TUG 2025 abstracts

Editor's note: Links to slides, videos and other information are posted at `tug.org/tug2025`.

— — ∗ — —

Doris Behrendt

The LaTeX Village at the Chaos Communication Camp — how it started, how it continues

The Chaos Computer Club (CCC) is a German "Verein" (a kind of NGO) that has been active since 1981. Among other things, it has organized a hacker congress every year since 1984 (`events.ccc.de/en/congress/`). In addition, a camp takes place every four years since 1999. It features talks, workshops, parties, and various community-grown projects that are usually organized into so-called villages (`events.ccc.de/camp/`). Similar hacker camps also exist in the Netherlands, the UK and, as far as I've heard, also in India.

A LaTeX Village has been held three times so far: in 2017 (Netherlands), 2019, and 2023 (Germany). In my talk, I'll share some behind-the-scenes insights and report on how, under my direction as a board member, DANTE e.V. successfully brought these villages to life.

Reports in German about the camps of 2017 and 2019 can be found in DANTE's journal *Die TeXnische Komödie*: "Still hacking anyway – vom Hacker-Festival in Zeewolde" in issue 2018/4 and "CCCamp19" in issue 2020/3, available at `dante.de/dtk/bisher-erschienen/`. The report on 2023 is still in progress ... the camp is quite rightly named "*Chaos* Communication Camp". I'll also talk about this in an informal way and show some pictures.

Erik Braun

The state of CTAN: 2025

CTAN is the source for TeX distributions and the place where TeX packages are kept to reach a wide range of users, which requires a lot of manual work. This talk gives an overview of the current software, hardware and people.

Frank Mittelbach

How accessible are typical PDFs and how accessible could they be?

In this talk we will listen together to read-outs of typical constructs found in PDFs (generated with LaTeX or otherwise) so that participants can get some first-hand experience of what we all have inflicted on readers with reading disabilities during the last thirty years. Documents with more complex content such as mathematical formula, tables, etc., are especially troublesome; but even fairly simple documents can cause the heuristics, as used in today's software, to fail in horrifying ways.

We then discuss how much better the results could and should be and how far we already get in this respect with the current implementation provided by the LaTeX Tagged PDF project.

An important aspect of this new software is that it works without any manual post-processing of the PDF — i.e., the accessibility is achieved fully automatically by just compiling the LaTeX source as usual.

Vaishnavi Murthy

Encoding Tulu-Tigalari: A journey to Unicode standardization

This talk traces the decade-long journey of proposing the Tulu-Tigalari script for Unicode encoding, culminating in the final proposal (L2/22-031). I will share the challenges of researching an archaic Southern Brahmic script, used historically for Tulu, Kannada, and Sanskrit, and the collaborative efforts to define its orthography based on manuscript analysis. Key milestones include addressing script disunification, standardizing glyph shapes, and navigating Unicode Technical Committee reviews. The talk will highlight the cultural significance of reviving Tulu-Tigalari for modern use and the technical intricacies of encoding a complex abugida, offering insights for script preservation advocates and Unicode enthusiasts.

[1] The accepted Unicode proposal for Tulu-Tigalari: `unicode.org/L2/L2022/22031-tulu-tigalari-prop.pdf`

[2] Naming the Tulu-Tigalari Script: `academia.edu/48672076/Naming_the_Tulu_Tigalari_Script`

Martin Osborne

`text2bib`: Converting plain text references into BibTeX format

`text2bib` is an open-source software system that converts files of plain text references into BibTeX files. The source code, which is written in PHP and uses the Laravel framework, is available at `github.com/osbornemj/text2bib`, and the system is implemented at `text2bib.org`.

A person might want to convert plain text references to BibTeX format because, for example, they are switching from hard-coding references to using BibTeX, or they want to import items into a reference manager like Zotero, using BibTeX as an intermediate format, or they run a journal that typesets papers using TeX and uses BibTeX for the references, but has authors whose papers contain lists of hard-coded references. I initiated the development of the system in the spring of 2007 because I fell into the last

doi.org/10.47397/tb/46-2/tb143abstracts

category. The initial development was done by an undergraduate at the University of Toronto. When the student graduated in the summer of 2007, I took over the project and have developed it since then. At some point I wrote a front end for public use. In the summer of 2023, I started re-coding the system using the widely-used Laravel framework. I launched the new version in March 2024.

For any given text reference, there are many possible BibTeX entries, depending on how the entry is going to be used: to generate a formatted reference using BibLaTeX or BibTeX, or to import the reference into a reference manager, for example. The system knows the properties of various BibTeX style files, and allows an administrator to record the properties of additional files. When a user uploads a file to `text2bib.org` for conversion, she is asked how she intends to use the BibTeX file that will be created, and if she selects BibTeX, she is asked to select the style she plans to use. The output of the script is tailored to her choices, with the correct fields populated.

The main features of a reference that the algorithm uses are (1) the capitalization of the letters — uppercase versus lowercase; (2) the punctuation — for example, commas, periods, and quotation marks; (3) whether the characters are letters or numbers; (4) the presence of words and phrases like "and", "in", "eds", and "retrieved from" — the list is long, and includes words in several languages; (5) the presence of words in an English dictionary (when deciding, for example, whether a word is likely to be the name of an editor or the first word of a booktitle); (6) some auxiliary data, like lists of von names and abbreviations used in journal names.

The algorithm first removes any doi, url, ISBN, and similar component based on pattern matching using regular expressions, determines where the author string ends, and breaks that string into authors (first checking whether the author list matches one of a set of patterns, and then, if no pattern matches, analyzing the string one character at a time). If the year is next, the algorithm extracts it. It assumes the title comes next. It determines where the title ends based on punctuation, capitalization, and whether the following string matches the pattern of publication information for some item type. It takes the remainder of the string to be the publication information. Based on the characteristics of the item, including the presence of various keywords, the system assigns a type to it. Then the publication information is parsed into its components. Once all the items in a file have been converted, a label is assigned to each item.

When a file is submitted for conversion, the user can ask the system to attempt to find each item at `crossref.org`. In principle this functionality could be useful, but it suffers from several problems. First, querying Crossref is very slow. Second, the Crossref data is not consistently formatted and its quality is low. Third, the crossref matching algorithm is poor — the best match can be the wrong item, even for a reference that is exactly correct.

Whenever I modify the algorithm to accommodate an item that is not converted correctly, I add the item to a table of examples and check that all of the existing examples are still converted correctly. Currently there are over 1,200 such examples.

I don't have a good estimate of the accuracy rate of the conversion algorithm. For source files that are free of errors, the rate seems high. But many source files contain errors, and even small ones, like missing spaces between words or mistakes in punctuation, or superfluous text at the end of the reference, can cause the algorithm to produce completely incorrect results. The reason is that the algorithm doesn't understand language, but rather implements a set of rigid rules. I believe that the next big improvement will come with use of an AI system that does understand language. Large language models like ChatGPT and Gemini currently do an excellent job of converting references, and it may be possible to train a much smaller model to do just as well.

`text2bib` has converted over a million references since I re-launched it in March 2024, and is currently used to convert around 2,000 items per day. I ask every new user who registers how they found the site, and currently, on average two per day say that ChatGPT or another AI system referred them to it. But I'm sure that the future lies with an AI system doing the conversions itself.

Salil B, निरंजन (Niranjan)

Symbols to improve the processes of construction and comprehension of tables

While constructing a table, the elements in a column (or a row) are ordered. However, no symbol is used to explicitly state the presence of the order. A set of symbols is proposed for the same. The same facilitates an efficient, easy, and clear comprehension of a table. The set of proposed symbols makes the process of construction of a table academically sound and relatively less prone to errors. Thus, the set of symbols helps both the processes: construction and comprehension of a table. The set of proposed symbols is not necessarily comprehensive or complete — it is open for addition, deletion, and modification.

Saravanan Murugaiah, Mukesh Subramaniam
Typographic branches from a (LaTeX2)Word root
The world of LaTeX has evolved over the years with technology advancements and integrations. And yet, the struggle remains for a large community of non-TeX users in converting a structured LaTeX document into MS Word for any reason, and this remains a major pain point in any workflow. Introducing emDoX, our in-house solution that addresses this challenge.

Rob Schrauwen
Keynote: *True but irrelevant: Twenty-five years of scientific content standards*
Rob Schrauwen has been involved with Elsevier's content processing and markup standards for more than a quarter of a century. In this talk he looks back to the implementation of an XML-first workflow 25 years ago and discusses if this delivered on its promises. Some of the questions that were discussed in the 1990s [1] are to an extent still being asked.

Also there is increasingly a debate about 'what's next?', suggesting it is time for moving away from XML — a debate that has many parallels with the transition from SGML to XML in the early 2000s, when the benefits that were touted for XML were in fact already benefits of SGML and the real motivation for change lay elsewhere. He discusses if we should move away and, if so, then to what?

[1] Sebastian Rahtz, Another look at LaTeX to SGML conversion, *TUGboat* 16:3 (1995), pp. 315–324. `tug.org/TUGboat/tb16-3/tb48raht.pdf`

Vít Starý Novotný
Expltools: Development tools for expl3 programmers
Static analysis tools are invaluable for catching bugs early and improving code quality. However, the highly dynamic nature of TeX has long posed a barrier to static analysis.

Expl3 is a high-level programming layer built on top of TeX. Originally developed to simplify the LaTeX kernel, it is now widely used by package authors across LaTeX and other TeX formats. While it retains much of TeX's flexibility, real-world expl3 code tends to follow more predictable patterns, making it far more amenable to static analysis.

In this talk, I introduce *expltools* — a long-overdue addition to the expl3 programmer's toolbox. Designed to support semantic analysis, it brings clarity and structure to a domain traditionally resistant to static tooling. I present the current state of the project, focusing on its design, practical usage, and validation on real-world expl3 packages.

A current work-in-progress version of the article is available at `github.com/witiko/expltools-tug25-paper`.