LaTeX classes for doctoral theses in Ukraine: Interesting tips and painful problems

Oleksandr Baranovskyi

Abstract

In the talk, I introduce `vakthesis`, a bundle of LaTeX classes for typesetting doctoral theses according to official requirements in Ukraine, discuss current status of the project and future development plans. Some LaTeX programming tricks that I have studied are considered.

1 Introduction

`vakthesis` is a bundle of LaTeX classes for typesetting doctoral theses in Ukraine.

It consists of the following main components: `vakthesis` and `vakaref` are traditional classes for a thesis and for a self-summary respectively; `mon2017dev` and `mon2017dev-aref` are modern classes for a thesis and for a self-summary respectively.

Traditional classes conform the now "obsolete" official format required by the VAK (Вища атестаційна комісія, i.e., Higher Attestation Commission). Now they are suitable for previously defended theses but are a base for modern classes also.

In 2017, the MON (Міністерство освіти і науки, i.e., Ministry of Education and Science) published own requirements on thesis typesetting. Modern classes conform this "new" official format by the MON. These classes are recommended for persons that defend their theses now. They are based on `vakthesis` and `vakaref` respectively and cannot work without them.

The bundle contains a number of example files: example of the main file of a thesis, example of introduction, chapter 1, and so on. So PhD student can use these example files as a template for his/her thesis.

2 Tips and problems

2.1 Command `\speciality` with two optional arguments

In Ukraine, any thesis should be classified according to the so-called List of Specialities.

This is a list that corresponds speciality code to speciality name and field of science. For example, 01.01.01 is a code for speciality "математичний аналіз" ("Mathematical Analysis"); and PhD student will be awarded by degree "кандидат фізико-математичних наук" ("Candidate of Physical and Mathematical Sciences").

In `vakthesis`, there is a command `\speciality` requiring one mandatory argument:

```
\speciality{01.01.01}
```

For a given speciality code the class will take a speciality name and a field of science from a special plain-text database and output corresponding information on the thesis title page.

The command `\speciality` has two optional arguments if, for some reason, user need to provide a speciality name and a field of science:

```
\speciality[математичний аналіз]{01.01.01}
  [фізико-математичних наук]
```

For example, there are specialities given in the list in the following form:

теорія та методика навчання (з галузей знань)
Theory and Methodology of Learning (on some field)

In this case PhD student should write on the title page

теорія та методика навчання математики
Theory and Methodology of Learning of Mathematics

On the other side, for some specialities, a degree can be awarded on various fields of science. For example, for speciality code 01.04.07, PhD student can obtain degree Candidate of Physical and Mathematical Sciences or Candidate of Technical Sciences according to specific nature of the thesis.

These cases cannot be processed automatically. But optional arguments can be used:

```
\speciality[теорія та методика навчання
  математики]{13.00.02}
```

and

```
\speciality{01.04.07}[технічних наук]
```

respectively.

Of course both optional arguments may be used simultaneously.

Hence command `\speciality` cannot have the following syntax:

```
% [\small\makevmeta]
\speciality[!<something>][!<something>]
  {!<code>}
```

and cannot be defined by the standard LaTeX tools.

How to define the command with two optional arguments around of mandatory argument? To this end we can use `\@ifnextchar`. I will give a simplified pseudocode to demonstrate a main idea.

```
\def\speciality{%
  \@ifnextchar[\a@cmd\aa@cmd
}
\def\a@cmd[#1]{%
  % Process speciality name #1
  \@speciality
}
\def\@speciality#1{%
  % Process speciality code #1
  \@ifnextchar[\a@cmd@b\a@cmd@bb
```

```
}
\def\a@cmd@b[#1]{
  % Process field of science #1
}
\def\a@cmd@bb{%
  % Find field of science in the DB
}
\def\aa@cmd#1{%
  % Process speciality code #1
  \@ifnextchar[\aa@cmd@b\aa@cmd@bb
}
\def\aa@cmd@b[#1]{%
  % Process field of science #1
  % Find speciality name in the DB
}
\def\aa@cmd@bb{%
  % Find speciality name in the DB
  % Find field of science in the DB
}
```

This is a very simple idea; and this idea just works. But this code is difficult to maintain as well as to expand for three or more optional arguments.

This is a reason why I reject this idea in a modern version of the classes. Now I use a simple key/value interface.

2.2 Environment `bibset` to make two reference lists in a thesis

The classes `vakthesis` and `vakaref` provide the environment `bibset` that supports two reference lists in a document in the case of using BibTeX.

Why there is a need for such environment? It is known that BibTeXcan generate only one reference list in a document, i.e., only one command `\bibliography` can be processed in standard situation.

To have more than one reference list is a usual need. There are many solutions for this problem but they are not suitable for me.

[TODO: Should I give an overview of the known solutions?]

In Ukrainian thesis, the following two lists may exist: the list of referenced sources and the list of author's publications.

Then the environment `bibset` is used in the following form.

```
\begin{bibset}{Список використаних джерел}
 \bibliographystyle{gost2008s}
 \bibliography{xampl-thesis}
\end{bibset}

\begin{bibset}[a]{Список публікацій автора}
 \bibliographystyle{gost2008}
 \bibliography{xampl-mybib}
\end{bibset}
```

This environment redefines standard commands `\bibliography` and `\bibliographystyle` such that they write commands `\bibdata` and `\bibstyle` to the `.aux` file if they appear in the first environment `bibset` and do not write otherwise.

Then during the first run of the usual procedure: latex, bibtex, BibTeX see only one copy of `\bibdata` and `\bibstyle` corresponding to the first environment `bibset`. At this point generated file `xampl-thesis.bbl` should be renamed to `xampl-thesis1.bbl` (manually or programmatically).

During the second run first and second environments `bibset` are interchanged, i.e., commands `\bibdata` and `\bibstyle` are written to the `.aux` file if they appear in the second environment `bibset` and are not written otherwise.

So BibTeX see commands `\bibdata` and `\bibstyle` corresponding to the second environment `bibset`. At this point generated file `xampl-thesis.bbl` should be renamed to `xampl-thesis2.bbl`.

At the last step LaTeX includes files `xampl-thesis1.bbl` and `xampl-thesis2.bbl` at the corresponding places. Hence author have two reference lists in the thesis.

In earlier versions of the official requirements by VAK there was not any mentions of the two reference lists. I just realised this for myself. However modern official requirements by MON contain an explicit recommendation to prepare even three reference lists in a thesis. This solution works in modern theses too.

2.3 `casus` package and UTF-8

In Ukrainian language, a noun (as well as some other parts of speech) can change its form to express its syntactic function in the sentence.

There exist seven cases: nominative, genitive, dative, accusative, instrumental, locative, and vocative.

For example, "університет" is a university in Ukrainian. This is the form of nominative case. If I want to write that I study at a university, I should use the form of locative case: "я навчаюся в університеті". That is the ending is changing and there is also a preposition.

How is it related to doctoral thesis? In the self-summary, authors write the name of the institution, where they studied and worked on their thesis. So a command `\institution` provided by classes whose argument is the name of this institution. This name (in the nominative case) is appeared on the cover page. On the same time, there is a sentence on the reverse side of cover page, where author states that

this work is performed at this institution. Here the name of institution is in the locative case.

I think it is redundant to ask author to provide different forms of the same institution's name if a command `\institution` is already gives a nominative form of the name. The `vakthesis` or `vakaref` class can "compute" genitive, dative or other form.

This is exactly what an auxiliary package `casus` do.

Briefly, the algorithm is the following. Suppose the institution name is "Національний педагогічний університет імені М. П. Драгоманова". Here there is a special word "університет" from the list of "known words". All words before the known word are adjectives. The `casus` package has rules to decline adjectives as well as rules to decline nouns. All words after the known word should not be changed. The algorithm should split a given sentence to words, find a known word ("університет", "інститут", "академія", "школа", "бібліотека", etc.), then decline adjectives and known word and then stop, i.e., do not touch the words after the known word.

In Ukrainian language, any name of institution has this form. So this algorithm works.

Unfortunately, some day some author reported me a problem he encountered in his thesis. He just re-encoded all `vakthesis` files and his thesis files from Windows-1251 encoding to UTF-8 encoding.

After this operation he received some mystic error messages such as

```
Missing number, treated as zero.
```

or

```
Undefined control sequence.
```

Skipping non-essential details the main problem is in the `casus` package. To decline a word (in particular, an adjective "педагогічний") the algorithm runs through the word until it find an ending from a given list of endings. This is the ending "ий" for this word. This ending is skipped and other ending that corresponds to a given case is added. So the words "педагогічного", "педагогічному" and so on are received.

This simple idea should not depend on file encoding. However my implementation of the algorithm does not work if file has an UTF-8 encoding. My conjecture is that the algorithm implementation fails for characters encoded by two or more bytes.

Maybe the better solution would be use a known stable package for string manipulation instead of my quick and dirty solution.

2.4 Incorrect checking if `hyperref` is loaded

Traditionally, theses are printed on the paper and stored at usual libraries. However LaTeX can generate an electronic document with hyperlinks. In particular, a `hyperref` package can be used to this end. Some authors of theses want to use this possibility.

Generally speaking, `vakthesis` bundle is not compatible with `hyperref` package. In particular, `vakthesis` classes redefine some internal commands such as `\@spart`, `\@schapter`, and `\@ssect`. The number of arguments are even changed. The `hyperref` package make its modifications carefully but cannot predict that these commands have more arguments now.

As a result simultaneous using of `vakthesis` and `hyperref` causes hard for diagnostics errors. Sectioning commands does not work as expected. The above-mentioned `bibset` environment cannot be used with `hyperref` too. In this case commands `\cite` are not hyperlinked.

To carefully interact with `hyperref` package `vakthesis` classes should check if `hyperref` is loaded:

```
\@ifpackageloaded{<package name>}
  {<true branch>}
  {<false branch>}
```

But internal command `\@ifpackageloaded` can be used in a preamble only. I cannot use this command inside of `bibset` environment to check if `hyperref` is loaded. So I check if internal `hyperref`'s command is defined:

```
\@ifundefined{hyper@warn}
  {<true branch>}
  {<false branch>}
```

It is easy; and it works. But, after a few years, I received a bug report from the `vakthesis` user. If he use `bibset` environment with `hyperref` loaded, then `\cite` is not hyperlinked.

The reason is that modern version of `hyperref` does not define `\hyper@warn` anymore. The command `\Hy@WarningNoLine` is defined instead of it. Hence the true branch is not executed at all.

The quick patch is to restore the definition in the document preamble:

```
\@ifundefined{hyper@warn}
  {\let\hyper@warn\Hy@WarningNoLine}
  \relax
```

In the next version of `vakthesis`, the `hyperref` check is also fixed.

However to check if a package is loaded inside of a command/environment is a bad thing anyway. To make a good software, I should prepare two versions of a command/environment (with and without

hyperref) and then choose the corresponding version.

2.5 Overwriting and overloading

I have started to develop vakthesis bundle in 2003 approximately. I was a PhD student at that time. My experience with TEX and LATEX was very limited. Sure, I used LATEX for preparing my papers, slides, etc.; but I did not try LATEX programming.

I should to say that access to Internet was unstable and expensive at that time. So I did not find any suitable templates or LATEX classes for thesis typesetting compatible with Ukrainian requirements. It was a unique solution to develop a LATEX class that I need by myself.

I decided to start from the standard report class and modify it according to my needs.

On the one hand, to overwrite existing class is a more clear solution. It is easy for developer. There exist a ready-to-use class that almost comply with my requirements. I just need to patch some parts of the code that do not agree with the requirements.

However this approach is more complicated for maintainer. First of all, I should look on report class development and update my code. At that time, I thought that changes of report are not very often.

Moreover, careless overwriting may cause problems. For example, in the report class, the command \part typesets part headings on separated pages. But vakaref class use this command to typeset structural parts of self-summary; and they are typeset as usual headings. So I removed any \newpage commands and stopped.

Someday a user encountered a problem when structural part heading occurs on the last line of the page. As a result a page break is appeared between the heading and the further text. This is unwanted behavior, of course; and fixing of the command is required.

More recent classes mon2017dev and mon2017dev-aref are designed as a "building" above the base classes vakthesis and vakaref respectively. They load base classes and then redefine some commands.

But overloading may cause problems too. Because casus package works with Cyrillic letters it is loaded after inputenc package. So in the vakaref class there is a corresponding line:

```
\AtBeginDocument{\usepackage{casus}}
```

For the new mon2017dev-aref class, some modifications in the casus are needed. So this class load a modified version:

```
\AtBeginDocument{\usepackage{casus2017dev}}
```

Since mon2017dev-aref is a building above the vakaref, we have also the line

```
\LoadClass{vakaref}
```

in the mon2017dev-aref.

Of course casus2017dev cannot work without casus. But we have the following chain. Firstly, mon2017dev-aref class loads vakaref class, then vakaref adds casus to \AtBeginDocument hook. Later mon2017dev-aref class adds casus2017dev to \AtBeginDocument hook. As a result, in a proper place, casus is loaded, and then casus2017dev is loaded.

This colossus with feet of clay do its excellent work... till some day when it fell. User is in a slight panic. Declension does not work in his thesis, and there is a waste instead of a title page in the thesis! Sure, he is ready to submit his thesis today! More interesting, I do not see this problem on my system.

The reason is in a new release of LATEX. In the version 2020-10-01, a general hook management system was provided. This affects a standard hooks defined by command \AtBeginDocument too.

I am not sure if I understand correctly what exactly happens. I suppose that, since casus and casus2017dev are loaded in different classes, we have that they are added to hooks with different labels. As a result, either code is executed in changed order or second \AtBeginDocument just executes code instead of adding to hook. Visible result is that casus2017dev is loaded before casus. It cannot work in this situation.

It is easy to fix the problem. It is enough to add the line

```
\RequirePackage{casus}
```

to the casus2017dev package.

Anyway, such many-level overloading may be the real headache for user as well as for maintainer.

3 Current status

- Now there are two separate modules ("obsolete" vakthesis and "modern" mon2017dev) and there exist many problems related to this separation.
- Alternative solutions exist. But vakthesis is supported and maintained.
- Anyway there exists an interest to vakthesis. Many people use these classes to typeset their theses belonging to various fields: mathematics, physics, computer science, etc.

  [TODO: Add more details.]

4 Future plans

- Combine `vakthesis` and `mon2017dev`. It is not just mechanical work if we want to keep compatibility.
- Make fully UTF-8 compatible.
- Provide more user-friendly documentation (and examples without real persons' information).
- Upload to CTAN (and therefore to main TEX distributions)
- Use VCS (and then upload code to GitLab, GitHub, own server, or other solution).
- Fix some known bugs.

[TODO: Add more details.]

◇ Oleksandr Baranovskyi
Doctor Barbarus Services
& Institute of Mathematics
of the National Academy of
Sciences of Ukraine
ombaranovskyi (at) gmail dot
com
https://sites.google.com/view/drbarbarus/