Baltic J. Modern Computing, Vol. 11 (2023), No. 1, 161-180
https://doi.org/10.22364/bjmc.2023.11.1.10

Towards Computer-Assisted Latvian ICT Terminology Development: from Theoretical Guidelines to Case Studies

Dace ŠOSTAKA, Juris BORZOVS

University of Latvia, Raiņa bulvāris 19, Riga, LV–1586, Latvia

{dace.sostaka, juris.borzovs}@lu.lv

Abstract. Currently, Information and Communication Technology terms are mainly created in English, and the secondary term formation is carried out in Latvian. Nevertheless, in our rapidly changing world, the number of ICT terms that need to be formed in Latvian exceeds the current capacity of ICT terminologists. Thus, this paper aims to provide insight into the historical context of ICT term formation in Latvia in general, describe the current ICT secondary term formation process in Latvia in particular, provide an insight into the case study of terminology commission meeting and terminology discussion process, and explain the need for more productive secondary term formation of ICT terms in Latvian. The paper emphasises the need for further research on determining the parts of a secondary term formation process that could and should be automated as soon as possible to increase the rate of secondary term formation in Latvian.

Keywords: terms, terminology, secondary term formation, English, Latvian, Information and Communication Technology, automation, case study

1. Introduction

What changes, endures (Pastāvēs, kas pārvērtīsies)
Rainis, Latvian national poet

The article deals with Information and Communication Technology (further in the text - ICT) terminology development in Latvia in general and with the process of secondary term formation in the ICT field in particular.

The article provides insight into the history of ICT terminology development in Latvia, briefly describes the current secondary-term formation process in Latvia, and introduces the current challenges and solutions; this is an extended and updated version of our article (Šostaka and Borzovs, 2022) "Towards Computer-Assisted Latvian ICT Terminology Development", presented at the conference Baltic DB&IS 2022 Doctoral Consortium and Forum, July 03–06, 2022.

The primary focus is on describing the decision-making process of the terminologists while looking for appropriate secondary-term in Latvian.

A secondary term will be formed when a new concept and term denoting the concept is already created in a source language (predominately English in the ICT field), and the need has arisen to create a term in a target language (in this article - Latvian). For example, in English, there is a concept describing an "electronic machine [..] for doing calculations [..]", and the term formed in English is "computer" (WEB, g); this is called a primary term formation. Then, in Latvian via, first of all, comprehension of the concept, namely "Tehniska sistēma [..], kas veic automātisku datu apstrādi [..]" is formed the term "dators" (WEB, f); this is called a secondary term formation.

The secondary focus is on the case study of the terminology creation process during the terminology commission meeting.

The topicality of the current research lies in several aspects.

First of all, in the relatively short run, the next few years, the current research could be helpful as terminology formation guidelines for ICT field professionals and translation companies that deal with ICT terminology, usually within a limited time frame (when it is necessary to decide within a few hours maximum what terminology unit will be created, in order to comply with real-work deadlines where defined work-output per day for is steadily increasing but human resources available keep dwindling due to numerous reasons, among them financing allocated for the translators in Latvia and European Union) and human resources, when it is little or no time to choose official terminology creation process that usually takes at least several days.

Next aspect, the same principles that are discussed in at least partially automated ICT terminology creation might be transferred and of good use for other fields – economics, music, forest management and others, thus facilitating the terminology and language development in these fields.

Last but not least, in the long run, if we speak for the next few decades, we would like to remind the significance of terminology in cross-culture and cross-industry communication. It is well-known that for academic communities and various industries, precise communication and understanding among different languages and cultures, exact usage of the terminology is of utmost importance in order to communicate concepts correctly and thus understand each other. Although, in the last few years, there has been made significant progress in the quality of machine translation (*GoogleTranslate, AmazonTranslate, Tilde, DeepL* and others), still one of the fundamental aspects of preciseness provided by machine translation is the human work invested in the creation of terminology, which is used in the high-quality machine translation.

The current work of secondary term creation could be more productive and effective if the mechanical, most time-consuming and repetitive parts of the secondary term preparation process were automated, thus freeing up resources for creative work. When it is impossible to describe precisely and relatively quickly the specific concept needed for the communicative situation, people will mainly switch to the language where these words and terms already exist for describing the concepts.

Currently, ICT terms are predominantly primarily formed in English. When there is a severe lack of necessary ICT terms in Latvian for communicating new concepts, speakers will use English terms to understand each other as quickly as possible. Still, the contemporary demand for creating high-quality secondary formed terms exceeds the current capacity of ITTEA, Information and communication technologies sub-commission of the Terminology Commission of the Academy of Science of Latvia (Latvijas Zinātņu akadēmijas Terminoloģijas komisijas Informācijas tehnoloģijas, telekomunikācijas un elektronikas terminoloģijas apakškomisija).

Thus, to facilitate and guide the systematic secondary term development process, we will research the secondary term creation process the ITTEA commission employs right now, intending to determine the most time-consuming, mechanical and redundant parts of the secondary term creation process.

In conclusion, insight into possible ICT terminology development scenarios for Latvia and possibilities for future research are provided. The need to identify and automate as much as possible parts of secondary term formation is emphasised.

The structure of the paper is as follows: basic concepts of term formation (Section 2), related research on ICT secondary term formation and editing the text with CAT tools (Section 3), the origins of ICT terminology in Latvia (Section 4); description of the actual ICT term formation methodology and process (Section 5) and a case study (Section 6), future research (Section 7), conclusion (Section 8).

2. Terminology unit, primary and secondary term formation

When we discuss term formation, it is essential to distinguish the concepts "term formation", "primary term formation", and "secondary term formation".

Terminology unit or term, as I. Zauberga explains, is the word used for "describing a concept in a specific area" (Zauberga, 2016); the monolingual English-English dictionary (WEB, l) defines "term" as "a word or expression that has a precise meaning in some uses or is peculiar to a science, art, profession, or subject".

Primary term formation, according to J. C. Sager, accompanies "concept formation and is therefore monolingual" (Sager, 1990). The further described process will reflect forming a terminology unit in the target language.

In the context of this article, to understand secondary term formation better, let us look at the term "computer" in the source language (English). It will be a term with the following definition "an electronic machine that is used for storing, organizing, and finding words, numbers, and pictures, for doing calculations, and for controlling other machines" (WEB, g).

According to J. C. Sager, *secondary term formation* "occurs when a new term is created for a known concept [..] as a result of knowledge transfer to another linguistic community" (Sager, 1990). Thus, when in Latvian term "dators" is created with a definition "Tehniska sistēma (ierīču komplekts), kas saskaņā ar uzdotu programmu veic automātisku datu apstrādi un ievadizvadi." (WEB, f), then the secondary term formation in the target language (Latvian) takes place.

3. Secondary term formation in the ICT field and usage of CAT tools

To speak about a specific matter, we need to agree on the meaning of the words or, in specific usage, terms. It is possible to describe an object or a process straightforwardly, but it takes much time and effort. Thus, term and terminology is a time-saving way to agree on how we will name things in a specific field.

There is extensive research and information on neology, or, to be more specific, on term creation or primary term creation available (Sager, 1990), (Kageura, 1998), (Zauberga, 2016); there is available research, theoretical and methodological

considerations on secondary term creation in Greek (Floros and Grammenidis, 2012), Greek and German (Koliopoulou, 2020) and other languages.

Nevertheless, there can be found relatively little research on *ICT* secondary term formation - in Romanian (Postolea, 2012) and Sinhala language (Widyalankara, 2015); ICT term formation is also discussed in research about term creation in Northern Sotho (Mojapelo, 2018). There is related research in translation automation and the use of ICT and CAT tools (Garcia, 2014) - various computer-assisted translation tools - TRADOS, Memsource and others that enable the transformation of translation activity from translating manually to the next technocratic level.

Namely, if we draw parallels with the history of the translation process and its automation, then now we have evolved (at least for technical texts, manuals and legal texts) from "monk-scribe", as Umberto Eco once upon a time called medieval writers and translators, to translators who translated writing by hand and typewriter, then moved to various incarnations of word documents where already it was an achievement that written text could be changed with "find and replace" (for example, person name, place name, spelling). Then there was the development of the translation programs where text segments could be translated (taken from the translation memory), and the translator's work was made more effortless– the translator had to evaluate the offered translation of the segment and either choose and confirm the offered version or decline and enter translator's version.

Even if the partially automated, carried out either by CAT tools or with the help of machine translation; even if the translation process is not perfect, it still speeds up the translation process significantly. We have arrived at the moment when human involvement in the translation process (as long as we speak about technical texts with a large percentage of repetitiveness, not belles-lettres, poetry and other types of creative texts) consists mainly of pre-editing and post-editing.

Various aspects, namely editing (on the macro-level and micro-level) of the translation process as well as guidelines and evaluation for the post-editing, were already described nine years ago (O'Brien et al., 2014), and the current and actual state of pre-editing and post-editing for machine translation is described three years ago (Arenas, 2020). Currently, there is the European Association of Machine Translation and conferences dedicated to machine translation; and the most recent one took place less than a year ago, in the spring of 2022 (WEB, a).

Thus, we envision that what can be applied to the translation process in order to speed it up and make it more consistent can and should be applied to the terminology creation process in order to free up more human resources for the creative part of the term formation and reduce the manual, mechanical work of preparing the English terms for the discussion within Latvia's Information and communication technologies sub-commission.

4. The origins of ICT terminology in Latvia

Although the rapid development of terminology in Latvia started more than a hundred years ago, in the first decade of the 20th century (Veisbergs, 2021), the ICT terminology developed in the 1960s with the beginning of the IT-based industry, namely, with the establishment of the Institute of Mathematics and Computer Science, the University of Latvia (WEB, b) and the Institute of Electronics and Computing (WEB, c).

Nevertheless, as the computing devices were accessible to a very limited number of people (WEB, n), the terminology was mainly used by a limited number of professionals as well mathematicians, students of the Faculty of Physics and Mathematics, and researchers who worked in the field of computing. Therefore, no systematic term formation work was carried out.

With the collapse of the Soviet Union and economic and political changes around 1992, personal computers and the Internet became accessible to the broader community. This was the turning point when user-guided term-creation began.

If there was a computer and a modem connected 9600 bytes per second, then the owner of the computer and modem had to be also a terminologist and somehow name the device and the action taken. Usually, naming resulted in creating calque, transliterating or transcribing the English ICT term and adding Latvian ending, for example: "mouse" – "pele", "computer" – "kompjūters", "file" – "fails", "router" – "rūteris", "save" – "seivot", "start" – "startēt", "connect" – "konektēties".

The determinant moment in primary and secondary ICT term development came in 1992 when systematic terminology work started with the foundation of ICTS. Three decades later, in February 2023, more than 9000 terms are currently approved. The existing terminology-formation work, its typical challenges and solutions are analysed in (Šostaka et al., 2023).

Let us look closer at the current ICT term development situation in Latvia.

In the last decade, there has been a rapid evolution of the IT field in the world (Jaeger, 2019), (Desbiens, 2023), (Xiaoying et al., 2023), (Gunjan and Zurada, 2023) and Latvia (Bāliņa, 2022), (WEB, e) and prognoses for near future foresee continuous IT field development (WEB, d). With the evolution of the IT field, the terminology describing new concepts evolves as well; therefore, the need remains and increases for secondary-formed ICT terms in Latvian. Therefore, there is a need for ICT terms in Latvian; if we do not develop the terms, the users will form them, and we will return to the year 1992, with calques, transcriptions and borrowings as the prevailing method of term formation.

Presently, terminology formation work is done by ITTEA sub-commission members, largely manually: looking up the definitions of the term, searching for existing translations in dictionaries, thesaurus, parallel texts, et cetera and combining them. We will look (in section 5) in detail at existing algorithms used by two ITTEA sub-commission members while searching for possible variants of rendering English ICT terms in Latvian.

ITTEA mainly deals with the following issues:

1. ISO 2382 (Information technology - Vocabulary, 3000+ terms);
2. ISO/IEC/IEEE 24765:2017(E) ISO Systems and software engineering vocabulary (more than 4500 terms);
3. Machine Learning and Computer Linguistics;
4. Internet of Things and Smart Technology from the European Union;

5. Current issues – answering questions regarding terminology formation we receive from the Latvian Language Agency, European Commission and other institutions.

Secondary term formation methodology guidelines for ICT terms in Latvian, in general, are as follows: we aspire to maintain a systemic approach in the development of the Latvian ICT terminology, defined by Latvian terminologist Valentīna Skujiņa in her monography (Skujiņa, 1993):

1. Using already existing terms;
2. Using existing terms as a part of the new term – compounds, hyphenated compounds and others;
3. Coining new terms (neology: primary and secondary term formation).

When preparing terms for secondary term formation, we:

1. Analyse terms in the context of a specific sub-domain;
2. Consult industry professionals;
3. Take into account the frequency of use of the term.

5. Secondary term formation methodology and process: in particular

Dr. math. Jānis Cīrulis (WEB, h), a member of ITTEA, narrates his terminology formation process as follows.

For an easier understanding of the terminology formation process, this narration is structured in the tables (1., 2., 3.,4.), where the "user demand" is described in the title of the table and the actions taken are listed in the table contents.

J. Cīrulis says that in case the English ICT terminology unit is not known to him, and neither has he understood the meaning of the concept described (please, check Table 1), he begins by looking up monolingual, general dictionaries: Oxford English Dictionary (British English) and Merriam-Webster (American English). He also uses search engines, for example, OneLook Dictionary Search (it indexes more than 900 dictionary sites) and Google (Google Search), using the following request "<term>, the definition" to understand the general meaning or several meanings of the word (or part of the collocation).

Table 1. Searching for a general concept of terminology unit

Source type	Source title	Source URL
Dictionary	Oxford English Dictionary	https://www.oed.com/
Dictionary	Merriam-Webster	https://www.merriam-webster.com/
Search engine	OneLook Dictionary Search	https://onelook.com
Search engine	Google Search	https://www.google.com

Then, to comprehend the specific meaning and usage (please, check Table 2), he uses search engines (Google, Google Scholar), as well as Wikipedia, which is useful for quick reference and general insight into a new field.

Table 2. Searching for the specific concept of terminology unit.

Source type	Source title	Source URL
Search engine	Google	https://www.google.com/
Search engine	Google Scholar	https://scholar.google.com/
Online encyclopedia	Wikipedia	https://www.wikipedia.org/

When he has achieved the moment when it seems that the general meaning is understood, it is possible to start looking for a corresponding functional analogue in Latvian. Still, there is know-how: if the term is related to the ICT field, he waits for the opinion of more knowledgeable colleagues.

Looking for functional analogues differs significantly for terms he:

1) is already familiar with;

2) encountered for the first time.

In the first case, he searches his memory, looking for existing solutions in the collateral branches: mathematics, physics, electronics and others. Is the term already translated into Latvian? If it is translated, then how exactly?

For the second case, he remarks on frequent challenges and possible solutions.

First, when dealing with word-group terms or collocations, even if the components of word-group are familiar, it is essential to understand the syntactic structure of the English term.

Second, a precise comprehension of the term (although definitions tend to be quite different) is facilitated by using a clear-cut definition of the term (if it can be found) and checking the usage of the word-group term in the actual context, namely searching for the term in parallel texts.

Third, when looking for the possible functional analogue of the term in the target language (Latvian), it helps to find the possible solution by looking up and comparing existing secondary ICT terms in other synthetic languages (French, German and others), which use inflexions to express syntactic relations in the sentence. Examples of other languages are also helpful for creating a more appropriate term for its definition.

For example, "computation data use". After thorough research of the term and its definition, it is possible to firmly insist that "computation" is not an adjective for "data", but it is an adjective for the established word-group term "data use"; thus, it is not related to "computing data", "actual data" or any other kind of "data".

In conclusion, J. Cīrulis remarks he has benefited the most from Wikipedia (having found English–origin word for term creation in Latvian) when looking for appropriate terminology units in Latvian that would correspond to the concept of the term in English, even if it sometimes means giving up the Latvian origin for terminology unit (this attitude contradicts the opinion of part of the ITTEA commission that terminology should be mainly created from words of Latvian origin).

Agnese Apse-Apsīte, also ITTEA commission member, translator and terminologist in EC DGT.C.LV.2 (Latvian Language Unit) European Commission, describes the term formation as follows.

She states that DGT.C.LV.2 are guided by the principle that creating new terms is the last resort, and thus we must first try to find out if the term already exists.

According to her, ICT terms could be divided into two parts:
1. industry-specific, "technical" terms (used by professionals in various fields - programmers, electronics manufacturers, security authorities, and others);
2. popular-science terms widely used in society (such as in conversation, on the website, and others).

They are working with both parts, as the Commission produces
1. detailed technical specifications (LSP – language for specific purposes, professionals writing for professionals) and
2. press releases and teaching materials (popular-science texts).

When encountering ICT terms in the text, the terms searching and terminology formation processes usually are as follows.

In Table 3 term searching process used by DGT.C.LV.2 is structured in order of importance:
1. Interactive Terminology Database for Europe (IATE, 2023) and Latvian National Terminology Portal (WEB, j) as well as the Academic terms database (WEB, f) - all three are considered trustworthy primary sources;
2. the Microsoft language portal (WEB, k) - Microsoft terminology can be found, and results can be evaluated with a more critical approach;
3. Online bilingual concordance (WEB, p) – the appearance of the term in already translated legal acts, in parallel texts, for example; if it is not present or is translated, but it seems to be incorrect;
4. the Latvian National Library (WEB, i) - books in the digital format or physical format; this point does not usually apply to IT terminology because books do not keep up with the times, but we use them a lot in other industries, e.g. chemistry, energy, and others; it is mentioned for the sake of order;
5. the sources found on the Internet - most often, there will be publications in scientific journals, also dissertations, master's theses, and others; they usually have a summary in a foreign language, or the authors, in many cases, indicate in brackets the terms in English.

Table 3. Searching for already existing terminology units

Source type	Source title	Source URL
Terminology database	Interactive Terminology for Europe	https://iate.europa.eu/
Terminology database	Latvijas Nacionālais terminoloģijas portāls	https://termini.gov.lv
Terminology database	Akadēmiskā terminu datubāze AkadTerm	http://www.akadterm.lv/term.php
Terminology database	Microsoft language portal	https://www.microsoft.com/en-us/language
Search engine	Linguee	https://www.linguee.com/

If the term is not found, then the terminology formation process begins.

Let us take a closer look at the way the terminology formation process in the EU is carried out (please, check Table 4).

First of all, we search for a definition (if it is already given in the law) or look for it in:

1. Interactive Terminology for Europe (it may already be created in other languages, and the English term is already defined);
2. the ISO standards database (ISO, 2022);
3. Technology Dictionary (WEB, n), The Government of Canada's terminology and linguistic data bank (WEB, o), Encyclopedia Whatis (WEB, q), in Wikipedia (WEB, r) and similar databases and glossaries;
4. if the definition cannot be found, we look for articles and other information; in other words, we are trying to understand what it is.

Table 4. Terminology formation process

Source type	Source title	Source URL
Terminology database	Interactive Terminology for Europe	https://iate.europa.eu/
Terminology database	The Government of Canada's terminology and linguistic data bank	https://www.btb.termiumplus.gc.ca
ISO standards database	International Organization for Standardization	https://www.iso.org/obp/ui/#home
Dictionary	Technology Dictionary	https://www.techopedia.com/dictionary
Encyclopedia	WhatIs.com	https://www.techtarget.com/whatis
Encyclopedia	Wikipedia	https://www.wikipedia.org

Regarding the secondary term creation process, DGT.C.LV.2 almost always creates "technical" terms using calque or borrowings. In addition, sometimes calques are created not from the source language (English) but from other languages (German, French and others) if it is possible to create a more understandable term. An important aspect here is the translatability of the new term, as professionals often already know the English term. When rendering such terms, the aspect of euphony seldom is considered, but only whether it will be understandable to the target audience and usable in sentences (which sometimes leads to the creation of slightly strange collocations).

Summing up the current secondary terminology-creation processes described by Jānis Cīrulis and Agnese Apse-Apsīte, the activities listed below could be automated to optimize the term-creation process.

In the source language (English), searching and returning results for:
1. the term definition in monolingual English-English dictionaries;
2. the source-language texts;
3. various databases – ISO, IATE and others.

In the target language (Latvian), searching and returning results:
1. English-Latvian dictionaries;
2. Parallel texts in Latvian from search engines;
3. Synonym dictionaries in Latvian (Letonika, Tezaurs and others).

6. Secondary term formation methodology and process: a case study

We have looked at the term-creation strategies used by commission members in general. In order to understand better the general principles applied to the terminology creation process, we will look at the terminology discussion process during the terminology commission meeting.

The discussion in terminology meetings usually is carried out in several stages.

First, the terminology list (as a word document, accessible and modifiable) is prepared, with terms in English, their definitions, already existing terms or parts of the term and suggested Latvian equivalent or equivalents. Second, the schedule for the meeting is sent to the commission members, with a link to the abovementioned document, where commission members can comment upon terms proposed for discussion and express their suggestions for Latvian equivalents. Third, we discuss the proposed terms during the commission meeting each fortnight.

Examples of the case study are from commission meeting number 605. which took place on the 21st of October, 2022. For translated and edited excerpts of the transcript, please see Appendix.

We started the discussion with the terms related to object-oriented programming (further in the text - OOP), proposed for discussion by our commission member Viesturs Vēzis. Twenty terms were selected for analysis. The terms we have decided to discuss in the current meeting are a high-impact factor to the broader society because they will be used in educational materials in the education system: both the school and higher education systems. Thus, the main factors when discussing terminology units are consistency and recognisability.

The first term discussed in the current meeting is the English term “framework”. There is already a secondary formed Latvian term, “satvars” (WEB, f), formed in 2014. Commission members agree to keep a term that was already accepted, namely “satvars”, because it is productive to change it only if there are significant arguments for change.

The following term to be discussed is the term “modifier”. It has not been officially accepted as a term in a programming language. We decided to accept the proposed Latvian term “modifikators”, already in everyday use.

The following terms to be discussed are “static**”** – **“**statisks**”** (WEB, f) and “static variable**”.** The word group (also – collocation) is not approved in Latvian; however, a part of the term already exists, namely the term “variable” (WEB, f), which has been approved three times already: in 2006, 2008 and 2016. Thus, it is decided to recreate the term, combining already existing Latvian terminology units, namely “statisks mainīgais”. There is a brief discussion where Jānis Cīrulis expresses his quest for understanding the term. Namely, the question was how the variable (something changeable) could be “static” (unchangeable). He sums up his comprehension that the "static variable" in the programming environment meant something else. Dace Šostaka sums up that it is agreed upon that the word “static” in the everyday lexicon differs from the word “static” in use as a term in the programming environment.

Next is the term group with the term “member” – “class member”, “object member” and “instance member”. The English term’s **“**member**”** equivalent within the ICT context in Latvian is “elements” (WEB, f); it can be found in the Microsoft Terminology database 2022. Corresponding for the English term **“**class**”** (WEB, f) Latvian term is “klase”; for the English term “object” (WEB, f) corresponds Latvian term “objekts”, the

English term “instance” (WEB, f) already has been approved by ITTEA as “instance” in Latvian. In the written part of the discussion in our online document, different colleagues, who are authorities in the field, agreed that the term “class member” should be rendered as a **“**klases elements**”.**

Still, Jānis Cīrulis opposes the majority; he draws ITTEA member attention to the fact that the comments in the online document at the entry “class member” indicate disagreement among colleagues. He asks to scroll through the electronic document so that it is possible for all the participants to read one more time the place in the document where he has written some excellent examples about the term “member” and its usage in context.

After further detailed discussion and reading the examples, ITTEA members agree that for the clarity of term understanding, it is definitely necessary to indicate the context of the term “class member” in the terminology portal AcadTerm.

The entry will read as follows: the term "class member" is rendered as the term "klases elements" in the context of OOP.

It shall be noted that there are cases in the terminology creation process when it is impossible to create a secondary term that will correspond to all the criteria in the target language, namely, Latvian. Thus, by specifying the context the term will be used, we resolve this possible ambiguity in understanding and using the term; it will help translators to find the necessary terminology unit. The term group with the term "member" are recreated in Latvian as “klases elements", “objekta elements” and “instances elements”.

Although the next term, “field”, has already been rendered in Latvian as “lauks” in 2006 (WEB, f), there is still a rhetorical question about whether to choose another Latvian term for rendering the term "lauks" in Latvian. Jānis Cīrulis suggests keeping the already existing equivalent, “lauks”, while Viesturs Vēzis proposes and asks if it is necessary to add an indicator that the rendition “lauks” will be used in the OOP context. Jānis Cīrulis argues that “field” as “lauks” is also used in numerous industries. Thus, it is unnecessary because if we write the indication that this is “lauks” in the OOP context, we might create a misleading impression that it is not rendered as a “lauks” in other programming languages.

The subsequent two terms, “member variable” and “member function”, are rendered in Latvian as “iekšējais mainīgais” and “iekšējā funkcija”.

It is important to emphasize that using already separately existing parts of terminology word groups to create new terms is possible in that case only when the terms reflect the concept. It is not possible to mechanically take already existing terms and create from them new terminology word-group.

Further, we discuss the subjective understanding of the term “function”. The term “function” is already rendered in Latvian as “funkcija” (WEB, f), term “attribute” is rendered as “atribūts” (WEB, f).

The terms “class variable” is approved as “klases mainīgais” with an indication that is used in OOP context only, “class field” is approved as “klases lauks”, and “class method” is approved as “klases metode”.The term “instance variable” is approved as “instances mainīgais”, “instance field” is approved as “instances lauks”, and “instance method” is approved as “instances metode”. “Polymorphism” can be found in the Akadterm database (WEB, f) as discussed already in 2008, is approved as “polimorfisms”

There is a detailed discussion regarding “encapsulation**”** that has already been approved (WEB, f) in 2008, but at the end of the discussion, we keep the already accepted version “iekapsulēšana”.

In Table 5, there is an excerpt from protocol 605 with terms accepted.

Table 5. Terminology units discussed and accepted

1.	framework	satvars
2.	modifier	modifikators
3.	static	statisks
4.	static variable	statisks mainīgais
5.	class member	klases elements (OOP)
6.	object member	objekta elements
7.	instance member	instances elements
8.	field	lauks
9.	member variable	iekšējs mainīgais
10.	member function	iekšēja funkcija
11.	function	funkcija
12.	attribute	atribūts
13.	class variable	klases mainīgais (OOP)
14.	class field	klases lauks
15.	class method	klases metode
16.	instance variable	instances mainīgais
17.	instance field	instances lauks
18.	instance method	instances metode
19.	polymorphism	polimorfisms
20.	encapsulation	iekapsulēšana

In conclusion, from the brief insight in the case study, it can be seen from the actual term discussion process during the ITTEA meeting that sometimes it can take relatively much time to define one term.

For example, coming to an agreement among commission members regarding the term "class member" took time, explaining the personal understanding of the concept in the source language (English) and target language (Latvian). It was a joint effort to agree on "klases elements" within the context of OOP as an appropriate secondary ICT term in Latvian. Nevertheless, such discussions significantly facilitate consistent and systematic secondary term formation work in the term system, namely when recreating terms within word-group "class".

Thus, even if lengthy discussions take up much time in the short term, they are of high importance for creating and maintaining the systematic approach in recreating the secondary ICT term system in Latvian.

7. Future research: potential development of the ICT secondary-term formation

There exist several avenues for future research of at least partially automating ICT terminology creation into Latvian. First of all, although there have been successful voting for various terms on the *AcadTerm* homepage (for example, the term "clickbait") and Facebook group for translators, it could be helpful to create and use an online voting mechanism designed especially for ICT field professionals here in Latvian, in Baltic countries and the European Union. Second, even though, as far as we are aware, we have not adopted anything from Estonia or Lithuania in the term formation and introduction process, it would still be interesting and useful for future research to contact colleagues from the Baltic and other countries and inquire about their experience in term formation and dissemination process. Third, we are determined to examine existing research on translation automation (and translation automation per se) in more detail to determine what could be applied for partial automation of the secondary terminology creation process. Thus, it would be rather valuable to research CAT tools (Translations SDL Trados and others) to evaluate what could be used from their experience to facilitate Latvian ICT terminology development.

8. Conclusion

We conclude the article by emphasising the necessity for further research in the automation of secondary-term creation.

In order to keep the Latvian language alive and developing, to keep up with the constant influx of new ICT terms, we have to keep creating terms in Latvian. We have to change and optimise the way we prepare terms for secondary term formation, automating as much as possible from the abovementioned processes both in the source and target languages.

For example, when creating secondary ICT terms in the past three years (since 2019), we have worked mainly with the current ISO standard (ISO/IEC/IEEE 24765:2017(E) ISO Systems and software engineering vocabulary), consisting of more than 4500 term units. We have two ITTEA commission meetings per month; in each meeting, we accept, on average, 20 terms. Hence, each month, approximately 40 secondary terms in Latvian are created, thus meaning approximately 400 terms per year. It means we can estimate the completion of the current ISO standard around the year 2030.

It can be presumed that even if we are highly selective about the terms we choose to form in Latvian, the exponential development of terms in the IT field exceeds our capacity to re-creating them. Therefore, further research into the possibilities of pre-processing terms for secondary term formation could: reduce the manual effort when searching for term definition and context, enable faster decision-making regarding terminology and, in the future, might provide the possibility for improving the maintenance of term consistency.

Thus, to be fluent in the ever-changing and rapidly developing world, aiming to create and integrate the appropriate ICT terminology as much as it is needed (and not only as much as possible right now) is one of the possible aspects that might facilitate more precise machine translation in the future and thus enable continuous survival and development of Latvian, Lithuanian, Estonian and other languages.

As seen from the case study, it becomes evident that there are parts of term creation that cannot be automated and will not be automated because there is a creative aspect of understanding the term in the current context in numerous sub-contexts of different programming languages.

The actual parts that could be automated can be looking for term definitions, extracting them from dictionaries, parallel texts, and encyclopaedias, providing parallel texts from reliable resources and evaluating the actual use of the already created term (when considering if the accepted term should be used), extracting various definitions of Latvian words (e.g.Tezaurus), extracting and copying in one document (environment) corpus examples of usage.

In conclusion, we would like to emphasise that as audio recognition speeds up the transcription process and reduces manual effort by some degree; very much in the same way, the reasonable expectations from the terminology creation tool are that of facilitating the following aspects in the secondary term creation process: definitions of terms both in English and Latvian, looking up contextual examples, extracting already existing term equivalents in Latvian.

It is of utmost importance to keep this terminology creation tool as simple as possible, keeping in mind the main aim: make the term creation process more manageable, not complicate it beyond recognition.

As with post-editing of the machine translation and recognition of the voice records in written form, this tool will be designed and used to facilitate the term-creation process for terminologists in the ICT field and others in Latvian and other languages concerned with secondary term formation.

References

Arenas, A. G. (2020). *Pre-Editing and Post-Editing.* https: //www.researchgate.net /publication /338310852_Pre-editing_and_post-editing, last viewed <19.02.2023>

Bāliņa, S. (2022). LIKTA: digital Essentials 2023 - *Latvian Information and communications technology association*, last viewed 01.03.2023, https://likta.lv /wp-content /uploads /2022/12 /Signe-Balina-LIKTA_kopsavilkums_06122022_3.pdf

Desbiens, F. (2023). *Building Enterprise IoT Solutions with Eclipse IoT Technologies*: *An Open Source Approach to Edge Computing*, Apress Berkeley, CA, 364.

Floros, F., Grammenidis S. (2012). Secondary term formation in Greek, *Terminology 18:1* 86–104. doi: 10.1075/term.18.1.05flo., last viewed <16.05.2022>

Garcia, I. (2014). Computer-Aided Translation, *The Routledge Encyclopedia of Translation Technology*, https://www.routledgehandbooks.com/doi/10.4324/9781315749129.ch3, last viewed <19.02.2023>

Gunjan, V.K., Zurada, J.M., (2023). *Proceedings of 3rd International Conference on Recent Trends in Machine Learning, IoT, Smart Cities and Applications*. Springer Singapore, 689.

IATE (2022). *Interactive Terminology for Europe* https://iate.europa.eu, last viewed <02.06.2022>

ISO (2022). *ISO standards database* https://www.iso.org/obp/ui/#home, last viewed <02.06.2022>

Jaeger, L. (2019). *The Second Quantum Revolution: From Entanglement to Quantum Computing and Other Super-Technologies*. Copernicus Cham, 339.

Kageura, K. (1998). Theories 'of' terminology: A quest for a framework for the study of term formation, *Terminology 5:1* 21-40. doi:10.1075/term.5.1.04kag., last viewed <16.05.2022>

Koliopoulou, M. (2020). Term formation under the prism of linguistic purism, *International Journal of Lexicography*, 1–15. doi: 10.1093/ijl/ecaa003., last viewed <16.05.2022>

Mojapelo, W. M. (2018). *An investigation of term creation in Northern Soho*. http://ulspace.ul.ac.za /bitstream /handle /10386 /2327 /mojapelo_wm_2018.pdf ?sequence= 1&isAllowed=y., last viewed <16.05.2022>

O'Brien, S., Balling, L. W, Carl, M., Simard, M., Specia L. (2014). *Post-editing of Machine Translation: Processes and Applications*. https://termcoord.eu/wp-content /uploads /2013/08 /Post-editing-of-Machine-Translation.pdf, last viewed <19.02.2023>

Postolea, S. (2012). From regularities to norms in the secondary formation of Romanian ICT terms, *Linguaculture*, 2. doi: 10.1515/lincu-2015-0004., last viewed <16.05.2022>

Sager, J. C. (1990). *A Practical Course in Terminology Processing*. Amsterdam/Philadelphia: John Benjamins.

Skujiņa, V. (1993). *Latviešu terminoloģijas izstrādes principi* [Principles of Development of Latvian Terminology]. Rīga, Zinātne, 224.

Skujiņa, V. (2002). *Latviešu terminoloģijas izstrādes principi* [Principles of Development of Latvian Terminology]. http://www.vvk.lv/?sadala=217 , last viewed <19.02.2023>

Šostaka, D., Borzovs, J. (2022). Towards Computer-Assisted Latvian ICT Terminology Development. In *Proceedings for 15th International Baltic Conference on Digital Business and Intelligent Systems (DB&IS) Forum.* 2022.

Šostaka, D., Borzovs, J., Cauna, E., Keiša, I., Pinnis, M., Vasiļjevs, A. (2023). The Semi-Algorithmic Approach to Formation of Latvian Information and Communication Technology Terms. *Baltic Journal of Modern Computing*, vol. 11, No. 1 (67-89). https://doi.org/10.22364/bjmc.2023.11.1.05

Veisbergs, A. (2021). *Latviešu valoda kā tulkojumvaloda* [Latvian as a Target Language]. Rīga: LU Akadēmiskais apgāds, 218.

Widyalankara, R. (2015). *The Expediency Hypothesis and modes of secondary term formation in ICT terminology in Sinhala: A bilingual corpus analysis*. https://www.researchgate.net /publication/282656961_The_Expediency_Hypothesis_and_modes_of_secondary_term_fo rmation_in_ICT_terminology_in_Sinhala_A_bilingual_corpus_analysis, last viewed <16.05.2022>

WEB (a). *Machine Translate Website*. https://machinetranslate.org/eamt2022, last viewed <19.02.2023>

WEB (b). The Institute of Mathematics and Computer Science, *University of Latvia Website*. https://www.lumii.lv, last viewed on <19.02.2023>

WEB (c). *Institute of Electronics and Computing Website*. https://www.edi.lv/, last viewed on <19.02.2023>

WEB (d). 10 Future IT Trends & Predictions for 2022/2023 You Should Be Thinking About – *FinancesOnline*. https://financesonline.com/it-trends/ , last viewed on <28.02.2023>

WEB (e). Projects - *Latvian Information and communications technology association*. Last viewed <01.03.2023>, available on https://likta.lv/projekti/

WEB (f). *Akadēmiskā terminu datubāze* [Academic term database] http://www.akadterm.lv, last viewed <08.03.2023>

WEB (g). *Cambridge Dictionary*. https: //dictionary .cambridge .org /us /dictionary /english /computer, last viewed <16.02.2023>

WEB (h). *Latvijas Enciklopēdiskā vārdnīca* [Latvian Encyclopedic Dictionary]. https://www.letonika.lv/groups/default.aspx?g=1&r=1&f=1, last viewed <19.02.2023>

WEB (i). *Latvijas Nacionālā bibliotēka* [National Library of Latvia]. https://www.lnb.lv, last viewed on <19.02.2023>

WEB (j). *Latvijas Nacionālais terminoloģijas portāls* [Latvian National Terminology Portal]. https://termini.gov.lv , last viewed on <19.02.2023>

WEB (k). *Microsoft Language Portal*. https://www.microsoft.com/en-us/language, last viewed <02.06.2022>

WEB (l). *Merriam-Webster Dictionary*. https://www.merriam-webster.com /dictionary /term, last viewed <16.02.2023>

WEB (m). *Nacionālā enciklopēdija* [Latvia's National Encyclopaedia]. https://enciklopedija.lv, last viewed <19.02.2023>

WEB (n). *Technology Dictionary.* https://www.techopedia.com/dictionary, last viewed <02.02.2023>

WEB (o). *The Government of Canada's terminology and linguistic data bank*. https://www.btb.termiumplus.gc.ca, last viewed <19.02.2023>

WEB (p). Linguee, *Online bilingual concordance.* https://www.linguee.com/english-latvian/, last viewed <19.02.2023>

WEB (q). Whatis, *TechTarget's free encyclopedia and learning center for IT and business professionals*. https://www.techtarget.com/whatis/, last viewed <02.11.2022>

WEB (r). Wikipedia, *Multilingual free online encyclopedia*. https://www.wikipedia.org/, last viewed <19.02.2023>

Xiaoying D., Yan Y., Jiali Z. (2023). *Cisco: Integration of Innovation and Operation*. Springer Singapore, Peking University Press, 221.

Zauberga, I. (2016). *Tulkošanas teorija profesionāliem tulkiem un tulkotājiem* [Theoretical tools for professional translator]. https://www.rakstiskatulkosana.lu.lv /fileadmin /user_upload /lu_portal /projekti /rakstiskatulkosana/Tulkosanas_teorija_Zauberga_LAT.pdf., last viewed <19.02.2023>

Appendix.
Fragment of the meeting transcript on 21st October 2022.

The transcript below is translated and edited for grammatical clarity by Dace Šostaka.
NB! Please take into account that the discussion is verbal and reflects the conversation of various ITTEA members. Therefore, there is numerous elliptical grammatical and stylistic constructions characteristic of verbal communication, even after editing.

DS: Let us start with the terms proposed by VV, which are mainly related to object-oriented programming (further in the transcript, OOP).

At the beginning of the meeting, we discuss the first term, "framework", which we then retain as the Latvian equivalent "satvars". We agree with the "satvars". It stays that way, and we are not inventing anything radically new, are we? Yes, we will have enough to discuss.

Similarly, for the term "modifier", we keep the Latvian equivalent, accepted earlier - "modifikators", the term "static", is rendered as "statisks".

JC: when I saw this term for the first time, I wondered for a long time how it could be that the "mainīgais" (variable) is "statisks" (static). That is, how is it possible that variable does not change?

However, then I read about it, thought about it and realized that it meant something else. Then it should stay like that – "statisks mainīgais".

DS - Well, yes, it is clear that, in this case, the meaning of the word "static" will differ from what we usually understand by the term "statisks" in Latvian language. Let us continue to sum up remarks upon the term, "static variable". We render "static variable" as "statisks mainīgais".

Next is "class member". We have a written discussion when discussing this term in an online document. In this discussion, colleagues well-versed in this field have agreed that we will render it in Latvian as a "klases elements".

JC: Objects that all colleagues have yet to agree.

DS: In such a case, if there is a technical inaccuracy - namely, if one of the colleagues highlighted the term incorrectly and such colleagues have yet to agree, then we discuss the term again.

JC: Please scroll to the place in the document where I have some excellent examples.

Now, about "member" in English, as it is used. Next, the last example - I did not search much for "elements" in Latvian, how it is used.

Now I would like to understand if all committee members agree that in the last example, the word “elements” in Latvian has the same meaning as "member" in English.

It is not the best use of “elements”.

In principle, I agree with VV, as the term “elements” has already entered Latvian usage and could not change anything there and let us keep it that way.

At the everyday language level, that is, in the language's general use, the term's meaning is not of such great significance.

However, on a technical level, the term “elements” has another use and a different meaning.

At least, as far as it is my understanding.

VV: If there is no longer explanation, I am not ready to say that something other than “elements” could be appropriate.

It is like describing an item, right? Is it so, or am I wrong?

JC: Well, it is hard to say.

VV: Well, “elements” has some type of item in it. It means that it is an item - a field; item - it is not correct to call it one.

However, it could be an item for - field, property, or method. Also, here some similar item is described. I understand that all of the mention are, could be called an “elements”.

JC: Isn't here the class body the class declaration or description?

VV: No, all the members are already in the body.

JC: What is a class body? Is it the class body, a class description or a class declaration fragment? How is it? Does it refer to the class itself or the class declaration?

VV: So, it is as follows: these members, these elements are declared inside the class, in the body of the class.

JC: When the declaration is written on paper, a list of elements comes out. Otherwise, I do not understand why you are talking about a list. So I think it's about the "class of body", and it is simply a definition fragment from the description of the class declaration in the last example.

VV: It is hard for me to tell in which (programming) language this is described. However, let's say I know that there are separate solutions, when at the beginning, you immediately write what will be inside in the head and then describe it in detail in the body.

I know that there are such constructions. But it is necessary to study whether it is in this particular case; it is very difficult to say.

JC: I agree with what was just said.

VV: I am not ready to discuss this issue at the moment.

JC: Yes, because the question is already, is it an “elements” for the list? Or is it an “elements” in the class? It is not really clear.

VV: A class is when a relation among various objects is written, for example, among various animals, which in turn will already create an object from that class according to such an “animal template”, create an instance in the corresponding field next to it.

JC: Actually, I need to understand something in this explanation. A class is not something in which something is written.

VV: I clarified. A class is exactly what, in most cases, describes what the instance is created from.

JC: In my opinion, the class is a set of data and activities, and nothing can be written in the class. You can write on paper.

VV: Well, let's say if we talk in analogies; with simple ones, such as structured programming languages, and using common concepts accordingly, then there is a variable, and variables have a type.

According to a class, it could be said to be similar to an object type. For example, there is a dog class and a cat class, but that class describes how, according to what principles, we produce these dogs, cats and so on. Cats and dogs are those objects or instances of the class. This is the nature of that class in object-oriented programming.

JC: As we discussed with US - if you take your “animal class”, namely “poultry” class, then the question arises about the “rooster”. Is the rooster an element of this class or a member of this class?

VV: Sorry, then the “rooster” or “hen” or “chicken”, respectively, is neither a member of the class nor an element of the class. But in this case, the “rooster” is an instance of the class. An instance of this class is instance in relation to the “rooster” template, and this template is instance to this “rooster”. Seven, ten, or a thousand “roosters” can be produced in this way.

This is what we understand in a classical way. This would be the understanding of a class in object-oriented programming.

JC: Okay, I agree with the programming at the moment. But tell a farmer that a “rooster” is an instance of him as a poultry farmer. He won't understand you.

VV: Well, that would be like arguing about the word “mouse”. Well, for a cat, the understanding of the word “mouse” is one thing. For us in the computer field, the understanding of the word “mouse” is another thing. Context is important.

JC: Okay, that was a bad example. I will stop speaking on the matter.

DS: So, summarizing the discussion we just had - do I understand it correctly that it will not be too much of an inaccuracy if we render a “class member” as a “klases elements”?

Thus, as a matter of fact, is it possible to accept “klases elements” as Latvian term for English term “class member” now?

JC: Until the moment when the “klases elements” appear in English.

VV: So far I have to so, I have not found it (“klases elements” in English) until now.

JC: I must say, I understand you. My problem is that I have found the “klases elements” in English.

EC: Well, there will be no “klases elements” in classical programming. Because typical examples of “klases elements” are class variables, class methods.

In other words, all those - as we have already chosen the term - elements that appear in the class description.

These are the various elements of the class because when describing the class, the variables are described, the methods are described, and various other elements are described, which allow this class to be used in different ways.

JC: Yes, good.

DS: So, for the clarity of the term usage, we will definitely indicate, we will point out in the protocol the context for the term: the term “class member” is rendered as “klases elements”, in Latvian, in the context of object-oriented programming. We indicate that it is used specifically in object-oriented programming.

Because there are cases in secondary term creation when it is impossible to find a term that would be appropriate for all possible contexts. However, by specifying the context of the term's use, we resolve this ambiguity described above.

Thank you, let us move on to the following term.

The next term is an “object member”, and the collocation with the term “object”. So, there is an object and its elements within the object.

VV: Wherever there is a “member”, we use the Latvian term “elements”.

DS: And let us remember that the next will be the “instance member”.

The next question about the term “field” - are we particularly creative with the representation of the term “lauks” in Latvian, or will it still be a “lauks”?

JC: We probably keep the Latvian equivalent of field, “lauks”.

VV: Actually, the only thing we can fix here again is that it refers to OOP again because exactly the same fields’ match as a field is in other sub-fields, which can also be found in other contexts. Of course, if we really want to emphasize it. Of course, we can also choose not to emphasize it.

JC: “Field” as a “lauks” is used not only in OOP but also in other industries.

VV: Because we have definitions of the “lauks” in other contexts in the previous terms.

JC: If we write next to the term field the indication that OOP is used, then we can create the wrong impression that it is not rendered as a “lauks” in other programming languages and environments, that the only context where “field” is rendered as “lauks” is OOP.

DS: Yes, I agree with the comment of JC, that if we emphasize that this term is used in the OOP context, then in order to have the correct use of the term, we should list the broadest possible

use of the term also in other contexts where the English term “field” is also used Latvian corresponding term “lauks”.

VV: Well, for example, we have previous definitions of terms that can be found in our term portal Akadterm, respectively, the terms are defined there, and those definitions are not applicable to this case. So I wanted to add that.

JC: I do agree with the definitions, but what about the term itself?

EC: That's the main thing.

The idea is that when we post a term on the Akadterm portal, we put a reference to a specific context if the term in this context is different from a term in another context.

But this is quite the opposite case. Well, I wouldn't say in all, but as many contexts as we can think of now, in all these contexts the “field” we know, the field is also called the “lauks”.

Therefore, there is no need to indicate the use of “lauks” in one context only, narrowing it down to one context only.

DS: Summary of the discussion - when we come across a term where the “field” cannot be represented as a “lauks”, then we will indicate a specific context of use.

DS: So we move on in our discussion to addressing the term “member variable”.

By commenting in written format on the shared document, Professor US agrees with the Latvian equivalent “iekšējs mainīgais”. Do the other members of the commission have any relevant comments? Will it be with indefinite ending - “iekšējs mainīgais” or with definite ending - “iekšēj*ais* mainīgais”?

VV: We have to choose the version used in the lectures.

DS: Ok, then will it be indefinite ending or definite ending: “iekšēja” or “iekšējā? “funkcija” or “funkcijas”? Will it be “iekšēja funkcija”?

JC: There are probably many “iekšējās funkcijas”. Because it is actually a generic name.

DS: Well, then there will be for “member function” the Latvian term “iekšējā funkcija”.

We now turn to the discussion of the next term.

VV: These are classic things that we probably don't need to discuss and don't need to change.

DS: It recommends that the “function” could be approved as “funkcija”, but I understand the procedure that they say that they could probably do without it and not carry out its approval.

JC: Function and Procedure. And US did not agree, but he again agreed with the attribute.

DS: Yes, but he has not said that he disagrees with the “funkcija” and the procedure. So, I think he has not commented on such an obvious matter.

VV: Because I think that the definition accepted at that time is accepted in that context.

DS: Thank you, sure. Now we turn to the term “attribute”, in Latvian rendered as “atribūts”. There is no fundamental disagreement there, either.

The next term in our list is a “class variable” whose representation is offered as: “klases mainīgais”. Do colleagues agree with the proposed term?

JC: Elsewhere, in a different context, it would be a “class variable”, but it is elsewhere.

DS: Yes, it is elsewhere.

A little further, I will have a question about the singular and plural forms of the term. I will wait for your comments, which option would be the most precise.

Let's now return to the term, “class field”.

VV: I agree with the offered solution.

DS: So it will be “klases lauks”.

JC: The only question is, what is the reason why professor JZ talks about cases and not about instances?

VV: I'm sorry, I don't know where this proposed option came from.

DS: Perhaps, when translating a term quickly, if there is little time, then usually the first translation of the term is word-for-word translation.

VV: But how should there be an "instance"?

DS: Yes, I also read that it is the "instance method".

But do we to the term “instances metode” add that the instance method will not be an instance but an instance?

VV: Somewhere further on, this term will be mentioned for discussion.

DS: Sure. Thanks for the clarification, I noted at this point that we need to talk about it.

Next, there is next to nothing discussions, no objections written to proposed terms in Latvian. So, we agree to keep the proposed terms:. “class method” remains “klases metode”, the “instance variable” is “instances mainīgais” and “instance field” as “instances lauks”, “instance method” is “instances metode”, and “polymorphism” remains “polimorfisms”.

Next, now we have the part where we have the opportunity to discuss the term. Let us look at the term "encapsulating" – “iekapsulēt”, “iekapsulēšana”.

JC: Question. Is the rendering of the term “iekapsulēšana”, which was once accepted, or does its Latvian equivalent remain the same? Because it's not OOP?

DS: I'm checking it now; it is “iekapsulēšana”.

JC: Because it has the usual ending–*tion*, which has two meanings in English.

DS: I agree. As it is now, I will not be able to comment, and this term will be marked for a further discussion.

VV: Yes, I think that in this case, in the context of use, we usually say “iekapsulēšana”, because then, as professor US has also agreed here, I think that we do not discuss this issue because it is something that is used on a daily basis.

JC: Well, if it is usually the case, then this is the case when OOP context should be indicated.

EC: Well, no, but here again, it is a little bit the other way around.

Because in what context is there a case where "encapsulation" isn't "iekapsulēšana"? Then we can ask the following question!

JC: “Iekapsulēšana” is already there.

We have ever noticed that the encapsulation result is also needed.

Take a look, four entries below, you can see an example that we have accepted term “encapsulation” some time ago. We have had it accepted already once, some time ago.

EC: Yes, but first argument is that we are not discussing it now. The second argument, this decision of ours about “iekapsulēšana” does not prevent us from forming the result form “iekapsulēšana” as well.

JC: Okay, I agree.

EC: There is no contradiction.

JC: It's just that some translators will quickly use the term “iekapsulēšana” as the Latvian equivalent when translating texts. But, well, so be it.

EC: But it will most often be correct, this correspondence, “iekapsulēšana”.

Received February 20, 2023, revised March 9, 2023, accepted March 11, 2023