to explain this fact? As we have seen, concepts-constructions such as $\lambda w\lambda t\lambda n\ [^0\neg\ [^0\mathrm{True}^{\mathrm{T}L1}{}_{wt}\ n\ {}^0L^1]]$ are definable by means of constructions explicitly employing the code L^1. It follows that such concept is *relative to language as code after all*. Indeed, $\lambda w\lambda t\lambda n\ [^0\neg\ [^0\mathrm{True}^{\mathrm{T}L1}{}_{wt}\ n\ {}^0L^1]]$ and $\lambda w\lambda t\lambda n\ [^0\neg\ [^0\mathrm{TrueIn}^{\mathrm{T}}{}_{wt}\ n\ {}^0L^1]]$ construct *one and the same property which is related to* L^1. (Hence, *all semantic properties and relations are relative to language as code.*) The purpose of any code is to discuss matters external to it; it is *not purpose of a code to discuss its own semantic features* (*cf.* Tichý 1988, 231). We thus concluded, similarly as in the previous section, that *every code is limited in its expressive, coding power* (*cf. ibid.*, 233).

The final conclusion. Tarski's famous *Undefinability Theorem* says that semantic predicates concerning L are not definable in L. The TIL-approach to semantic concepts fully confirms it. Of course, it is added that the respective concepts are definable (the constructions exist and they may even construct something), yet they cannot be coded-expressed in a sufficiently rich L.[7]

[6] Analogously for other semantic terms and concepts.

[7] A remark. A partial truth-predicate could be added only to that object language-code which has a *limited expressive power* (natural language is not such), i.e. a language not allowing to form a (meaningful) total *un*truth-predicate from the partial truth-predicate or a language not containing any equivalent of the total untruth-predicate. To illustrate

References

[1] Materna, P. (2004). *Conceptual Systems*. Berlin: Logos.

[2] Raclavský, J. (2008). Explications of Kinds of Being True (in Czech). *SPFFBU* B, *53*(1), 89-99.

[3] Raclavský, J. (2009). *Names and Description: Logico-Semantical Investigations* (in Czech). Olomouc: Nakladatelství Olomouc.

[4] Raclavský, J. (2009a). Liar Paradox, Meaning and Truth (in Czech). *Filosofický časopis*, *57*(3), 325-351.

[5] Raclavský, J., Zouhar, M. (2011). Paradoxes of Denotations and Reference. (*ms.*).

[6] Tichý, P. (1976). *Introduction to Intensional Logic*. (unpublished *ms.*).

[7] Tichý, P. (1988). *The Foundations of Frege's Logic*. Walter de Gruyter.

[8] Tichý, P. (2004). *Pavel Tichý's Collected Papers in Logic and Philosophy*. V. Svoboda, B. Jespersen, C. Cheyne (eds.), University of Otago Press, Filosofia.

the second possibility, consider $[^0\text{Babig}_{wt}\ n] \Leftrightarrow^o [^0\exists\ \lambda o\ [[o\ \ ^0= [^0\text{TrueIn}^{\text{P}}{}_{wt}\ n\ ^0L^1]\ ^0\wedge [o\ \ ^0= \ \ ^0\text{T}]]]$ (the definiens is in fact a total concept of truth of expressions); one cannot safely add the predicate "babig" so defined to the object-language L^1.