Final exhortation: Go forth now and create masterpieces of the publishing art!
Donald Ervin Knuth: *The TEXbook.*

Kapitola osmá: Závěrečné shrnutí

Petr Olšák

Touto kapitolou bych chtěl uzavřít své povídání o TEXu. Vezmeme-li v úvahu, že seriál vycházel v časopise, který se nespecializuje na problematiku sazby textů na počítači, musíme přiznat, že toho bylo řečeno až příliš mnoho.

V následujícím textu popíšeme další myšlenkové směry týkající se sazby pomocí programu TEX a ukážeme některé problémy, které s tím souvisejí. Jedná se o program METAFONT, dále se zmíníme o virtuálních fontech, o zařazování obrázků do TEXu a nakonec o často vytýkané otázce uživatelského rozhraní. O každé této problematice by se dala napsat minimálně jedna celá kapitola, ale jak už jsem se zmínil v úvodu této kapitoly, myslím, že toho bylo dost. Kdo bude mít hlubší zájem o TEX a elektronickou sazbu, možná se dočká knížky na toto téma, kterou se chystám napsat asi v průběhu tohoto roku.

Nejprve bych se chtěl všem čtenářům omluvit. Záměrně a vědomě jsem se totiž dopustil jednoho typografického prohřešku. Text jsem natáhnul na celou šířku zrcadla sazby. Tím se staly řádky neúnosně dlouhé, což snižuje čitelnost. Oko čtenáře musí pracně hledat začátek dalšího řádku. V oborové normě pro sazbu textů sice není definovaná maximální délka řádku, avšak takto dlouhé řádky, jaké jsem použil ve svém seriálu, jsou skutečně na hranici čitelnosti. Správně by se pro takovou šířku sazby mělo použít větší písmo. Tím by se ale na jednu stránku vešlo méně informací. Časopisy samozřejmě šetří plochou papíru a chtějí na ni umístit co nejvíce textu. Proto je v novinách a časopisech použito častěji drobnější písmo a plocha pro sazbu je rozdělena do sloupců.

Dlouho jsem zvažoval, zda se ve svém seriálu také nepřikloním k vícesloupcové sazbě. Kdybych navrhoval vzhled celého časopisu, byla by taková věc víceméně nutností. Já jsem ale navrhoval vzhled jediného článku a bylo mi dovoleno vysázet jej jakkoli, tj. bez návaznosti na vzhled ostatních stránek časopisu. Ukázky přes celou šířku stránky umístěné přesně tam, kde je potřebuji, a vysoká čitelnost Computer Modern fontů nakonec rozhodly pro ponechání dlouhých řádků. Zkuste si představit, že by byly vysázeny texty s takto dlouhými řádky v písmu Times (toto písmo je použito ve většině časopisů). Myslím si, že text by byl absolutně nečitelný. Toto písmo nedrží linku, protože bylo navrženo pro hrubou nekvalitní novinovou sazbu, prováděnou tehdy ještě klasickou technikou knihtisku v deníku Times.

V dalším textu jsem převzal šířku sloupců používaných v tomto časopise, aby čtenář mohl důkladně porovnat čitelnost Computer Modern fontů sázených programem TEX s ostatním obsahem časopisu, kde je nejčastěji použit font Times a sazba provedena programem Ventura Publisher.

O programu METAFONT.

Kromě programu TEX vyvinul autor ještě další program, který umožňuje návrh písem. Vstupem pro METAFONT je textový soubor (soubory), který vektorově popisuje tahy jednotlivých písem. Myšlenka je podobná, jako při PostScriptovém popisu fontů, ovšem jazyk je jiný, možná poněkud komplikovanější, ale na druhé straně je určitě silnějším nástrojem pro tvorbu písem.

Program je podrobně zdokumentován v třetím a čtvrtém díle Knuthovy monografie *Computers & Typesetting.* Třetí díl (The METAFONTbook) je manuál k programu a definice vstupního jazyka a čtvrtý díl (METAFONT — the program) obsahuje podrobně zdokumentovaný zdrojový text programu METAFONT. Pozorný čtenář tohoto seriálu jistě ví, že první a druhý díl monografie se analogickým způsobem věnuje programu TEX.

Ve zdrojových textech Computer Modern fontů najdeme jednu zajímavost, která myslím nemá v PostScriptových popisech fontů obdobu. Jedná se o to, že většina těchto fontů (např. antikva, bezpatkové písmo, strojopis) je generována jediným vstupním souborem, který popisuje tahy jednotlivých písmen pro *všechny* řezy naráz. Je to uděláno tak, že před zahájením vlastního generování písmen METAFONT načte tzv. parametrický soubor, v němž jsou napsány hodnoty několika desítek parametrů. Například tloušťka silného i vlasového tahu, parametry pro tvar a velikost serifů (patek), sklon písma, korekce optických klamů atd. S těmito parametry se pak dále v souborech popisujících tahy jednotlivých písmen pracuje jako s proměnnými.

Kdo si koupí pátý díl Knuthovy monografie *Computer & Typesetting* (Computer Modern Typefaces), může snadno modifikovat tyto fonty podle svých vlastních představ. Jsou tam totiž podrobně popsány nejen všechny parametry z parametrických souborů, ale i soubory popisující tahy písem. Komu se například jeví toto písmo příliš světlé, může pozměnit jediný parametr a vygenerovat si font s menší světlostí. Autor nabízí své fonty k volnému použití. Jediné omezení, na kterém autor trvá, je otázka názvu fontů. Modifikované fonty se nesmí jmenovat Computer Modern.

V METAFONTu je možno vygenerovat jakousi bázi — skupinu předzpracovaných maker, kterou pak při opakovaném volání METAFONTu nemusí program číst znova v textové podobě (podobně jako v TEXu jsou formáty). Např. základní báze *plain* obsahuje algoritmy, které umožňují pomocí několika málo parametrů zadat kvalitu výstupního zařízení (rozlišovací schopnost, jak moc se rozpíjí inkoust apod.) a skupiny těchto parametrů shrnout pod jedno slovo. Když se pak napíše na příkazový řádek mezi parametry např. `mode=hplaser`, METAFONT s bází plain bude vědět, že má generovat fonty pro laserovou tiskárnu.

Výstupy z METAFONTu jsou dva. Jednak bitová mapa fontu a jednak soubor `tfm` s metrickými údaji pro TEX. Protože bitové mapy jsou potřeba v různých velikostech a zabírají velké místo na disku (na rozdíl od souborů `tfm`), lze v procesu zpracování dokumentu volat METAFONT až v okamžiku, kdy je připravena definitivní sazba. To má smysl zvláště pro vysoká rozlišení. Většina programů, které pracují s TEXovským výstupem a s bitmapami fontů, umí vyvolat METAFONT podle potřeby. Bohužel rychlost digitalizace fontů pomocí METAFONTu je ve srovnání s PostScriptem nesrovnatelně pomalejší. Na druhé straně si ale můžete vytvořit bitové mapy jednou pro vždy a digitalizaci neopakovat znovu s každým novým dokumentem, jako v PostScriptu. Také ušetříte za strojový čas osvitové jednotky, protože si fonty připravíte v klidu doma na svém počítači METAFONTem a nezaměstnáváte procesor osvitové jednotky požadavkem na digitalizaci PostScriptových písem. PostScriptový výstup s takto připravenými už digitalizovanými fonty je samozřejmě v TEXu možný například pomocí konvertoru `dvips`.

Chtěl bych stručně provést srovnání METAFONTu a PostScriptu. V mnohém se obě myšlenky podobají. Například reprezentace tahů je v obou formátech řešena pomocí souřadnic jistých významných bodů bezièrových křivek. Na první pohled bohatší vyjadřovací repertoár pro tahy u PostScriptu (např. i části kružnic) je jen zdánlivý. Programátor v METAFONTu si takové věci, které nejsou v základním vybavení programu, může naprogramovat do procedur pomocí makrojazyka, který je v METAFONTu mnohem silnější a (bohužel) komplikovanější, než v PostScriptu.

PostScript přebírá starost o rastry a odstíny šedi (a ve verzi Level 2 dokonce o barvy). Programátor se o tyto věci nestará a tím pádem ani nemá absolutní jistotu, jak to na různých rozlišeních dopadne. V METAFONTu si takové věci musí na úrovni pixelů udělat programátor sám.

Jazyk METAFONTu není orientován pro popis celé stránky, ale jen pro návrh písma. Proto například v METAFONTu nehledejte možnost importu obrázků v bitmapových formátech. Také některé efekty související s manipulací kresby na celé stránce nelze v METAFONTu provést.

METAFONT nepracuje s jediným rastrem (image v PostScriptu), ale může jich mít libovolně mnoho a pracuje se s nimi jako s proměnnými typu `picture`. Tyto rastry lze pak podle různých pravidel kombinovat, ale mají omezení velikosti: od počátku souřadnic se nakreslí maximálně 4096 pixelů na všechny čtyři strany.

Typů proměnných je v METAFONTu daleko více než v PostScriptu. Kromě numerických proměnných máme proměnné typu „dvojice“ (`pair`), „cesta“ (`path`), „pero“ (`pen`), „transformace“ (`transform`) a další. Také lze deklarovat pole proměnných s libovolným počtem prvků. Zmíněné typy napovídají, že v METAFONTu naleznete některé zajímavé vestavěné operace. Např. lze definovat tvar pera pomocí proměnné typu `path`, transformovat jej lineární transformací podle hodnoty proměnné typu `transform` a uložit do proměnné typu `pen`. Aktuální pero může mít libovolný konvexní tvar. Je-li kruhové, pak METAFONT simuluje práci se známými „redis pery“. Je-li tvaru čtverce, pak jsou konce tahů hranaté. Můžete nasadit i „seříznuté pero“, kterým lze dosáhnout zajímavé kaligrafické efekty. Naproti tomu v PostScriptu lze definovat tvary písma pouze obrysem. (Poznamenejme, že to je nejběžnější způsob při návrhu písma a že i všechny Computer Modern fonty jsou definovány obrysem a nevyužívají vlastnost měnit tvar pera.)

O METAFONTu rád říkám, že to je jedna velká soustava lineárních rovnic. Do proměnných se totiž dají ukládat hodnoty jiných proměnných, které jsou v daném okamžiku ještě neznámé. Pokud jsou všechny takové vztahy lineární a těsně před použitím proměnné při digitalizaci jsou všechny vstupní proměnné známy, METAFONT nemá problémy. S tím souvisí i možnost velmi elegantně kreslit například různé náčrtky pro konstruktivní úlohy ze střední školy typu: je dán trojúhelník, najděte kružnici opsanou. Ve vstupním souboru zadáme numericky pouze souřadnice trojúhelníka a pak popíšeme celou konstrukci jazykem: „najdu střed úsečky, sestrojím kolmici, najdu průsečík, sestrojím kružnici apod.“ Popis takového náčrtku bude fungovat pro různá vstupní zadání.

METAFONT má zabudovaný důmyslný algoritmus na dopočítání umístění kontrolních bodů bezièrových křivek, zvláště pokud jsou napojovány za sebou. Je to výsledek jedné dizertace, kterou Knuth cituje ve své knize o METAFONTu. Návrhář fontu nemusí vypisovat kontrolní body, ale místo nich třeba požadovanou směrnici, nebo třeba nic — v takovém případě se dopočítají kontrolní body tak, aby veškerá napojení byla hladká. Tuto vlastnost jsem v PostScriptu nenašel.

O virtuálních fontech.

TEX dokáže pracovat s jakýmikoli fonty, které mohou mít libovolnou datovou reprezentaci, například i takovou, která ještě nebyla vůbec navržena a objeví se až v budoucnu. Proč? Vzpomeňme si, že samotný program TEX pracuje pouze s rozměrovými údaji fontů a že interpretaci tvarů písmen mají na starosti až další programy, které zpracovávají výstup TEXu do čitelné podoby.

Stačí proto mít metrické údaje fontu. Konverzní programy např. z PostScriptových fontů `afm` (Adobe font metric) do formátu `tfm` (TEX font metric) samozřejmě existují. Také existují konverzní programy z nečitelného `tfm` do čitelného tvaru a zpět. Odborník tedy může provést dodatečné korekce a třeba i zařídit jiné kódování fontu.

Jak si takto vytvořený výsledek sazby TEXu prohlédneme nebo vytiskneme? Existuje konvence, že všechny programy, které pracují s výstupem TEXu a s bitmapami fontů musí umět při nenalezení fontu v klasickém formátu hledat v tzv. virtuálních popisech fontu. Tyto virtuální popisy mohou odborníci editovat (existuje konverze do čitelné podoby a zpět). V těchto popisech se praví, jakým způsobem se má reprezentovat tvar každého písmene.

Například pro tvar písmene lze použít náhradní font, který ve tvaru bitmapy v počítači existuje. Přitom font může být zcela jinak kódovaný. Nebo se písmeno poskládá z jednotlivých částí, které se najdou v jiných fontech (např. pro akcentovaná písmena). Nebo se pro reprezentaci tvaru použije jazyk výstupního zařízení, například sekvence příkazů PostScriptu, máme-li výstup do PostScriptu. Nebo se to vše kombinuje dohromady. Například háček můžeme vyrobit „PostScriptově" a písmeno pod ním necháme „bitmapové".

Je možné použít virtuální popis fontů třeba při práci s drahým a v domácích podmínkách nedostupným fontem. Autor myšlenky, Donald Knuth, uvádí tento příklad: Má provést sazbu za pomocí drahého fontu, který je dostupný pouze v osvitové jednotce. Má k němu metrické údaje. Tyto údaje si zkonvertuje do `tfm` a provede sazbu TEXem. Výsledek si prohlédne na obrazovce pomocí nějakých náhradních bitmap (prohlížeč totiž umí číst virtuální popisy, kde je provedeno přesměrování na dostupné bitové mapy). Na své osobní laserové tiskárně si pak udělá kontrolní tisky za použití třeba jiných fontů (zase pomocí přesměrování metodou virtuálního fontu) a nakonec ostrý tisk proběhne až v osvitové jednotce.

Pro nás je důležité, že metoda virtuálních fontů otevírá TEXu možnost pracovat například s PostScriptovými fonty, kterých je na trhu nepřeberné množství. Přitom je dobré upozornit na to, že PostScript je jen jedna z alternativ pro výběr fontů a fonty různých datových reprezentací se dají míchat.

O obrázcích

Myslím si, že nyní už čtenáře nepřekvapím, když budu tvrdit, že do TEXu lze zařadit jakékoli obrázky v libovolném grafickém formátu, který ještě třeba ani nebyl navržen a spatří světlo světa v budoucnu.

Myšlenka je velmi prostá. TEX samotný se stará o sazbu, takže jeho úkolem je pouze vynechat v textu odpovídající místo pro obrázek (případně číslovat popisy obrázků apod.) a dále předat do výsledné sazby informaci o tom, jakým způsobem bude obrázek reprezentován.

Tato informace se předá pomocí příkazu `\special`, za nímž vše, co je ve složených závorkách, se doslova přepíše do výstupního souboru `dvi`. Jak si s touto informací poradí příslušný tiskový program, to už je jeho věc a nikoli starost TEXu.

Držme se při zemi a zkusme sumarizovat, co se tedy dá v současné době udělat pomocí nejčastěji dostupné instalace emTEXu s tiskovými ovladači pana Mattese. Tyto programy zpracují bitmapové formáty ve tvaru `pcx` a `msp`. Bohužel, nikoli v barvě. Máme-li obrázek v jiném formátu nebo negativní, musíme provést konverzi některým z dostupných programů.

Děláme-li výstup do PostScriptu pomocí programu `dvips`, můžeme samozřejmě zahrnout PostScriptové obrázky v „encapsulated PostScript form". Takový obrázek je „natahovací", tj. může mít jakoukoli velikost (digitalizuje se až ve výstupním zařízení). Ve standardní výbavě nalezneme TEXovské makro, které prostřednictvím příkazu `\special` předává ovladači požadavek na takové zvětšení, jaké odpovídá vynechanému místu. Umí-li finální tiskové zařízení pracovat s barvou, máme samozřejmě obrázky barevné.

Často pro obrázek postačí bitmapový formát fontu. Obrázek je reprezentován jako jedno písmeno, nebo více písmen na sebe navazujících. Pak s tím TEX i tiskové programy pracují jako s fonty. Velice pěkné „pérovky" (tj. náčrtky, matematické obrázky, grafy apod.) lze vytvořit přímo programem METAFONT. Druhá možnost je konvertovat bitmapové obrázky do formátu bitových map fontů pomocí programu `bm2font`. Zde lze využít možnosti jistých korekcí a zvýšit např. kontrast nebo jiné parametry.

Právě proto, že zařazování obrázků není v TEXu koncepčně a jednotně řešeno, mnozí sazeči se raději přikloní k jinému programu. Přitom ale nikdo netvrdí, že by obrázky nešly do TEXu v libovolném formátu zařadit. Chce to jen „sehnat" příslušný doplňkový software.

O uživatelském rozhraní

Jedním z nejvíce vytýkaných nedostatků TEXu je skutečnost, že autor předpokládal, že uživatel jeho programu bude vybaven určitou inteligencí. Tento předpoklad se při současném trendu „intuitivního" zacházení s počítačem jeví jako podstatný nedostatek. Ovšem uvědomme si, že program vznikl v době, kdy počítače byly pouze v rukou expertů a navíc s programem pro počítačovou typografii by měl *vždy* zacházet odborník a ne každý, kdo si chce s počítačem jen tak nezávazně pohrát.

Dnešní trend intuitivního zacházení s počítačem má samozřejmě i mnoho odpůrců, jejichž argumenty, zvláště v souvislosti s profesními programy typu počítačová typografie, nelze přehlížet. Je třeba si položit otázku, *komu* jsou takové programy určeny. Má jej užívat odborník na typografii nebo výtvarník, přičemž „počítačové vzdělání" se u něj nepředpokládá? Mám dojem, že přesně takto si autoři různých komerčních DTP programů vykreslili uživatele svých produktů. Uvědomme si ale, že tento předpoklad je poněkud mylný. Většina výtvarníků o počítač nikdy nezavadí, i kdybyste je k tomu chtěli donutit násilím a obrazovka se tvářila jakkoli vlídně. Mezi uživateli programů pro počítačovou sazbu tedy většinou najdeme lidi, kteří jsou častěji profesionály v oboru výpočetní techniky a jen tak trochu výtvarníky. Ideálními uživateli by ovšem měli být lidé znalí jak počítačů, tak typografie.

Mám dojem, že kdyby programy, které vycházejí z intuitivního způsobu ovládání, byly živé bytosti, asi by si pomyslely: Vážený uživateli, jsem rád, že sis mě koupil. Nepředpokládám od tebe vysokou inteligenci a proto nemusíš ani umět číst. Nabízím ti na obrazovce nejrůznější obrázky a ty si můžeš myší vybrat, co budeš dělat. Vím, klávesnice je pro tebe příliš komplikovaný vynález, má mnoho neznámých tlačítek a tak ji raději od sebe odsuň. Je tu přece myš. Stačí, když budeš používat jen jednu ruku a na ní pouze dva prsty. Hlavu samozřejmě používat nemusíš, to obstarám za tebe. Kdyby tě náhodou zajímalo, jak jsem naprogramovaný či jak přesně funguji, budeš mít smůlu. V manuálu to nenajdeš. Tam se akorát dozvíš, jaké tlačítko myši je nutné v té které situaci zmáčknout. Kdybys přece jen se mnou začal experimentovat tak, abys zjistil, jak jsem uvnitř udělaný, budiž ti taková činnost co nejvíce znesnadněna či přímo znemožněna. Že výsledek mé práce není vždy zcela dokonalý a neodpovídá profesně tomu, k čemu jsem byl určen? Vždyť vidíš, kolik práce mi dalo, abych ti usnadnil práci, tak takové „maličkosti“ snad omluvíš.

Tento přístup celkem názorně dokumentuje jedna zajímavá recenze na program Ventura Publisher. Tato recenze vyšla v Softwarových novinách minulý rok a byla následována diskusí, zda takový článek má či nemá být prezentován jako recenze. Přitom nejdůrazněji se tam diskutovalo o tom, zda příslušné tlačítko myši je nutné zmáčknout jednou nebo dvakrát.

Myslím si, že si žádný programátor nedovede představit, že by vytvářel program pomocí nějakého uživatelského rozhraní s nabídkami typu: teď se deklaruje proměnná, teď se udělá cyklus apod. Přitom fyzicky by zdrojový text programu nebyl nikde napsaný — program by byl rovnou ve spustitelném tvaru a byl by důsledkem zpětně nezjistitelného pohybu myšího kurzoru po obrazovce. Ačkoli by rozhraní nabízelo veliký komfort interaktivního ladění, nemít program fyzicky napsaný v nějakém jazyce je pro programátora nemyslitelné. Význam rčení „co je psáno, to je dáno“ je zde více než patrný. A totéž platí pro sazbu. Takže TEX svou filosofií nutně vyhovuje programátorsky založeným lidem.

Jediný problém masového použití TEXu je samozřejmě v tom, že neplatí tvrzení, co Čech to programátor a typograf. Zde má TEXovský software neustále co dohánět. Je třeba vytvářet filtry z textových procesorů do TEXu. Ať si autor napíše text v libovolném editoru, jakým je T602, Word-*cokoli*, AmiPro apod. Budou-li existovat filtry z těchto programů do zdrojového textu TEXu, může nakonec odborník zpracovat autorův text kvalitní sazbou.

Poznamenávám, že některé z těchto filtrů existují, ovšem žádný z nich nemůže být dokonalý. Při této „filtraci“ budou vždy existovat nějaké problémy. To určitě znají i uživatelé DTP programů. Například, pokud autor převezme dělení slov do svých rukou a bude na konci *svých* řádků „ručně a natvrdo“ psát rozdělovníky, pak je s takovým textem dosti obtížná práce. Toho můžeme být svědky v mnoha novinových článcích, kde občas nějaké slovo má tako-vý tvar. Nebo pokud si někdo pohraje s tabulkami v programu T602 a odpočítá si mezery, aby mu vše pěkně lícovalo, pak se takto vytvořená tabulka jen těžko převádí do datové reprezentace popisující sazbu. A to už vůbec nemluvím o matematických formulích.

Chtěl jsem naznačit, že by se při přípravě dokumentu měla vytvořit rozumná dělba práce. Autor text napíše v textovém procesoru a lidé, kteří se profesionálně zabývají sazbou, text vysázejí. Zde pak jistá vyšší komplikovanost TEXu a neexistence uživatelského rozhraní nevadí. Bohužel existuje mnoho profesionálů, kteří se živí sazbou, a odmítají se TEX učit, protože je příliš komplikovaný. To je ovšem škoda. Ať potom nenaříkají, že to či ono v jejich sázecím systému nejde realizovat. Takové nářky jsem slyšel pouze od uživatelů DTP programů ale nikdy ne od uživatelů TEXu. Pokud v TEXu něco nejde, pak to není proto, že to program neumí, ale proto, že uživatel ještě nepřišel na to, jak problém vyřešit.

O úpadku písma

V tomto seriálu jsem chtěl mimo jiné ukázat, jak by měl vypadat správný přístup k řešení návrhu programu pro počítačovou sazbu. Myslím si, že Donald Knuth nám může být vzorem. Než se pustil do programování, nejdříve podrobil důkladnému studiu tradice oboru, které zde byly po staletí vytvářeny. Snažil se co nejvíce těmto tradicím přiblížit. Zdá se mi, že tvůrci podobných programů si často řeknou, že jim stačí umět programovat. Vytvářejí přeci něco nového, tak proč se dívat na nějaké tradice. Tento „hurá-přístup“ vede k celkovému úpadku písma. Texty na počítačích lze totiž velice snadno kopírovat a v relativně velkém množství tisknout. Kdyby o těchto produktech věděli staří mistři v oboru typografie a písmařství, asi by se obraceli v hrobě. Písmo neslouží jen pro přenos informace, ale je též nositelem kultury, podobně jako třeba stavební slohy. Toho si byl Knuth velmi dobře vědom a myslím si, že je to vidět nejen v kvalitě programu, ale i písma, které vytvořil (inspiroval se ručně řezaným secesním písmem Modern Monotype 8a).

Pro ukázku toho, jak negativně mohou počítače ovlivnit lidskou kulturu, není třeba chodit daleko. Vezměme si u nás nejpopulárnější textový procesor. Samozřejmě jej nechci srovnávat s programy typu DTP. Když se ale v současné době podívám na každou druhou tiskovinu (úřední či jinou), je vytištěna v tomto editoru. Texty v tomto programu spáchané na mě křičí z nejrůznějších nástěnek, vývěsek a amatérsky vyrobených plakátků. To písmo je mírně řečeno strašné. Nepodobá se ani písmu psacího stroje, z něhož by měl program srovnatelných možností vycházet. Oko výtvarníka pláče a lidé si na to zvykají. Je to tisíckrát větší úpadek písma jako odrazu lidské kultury, než bylo na přelomu minulého století propůjčení sazečského řemesla merkantilním účelům (noviny, formuláře a reklamy). Pro autory tohoto procesoru budiž omluvou, že v okamžiku, kdy program vytvářeli, vůbec netušili, kam až to povede. Kdyby si na svůj projekt pozvali typografa, patrně by jim řekl, že jejich program může pouze nahradit psací stroj a není vhodné pro něj navrhovat další písma. Například písmo dvakrát tak vyšší než širší (nebo obráceně) je, s prominutím, zvrhlost. Stejně tak zarovnání textu na pravý i levý okraj pomocí manipulace s mezerami *pevné* délky je nehoráznost. Bohužel, trh si takové věci přeje a program s omezenými možnostmi by se těžko prodával.

Je vidět, že po světě chodí stále málo profesionálních typografů, kteří by byli ochotni zabývat se počítači. A to je velká škoda.