Proč nerad používám LaTeX

V tomto článečku bych chtěl uvést několik věcí, které se mi na koncepci LaTeXu nelíbí a vysvětlit, proč jsem zůstal u plainu. Nechci tímto článkem vyvolávat nepokoje v řadách CSTUGu. Vážím si samozřejmě lidí, kteří perfektně rozumějí LaTeXu a tento článek nechce nijak znevažovat jejich práci.

LaTeX je nadstavba postavená nad formátovacím programem TeX. Podle mého názoru hlavní cíle LaTeXu jsou následující:

- Odstínit poměrně složitou problematiku TeXu od koncového uživatele.
- Vytvořit vlastní jazyk vstupních textů.
- Umožnit formátovat jednoduché dokumenty podle předzpracovaných stylů.
- Umožnit snadnou výměnu dokumentů a jejich nové přeformátování.

Těchto cílů bylo dosaženo jen částečně. Pokusím se dokázat, že úplné dosažení stanovených cílů není možné z principiálních důvodů a že brzdou v této věci je, překvapivě, samotný TeX.

Nedostatečnost jazyka TeXu

TeX nebyl stvořen se záměrem, aby se nad ním dělaly univerzální nadstavby na úrovni jeho jazyka maker. Rozhodně nelze tyto nadstavby prezentovat veřejnosti jako uzavřené systémy. Na to není programovací jazyk TeXu stavěný. Proto nechápu, proč se o to autoři různých vylepšení LaTeXu neustále snaží.

Můžete argumentovat, že vždy musí existovat hranice, pod níž nemusím zkoumat, jak systém funguje, mám-li si zachovat duševní zdraví. Například při jakékoli činnosti s počítačem potřebujeme určitou míru abstrakce. Programujeme-li třeba databázovou aplikaci, zajímají nás elementární nástroje vývojového prostředí a nezabýváme se tím, jak putují elektrony po polovodičích centrálního procesoru.

Pro tento způsob odstínění problému byla na každém stupni abstrakce vytvořena v počítačových vědách dobrá koncepce, která odstínění umožní. Musíme si ale přiznat, že TeX takovou dobrou koncepci nenabízí. Nelze pomocí jeho makrojazyka vytvořit novou úroveň práce, při které není potřeba vracet se k vestavěným vlastnostem TeXu. Z toho pramení většina problémů LaTeXu (jak uvidíme jednotlivě dále). Základní chybou LaTeXu tedy je, že si stanovil cíle, pro které nebyly vytvořeny podmínky v TeXu samotném.

Uvedeme příklad. V běžných programovacích jazycích může programátor elementárními prostředky ošetřit všechny neočekávané situace programu. Všichni víme, že dobrý program by měl mít ošetřeny všechny alternativy vstupu a považujeme za ostudu, když program havaruje se systémovou chybou. Makrojazyk TeXu ale nemá potřebné nástroje na ošetření všech závad na úrovni „LaTeX error“ (srovnej: chyba ošetřená programem). Chyba často vyjde najevo až po činnosti vnitřních algoritmů TeXu a projeví se jako „TeX error“ (srovnej: systémová chyba programu).

Pokud by se autor makra snažil vše ošetřit na úrovni svého makra, byl by vzniklý kód z 90 % o tom, jak ošetřit uživatelské výstřelky a ve zbylých 10 % o vlastní sazbě. To je myslím zbytečná investice jednak lidského potenciálu (mám na mysli lidi, kteří taková makra dělají a jiní, kteří jejich výtvory musejí číst) a jednak strojového potenciálu (mám na mysli strojový čas a paměť počítačů).

Snaha LaTeXu o odstínění složitosti TeXu

Jedním z cílů LaTeXu bylo umožnit nepoučenému uživateli používat TeX, aniž by musel pronikat do složitostí tohoto programu. Nakonec ale sami LaTeXoví odborníci přiznávají, že bez TeXbooku to prostě nejde. Takže snaha o odstínění problematiky TeXu je vlastně zbytečná.

Představte si, že si nějaký uživatel přečte LaTeXovou příručku a nabyde dojmu, že mu bude stačit rozumět problematice sazby na úrovni této příručky. Pak se jednou překlepne třeba při sestavování tabulky a na terminálu na něj TeX křičí: `Extra alignment tab has been changed to \cr`. Uživatel začne znovu listovat ve své příručce a zjistí, že tam o žádném `\cr` není jediná zmínka. Má pak tři možnosti: (1) Zmáčkne Enter a podobně se zachová i u dalších chyb. Pomyslí si, že ten LaTeX je něco tajemného a mystického. (2) Propadne zoufalství a jde od toho. Dojde k závěru, že je lepší zůstat u Wordu. Vždyť stačí vzít tabulku v Excelu a jednoduše ji přemístit do Wordu a jaképak smolení se s nějakým podezřelým `\cr`. (3) Pořídí si TeXbook a po intenzivním studiu nakonec řekne: „aha“. V tuto chvíli ale už nepotřebuje, aby mu LaTeX zakrýval složitost TeXu.

Nebudu podrobně popisovat zoufalý výraz v očích každého začínajícího uživatele LaTeXu, když po použití `\vspace` tak, že nad značkou ani pod ní není prázdný řádek, dostane nečekaný výsledek. Ve své příručce nemá žádnou zmínku o tom, co to je `\vadjust`, takže mu připadá chování `\vspace` jako tajuplné. Podobně zoufale se uživatelé dívají na obrazovku svého počítače, pokud jim uplavou plovoucí objekty (tabulky a obrázky) úplně jinam, než předpokládali. Přitom oni chtěli jen použít značku `\caption` a to je přinutilo definovat objekt v prostředí, v němž může uplavat na jinou stránku.

Utajení skutečností

Utajení některých důležitých vlastností TeXu v LaTeXových příručkách v největší míře souvisí s předchozím bodem o snaze odstínit složitost. Ovšem některé chybějící informace jsou těžko odůvodnitelné. Za všechny uvedu jen dva příklady.

LaTeXový uživatel se nikde nedočte o makru `\choose`, které je elementární zkratkou použití primitivu `\atopwithdelims` a vytváří binomické koeficienty. Můžeme sice v LaTeXu napsat

```
\left(\begin{array}{c} n\\ k\end{array}\right)
```

ale lidé aspoň s nepatrnou mírou vkusu tuší, že to je podivná snůška TeXových primitivů (`\left`, `\right`) kombinovaná s LaTeXovým monstrem `array`.

Druhým příkladem je chybějící zmínka o primitivu `\eqno` a `\leqno` pro značkování rovnic. Skutečně, v celém Lamportovi to nenajdete a v jiných příručkách taky ne. Autoři těchto příruček zřejmě žili v domění, že jejich čtenáři si za všech okolností vystačí s automatickým značkováním rovnic, které jim poskytuje LaTeX. Ovšem chyba lávky. Zúčastnil jsem se jednoho LaTeXového kurzu, kde řešili problém, jak dostat na okraj rovnice konkrétní značku, nepřidělenou žádným automatem. Například značku (P). Řešení tohoto úkolu bylo impozantní. Nejprve se vysázela rovnice, pak se pomocí záporného `\vspace` vycouvalo zpět na úroveň rovnice, dále se použilo `\begin{flushright}`, napsala se značka a konečně se to uzavřelo pomocí `\end{flushright}`. To asi nepotřebuje dalšího komentáře.

Nerozlišování mezi LaTeXem a TeXem

Skoro v žádné LaTeXové příručce se nedozvíme přesnou souvislost mezi LaTeXem a TeXem. Občas v některé najdeme mlhavé přirovnání rozebírající vztah závodního vozu s pohodlným rodinným korábem, ale vyložení technických souvislostí považují asi autoři příruček za složité.

Z uvedených důvodů se v LaTeXových příručkách nerozlišuje mezi vlastnostmi TeXu a LaTeXu. Pro jednoduchost se všude mluví o LaTeXu. Například LaTeX formátuje odstavce, LaTeX rozdělí text na více stránek. Víme, že to není pravda. Když pak někteří LaTeXoví uživatelé přijdou na to, že kromě LaTeXu existuje ještě nějaký podivný TeX, jsou obvykle zmateni. Své zmatení dávají často najevo novotvarem (La)TeX.

Vlastní jazyk vstupních textů

LaTeX se pokusil definovat vlastní jazyk pro značkování vstupních textů. Snaha byla, aby značky podléhaly pokud možno jednotným principům a byly pro uživatele pochopitelné. Jazyk má také pomoci uživateli vyhnout se různým syntaktickým úskalím jazyka TeXu.

Srovnejme způsoby zápisu, které zhruba(!!) znamenají totéž:

TeX	LaTeX
`\hsize=30pc`	`\setlength{\textwidth}{30pc}`
`\vskip1cm`	`\vspace{1cm}`
`{a\over b}`	`\frac{a}{b}`
`\hbox{abc}`	`\mbox{abc}`
`\vbox{...}`	`\minipage{...}` nebo `\parbox{...}`

Bohužel (nebo bohudík?), v LaTeXu je možno psát i některé přímé TeXové konstrukce. To je další nedůslednost v odstínění problematiky TeXu před LaTeXovým uživatelem. Jakmile to LaTeXový uživatel zjistí, začne většinou míchat oba dva způsoby zápisu: TeXový i LaTeXový. To dokazuje, že snaha o vytvoření lepšího, přehlednějšího, a přitom stejně mocného jazyka, se zcela nepovedla.

TeX nám například hlásí `Overfull \hbox` a LaTeX jako makro nemá prostředky toto hlášení změnit na `Overfull \mbox`. Záznam o sazbě je tedy LaTeXovému uživateli předkládán pomocí TeXového a nikoli LaTeXového slovníku. Považuji to za matení nebohých uživatelů.

Strukturované značkování

Pod tímto pojmem rozumím značky, které člení dokument do struktury (například kapitola, citát, poznámka) a nezabývají se typografií dokumentu (zda použít `\medskip` nebo `\bigskip` a jaký zavést do sazby font).

LaTeXoví zastánci zdůrazňují, že koncepce vstupního jazyka LaTeXu vede uživatele ke strukturovanému značkování. To není pravda.

Uživatele, který teprve začíná pracovat s TeXem nebo s jakýmkoli jiným sázecím systémem, velmi těžko přesvědčíme o výhodách strukturovaného značkování. Chce to dlouhodobou praxi, než člověk na výhody tohoto způsobu značkování přijde. Většina lidí se rozhodne pořídit první dokument proto, aby jednotlivé jeho části vypadaly přesně tak a tak. Nasadí si tak klapky na oči, které se těžko sundávají.

Mé zkušenosti s dokumenty různých autorů ukazují, že v používání strukturovaného značkování není rozdílu mezi uživateli plainu a LaTeXu. Spíš bych řekl, že dlouhodobá praxe vede ke strukturovanému značkování bez závislosti na tom, jakými prostředky je text pořízen.

Domnívám se také, že člověk snáze dospěje ke koncepci strukturovaného značkování, pokud má vzhled dokumentu absolutně pod kontrolou svých vlastních maker. Používat značky vymezující strukturu dokumentu a nevědět, co se *přesně* stane v době, kdy mě bude zajímat vzhled dokumentu, může být pro uživatele, který je zodpovědný i za typografii dokumentu, frustrující.

Idea strukturovaného značkování se nejlépe projeví při dělbě práce: písařka a sazeč. Pokud sazeč chce postavit sazbu dokumentu na plainu, dohodne se s písařkou na některých značkách definujících strukturu dokumentu. Jestliže chce písařka průběžně vidět, jak dokument vypadá, dají se některé značky narychlo nadefinovat (například `\let\značka=\relax`). Při vlastním zpracování dokumentu pak sazeč definuje jednotlivé značky podle požadavků na typografii dokumentu. Vzhled knihy mu vzniká rovnou pod rukama a před očima.

Pokud sazeč při návrhu značek vychází z plainu, má písařka většinou podstatně méně práce. Sazeč totiž může s výhodou využít mocný nástroj v podobě separátorů parametrů v makrech TeXu, který není v LaTeXu pro potřeby uživatele využit. Ve smluvených značkách je pak daleko nižší výskyt kučeravých závorek a dalších droblinek, které nejsou zcela potřeba. Například pro název kapitoly se ti dva mohou dohodnout na:

`\kap Název kapitoly`
⟨*prázdný řádek*⟩

Je to jednak přehlednější a jednak to vyžaduje od písařky méně námahy, než kdyby postupovala podle LaTeXu.

Ideální dělba práce

V předchozí úvaze jsme naznačili ideální dělbu práce. Autor nebo písařka pořídí text se značkami podle pokynů sazeče. Sazeč, který se naučí TeX, pak sám doplní makra podle pokynů typografa. Domnívám se, že to byl původní záměr autora programu TeX.

LaTeX se snaží uvedenou dělbu práce obejít. Autor prý může napsat text podle jednoduché příručky k LaTeXu a sazečem se stává samotný LaTeX. Protože schází lidský faktor mezi autorem a konečným vzhledem dokumentu, trpí tím často autor i dokument.

Z jedné strany dává LaTeX autorovi najevo, že typografii zvládne automat sám a dobře (což vůbec není pravda) a z druhé strany nakonec nutí autora získat znalosti, které by při lidské dělbě práce nebyly pro něj potřebné. Pokud má autor realizovat přesně požadavky typografa, musí proniknout nejen do problematiky TeXu, ale též do problematiky složitých maker LaTeXu, ve kterém původně svůj dokument napsal. Tím se z autora stává jakýsi polyhistor. Autor na počátku pořizování svého dokumentu určitě netušil, že na něj budou nakonec kladeny tak vysoké požadavky.

Předzpracované styly

Mezi zastánci LaTeXu se může rozšířit klamná vize, že všechny požadavky na sazbu, které kdy budou potřebovat, zvládnou výběrem vhodného stylu. Vůbec tedy není potřeba rozumět TeXu, stačí zjistit, jaký styl použít, napsat jeho název do hlavičky a máme vystaráno.

Chtěl bych zdůraznit, že základní styly v LaTeXu jsou po typografické stránce poměrně dost nevhodné. Především jde o volbu velikostí nadpisů a jejich umístění v sazbě. Rovněž existence pružných vertikálních výplňků je ve většině případů naprosto nepřípustná. Zbavit se jich je práce poměrně zdlouhavá a ve svém důsledku znamená přepsat si aspoň `\output` rutinu podle vlastní koncepce. Jestliže to ale uděláme a provedeme to dobře, jsme na nejlepší cestě vytvořit si vlastní makro, které nazveme například MujTeX nebo jinak. LaTeX jako takový už nepotřebujeme.

Svými základními styly dělá navíc LaTeX TeXu negativní reklamu. Dovedu si představit, že kupříkladu nějaký nakladatel projeví nezávazný zájem o TeX a udělá povrchní průzkum věci. Přitom jako první narazí na LaTeX a prefabrikované dokumenty jednoho typu. Může se stát, že průzkum na této úrovni skončí a nakladatel o TeX přestane mít další zájem.

Dalším problémem LaTeXu je snášenlivost stylů různých výrobců mezi sebou. Toto není možné elementárními prostředky TeXu nikdy uspokojivě vyřešit. Vždy budou existovat detaily, ve kterých se mohou některé styly vzájemně pohádat. Přitom výsledný efekt a případné chybové hlášení je velmi často dosti daleko od podstaty věci a náprava se těžko hledá. Odstranění takových chyb pak vyžaduje nejen perfektní znalost TeXbooku, ale též důkladnou znalost kódů jednotlivých stylů a kódu samotného LaTeXu. A především notnou dávku trpělivosti. Dospějete postupně k závěru, že na úrovni plainu byste se stejnými znalostmi a s podstatně menším úsilím vytvořili makro, které veškeré typografické požadavky řeší také. Důležitá je efektivita práce. Hledání tajemných chyb ve stylech a bloudění po kódech stylů a kódu vlastního makra LaTeXu mi nepřipadá příliš efektivní.

Vždy budou existovat požadavky, které nepůjdou vyřešit pouze zavedením dalšího stylu do LaTeXu. Pak se uživatel musí rozhodnout modifikovat styl nebo si problém vyřešit sám.

Složitost maker LaTeXu

Většina stylů musí být aspoň částečně schopna pracovat univerzálně v různých podmínkách. Mají proto ošetřeno plno věcí, které by autor makra „na míru" nikdy nedělal. Kód takových stylů se pak stává zbytečně složitý a nepřehledný. Příkladem takového nepřehledného a zbytečně složitého kódu (protože chce být univerzální) je samotné jádro LaTeXu. Navíc, jak plyne z nedostatečnosti makrojazyka TeXu, je požadavek na univerzálnost LaTeXu principiálně těžko dosažitelný.

Ve většině případů složitost LaTeXu ve svém dokumentu uživatelé v plné šíři nevyužijí. Určitě plno maker LaTeXu zůstává při sazbě dokumentu v nečinnosti. Ta makra, která něco činní, by mohla stejnou práci vykonávat efektivněji a jednodušeji. To snese přirovnání o dělové kouli proti mouše. Přitom si ta moucha pohodlně uletí, protože než se tam ta dělová koule přivalí, to chvíli trvá.

Zkuste se namátkou zeptat nějakého LaTeXisty, kterými cestami se ubírá napsané „`ř`" ve vstupním kódu dokumentu, než se dopracuje k výstupnímu písmenu „ř" v `dvi`. Bazírujte na naprosto přesném popisu expanze všech maker, které jsou při této činnosti použity. Většina lidí vám na tuto otázku nebude schopna dát odpověď. Ona totiž ta odpověď vůbec není jednoduchá. Dal jsem si tu práci a jeden půlden jsem v kódu LaTeXu tuto cestu hledal. Když jsem zjistil jakými zákoutími a skulinami tato cestička vede, zhrozil jsem se a začal jsem vážně pochybovat, jak to může vůbec stabilně fungovat. Věc je závislá na velkém množství dalších okolností, takže v žádné chvíli nemáme absolutní jistotu, zda „`ř`" povede vždy na „ř". Zákonitě to musí zlobit při použití primitivu `\uppercase`. Proto LaTeX tento primitiv dost nepřehledně předefinovává. Člověk si pak není jistý vůbec ničím.

Když použiji věci, o kterých vím do posledního šroubu, jak fungují v TeXu samotném, nebudu mít nikdy jistotu, co to udělá v TeXu zavirovaném LaTeXem. Tato nejistota mě od LaTeXu poměrně značně odrazuje.

K veřejnému použití jsem například nabídl makro na čárové kódy EAN. Občas mi nějaký uživatel napíše, že to nefunguje, a přitom to použil v LaTeXu. Odpovídám stručně: „Použijte (cs)plain, za LaTeX já neručím. Pokud to náhodou funguje v LaTeXu, považujte to za zázrak. Stejně si nebudete jistý, zda to za půl roku v tom LaTeXu ještě bude fungovat." Umíte si představit, jak mi povstaly vlasy hrůzou na hlavě, když mi jeden dobrák ze zahraničí oznámil, že z mého makra `ean.tex` vytvořil odrůdu `ean.cls`!

Přenositelnost dokumentů

Je potřeba odlišit dva typy přenosu dokumentů. V první řadě se budeme ptát po přenositelnosti textu dokumentu včetně struktury, ale bez přesných specifikací formátu sazby. Příjemce pak dokument formátuje podle svých požadavků. Výsledek může vypadat docela jinak, než jak dokument viděl na obrazovce autor. Takový způsob přenosu se používá například při sestavování sborníků. V druhém typu přenosu budeme chtít přenést zcela věrně nejen text dokumentu, ale i formát sazby. Požadavkem je, aby příjemce viděl na obrazovce totéž, co autor.

LaTeX je poměrně dobře koncipován pro první typ přenosu. Svým jazykem totiž deklaruje jednotný způsob značkování dokumentů, který by se mohl s trochou nadsázky nazvat mezinárodním standardem. Praxe ale ukazuje, že nic není tak růžové, jak se na první pohled zdá. Už jste někdy vytvářeli sborník příspěvků rozličných autorů? Uvedu příklad ze své zkušenosti: 50 % příspěvků přišlo ve Wordu či něčem podobném. Po troše konverzní práce je využitelný většinou jen holý text. 40 % příspěvků přišlo v různých verzích LaTeXu. Posledních 10 % přišlo v plainu nebo jako holý ASCII text. Nakonec se ukázalo, že s těmito příspěvky bylo nejméně práce.

LaTeX jako standardní jazyk pro značkování struktury dokumentu nám příliš nepomohl. Příspěvky v LaTeXu totiž byly „každý pes jiná ves“. To je dáno jednak různými verzemi současného LaTeXu a jednak neschopností většiny přispěvatelů oddělit strukturu dokumentu od sazby. V jejich příspěvcích se to hemžilo značkami jako `\vspace{5mm}`, `\clearpage`, `\\`, `{\Large Nadpis}` atd. Aby to k něčemu bylo, museli by všichni přispěvatelé přistoupit na společný způsob značkování. Nestačí říci, že je to LaTeX. Je potřeba vyjmenovat všechny povolené značky vymezující strukturu a pro jistotu uvést též všechny obraty, které jsou pro konečnou typografii sborníku nevhodné. Ukazuje se, že těch pár obratů, které zůstávají povoleny, lze zpracovat jednoduchým makrem. Ačkoli tyto obraty vypadají LaTeXovsky, není nakonec nutno použít LaTeX.

Pokud jde o přenositelnost dokumentů včetně formátu, je na tom LaTeX dost špatně. Především různé verze LaTeXu mohou vést k různým výsledkům. Rovněž koncepce stylů v tuto chvíli naráží na problémy. Většina uživatelů LaTeXu totiž implicitně předpokládá, že příjemce jeho dokumentu má povinnost mít instalovány všechny stylové soubory, které jsou v jeho dokumentu použity. Méně znalí uživatelé se navíc chovají k záhlaví svého dokumentu jako k nějakému tabu, které někde obkreslili a někdo poučenější jim do toho dopsal volání nějakých nových stylů. Výsledkem často je, že zaslaný dokument používá v záhlaví dvacet různých stylů, přičemž se nakonec ukáže, že jsou potřeba jen tři. Při obdržení takového dokumentu je proto nejprve nutné minimalizovat počet použitých stylů, dále zvážit, zda kvůli jednomu efektu v dokumentu má cenu toulat se po Internetu a shánět styly, které ještě v naší instalaci TeXu nemáme. Mnohdy je jednodušší chyby způsobené nezavedením stylu prostě přeskočit, vždyť ono to nějak dopadne. Pak přijde velké překvapení: ono to dopadlo většinou uspokojivě. Proč tam ten styl vlastně byl?

Na druhé straně uživatelé plainu dobře vědí, jaká makra jsou v zasílaném dokumentu potřeba. Vycházejí z toho, že příjemce bude mít standardní a v jednotlivých verzích skoro neměnný plain nebo csplain. Všechny definice, které rozšiřují uvedené formáty, pak přibalí naprosto vědomě do svého dokumentu. Dokumenty pak jsou bezproblémově přenositelné. Schopnost přesně rozlišovat mezi jádrem formátu a nadstavbou je především vlastností uživatelů plainu, protože jsou sami nuceni tomu pořádněji rozumět.

Uvedu jeden kuriózní příklad. Návrh stanov CSTUGu byl napsán v plainu a bylo jej tedy možné vystavit na síti tak jak je. Bez obav, že bude mít příjemce při zpracování problémy. Na druhé straně volební kandidátka byla napsána v LaTeXu a na síti jsme ji v podobě zdrojového textu neviděli. Víte proč? Autor tohoto dokumentu použil pro čtverečky na zakřížkování zvolených jmen speciální font. To ovšem v LaTeXu znamená, že je potřeba mít správnou verzi souboru `fd` pro tento font a plno dalších záležitostí. Jinak to nebude fungovat. Autor kandidátky mě proto explicitně požádal, abych z důvodů komplikací s přenositelností dokument ve zdrojové podobě nevystavoval. Já sám jsem půl hodiny pracoval na tom, než jsem byl schopen kandidátku bezproblémově vytisknout. Z Internetu jsem musel přitáhnout použité styly, `fd` toho fontu a font samotný. Kdybych tušil, že se jedná o přihlouplé čtverečky, měl bych pro ně makro postavené na primitivech `\hbox`, `\vbox`, `\vrule` a `\hrule` uděláno během pár vteřin. A dokument by byl přenositelný.

Nestálost verzí LaTeXu

To je kapitola sama pro sebe. Divím se, že pohyb verzí a zmatek, který kolem toho vznikl, neodradil uživatele od používání LaTeXu. Všichni uživatelé LaTeXu 2_ε jsou vlastně pokusnými králíky a jsou zúčastněni na velkém projektu, který směřuje snad někdy k LaTeXu 3.

Jako poměrně velký podraz vůči uživatelům považuji moment, kdy se pojmem LaTeX začalo nazývat něco, co se od původního a hojně používaného makra tohoto jména (LaTeX 2.09) poměrně zásadně liší. Jaký terminologický chaos vznikl po tomto dravém nástupu LaTeXu 2_ε, to jistě každý čtenář dobře ví. Když nyní řeknu slovo LaTeX, nikdo neví, co tím mám na mysli.

Každý půlrok přichází na svět nová verze LaTeXu 2_ε, která je s předchozí slučitelná jen na uživatelské úrovni. Pokud ovšem uživatel přesáhne svými znalostmi běžné příručky k LaTeXu a tyto znalosti použije a podepře je vlastnostmi konkrétního jádra LaTeXu, má za půl roku smůlu.

Mnozí z nás byli svědky neslučitelnosti různých verzí počeštění LaTeXu s různými verzemi jádra LaTeXu 2_ε. Také víme, že tyto problémy vycházely najevo na naprosto neočekávaných místech (například použití makra `\ldots`). To samo o sobě dokumentuje obtížnou kontrolu nad složitými makry LaTeXu a nesnadnou implementaci národní nadstavby k LaTeXu.

Půlroční změny verzí LaTeXu musí udržovat uživatele i administrátory systémů v neustálé pozornosti. To je úmorné a vyčerpávající. Přiznávám se, že já, jako administrátor systému pro naši katedru, tuto rychlost pohybu nestačím sledovat a docela mě unavuje.

Jednorázová investice do vzdělání

Naučit se rozumět vestavěným algoritmům TeXu a správnému použití primitivů nebo maker plainu je činnost, kterou stačí udělat jednou pro vždy. Naopak, v případě stylů LaTeXu člověk musí být neustále ve střehu a musí se postupně stát pochodujícím slovníkem na styly, aby dokázal ke konkrétnímu účelu vybrat a správně použít ten správný styl té jedině správné verze. Přiznávám, že to přesahuje kapacitu mého mozkového potenciálu, a proto zůstávám při zemi: pouze se znalostmi algoritmů TeXu, jeho primitivů a maker plainu. Tyto věci nejsou v pohybu, takže se nemusím učit každý rok něco nového.

Závěr

TEX je koncipován jako otevřený systém. Kdo má chuť a sílu si osvojit jeho detaily, má v rukou mocný nástroj na vytváření nadstaveb podle svých představ. Není podle mého názoru férové vnucovat uživatelům „jedinou správnou nadstavbu" a v příručkách tvrdit, že při jejím použití nebudou muset vědět, co to je TEX. LATEX se tím snaží zastřešit otevřený základ uzavřenou nadstavbou, čímž neguje výhody základu. Protože je toto zastřešení navíc děravé z důvodů nedostatečnosti samotného základu, utrpí tím nakonec oba. Jednak základ a jednak nadstavba. A to je škoda.

Domnívám se, že kdyby se více úsilí věnovalo výkladům a vysvětlování principů samotného TEXu, bylo by to jen ku prospěchu věci. Udržovat neustále přehled o vlastnostech kódu LATEXu a všech jeho současných stylů a verzí je podle mého názoru daleko náročnější, než se jednou pro vždy seznámit se samotným TEXem.