Specifikace ročníkového projektu na téma:

Převod PDF dokumentů do textu

Autor: **Jonáš Bujok**

Motivace, aneb o čem to bude:

V dnešní době jsou PDF dokumenty velmi rozšířené a máme také nepřeberné množství programů s nimi pracujících. Jednou ze skupin takovýchto programů jsou ty, které převádějí PDF dokument na jiný formát dokumentu. Např. DOC, TXT, PS a jiné. Tyto programy se velice liší v mnoha faktorech jako uživatelské rozhraní, rychlost a způsob použití, ale většina z nich má společnou jednu špatnou vlastnost. U většiny programů je problém s převodem českých speciálních znaků. Dokonce starší verze Adobe Acrobat reader mají s tím problém. Výstupy bývají různé. Občas jsou čárky nebo háčky zvlášť vedle písmenka, nebo převádí na úplně divné znaky. Často je tak třeba celý dokument procházet a opravovat tyto chyby. Proto mně napadlo vytvořit převodní program z PDF do formátu TXT, který by umožnil uživatelům se tomuto převodnímu problému vyhnout a převáděl by tyto znaky správně.

Platforma: Linux nebo Microsoft Windows,
Vývojové prostředí: Microsoft Visual Studio 2008
Jazyk: C++

Jak to bude vypadat:

V tomto odstavci se moc rozepisovat nebudu, protože jsem zvolil řešení jako command line aplikaci. Výsledný program bude jeden soubor, který bude možné spustit z příkazové řádky a za název programu do prvního argumentu napsat vstupní PDF soubor a do argumentu druhého výstupní soubor TXT. Toto řešení se zdá být logické, protože pro samotný převod není potřeba žádné GUI. Pokud chceme převést PDF na text, tak není důležité, že ten program, který to provádí má pěkné tlačítka. Druhá výhoda tohoto řešení je, že se dá použít pro náročnější nasazení, kde bude volán nějakým skriptem a převádět velké množství souborů.

Něco málo o PDF:

Struktura souboru PDF je velmi komplikovaná a její popis rozsáhlý (viz. PDF specifikace, která má kolem 1300 stran), ale pokusím se vysvětlit alespoň základ jak je text v PDF uložen.
Zjednodušeně řečeno PDF soubor má v sobě uloženy objekty (kde každý má své identifikační číslo) a referenční tabulku, která popisuje, které objekty se mají použít (mohou tam být i nevyužité objekty třeba z dřívějších verzí dokumentu) a v jakém pořadí se mají načítat (mohou se použít i vícekrát). Referenční tabulka je vlastně jen seznam objektů a jejich poloha v souboru (číslo bytu kde objekt začíná).
Jeden z mnoha druhů objektů je „Text Object“ uzavřený mezi klíčová slova „BT“ a „BE“. Tento objekt, kromě samotného textu, obsahuje i množství jiných informací (fonty, rozložení, velikost atd.) důležitých pro jeho zobrazení. Samotný text je pak uložen v Unicode standardu jako „String Object“ nebo odkazem na jiný objekt. Pro speciální znaky se používají převodní tabulky (viz. Popis převodu dále). A aby toho nebylo málo, tak celý objekt může být zabalen do jiného objektu a zkomprimován (či zakódován) jednou, či několika kompresními metodami najednou. Na to se používá „Stream Object“. Na začátku tohoto objektu je slovník, kde jsou uloženy informace potřebné pro dekompresi (či dekódování) streamu (length, filter aj.) a pak následuje samotný stream uzavřený mezi klíčová slova „stream“ a „endstream“. Pro text se ve valné většině případů používá FlateDecode, který implementuje volně šiřitelná knihovna Zlib. Může být také použit ASCIIHexDecode, ASCII85Decode, LZWDecode, RunLengthDecode a Crypt.

Jak program bude fungovat:

Když už víme jak je text v PDF uložen, rozeberme si trochu postup jakým se text z PDF souboru získává. Program se nejdříve podívá na konec souboru, kde je uložen (za případnými komentáři) odkaz na referenční tabulku. Poté najde referenční tabulku a začne jí procházet řádek po řádku, a podle informací v ní uložených načítat objekty. Jednotlivé objekty pak prohledávat o výskyt textového objektu nebo stream objektu.

- Pokud objeví textový objekt, tak se pokusí v něm najít string objekt a ten pak bude procházet metodou pro převod /*ddd* (d je desítková číslice) posloupnosti na speciální znaky a výsledek se pošle na výstup. Obsahuje-li textový objekt informace o fontu, je třeba si jej také uložit, protože bude potřeba pro převodní metodu. Převodní metodu popíši níže.
- Pokud objeví stream objekt tak použije dekompresní metody popsané v parametrech objektu a pak pokračuje předchozí popsanou metodou pro hledání textového objektu uvnitř streamu. První verze programu bude schopná dekomprimovat pouze FlatDecode. Další verze by mohla umět i LZWDecode a další metody popsány výše.
- Ostatní objekty bude zahazovat, až na vyjímky, jako jsou číselné objekty, stringy, fonty a CMap objekty (viz dále).

Převod znakového kódu /ddd na Unicode hodnotu:

Jak jsem popisoval výše – speciální znaky jsou ve string objektu v PDF souboru nahrazeny posloupností „/ddd“, tedy lomítkem a třímístným desítkovým číslem. Tyto znakové kódy je třeba správně převést do Unicode formátu při získávání textu z stringu v textovém objektu. K tomu potřebujeme zjistit jestli Font Dictionary (informace o fontu) daného textového objektu obsahuje položku ToUnicode. Pokud ne, pak se použijí standardní převodní tabulky (což dělá většina převodních programů vždy a to je chyba) popsané v PDF specifikaci v appendix D. Jestliže font obsahuje položku ToUnicode (je to většinou odkaz na jiný objekt v PDF souboru) znamená to, že daný font má vlastní mapu (ve specifikaci PDF se jmenuje CMap) znakových kódů na Unicode hodnotu a proto je třeba použít tuto mapu místo standardních (což většina převodních programu pravděpodobně ignoruje).

Závěrem:

Kromě toho, že program by měl správně dekódovat českou (a taky jinou) diakritiku, bude viditelná i snaha o efektivitu jak paměťovou tak výpočetní (pro případ nasazení v oblastech kritických na rychlost). Nebudou se dekódovat objekty, které nejsou potřeba a bude se „uklízet“ paměť kdykoli to půjde (mazání objektů které, už byly zpracovány). Jelikož ale není celá operace získávání textu moc paralelní, bude velmi těžké najít části, které by mohly být zpracovávány paralelně a využít tak multiprocesorový potenciál dnešních počítačů. Proto aplikace pravděpodobně nebude vícevláknová.

Použité zdroje informací:

- PDF Reference and Related Documentation od firmy Adobe
- rfc1950, rfc1951 a další rfc dokumenty pro pochopení kompresních metod
- Internet – další informace ohledně Unicode, utf aj.

Chtěl jsem ještě na konec odůvodnit tak malý počet použitých zdrojů: Je to způsobeno tím, že PDF specifikace (první zdroj v seznamu) je natolik obsáhlá a vyčerpávající, že pro napsání převodního programu již není potřeba více zdrojů. Je v ní obsažena syntax PDF souboru, popis všech objektů a standardní převodní tabulky a všechny další nutné informace.