Hyphenation patterns: Licensing and stability

Arthur Reutenauer

Abstract

New thoughts on old questions: hyphenation patterns, licensing, and stability.

1 Don Knuth's question

The package hyph-utf8, started in 2008 by Mojca Miklavec and myself to collect all known hyphenation patterns for different languages, has already been the subject of two *TUGboat* articles [2, 3], and the talk I gave during TUG'19 was a summary of those. Having worked on that project for over a decade, I was nonetheless caught by surprise when Don Knuth, who took the first question, asked me how we dealt with archiving and the need to keep page breaks stable over time. I improvised a reply that I'm afraid was not very convincing, and partly missed the point. Here is the answer I wish I had given.

2 Initial answer

There is no policy on stability and backward compatibility for hyphenation patterns in TEX distributions. When Mojca and I took over the existing patterns, we found no evidence of a strategy, or even a rule of thumb, to decide how to update them, and in particular no safeguard against incompatibilities introduced by correcting errors in existing patterns. Depending on contributors' availability, there could be regular updates over a period of time (usually not exceeding a few years), small improvements at irregular intervals, or — most often — no changes at all after the initial development effort.

A practical issue arose soon after we got started on hyph-utf8, as the German patterns were being worked on very actively, in an extensive effort that was guaranteed to introduce incompatibilities. There was however no doubt that such an update would be beneficial to German-speaking users, as it addressed many earlier misses and mistakes. Because we needed a decision, we followed the sensible piece of advice by Karl Berry (who was also the only person to venture an opinion), that we simply keep the patterns as-is for TEX and pdfTEX, and only use the new patterns, as well as any later updates, for XETEX and LuaTEX. (The same team that updated the German patterns produced a package to optionally use the "experimental" patterns in the 8-bit engines as well.) The sentiment was that it was essential to commit to some kind of stability for a major language like German, where incompatible changes would affect more users; and for the older engines, that were already mature.

All in all, however, surprisingly little discussion has ever taken place about how to achieve stability in such an essential part of any TEX installation. The main topic of the conversations we've had has indeed been rather different ...

3 Interrogation

The problem of licences has already been discussed in [3] in connection with other projects interested in the patterns from hyph-utf8, which often had reservations about the LPPL (LATEX Project Public License) for one reason or another. Since that licence is quite central to the TEX world, and it's been used for many pattern files, I will discuss it in the next two sections.

4 What's in a licence

The LPPL was written in order to formalise the conditions that Don put on distributing TEX — anyone may freely use the idea and even the code, but a program may not call itself TEX unless it passes the `trip.tex` torture test — and boils down to:

- Any derivative work, whenever it "identifies itself to the user ... clearly and unambiguously identifies itself" as such [clause 6(a)].
- ... except when made by a specific person, the *maintainer*, in which case the derivative work is considered an updated version of the original work [clause 4]. A work under the LPPL can be either *author-maintained* (only the original author can ever be maintainer), *maintained* (a new maintainer could take over in the future), or *unmaintained* [section "Maintenance of The Work"].

I explained in [3] why the latter point isn't really suitable for hyph-utf8, and forgot to mention that the former wasn't either: except with LuaTEX, patterns are dumped into the format, and thus never "identif[y] [themselves] to the user" during a normal TEX run. That clause of the LPPL is simply moot for hyphenation patterns.

In other words, even if all the patterns were under the LPPL, one could very well produce a new, completely incompatible set of patterns while respecting the letter of the licence, and users wouldn't notice from looking at their terminal or log files.

It's also striking that the only mechanism the LPPL provides to ensure stability is to put everything into the hands of an all-knowing maintainer, who gets full control over the successive versions (and I do mean *full*: the only duty of a maintainer is to publish up-to-date contact details, not even to acknowledge bug reports sent through this contact).

5 What isn't

The advantage to having one person, or a few people, designated as solely responsible for a package, is clear: distributions need to know who is entrusted with making updates, with collecting bug reports and (hopefully) fixing them, etc. Nor can too many formal duties be attached to that responsibility, as that would be unfair to the often overburdened volunteers who put their time and effort at the service of the community.

Beyond these practical considerations, however, it's not clear how maintainers are supposed to ensure stability. They could of course be the ones to gather user wishes and strike compromises, but equally they could just make decisions without consulting anyone else, and in our experience the latter is much more common than the former. It even takes a more sinister turn, as with the recurring case of one prolific package maintainer renouncing his production and dumping it on the lap of hapless volunteers, only to later claim it back and try to assert what he considers his rights, by among other means declaring his package author-maintained. (It's a real problem. I am among the people trying to deal with the situation. We don't know what to tell this person.)

Clearly, no licence is in and of itself going to help with difficult people who abuse it. It can only lay out conditions and principles that its adopters will follow.

The problem is that the LPPL does not even do that. As summarised in the previous section, it deals in great details with name and maintainer changes, but doesn't actually offer any explanation of what maintainers can do to guarantee compatibility. In fact, it doesn't even use the words "compatibility", "stability", or any related ones, except in the preamble and a paragraph near the end, both of which state without explanation that the licence helps with compatibility and stability. Readers are referred to [1] to check for themselves.

Even more than specific words, what I'm missing in the LPPL is some sense of a commitment to stability — an encouragement to package maintainers to produce equivalents to `trip.tex` for example — instead of rights without corresponding duties. (It's all the more surprising as `modguide.tex`, a precursor document, did in fact mention regression tests prominently.) This spirit of the LPPL has in my opinion contributed to a certain complacency in the TEX world, an unwarranted feeling that because of its venerable origins our community is "better at compatibility".

6 A new answer

In light of the above, I hope I can be forgiven for not having a ready answer to this essential question: how can authors ensure that their linebreaks are going to be stable in the future? At this point I need to soften my initial response (lest Don should have a heart attack!) because there is *some* policy about stability: the original `hyphen.tex` will never change and will always available as `\language0` (that is hardcoded in all our software); and there is, as mentioned, a general feeling that patterns for "major languages", whatever those are, shouldn't change too much (although big changes were made to the Spanish patterns under our watch a few years ago). Apart for that, it's pretty much free-for-all. We do of course monitor the situation and discourage authors from changes that are too extensive, but decisions are made on an as-needed basis.

A more systematic approach would require an actual discussion about what compatibility means for hyphenation patterns, and how to achieve it. Among the ideas usually mooted are a complete pattern freeze; lists of hyphenated words whose breakpoints are guaranteed not to change; and a versioning system for patterns. I can't imagine that a single strategy is going to fit every language, but there could be a multiple-tier system of pattern stability, with each language mapped to one tier.

Most important, we need decisions. It's not clear to me that Mojca and I should be the ones to make them, since we're only the implementors, but we'll be more than happy to help along the way, and to execute any vision that can be had in that area. As indeed we have, for over a decade.

References

[1] LaTeX3 Project. The LaTeX Project Public License, version 1.3c, 2008. `http://mirror.ctan.org/macros/latex/base/lppl.tex`

[2] M. Miklavec and A. Reutenauer. Putting the Cork back in the bottle — Improving Unicode support in TeX. *TUGboat* 29(3):454–457, 2008. `https://tug.org/TUGboat/tb29-3/tb93miklavec.pdf`

[3] M. Miklavec and A. Reutenauer. Hyphenation in TeX and elsewhere, past and future. *TUGboat* 37(2):209–213, 2016. `https://tug.org/TUGboat/tb37-2/tb116miklavec.pdf`

◇ Arthur Reutenauer
Storgatan 28B
753 31 Uppsala
Sweden
`arthur (at) hyphenation dot org`