Příloha A

Úvod do univerzálního vyznačování

(Tato příloha netvoří nedílnou část této mezinárodní normy)

A.1 Proces vyznačování

Systémy zpracování textu obvykle vyžadují, aby do základního textu, který má být zpracován, byly zahrnuty dodatečné informace. Tyto dodatečné informace nazývané „vyznačení" slouží ke dvěma účelům:

a) k separaci logických prvků dokumentu a

b) k specifikování funkcí zpracování, které jsou s těmito prvky prováděny.

V publikačních systémech, jejichž formátování může být velmi složité, provádí vyznačování obvykle sám uživatel, který byl pro tuto činnost speciálně vyškolen. V textových procesorech mají formátovače textu obvykle menší význam, takže vyznačování (v omezenějším rozsahu) může být bez vědomého úsilí generováno uživatelem. Protože se však objevily cenově dostupné tiskárny vyšší třídy, budou muset kancelářská pracoviště převzít větší podíl funkcí publikačního systému a „bezděčné" vyznačování bude možné pouze v části zpracování kancelářského textu.

Je proto důležité si uvědomit, jak vyznačuje dokument uživatel systému vyšší třídy. Existují tři odlišné stupně, i když nemusí být jako takové uživatelem vnímány.

a) Nejprve analyzuje strukturu informace a jiné atributy dokumentu; tj. identifikuje každý smysluplný oddělený prvek a charakterizuje ho jako odstavec, nadpis, uspořádaný seznam, poznámku pod čarou nebo nějaký jiný typ prvku.

b) Potom ze své paměti nebo z příručky zvolí instrukce zpracování („ovladače"), které budou vytvářet formát požadovaný pro tento typ prvku.

c) Nakonec vloží vybrané ovladače do textu.

Nyní si ukážeme, jak vypadá začátek této stránky vyznačené ovladači v typickém formátovacím jazyku pro zpracování textu:

```
.SK 1
Systémy zpracování textu obvykle vyžadují, aby do základního textu, který má být zpracován, byly zahrnuty dodatečné informace. Tyto dodatečné informace nazývané „vyznačení" slouží ke dvěma účelům:
.TB 4
.OF 4
.SK 1
1. k separaci logických prvků dokumentu a
.OF 4
.SK 1
2. ke specifikování funkcí zpracování, které jsou s těmito prvky prováděny.
.OF 0
.SK 1
```

Ovladače .SK, .TB a .OF příslušně vyvolají posunutí o řádku, nastavení tabulační zarážky a způsob formátování odsazením nebo „visícím odsazením" („hanging indent"). Znaménko negace (-) v každé položce seznamu představuje kód tabulátoru, který by jinak nebyl vidět.)

Procedurální vyznačování, jako je toto, má ovšem řadu nevýhod. V prvé řadě se obvykle ztratí informace o atributech dokumentu. Jestliže se například uživatel rozhodne při formátování centrovat jak jednotlivé nadpisy, tak i názvy obrázků, nebude ovladač „centrování" udávat, zda text, na nějž působí, je nadpisem nebo názvem obrázku. Jestliže tedy uživatel chce použít dokument v aplikaci pro vyhledávání informací, nebudou vyhledávací programy schopné odlišit nadpisy, které mohou být z hlediska informačního obsahu velmi významné, od jiného textu který byl vycentrován.

Upraveno z Goldfarb, Charles F., „A Generalized Approach to Document Markup", *SIGPLAN Notices, June 1981*, se souhlasem autora a sdružení Association for Computing Machinery.

Procedurální vyznačování není kromě toho pružné. Jestliže se uživatel rozhodne změnit styl svého dokumentu (například proto, že používá jiné výstupní zařízení), bude muset proces vyznačování zopakovat, aby respektoval tyto změny. To mu například umožní netisknout kopie konceptu s dvojitým řádkováním na levné počítačové řádkové tiskárně, ale získat vysoce kvalitní konečný výtisk na drahém fotografickém kopírovacím stroji. A jestliže si přeje vyhledat konkurenční nabídky pro sazbu svého dokumentu, bude omezen na takové dodavatele, kteří používají identický systém zpracování textu, pokud nechce vynaložit náklady na opakování procesu vyznačování.

Kromě toho může být vyznačování řídicími slovy časově náročné a náchylné k chybám a vyžaduje vysokou kvalifikaci obsluhy zvláště v případech, kdy se požadují složité typografické produkty. Je tomu tak (i když v menší míře) i v případě, že systém připouští definované procedury („makra"), protože těmito procedurami musí být doplněn uživatelský slovník základních ovladačů. Například elegantní a účinný systém TeX (2), který se široce používá pro matematickou sazbu, obsahuje ve své základní implementaci na 300 takových základních ovladačů a maker.

Těmto nevýhodám procedurálního vyznačování se lze vyhnout, použije-li se schéma vyznačování, které vytvořili C. F. Goldfarb, E. J. Mosher a R. A. Lorie (3,4). Nazývá se „univerzální vyznačování", protože neomezuje dokumenty na jedinou aplikace a na jediný způsob formátování nebo systém zpracování. Univerzální vyznačování je založeno na dvou nových postulátech:

a) Vyznačování by mělo popisovat strukturu dokumentu a jiné atributy, a nikoliv specifikovat procesy, které se mají provést, neboť popisné vyznačování vyžaduje, aby bylo zapsáno pouze jednou, a bude postačovat pro všechna budoucí zpracování.

b) Vyznačování by mělo být přesné, aby bylo možno rovněž požívat techniky vhodné pro zpracování přesně definovaných objektů, jako jsou programy a datové báze.

Tyto postuláty budou rozvíjeny intuitivně při vyšetřování vlastností tohoto typu vyznačování.

A.2 Popisné vyznačování

Při univerzálním vyznačováním se proces vyznačování zastaví na prvním kroku: uživatel zjistí místo každého signifikantního prvku dokumentu a vyznačí jej mnemotechnickým názvem („generický identifikátor"), který jej podle jeho představy nejlépe charakterizuje. Systém zpracování asociuje vyznačení ke instrukcím zpracování způsobem, který krátce popíšeme.

Notace univerzálního vyznačování, známá jako Standardní univerzální vyznačovací jazyk (SGML), byla vytvořena pracovní skupinou Mezinárodní organizace pro normalizaci (ISO). Použijeme-li vyznačení podle SGML, bude začátek této přílohy vypadat takto:

<p>
Systémy zpracování textu obvykle vyžadují, aby do základního textu, který má být zpracován, byly zahrnuty dodatečné informace. Tyto dodatečné informace nazývané <q>vyznačení</q> slouží ke dvěma účelům:

k separaci logických prvků dokumentu a
ke specifikování funkcí zpracování, které jsou s těmito prvky prováděny.

Každý generický identifikátor (GI) je oddělen symbolem menší než (<), jestliže je na počátku prvku, nebo symbolem menší než následovaným lomítkem (</), jestliže je na konci. Symbol větší než (>) odděluje GI od jakéhokoliv následujícího textu.[1)] Mnemotechnické zkratky P, Q, OL a LI příslušně platí pro odstavec typů prvků (paragraph), uvozování (quotation), uspořádaný seznam (ordered list) a položku seznamu (list item). Kombinace GI a jeho oddělovačů se nazývá „počátečním příznakem" („start-tag") nebo „koncovým příznakem" („end-tag") podle toho, identifikuje-li počátek nebo konec prvku.

Tento příklad má několik zajímavých vlastností:

a) V textu nejsou uvozovky; generuje je zpracování určené pro prvek v uvozovkách, a jestliže to dovolí výstupní zařízení, bude rozlišovat mezi počátečními a koncovými uvozovkami.

b) Čárka, která následuje za prvkem v uvozovkách, není ve skutečnosti jeho součástí. V tomto případě byla během formátování ponechána mimo uvozovky, ale mohla by právě tak snadno být vložena dovnitř, kdyby bylo upřednostněn tento způsob.

c) Neudávají se pořadová čísla položek uspořádaného seznamu; jsou generována během formátování.

[1)] Ve skutečnosti jsou právě tyto znaky předurčené. SGML připouští výběr znaků oddělovače.

Jinými slovy řečeno, zdrojový text obsahuje pouze informaci. Znaky, jejichž jedinou funkcí je zdokonalit prezentaci, se generují během zpracování.

Jestliže, jak je postulováno, popisné vyznačování, jako je toto, vystačí k veškerému zpracování, musí následovat, že zpracování dokumentu je funkcí jeho atributů. Způsob, jakým je text sestaven, nabízí pro tento předpoklad intuitivní podporu. Takové techniky, jako je zahajování kapitol na nové stránce, psaní zdůrazněných vět kurzívou a seznamy odsazení se používají k tomu, aby zdůrazněním strukturálních atributů dokumentu a jeho prvků napomohly čtenářovu pochopení.

Z této analýzy může být vytvořen třístupňový model zpracování dokumentu:

a) Rozpoznání: rozpoznává se atribut dokumentu, např. prvek s generickým identifikátorem „poznámka pod čarou" („footnote").

b) Mapování: atribut je asociován s funkcí zpracování. Poznámka pod čarou GI by například mohla být asociována s procedurou, která tiskne poznámky pod čarou ve spodní části stránky, nebo která je shromažďuje na konci kapitoly.

c) Zpracování: provádí se vybraná funkce zpracování.

Programy formátování textu jsou ve shodě s tímto modelem. Nejprve rozpoznávají takové prvky, jako jsou slova a věty tím, že interpretují mezery a interpunkce jakožto implicitní vyznačení. Mapování se obvykle provádí pomocí tabulky větvení. Zpracování slov typicky zahrnuje určení délky slova a test překročení řádky; zpracování vět by mezi věty mohlo vkládat mezeru.[1)]

V případě prvků nižší úrovně, jako jsou slova a věty, je uživateli obvykle dána malá možnost ovládat zpracování, a téměř žádná možnost ovládat rozpoznávání. Některé formátory nabízejí větší pružnost, pokud jde o prvky vyšší úrovně, jako jsou odstavce, zatímco formátory s výkonnými jazyky maker mohou mít úspěch, pokud podporují popisné vyznačování. Z hlediska modelu zpracování dokumentu spočívá přednost popisného vyznačování v tom, že uživateli dovoluje definovat atributy - a tudíž typy prvku -, které nejsou známé formátoru, a specifikovat pro ně zpracování.

Například právě popsaná ukázka SGML obsahuje kromě běžnějšího „odstavce" („paragraph") též typy prvku „uspořádaný seznam" („ordered list") a „položka seznamu" („list item"). Zabudované rozpoznávání a zpracování takových prvků je nepravděpodobné. Místo toho bude každý rozpoznáván podle svého explicitního vyznačení a pro konkrétní průběh zpracování mapován do k němu asociované procedury. Jak vlastní procedura, tak asociace k GI by se vyjádřily v jazyku maker systému. Při jiných průbězích zpracování nebo v jiném okamžiku téhož průběhu zpracování se může asociace změnit. Položky seznamu by se například mohly v těle knihy očíslovat, ale v příloze označit písmeny.

Až dosud jsme hovořili pouze o jediném atributu, generickém identifikátoru, jehož hodnota charakterizuje sémantickou funkci prvku nebo účel. Některá schémata popisného vyznačování činí odkazy na vyznačení, jako je „generické kódování" („generic coding"), protože GI je jediným atributem, který rozpoznávají (5). Ve schématech generického kódování může být rozpoznávání, mapování a zpracování uskutečněno najednou jednoduchým způsobem tak, že se generické identifikátory použijí jako názvy řídicích procedur. Ze stejného vyznačení lze získat různé formáty vyvoláním jiné množiny homonymních procedur. Tento přístup je dostatečně efektivní k tomu, aby jedna význačná implementace, systém SCRIBE, mohla úplně zakázat procedurální vyznačování (1).

Generické kódování při praktickém použití znamená značné zlepšení, oproti procedurálnímu vyznačování, avšak je koncepčně nedostačující. Dokumenty jsou složité objekty a obsahují jiné atributy, které musí být vyznačovací jazyk schopen popsat. Předpokládejme například, že se uživatel rozhodne, že jeho dokument zahrne prvky typu „obrázek" („figure") a že musí být možné činit odkazy na jednotlivé obrázky pomocí názvu. Označení pro prvek určitého obrázku známé jako „obranděl" („angelfig") by mohlo začínat tímto počátečním příznakem:

<fig id=angelfig>

„Fig" ovšem zastupuje „obrázek" („figure"), hodnotu atributu generického identifikátoru. GI identifikuje prvek jako člen množiny prvků majících stejnou funkci. Naproti tomu atribut „jedinečného identifikátoru" („unique identifier") (ID) odlišuje tento prvek od všech ostatních, dokonce i těch, které mají stejný GI. (Není nezbytné říkat „GI=fig", jak tomu bylo v případě ID, protože v SGML se rozumí, že první částí vyznačení prvku je hodnota jeho GI).

---

1) Model se nemusí odrážet v architektuře programu; například zpracování slov může být pro zlepšení funkce zabudováno do hlavní rozpoznávací smyčky.

Atributy GI a ID se nazývají „primární", protože je může mít každý prvek. Existují také „sekundární" atributy, které náležejí pouze jistým typům prvků. Například, kdyby uživatel chtěl, aby některé z obrázků v jeho dokumentu obsahovaly ilustrace, které má vytvořit umělec a které mají být přidány ke zpracovávanému výstupu, mohl by definovat typ prvku „výtvarné dílo" („artwork"). Protože by mohla být důležitá velikost externě generovaného výtvarného díla, mohl by definovat prvky výtvarného díla tak, aby měly sekundární atribut „výška" („depth").[1] Výsledkem by byl následující počáteční příznak pro část výtvarného díla o výšce 24 pika:

<artwork depth=24p>

Vyznačení obrázku by též mělo popsat jeho obsah. „Obsah" („content") je ovšem primárním atributem, t.j. tím, který popisují sekundární atributy prvku. Obsah se skládá z uspořádání jiných prvků, z nichž každý může dále mít ve svém obsahu jiné prvky atd., až už další dělení není možné. [2] Jeden způsob, kterým se SGML odlišuje od schémat generického kódování, spočívá v koncepčních prostředcích a v prostředcích notace, které poskytuje pro zacházení s touto hierarchickou strukturou. Ty jsou založeny na druhé hypotéze univerzálního vyznačování, že vyznačování může být přísné.

A.3 Přísné vyznačování

Předpokládejme, že obsah obrázku „angelfig" se skládá ze dvou prvků, těla obrázku a názvu obrázku. Tělo obrázku samo dále obsahuje prvek výtvarného díla, zatímco obsah názvu obrázku tvoří znaky textu bez explicitního vyznačení. Vyznačení pro tento obrázek by mohlo vypadat takto:[3]

<fig id=angelfig>
<figbody>
<artwork depth=24p>
</artwork>
</figbody>
<figcap>Three Angels Dancing — Tři tančící andělé —
</figcap>
</fig>

Vyznačování přísně vyjadřuje hierarchii tím, že identifikuje začátek a konec každého prvku v klasickém levém uspořádání seznamu. K interpretování struktury nejsou potřeba žádné dodatečné informace a bylo by možné implementovat podporu pomocí jednoduchého schématu vyvolání makra uvedeného výše. Nicméně za tuto jednoduchost je třeba zaplatit tím, že pro každý prvek musí být přítomen koncový příznak.

Tato cena by byla zcela nepřijatelná, kdyby musel uživatel všechny příznaky zapisovat sám. Ví například, že začátek odstavce ukončuje předchozí odstavec, takže by se mohl zdráhat si přidělávat potíže a náklady tím, že by zapisoval koncový příznak pro každý jednotlivý odstavec a raději se o tuto svoji znalost podělí se systémem. Rovněž by ho asi velmi rozčilovalo definovat si jiné typy prvků, pokud by se vyskytovaly příliš často.

V případě SGML je však možné mnohá vyznačení vynechat tím, že se systém informuje o struktuře a atributech každého typu prvku, který uživatel definuje. To se provádí sestavením „definice typu dokumentu" (document type definition), k čemuž se používá konstrukce jazyka nazývaná „deklarace prvku" (element declaration). Zatímco vyznačení v dokumentu se skládá z popisů jednotlivých prvků, definice typu dokumentu stanoví množinu všech možných platných vyznačení typu prvku.

Deklarace prvku obsahuje popis přípustného obsahu, který je obvykle vyjádřen ve variantě notace řádných výrazů. Předpokládejme například, že uživatel rozšíří svoji definici „obrázku" („figure") tak, aby tělo obrázku mohlo obsahovat buď výtvarné dílo nebo jisté druhy textových prvků.

---

[1] „Depth=" není pouze ekvivalent řídicího slova meziřádkové vzdálenosti. Ačkoliv by program sestavení plné stránky mohl vytvořit skutečnou mezeru, formátor sazebnice by mohl vytisknout zprávu s pokynem pro uspořádávajícího umělce, že ji má vynechat. Program vyhledávání by jednoduše mohl indexovat obrázek a zcela ignorovat výšku.

[2] Může se tudíž hovořit o dokumentech a prvcích téměř zaměnitelně: dokument je prostě prvek, který je na vrcholu hierarchie pro daný průběh zpracování. Technická zpráva například by mohla být formátována jako samostatný dokument i jako prvek časopisu.

[3] „obsah =" („Content=") podobně jako „GI=", může být bezpečně vynechán. Není nezbytný, je-li obsah generován externě, je mu rozuměno v případě, kdy je obsah tvořen pouze prvky vyznačenými příznakem, v případě znaků dat je doplněn oddělovačem (>), který ukončuje počáteční příznak.

Deklarace prvku by mohla vypadat takto:[1)]

```
<!— PRVKY          MIN OBSAH (VÝJIMKY)—>
<!ELEMENT fig      - - (figbody, figcap?)>
<!ELEMENT figbody  - 0 (artwork | ol | ul)+)>
<!ELEMENT artwork  - 0 EMPTY>
<!ELEMENT figcap   - 0 (|PCDATA)>
```

První deklarace znamená, že obrázek obsahuje tělo obrázku a že může volitelně obsahovat název obrázku, který následuje za tělem obrázku. (Spojovníky vysvětlíme zanedlouho.)

Druhá deklarace praví, že tělo může obsahovat buď výtvarné dílo, nebo smíšené seskupení odstavců, uspořádaných seznamů a neuspořádaných seznamů. „O" v poli minimalizace vyznačení („MIN") indikuje, že koncový příznak těla lze vynechat, je-li jednoznačně implicitně určen začátkem následujícího prvku. Předcházející spojovník znamená, že počáteční příznak *nemůže* být vynechán.

Deklarace vztahující se na výtvarné dílo je definuje, jako by mělo prázdný obsah, protože výtvarné dílo bude vytvořeno externě a do těla bude vloženo. Ukončujícího vyznačení není třeba, protože dokument nemá obsah.

Poslední deklarace definuje obsah názvu obrázku jako 0 nebo více znaků. Znak je konečný a nepřipouští další dělení. „O" v poli „MIN" indikuje, že se může vynechat koncový příznak názvu obrázku. Navíc je z již uvedených důvodů vynechání možné v případě, že je koncový příznak jednoznačně implicitně určen koncovým příznakem prvku, který obsahuje název obrázku.

Předpokládá se, že p, ol a ul byly definovány v jiných deklaracích prvku.

Po takto provedené formální definici dostupných prvků obrázku je nyní přijatelné následující vyznačení pro „obranděl" („angelfig"):

```
<fig id=angelfig>
<figbody>
<artwork depth=24p>
<figcap>Three Angels Dancing      — Tři tančící andělé —
</fig>
```

Protože už nejsou potřebné koncové příznaky pro tři z prvků, bylo dosaženo 40% redukce vyznačení.

– Jelikož deklarace prvku definovala název obrázku jako část obsahu obrázku, tím, že ukončuje obrázek, automaticky ukončila název obrázku.

– Počáteční příznak <figcap> implicitně ukončuje tělo obrázku, protože samotný název obrázku je na stejné úrovni jako tělo obrázku.

– Prvek výtvarného díla byl samoukončující, jelikož deklarace prvku definovala jeho obsah jako prázdný.[2)]

Definice typu dokumentu také obsahuje „deklaraci seznamu definic atributů" pro každý prvek, který má atributy. Definice obsahuje možné hodnoty, kterých může atribut nabývat, a rovněž předurčenou hodnotu, pokud je atribut volitelný a není v dokumentu specifikován.

Toto jsou deklarace seznamu atributů pro „obrázek" („figure") a „výtvarné dílo" („artwork"):

```
<!—          PRVKY    NÁZEV    HODNOTA    PŘEDURČENÍ —>
<!ATTLIST    fig      id       ID         #IMPLIED>
<!ATTLIST    artwork  depth    CDATA      #REQUIRED>
```

Deklarace týkající se obrázku indikuje, že může mít atribut ID, jehož hodnotou musí být název jedinečného identifikátoru. Atribut je volitelný a nemá předurčenou hodnotu, není-li specifikována.

Naproti tomu se vyžaduje atribut výšky prvku výtvarného díla. Jeho hodnotou může být jakýkoliv řetězec znaků.

[1)] Otazník (?) znamená, že prvek je volitelný, čárka (,) znamená, že prvek následuje po předcházejícím prvku v posloupnosti, hvězdička (*) znamená, že se prvek může vyskytovat 0 nebo vícekrát a plus (+), že se prvek musí vyskytnou 1 nebo vícekrát. Svislá čára (|) se používá k oddělení alternativ. Závorky se používají k seskupování jako v matematice.

[2)] Ve skutečnosti SGML umožňuje zredukovat počet vyznačení ještě více.

Definice typu dokumentu se používají jako doplněk k minimalizaci vyznačení.[1] Mohou se používat ke kontrole platnosti vyznačení v dokumentu, ještě před tím, než se vynaloží náklady na jeho zpracování, nebo k řízení dialogových výzev uživatelům, kteří nejsou seznámeni s typem dokumentu. Například aplikace zapsání dokumentu by mohla číst popis prvku obrázku a vyvolat procedury pro každý typ prvku. Procedury by mohly vysílat zprávy k terminálu, vyzývající uživatele, aby zapsal ID obrázku, výšku výtvarného díla a text názvu obrázku. Procedury by rovněž mohly zapisovat vlastní vyznačení do vytvářeného dokumentu.

Definice typu dokumentu umožňuje, aby SGML minimalizoval uživatelovo úsilí při zapisování textu bez spoléhání se na „inteligentní" program na úpravu textu. Tím se dosahuje maximální přenositelnost dokumentu, protože může být pochopen a revidován lidmi používajícími jakoukoliv z miliónů existujících „němých" klávesnic. Definice typu společně s dokumentem opatřeným vyznačeními však přesto stále vytvářejí přísně popsaný dokument tak, jak jej vyžaduje strojové zpracování.

A.4 Závěr

Bez ohledu na stupeň přesnosti a flexibility, který při popisu dokumentu umožňuje univerzální vyznačování, zůstává pro uživatele, který připravuje dokumenty pro publikování, důležité toto: může standardní univerzální vyznačovací jazyk nebo nějaké schéma popisného vyznačování dosáhnout typografických výsledků srovnatelných s procedurálním vyznačováním. Poslední publikace vydaná Prentice-Hall International (6) představuje empirické potvrzení hypotézy univerzálního vyznačování v kontextu této praktické otázky.

Je to učebnice programování, která obsahuje stovky autorem vytvořených vzorců v symbolické notaci. Přes typografickou složitost materiálu (v mnoha řádcích je například tucet i více změn fontů) nebylo v celém textu knihy potřeba použít žádné procedurální vyznačení. Text byl vyznačen použitím jazyka, který se přibližoval principům univerzálního vyznačování, ale který byl méně flexibilní a úplný než SGML (4).

Dosažitelné procedury podporovaly pouze výstupní zařízení počítačů, která byla vhodná pro předběžné verze knihy, které se používaly jako skripta. O otázkách sázení písma se neuvažovalo, až do chvíle, kdy byla kniha přijata k publikaci. V tomto okamžiku se její autor zarazil nad časem a úsilím nezbytným k přepsání a korektuře asi 350 stránek. Začal hledat alternativní řešení ve stejné době, kdy autor tohoto článku hledal předmět, na němž by experimentálně ověřil použitelnost univerzálního vyznačování pro komerční publikace.

Po čase, kdy hledání obou úspěšné skončilo, byl zahájen neobvyklý projekt. Jelikož autorův procesor přímo nepodporoval fotosázecí stroje, byly napsány procedury, které vytvářely zdrojovou množinu s procedurálním vyznačováním pro zvláštní program typografické sazby. Specifikace pro formátování poskytl vydavatel a přesto, že vyznačený dokument již existoval před předáním specifikací[2], nebylo třeba žádných ústupků, aby se vyhovělo použití univerzálních vyznačení.

Experiment byl ukončen včas a vydavatel ho považoval za naprostý úspěch (7).[3] Procedury po provedení některých modifikací stylu formátování nalezly další využití při výrobě rozmanitých vnitropodnikových publikací.

Univerzální vyznačování má tedy jak praktický, tak akademický význam. U vydavatele redukuje náklady na vyznačování, zkracuje dobu při výrobě knih a nabízí maximální flexibilitu plynoucí z textové databáze. V úřadech umožňuje vzájemnou výměnu mezi různými druhy textových procesorů s rozmanitými funkčními možnostmi a dovoluje, aby se z příslušných prvků základního dokumentu, jako je memorandum, automaticky vytvářely pomocné „dokumenty", jako jsou záznamy „mail log entries".

Přísné popisné vyznačování jazyka SGML současně činí text přístupnějším pro počítačovou analýzu. Na rozdíl od procedurálního vyznačování (nebo vůbec žádného vyznačování), zanechávajícího dokument jako řetězec znaků, který má pouze takový tvar, jaký může být odvozen z analýzy smyslu dokumentu, univerzální vyznačování redukuje dokument do správného tvaru ve známé gramatice. To umožňuje, aby se zavedené techniky počítačové lingvistiky a kompilačního výzkumu používaly v aplikacích zpracování přirozeného jazyka a jiných dokumentů.

---

[1] Některé kompletní praktické definice typu dokumentu lze nalézt v (4), i když nejsou kódovány v SGML.

[2] Naopak, vydavatel využíval univerzální vyznačování a měnil některé specifikace poté, co viděl stránkové korektury.

[3] A to navzdory určitým zeměpisným komplikacím: vydavatel byl v Londýně, autor knihy v Bruselu a autor tohoto článku v Kalifornii. Téměř veškerá komunikace se děla přes mezinárodní počítačovou síť a když se všichni účastníci poprvé sešli, byl projekt již téměř dokončen.

.5 Poděkování

utor je zavázán E. J. Mosherovi, R. A. Loriemu, T. I. Petersonovi a A. J. Symondsovi - spolupracovníkům při počátečním vývoji univerzálního vyznačování - za řadu příspěvků k myšlenkám popsaným v tomto článku, N. R. isenbergovi za jeho spolupráci při návrhu a vývoji procedur používaných při ověřování použitelnosti univerzálního yznačování v komerční publikační činnosti a C. B. Jonesovi a Ronu Decentovi za to, že riskovali svou vynikající nihou pro některé nové myšlenky.

.6 Literatura

B. K. Reid, „The Scribe Document Specification Language and its Compiler", *Proceedings of the International Conference on Research and Trends in Document Preparation Systems*, 59-62 (1981).

Donald E. Knuth, *TAU EPSILON CHI, a system for technical text,* American Mathematical Society, Providence, 979.

C. F. Goldfarb, E. J. Mosher, and T. I. Peterson „An Online System for Integrated Text Processing", *Proceedings fthe American Society for Information Science,* 7, 147-150 (1970).

Charles F. Goldfarb, *Document Composition Facility Generalized Markup Language: Concepts and Design Guide,* Form No. SH20-9188-1, IBM Corporation, White Plains, 1984.

Charles Lighfoot, *Generic Textual Element Identification-A Primer,* Graphic Communications Computer Association, Arlington, 1979.

C.B.Jones, *Software Development: A Rigorous Approach,* Prentice-Hall International, London, 1980.

Ron Decent, *osobní komunikace s autorem* (7. září 1979).