A Format Compilation Framework for European Languages

Laurent Siebenmann
Mathématique, Bât. 425
Université de Paris-Sud
91405-Orsay, France
Internet: lcs@matups.matups.fr

Abstract

In the third and final version of TEX (fall 1989), Donald Knuth provided his TEX with facilities for multilingual typography: fonts of 256 characters, hyphenation for many languages, and virtual fonts — to mention three of many wonderful new features. However, an arbitrary limitation of TEX forces one to load all language hyphenation patterns with naked INITEX, a primitive and notoriously unfriendly version of TEX that has hitherto been mostly reserved for TEX wizards. This article presents FORMAT-DUMPER, a TEX macro package in the form of a directory of inter-related .tex files; it offers a pleasant interactive environment for the creation of multilingual formats and should thus enable ordinary TEX users to build precompiled formats with a cocktail of language and other features appropriate to their needs. FORMAT-DUMPER was originally posted in April 1992 in a bilingual French-English version that nevertheless already served the major TEX format cores: Plain, LATEX, $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX, and LA$\mathcal{M}\mathcal{S}$-TEX. Along with some current features, this article mentions numerous possible improvements. To allow FORMAT-DUMPER to be fully multilingual within the realm of European languages that use a basically Latin alphabet, I propose use or adaptation of Johannes Braams' babel, assimilating its style-independent features. Thus FORMAT-DUMPER should become a useful adjunct to an updated babel. With this in view, the article concludes with a carefully motivated discussion of the stellar language switching mechanism of Babel using however modified nomenclature based on the two-letter language codes recently introduced by Haralambous.

Introduction

Since the development of the TEX kernel was terminated by Knuth in 1989, a number of important problems have come into focus that must now be solved within the somewhat arbitrary constraints of TEX version 3. This article discusses one of them, the compilation of non-English or multilingual formats — restricting attention to the problems raised by European languages that use a (possibly accented) Latin alphabet.

Version 3 of TEX forbade introduction of new hyphenation patterns during the normal operation of the program TEX. More precisely, only the special setup of TEX called INITEX is henceforth able to interpret the command \patterns. What is more, although Knuth's documentation gives no warning, even INITEX is unable to interpret \patterns in case a precompiled format has been loaded by INITEX.

It seems to me to be a perfectly reasonable and worthy challenge to modify TEX so as to obviate this limitation. However, I suspect that no such modification of TEX will be in wide use before the next century dawns. We should therefore endeavor to live comfortably with TEX as Knuth left it.

[*A note for TEXperts*: One can of course wonder whether this is a documentation bug, a program bug, or a widespread implementation bug. I believe it is an accidental documentation bug and an intentional program limitation. As anecdotal evidence I point out that, on page 453 in *The TEXbook* it is asserted that "... the pattern dictionary is static: To change TEX's current set of hyphenation patterns, you must give an entirely new set ...". In fact, this statement is a hangover from earlier versions. Indeed, since version 3 appeared, \patterns (while available) behaves cumulatively as \hyphenation always has, and Knuth says as much in his *TUGboat* article (Knuth, 1989) introducing version 3 of TEX. Bernd Raichle <raichle@informatik.uni-stuttgart.de> confirmed on the basis of his reading of *TEX The Program* that the

limitation lies in the program. Further, Raichle points out another mild inaccuracy in the documentation of `\patterns`: The statement in Appendix H of *The TEXbook*: "All `\patterns` for all languages must be given before a paragraph is typeset" is not strictly accurate. Only a *second* linebreak pass (that for hyphenation) invalidates `\patterns`. Indeed, like `\dump`, it causes the hyphenation trie to be packed; and a second pass can be inhibited, if you set `\pretolerance` to a high value.]

The upshot of this is that the *precompiled* versions of standard formats such as Plain, LaTeX, $\mathcal{A}\mathcal{M}\mathcal{S}$-TeX and LA$\mathcal{M}\mathcal{S}$-TeX that one encounters *cannot* be enhanced to handle a language whose hyphenation patterns have not been installed at the time of the original format compilation. Under TeX in version 2 there was the possibility of flushing out existing patterns at any moment and replacing them with new; these patterns could be arranged to serve two languages at once using a trick that Knuth attributes to J. Hobby (see early versions of *The TEXbook*).

In practice, this presently means that the average TeX user has available only one or two languages chosen by the TeX guru who compiled the formats at the given TeX site. In continental Europe most sites support English and (hopefully!) the national language. Current TeX implementations usually have space for a third language, but — for lack of hyphenation pattern capacity (`trie_size`, and `trie_op_size`) — not more. There being usually no unanimous choice for a third language among a multitude of choices,[1] it is rarely present. But most European users would greatly appreciate having their TeX 'ready to go' in some other language or languages they know. This situation is a deplorable obstacle to the optimal use of TeX in languages other than English, and to wide dissemination of non-English documents in `.tex` format once they have been created.

There is another problem of which *all* scientists and scholars are at least subliminally aware. In a high proportion of scientific or scholarly bibliographies, some titles are in 'foreign' languages. The current custom of using tiny fonts for such bibliographies and inhibiting hyphenation is a barely tolerable stop-gap measure, and, for multicolumn styles, it fails badly, causing incorrect linebreaks. In the near future, authors will, I hope, make it a habit to specify the languages using conventional control sequences (Haralambous, 1992, 1993). Those who employ a multicolumn style have strong reason to demand the ability to *quickly* create a format supporting the required foreign languages. Publishers and others who care about the finer points of typography should *always* demand this ability. This need suggests that, in the near future, *big* TeX implementations should routinely be able to handle somewhat more than the half dozen or so languages they do currently with `trie_size=32` kilobytes. It also reveals a need for just 'basic' services for *many* foreign languages.[2]

The Format-Dumper solution in outline. I feel the best way to offer appropriately designed multilingual formats with TeX 3 is to make the use of INITEX for format compilation an easy matter that any confirmed TeX user will happily undertake on his own. My view is not yet widely shared, probably because INITEX is reputedly an unfriendly animal of which ordinary users should steer clear. More widely accepted is the notion that just tomorrow we all will have computers of infinite capacity and infinite speed, equipped with precompiled universal formats containing all the language features we need. I am skeptical of this optimism and suggest that Knuth is another skeptic:

Suppose you were allowed to rewrite all the world's literature; should you try to put it all into the same format? I doubt it. I tend to think such unification is a dream that's not going to work.

TUGboat, vol 13 (1992), page 424.

I have constructed a provisional setup for exploiting INITEX, called FORMAT-DUMPER. It currently comes in two flavors "-cm" and "-ck", serving Computer Modern and Cork norm font encodings respectively. The master posting is on `matups.matups.fr`, and mirror postings are available on faster CTAN servers (see these proceedings).

On the basis of this experience, which is just beginning to branch out from its bilingual French-English core, I hope to be able to suggest here features for a general framework that should apply to all European languages. It is surely too early to

1 In Haralambous (1992), where two-letter language codes were proposed, the existence of hyphenation pattern files for the following is asserted: Croatian HR, Czech CS, Danish DA, Dutch NL, Finnish FI, French FR, German DE, Hungarian HU, Italian IT, Norwegian NO, Portuguese PT, Romanian RO, Slovenian SL, Spanish SP, Swedish SW, UK English UK, US English US, all these supported by babel (Braams, 1991); and moreover Catalan CA, Estonian ES, Icelandic IS, Lithuanian LT, Polish PL and Slovak SK.

2 Titles include a variety of punctuation, so punctuation is clearly 'basic'.

pick best possible choices, but it is high time to evoke possibilities before a critical public.

I discuss FORMAT-DUMPER under the headings:

- exploitation of standard format files rather than *ad hoc* versions modified for the needs of European languages;
- assemblage of diverse tools in a directory rather than in a monolithic file;
- an interface based on programmed 'dialogs' between INITEX and the user; and
- a stellar protocol for switching language conveniently in multilingual formats.

Interspersed among these are numerous comments on problems raised by the use of active characters in multilingual formats.

Even if FORMAT-DUMPER develops little or ultimately disappears, I expect that some of the new features I discuss will be adopted in fully developed multilingual frameworks of the future.

Format-Dumper plus Babel. FORMAT-DUMPER is, I believe, the first system to address the problem of systematizing format building under INITEX. On the other hand, the first system for organizing multilingual TEX is babel by Johannes Braams (Braams, 1991), a system one finds posted on many servers. It provides adaptations of LATEX and Plain TEX to an impressive list of European languages.

What is the relation of FORMAT-DUMPER to babel? In its bilingual form FORMAT-DUMPER was independent of babel because the French language features were developed by Desarmenien and others before babel was conceived. Much of the effort in the bilingual version of FORMAT-DUMPER went to serving French users of $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX and LA$\mathcal{M}\mathcal{S}$-TEX who were not served by babel. FORMAT-DUMPER offers no services that are specific to LATEX or to any format — although there are a good many patches for the major formats (notably $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX and LA$\mathcal{M}\mathcal{S}$-TEX) in order to permit character activation. Its emphasis is on low-level things: the installation of patterns, the character activation just mentioned, font and accent administration (this part called Caesar).

babel gives little help in managing INITEX, perhaps because babel was designed before the new strengths and limitations of TEX 3 were fully assimilated. On the other hand, after four years of development, babel covers an impressive spread of languages, while FORMAT-DUMPER is just a prototype that is clearly incomplete. babel's emphasis has been on serving LATEX.

From the users' point of view in 1993, the sum of both and much more are wanted, since so much remains to be done for multilingual TEX.

Unfortunately, in a fully multilingual context, this sum cannot remain disjoint. Certainly FORMAT-DUMPER must exploit the vast compilation of macros for the typography of many nations that babel has built up. Further, the examination of language switching that I make in the closing section of this article leads me to conclude that babel's stellar language switching mechanism is effective, natural, and even capable of extension for worldwide use. Indeed, I see no reasonable alternative to it. Consequently, to become truly bilingual, FORMAT-DUMPER must in some manner join forces with babel.

I believe most of the existing and proposed features of FORMAT-DUMPER could be happily married with most low-level features of babel. This seems an attractive way to progress rapidly; in practice this means that the features of babel not specific to LATEX should be physically separated for ready use by FORMAT-DUMPER. Perhaps the LATEX features will be absorbed into version 3 of LATEX, (cf. Haralambous, 1993).

It is still quite unclear to me whether FORMAT-DUMPER should dissolve into the babel system or whether it should continue an independent life by interfacing with a future version of babel.

Contributions from Haralambous, 1992. In a 1992 article that indicates directions for multilingual developments centered on Norbert Schwartz' DC fonts, Y. Haralambous proposed a succinct language switching syntax \FR, \DE, \UK, \US, ... that already has an intriguing history. It is worth quoting the relevant passage:

"The DC fonts and TEX 3.xx's language switching features require new macros, which will also have to be standardized.

"These macros are of two kinds:

" 1) macros for accessing accented or special characters which are not available in the Computer Modern fonts [... omitted]

" 2) macros for language switching (see 'Language-switching macros', below). Since it is now possible to typeset a multilingual text where each language uses its own hyphenation rules, its own fonts and eventually its own direction of script, there must be a standard (and simple) way to switch between these languages.

"So if, for example, you want to include German or French words in your English text — 'Wagner's *Götterdämmerung* was very appreciated by the 19th century's *bourgeoisie*' — they will have to be hyphenated according to German and French rules. To indicate this to TEX, macros have to be selected which are easy to remember, short, and universally acceptable.

"For this, the standard has basically been taken from the standard 2-letter ISO 639 language codes as control sequence names [footnote on exceptions omitted] (see 'Language-switching macros', [omitted]).

"The previous example is then written:

```
\US Wagner's
\DE {\it G\"otterd\"ammerung\/}
\US was very appreciated
by the 19th century's
\FR {\it bourgeoisie}
```

I considered these recommendations very positive and felt that they deserved to guide the development of language switching in FORMAT-DUMPER. When presented orally at the EuroTEX conference in Prague (September 1992), they raised mild objections from persons who felt that more explicit names such as \french, \german, \ukenglish, \usenglish, ... would be more suitable. On the other hand, the idea of using an ISO standard for these new lower-level macros had the great advantage of leaving undisturbed existing higher-level macros like \french and \german. The use of capitals (\FR not \fr) further reduces the probability of conflict with existing macros.

Recall that in becoming truly multilingual FORMAT-DUMPER is faced with the awkward matter of assimilating the low level language features from the present babel that constitute a core of essential services which all users will want, while leaving aside a plethora of optional high level features that belong more to a style file than to a base format. I therefore set out to implement Haralambous' syntax in a simplest possible fashion. However, I quickly found myself skating on thin ice; the difficulties described in the closing section made me fall back on the approach of babel.

In retrospect, one can perceive warning signs even in the above quotation. Why was the syntax not as follows?

```
\US Wagner's
{\DE\it G\"otterd\"ammerung\/}
was very appreciated
by the 19th century's
{\FR\it bourgeoisie}
```

This syntax is more natural and and reduces the uses of \US from two to one. For this to work, \US, \FR and \DE must respect TEX grouping.

I surmise that Haralambous intended at the time of publication (Haralambous, 1992), that \US, etc., cause *global* changes which do *not* respect TEX's grouping. This turns out to be rather objectionable, because such global changes are confusing to the user in the presence of pre-existing language macros that do respect grouping, beginning with those of Desarmenien for French in the 1980's. Furthermore, Knuth warns (*The TEXbook*, p. 301) that use of *both* local and global changes of the same entity can lead to a slow poisoning of TEX's grouping mechanism.[3]

But this answer raises a second question. Why would Haralambous have proposed \global changes? There are known approaches to implementing language changes by using grouping (and respecting grouping); they do work but they encounter difficulties that I will explain below in motivating the stellar protocol.

When the French translation (Haralambous, 1993) appeared in spring 1993, the macro syntax proposal was omitted without comment. One unfortunate consequence of this disappearance of \US, \FR, \DE, etc. is that the related internal macros I will propose for language switching may not be welcomed. On the other hand I would be very pleased to contribute to rapid rehabilitation of these handy control sequences in a form respecting grouping.

The reconstruction of babel's language switching in the closing section offers somewhat more than modernized notation. I feel that use of a virtual language EC at the center of the star in place of US is a step in the right direction — the 'communal' organization functions slightly better than the 'parental' one. It also suggests a tree-like structure for a worldwide system.

Use of Standard Formats

The use of *unmodified* standard versions of the .tex files defining standard formats is recognized to be a feature vital for the stability, compactness and clarity of any TEX setup. This discipline is accepted by FORMAT-DUMPER; it does *not* mean that undesirable

3 At one point there was also a failure of babel's system to respect grouping; see the \gdef on page 294 of (Braams, 1991).

behavior of a format cannot be modified; but rather that modifications will all be done by 'patch' files external to the standard format files.

By dint of somewhat 'dirty' patching trickery it becomes possible to use standard `plain.tex` even when Cork norm fonts are used. The read-only memories (=ROMs) that provide the low-level routines for micro-computers usually provide a protocol for patching; similarly it would be a good thing if future versions of basic formats were built to make patching less tricky.

Just a few kilobytes specific to FORMAT-DUMPER are needed to compile a typical individual's desired format. Thus it is possible, for example, for a student in Australia with a microcomputer and a modem of speed only 2400 baud to obtain, by `ftp` from a European site, enough of FORMAT-DUMPER in just a few minutes to automatically compile a good French-English bilingual format with the help of standard formats he already possesses.

Assemblage of Diverse Tools as a Directory

Format files, language hyphenation files, patch files, language files for typography and typing, and, finally, dumper files organizing the foregoing — these are the main constituents of the FORMAT-DUMPER directory. There is even a catch-all `initexme.tex` dumper file which leads the user to all possible format choices.

The patch files for French required to accommodate possibly active punctuation are non-trivial (although compact) with 1993 $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX, and LA$\mathcal{M}\mathcal{S}$-TEX. I expect that optimal use of many other languages will require some character activation, but hopefully less than for French.

As FORMAT-DUMPER covers more languages and formats, the number and total volume of files in the FORMAT-DUMPER directory will become considerable. On the other hand a single user wanting to build a specific format needs only a few kilobytes of files. How can one enable the user exploiting ftp to get just what he needs? One interesting possibility (among many) is to have a one-file 'scout' version of FORMAT-DUMPER which produces not a format but rather a 'roster' file, each line of which is of the form

`mget` ⟨*filename*⟩

This roster will serve under most operating systems to fetch automatically by ftp all the required files, and not more.

The thorny issue of character activation. Without this challenge, I might never have bothered to create FORMAT-DUMPER! However, only the impact on the design of FORMAT-DUMPER is of interest here.

Some TEXperts (myself included) feel that activation of several characters, notably `;:?!` (which in French typography should yield subtle preceding spacing) is the only way to make French typing for TEX at once convenient, portable, and typographically correct. French is perhaps the most troublesome language in this regard, but others have similar problems; for example, German TEX formats often activate " for a variety of reasons. Such activation unfortunately requires that formats and macro packages be 'cleaned up'; FORMAT-DUMPER does the necessary patching non-invasively. Hopefully, someday soon, the standard versions of major formats will be 'clean' in this sense.

Other TEXperts, firmly believe that category change is too troublesome to cope with — arguing that there will always be macro packages that have not been 'cleaned up'. They accept less convenient typing, for example `\;` for semicolon in prose, or they have to use a preprocessor. For them, an option keeping `;:?!` etc., *inactive* (i.e., of category 12 = other) will be provided by FORMAT-DUMPER.

There is a residual category problem to be investigated in multilingual TEXs. It is known to be troublesome to switch category codes at times when TEX is reading ahead during its macro expansion process, for instance when it comes to expand the middle of a big macro argument. This arises because category code change cannot influence what has already passed TEX's lips. If the act of changing category code is dangerous, then we should perhaps not change category code in the process of changing language! That has indeed always been a policy of FORMAT-DUMPER.

Then what category choice is appropriate for an English-French-German format? There is a practical answer in this known case: have `;:?!` active and " inactive, but allow users of German to activate " *with due care* when they wish — using a command `\flexcat!\activate"`. An error warning is issued by `\flexcat!` in the rare cases when category switch is dangerous at that point.[4] A language change to French there could cause bad punctuation (*n'est-ce pas ?*) — if one were to change category

4 Such dangerous points cannot be predicted on general principles. But, it may be helpful to note that they are more or less the points where 'verbatim' macros break down.

code along with language. A 'danger warning' macro for category change is perhaps a new idea; here is a quick explanation. `\flexcat` temporarily changes the category of ! to 11 (letter) and then examines the category of the following !; if it is not 11 an alarm goes off. For more details see my July 1993 posting mentioning `\flexcat` in the GUT TEX forum archived on `ftp.univ-rennes1.fr`.

I am therefore optimistic that category activation and multilingual TEX formats can be mixed if due care is taken. There always remains an 'all inactive' option, but it would be sad to see TEX become less friendly in an international context!

A User Interface Based on Programmed 'Dialogs'

This means dialogs between INITEX and the user orchestrated by a 'dumper' script. Mike Downes has written a very valuable guide for building such dialogs (Downes, 1991).

This interface makes the FORMAT-DUMPER system nearly self-documenting and an incredible number of distinct compiled formats becomes readily available to a diverse population of users. FORMAT-DUMPER is a self-serve 'format supermarket'!

The user will be able to select languages at will from a list of languages currently supported, subject only to limitations of TEX capacity. For each language the user will specify (from a short list of options) which characters he will make active. One of the options should always be *none* (i.e., no new active characters)! The format being produced will, for reasons explained above, have as active characters all those characters designated as active by the user for *one or more* of the languages included. Long experience in having `;:?!` active for English in a French-English format encourages me to believe that this *modus vivendi* will be acceptable.

There are many options it is wise to include into a format dumped by FORMAT-DUMPER rather than `\input` them repeatedly later. The $\mathcal{A}\mathcal{M}\mathcal{S}$ math fonts are one of my steady personal choices. It will be possible and desirable to put more and more good things into a compiled format, in order to to fit the users special needs.

To prevent this wealth from bewildering the user (and those who have to handle his or her `.tex` files at a later date) devices are needed to recall the salient features of the format built and permit an equivalent format to be created in any other environment. Even the casual user will want this under MSDOS operating systems, where the format name is limited to 8+3 letters and hence inadequate as a source of information.

Here are two devices for format identification. By using the `\everyjob` primitive, a telegraphic summary of the format can be placed on the log at each launch. Further information should be provided by an optional command `\ShowFormatInfo`. This should indicate all the essential choices made in creating the format. This information should be formatted as `.tex` comments suitable to become part of the header of any `.tex` file. This should assure world-wide portability of the `.tex` files for any format produced.

Format compilation, for a book say, is a process that I tend to repeat often until just the right recipe is found: just the right fonts and macros, no more. There will be a way to have FORMAT-DUMPER *learn* to anticipate what one wants, so that most multiple choices can be bypassed as the default choice, i.e., by simply striking the return key. Here is the idea: FORMAT-DUMPER should learn to make the right choice the 'default' choice. For this, one can have FORMAT-DUMPER make the default choice be the choice made on the previous dumper run; choices would therefore be written to an auxiliary file. (If the last run is not recent, as measured by `\today`, one might reasonably revert to 'default defaults'.)

Using compiled formats to good effect. At a later stage, it saves much time to build upon formats produced by FORMAT-DUMPER, say `fe-LaTeX`, to obtain a more temporary format that attacks your typesetting *almost instantly*. If your (LA)TEX often spends many seconds chewing through macros before starting genuine typesetting, you have some tricks to learn.

They will be illustrated by an example. Given a `.tex` file for (say) `fe-LaTeX`, give the name `x.tex` to the segment of up to (and possibly including) `\begin{document}`, and let `y.tex` be the rest. Then add `\dump` to the end of `x.tex` and compile with INITEX, normally using the syntax

```
initex  &fe-LaTeX  x          (1)
```

A format is dumped; let it be called `x-LaTeX`. This new format is the one that then attacks `y.tex` instantly, using VIRTEX with the syntax

```
virtex  &x-LaTeX  y           (2)
```

On a unix system, one can alias this ephemeral but oft repeated command as (say) `ty`; further one might use 'piping' to make `ty` also preview the result.

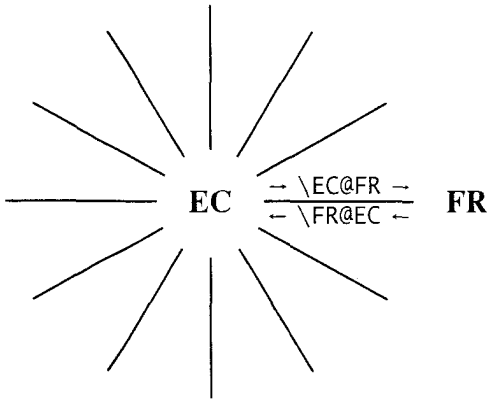
Regrettably, a number of routine but system dependent technicalities may necessitate help from your system manager. The most frustrating is perhaps difficulty in accessing a freshly dumped format. I recommend wider user education concerning the way TEX accesses the various files it needs.

It is a praiseworthy initiative of *Textures* on Macintosh computers to incorporate into VIRTEX all the functionality of INITEX. This means that *Textures* users are less intimidated by format dumping. Instead of the command (1), they have *Textures* compose `x.tex` using format `fe-LaTeX`; and instead of (2), they have *Textures* compose `y.tex` using `x-LaTeX`.[5]

Switching Language

The stellar protocol described here is morally that of babel, but I have permitted myself liberal changes of detail in the hope of clarifying ideas and influencing future versions of babel.

The proposed center of the star is not English but rather a virtual language whose two-letter tag should be EC (for European Community); it would have neither hyphenation patterns nor exceptions. (More details on making EC a convenient institution come later.)



For each language, say French, a number of language-change control sequences should be defined.

(a) `\FR` for a middle-level language change, as proposed by Haralambous (1992).

(b) `\EC@FR` and `\FR@EC`, macros which are of a lower level than (a).

(c) `\l@FR` is TEX hyphenation system number for FR. `\language=\l@FR` switches to it. This is the lowest level.[6]

The shape of this protocol is *stellar* since all languages are explicitly related to the virtual central language EC as the figure above indicates and *not* directly among themselves.

To motivate this design we now examine the nature of the switching problem and the weaknesses of simpler designs.

Unlimited complexity of language. The macro `\FR` must be more complicated than one first thinks because its action in general should *depend on the current language at the time it is called.* Indeed, special features for the current language have to be neutralized. The features I have in mind are chiefly the behavior of punctuation and spacing that are peculiar to the current language, but they may include typing conventions and there are many additional bits and pieces. (The hyphenation patterns and exceptions themselves do not pose any problem, since by calling new ones we automatically put aside the old.)

These bits and pieces are sufficiently important that they cannot be ignored, and sufficiently numerous that one does not have time to reset all the features and parameters that some other unspecified language might conceivably have altered. There is an attempt in Haralambous (1992) to put an *a priori* bound on the things that that will change under language switching; I feel that this is impractical, and fortunately we will find it unnecessary. Let us look at some bits and pieces.

The Czechs have a rare but logical and unambiguous convention for splitting hyphenated words at line ends: the hyphen is programmed to survive at the beginning of the following line. There is the question whether the parts of a word spelled

5 There was an unfortunate bug in versions 1.4 and 1.5 of *Textures* that virtually forced one to quit *Textures* before using a (re)compiled format. The long life of this bug, now fixed by version 1.6, confirms my impression that even in the exceptionally good *Textures* environment, format compilation has been underused!

6 (a) babel's `\selectlanguage {francais}` corresponds roughly to `\FR`. In fact the syntax `\selectlanguage {FR}` should be retained in programming.
(b) babel's `\extrasfrancais` corresponds roughly to `\EC@FR`, and `\noextrasfrancais` to `\FR@EC`.
(c) babel's `\l@francais` corresponds to `\l@FR`.

with a hyphen should be subject to hyphenation at linebreaks. French spacing around punctuation involves \frenchspacing which *suppresses* extra spacing after punctuation (conventional in English) and *adds* peculiar extra spacing *before* the four punctuation points ;:?!. Another detail is the positive \lccode the French assign to the apostrophe (English gives it \lccode=0). I expect the list of such language dependent features will never be complete.

Accent style becomes a nightmare for TEX. With CM fonts, French users usually prefer to suppress accents on capital letters, although with a suitably designed font family, French typography recommends their use. Incidentally, suppression can be gracefully handled by adding a few hundred octets of macros to the accent administration package 'Caesar' currently used by FORMAT-DUMPER.

Far more vexing for multilingual format building are disagreements in accent positioning between, for example, French and Czech (Zlatuška, 1991); thus I permit a digression here. At present this disagreement seems to require that different font systems be used for French and Czech, at considerable cost — simply by reason of a small number of misplaced accents. Although this is wasteful, let it be conceded that the setup can be gracefully handled by the NFSS (New Font Selection System) of Mittelbach and Schöpf. Indeed, 'language' can be one more 'property' analogous to 'shape' or 'size'. This solid but weighty solution is recommended by Haralambous (1993) and is materially supported by NFSS version 2.

I mention that there may perhaps be an efficient solution based on the stellar language switching we are studying, and the notion of a SPECIAL within a character description in a virtual font.[7] Consider the task of enhancing a future virtual version of the DC fonts of N. Schwartz to allow the positioning of accents favored by Czech typography (Zlatuška, 1991). Each language change into or out of Czech should be marked in the .dvi file by a TEX \special command. For implicit language switches induced by TEX's grouping, the \aftergroup primitive helps to insert the \special command. Nevertheless, the macros \CS@EC and \EC@CS for Czech are the only macros in the switching scheme that need to be enhanced. The virtual DC fonts would then be enhanced by including SPECIALs within the MAP description of each accented character altered for Czech, indicating the modified accent positioning. This would, in turn, require that drivers learn to interpret these specials in the intended order. (These SPECIALs and their modifications are ignored by current drivers.) By this method, single virtual font can become a sheaf of virtual fonts indexed by a parameter or parameters. In this context, the parameters correspond to language related variations, and a language change typically causes tiny alterations on a few font characters. This solution is a 'pipe dream' — but hopefully one with real potential.[8] (I believe it worthwhile to dream up SPECIAL extensions of virtual font mechanisms, as that will, in the long run, stimulate the improvement of drivers and driver standards.)

I have argued (with digressions) that there is a limitless number of things \FR would have to do if it were not known what language we are switching out of. Presently, I will observe that it is easy to have TEX keep track of the current language and that an efficient way to switch fonts is obtained by going between any two languages *via* the central language EC using the macros (b). But, before plunging into more detail, we really must be convinced that simpler pre-existing schemes are inadequate.

Language switching based on grouping. Many readers will surely have thought of a simpler language switching scheme based on grouping. Suppose we begin each typescript always in a preferred 'base' language, say US (where TEX began). Then, if we define \FR to switch from the base language to FR and similarly for \DA and \CS, the syntax

{\FR ⟨*French text*⟩}
{\DA ⟨*Danish text*⟩}
{\CS ⟨*Czech text*⟩}

will provide correct language switching. Alternatively, the LATEX 'environment' syntax:

\begin{FR}⟨*French text*⟩\end{FR}
\begin{DA}⟨*Danish text*⟩\end{DA}
\begin{CS}⟨*Czech text*⟩\end{CS}

could accomplish the same. In essence, why not rely on the powerful grouping mechanism of TEX to get us back to the base language at the end of a group, thereby economizing half the macros proposed in (c)? My answer is that this well-known approach is indeed valid but often *inadequate*!

7 There is another approach via an enhancement of the \charsubdef addition to TEX by Mike Ferguson <mike@inrs-telecom.uquebec.ca>, but Mike warns me it is not easy to implement.

8 Character shapes can change arbitrarily; the main restriction is that the font metric data cannot change.

It is inadequate when the current language has been declared at a higher grouping level than at the spot where we propose to switch language; indeed closing groups before switching — in order to return to the base language — is not an option there, since it has, in general, disastrous side effects. (In addition, one would not even know how many groups to close!) For example, the desirable syntax

```
\FR ⟨French text⟩
\begin{anyenvironment}
⟨French text⟩
{\DA⟨Danish text⟩}
⟨French text⟩
\end{anyenvironment}
⟨French text⟩
```

would be invalid, and clumsy to fix. Such clumsiness is unacceptable in many applications, such as literary criticism, history, art, travel, music, etc., where many quick language switches are required.

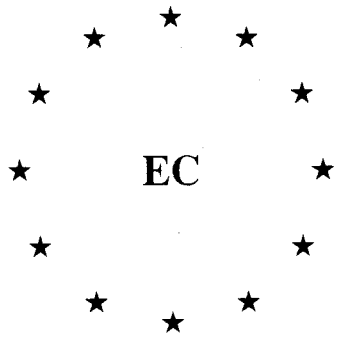
Note that the above example would pose no problem *if* the notion of language were encompassed by a fixed finite number of features such as hyphenation rules, fonts, and direction of script. Indeed it would suffice to have \FR, \DA, etc., each adjust all of these features; the change to Danish would *automatically* undo all the features of French. Unfortunately, this 'Cartesian' view of language seems inadequate, as was explained above.

Functioning of the stellar protocol. I hope the above analysis of alternatives will convince readers to take the stellar protocol seriously. Here are some details to explain how it provides valid language change, say in this last example.

At any moment, a reserved register \l@toks contains the tag of the current language, say FR, and there exists a macro \FR@EC pre-defined to cause reversion from FR conventions to the EC conventions. Its first duty is of course to change the value of \language from \l@FR to \l@EC. Its multiple other duties are to dismount all the special features that \EC@FR will have introduced at an earlier time.

The action of \DA is then double: First, it looks at the current language register \l@toks and uses what it finds (FR say) to revert to EC by \FR@EC. Second, it applies another pre-programmed macro \EC@DA to pass from EC to Danish, and replaces FR by DA in the current language register.

Thus, for each language (FR say), the crucial problem is to define two macros \EC@FR and \FR@EC to introduce and suppress national features.



Eurocentrism. The shape of this solution recalls the twelve-star circle on the European flag. It installs eurocentrism — since change from one national language to another passes through the formal center EC.

What should be the features of the artificial language EC? One wants it to be a sort of center of gravity to which one can easily revert. Thus, I would give it the consensus \lccodes used by Ferguson in his MLTEX. Null hyphenation patterns let EC serve in a pinch in lieu of a missing language, say \DA, at the cost of introducing soft hyphens by hand; \DA should then be \let equal to \EC, which prevents incorrect hyphenation for this language (and no more). Incidentally, \EC is defined like \DA, but of course \EC@EC does nothing.

For the rest, Knuth's features for English probably make a reasonable choice; then at least we all can remember what EC means!

Adaptability and extensibility of the switching scheme. For the purposes of FORMAT-DUMPER, the language change macros \FR, \US, etc., should initially be low to middle level. But, since various complex styles may be loaded later, there must be possibilities for enhancement of the features for any language.

Permanent enhancement is straightforward. On the other hand, a macro package might want to provide a macro \myfrenchextras to *conveniently add* French language features that will *go away* when we next leave French. This suggests a possible enhancement to the language change protocol described, which we now explain for the switch from French to Danish. \myfrenchextras should both introduce the extra features and add to a reserved token sequence \@ECtoks[9] an action \@undomyfrenchextras. Then, when we leave French by \DA, the enhanced mechanism would first

9 babel's \originalTeX corresponds roughly to \@ECtoks.

make \the\@ECtoks act, then empty \@ECtoks, and finally execute (as normally) \FR@EC\EC@DA. There are many possible variations to explore. Even the naive use of grouping will often produce adequate results.

One could allow more than one manifestation of a given language, say basic French FRb as presently supported by FORMAT-DUMPER, or classical French FRc as supported by `french.sty` of B. Gaulle (available on `ftp.univ-rennes1.fr`), or again FRx a French publisher's distinctive typography. A new manifestation of a language can clearly be added 'on the fly' by a user (without dumping using FORMAT-DUMPER), so long as the hyphenation scheme coincides with that of a manifestation already installed. It seems unlikely that rapid switching between such variants of French would be of any interest; hence any one could appropriate the standard macro \FR.

The opposite is true where dialects of a language are concerned, since conversations (say in a play) would require rapid switching. The 'closeness' of the dialects of one language could be most efficiently exploited using a treelike rather than stellar protocol (below).

Incidentally, to facilitate the above extensions, TEX should maintain a list in a standardized form of names of the installed hyphenation systems and another of the installed languages (with switching macros).[10]

A world-wide system. Naturally, one dreams of a world-wide system with the virtual language UN (for United Nations) at its center. Its natural shape would, I imagine, be treelike rather than stellar; EC might be one of several internal nodes corresponding to a language grouping. Two points of the tree are always joined by a unique shortest path (usually shorter than the one passing through UN) and this may permit a natural and efficient generalization of the stellar switching scheme that has been sketched above.

Such are the simple but powerful ideas underlying the language switching in babel. It seems clear that there are no insuperable obstacles to having future versions of babel and FORMAT-DUMPER collaborate — in order to provide convenient compilation of multilingual formats using INITEX.

I am pleased to thank Johannes Braams, Klaus Lagally, Frank Mittelbach, Bernd Raichle, and the referee for helpful comments on the preliminary version of this article.

10Beware that the correspondence of the two lists need not be exactly one-to-one. There may be different hyphenation systems for the same language (commonly minimal and maximal). And there may be more than one language manifestation/dialect using the same hyphenation system. In babel, an ephemeral external file `language.dat` bears this information at format compilation time; something more robust and accessible is called for — see (Taupin, 1993).

Bibliography

Braams, Johannes. "babel, a multilingual style-option system for use with LaTeX's standard document styles". *TUGboat*, **12**(2), pp. 291–301, 1991; the babel system is available on the major TEX archives.

Downes, Michael. "Dialog with TEX". *TUGboat*, **12**(4), pp. 502–509, 1991.

Haralambous, Yannis. "TEX Conventions Concerning Languages". TEX and TUG News, **1**(4), pp. 3–10; this article with its useful tables is available in digital `.tex` form on `ftp.uni-stuttgart.de`.

Haralambous, Yannis, "TEX Conventions concernant les polices DC et les langues". Cahiers GUTenberg **15**, pp. 53–61, 1993.

Knuth, Donald. "The New Versions of TEX and METAFONT". *TUGboat*, **10**(3), pp. 325–328, 1989.

Taupin, Daniel. "Commentaires sur la portabilité de TEX". Cahiers GUTenberg, **15**, pp. 3–31, 1993.

Zlatuška, Jiří. "Automatic generation of virtual fonts with accented letters for TEX". Cahiers GUTenberg, **10–11**, pp. 57–68, 1991.