Spoločnosť pre otvorené informačné technológie

OTVORENÝ SOFTVÉR VO VZDELÁVANÍ, VÝSKUME A V IT RIEŠENIACH

Zborník príspevkov medzinárodnej konferencie
OSSConf 2014

2.–4. júla 2014
Žilina, Slovensko

SPOLUORGANIZÁTOR KONFERENCIE

SPONZORI KONFERENCIE

PARTNERI KONFERENCIE

Spoločnosť pre otvorené informačné technológie

OTVORENÝ SOFTVÉR VO VZDELÁVANÍ, VÝSKUME A V IT RIEŠENIACH

Zborník príspevkov medzinárodnej konferencie
OSSConf 2014

2.–4. júla 2014
Žilina, Slovensko

Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
2.–4. júla 2014, Žilina, Slovensko

Vedeckí garanti konferencie:

prof. Ing. Karol Matiaško, PhD., Žilinská univerzita, Žilina
prof. Ing. Miloš Šrámek, PhD., Austrian Academy of Sciences, Wien (AUT); SOIT, Bratislava

Vedecký a programový výbor:

prof. Ing. Miloš Šrámek, PhD., Austrian Academy of Sciences, Wien (AUT) – **predseda**
prof. Oleg Černojarov, DrSc., Moscow Power Engineering Institute (RU)
doc. Ing. Karol Grondžák, PhD., Žilinská univerzita, SOIT, Žilina
doc. RNDr. Štefan Peško, CSc., Žilinská univerzita, SOIT, Žilina
doc. Ing. Jiří Rybička, Dr., MZLU Brno (CZ)
RNDr. Rudolf Blaško, PhD., Žilinská univerzita, SOIT, Žilina
RNDr. Ján Buša, CSc.,FEI, Technická Univerzita Košice, SOIT, Košice
Mgr. Peter Czimmermann, PhD., Žilinská univerzita, SOIT, Žilina
Mgr. Michal Kaukič, CSc., Žilinská univerzita, SOIT, Žilina
RNDr. Aleš Kozubík, PhD., Žilinská univerzita, SOIT, Žilina
Dr inż. Remigiusz Olejnik, Zachodniopomorski Uniwersytet Technologiczny w Szczecinie (PL)
Ing. Pavel Stříž, Ph.D., Nakladatelství Martin Stříž, Bučovice (CZ)
RNDr. Ladislav Ševčovič, PhD., FEI, Technická Univerzita Košice, SOIT, Košice
Ing. Jiří Eischman, Red Hat CZ, SOIT, Brno (CZ)

Organizačný výbor:

Aleš Kozubík, Žilinská univerzita, SOIT, Žilina – **predseda**
Rudolf Blaško, Žilinská univerzita, SOIT, Žilina
Roman Hajtmanek, SOIT, Žilina
Michal Kaukič, Žilinská univerzita, SOIT, Žilina
Tomáš Majer, Žilinská univerzita, SOIT, Žilina
Ľubica Michálková, Žilinská univerzita, Žilina
Peter Mráz, DRU, SOIT, Kremnica
Miloslav Ofúkaný, GeoCommunity, SOIT, Bratislava
Michal Páleník, Freemap Slovakia, SOIT, Bratislava
Martin Šechný, SOIT, Prešov
Peter Štrba, Gymnázium Mikuláša Galandu, SOIT, Turčianske Teplice

Vydavateľ: Spoločnosť pre otvorené informačné technológie, Bratislava

ISBN 978-80-970457-4-6

Recenzenti:

Blaško, Rudolf, RNDr., PhD.
Czimmermann, Peter,Mgr., PhD.
Grondžák, Karol, doc., Ing., PhD.
Kaukič, Michal, Mgr., CSc.
Kozubík, Aleš, RNDr., PhD.
Kozubíková, Zuzana, Ing., PhD.
Majer, Tomáš, Ing., PhD.
Peško, Štefan, doc., RNDr., PhD.
Rybička Jiří, doc., Ing., Dr.
Segeč, Pavel, doc., Ing., PhD.
Stříž, Pavel, Ing., Ph.D.

Editori:

Rudolf Blaško
Aleš Kozubík

Všetky práce, uverejnené v zborníku, boli posúdené dvomi nezaujatými recenzentmi.

Práce, uverejnené v zborníku, neprešli jazykovou úpravou.

POZVANÍ PREDNÁŠATELIA

6. ročník Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf
OSSConf 2014: 7–10

Otevřený svět Geo*

JÁCHYM ČEPICKÝ (CZ)

Abstrakt. Open Source Geosvět se od toho „normálního“ IT svět trochu v něčem liší. Co je to ale otevřenost, jako koncept, kde se vzala tak najednou? A nebo to nebylo najednou? A proč vlastně veškerý software není otevřený? A měl by být?

Klíčová slova. GIS, Open Source, Free Software.

OPENED WORLD OF GEO*

Abstract. Open Source for geospatial is always little bit different from „standard“ IT world. What is it **to be open**, where does it come from? Or it was not all at sudden? Why isn't all software open sourced? Should it be?

Keywords. GIS, Open Source, Free Software.

Více než deset let se zabývám vývojem open source software pro geoinformatiku. Úzce s tím souvisí i to, že trochu déle se počítám mezi uživatele operačního systému GNU/Linux. To obyčejně nepovažuji za důležité zmiňovat, ale v kontextu této přednášky to považuji za významnou skutečnost.

Během studií jsem u svého kamaráda našel v jeho knihovničce knížku „Učíme se Red Hat Linux“ a při jejím čtení mi došlo, že vedle počítačů jak je znám existuje jiný svět, který bych rád poznal. V té době pro mě počítač bylo PC s MS Windows 98, používal jsem Excel na tvorbu protokolů a prací pro svůj obor (kterým bylo lesní inženýrství). Nikdy jsem neměl ambici programovat, počítače jsem chápal jako nástroje, které by mi měli v ideálním případě usnadnit práci. V knížce jsem se seznámil se základními idejemi open source a free software a byly mi sympatické, jakkoliv jsem nedokázal pochopit, proč by někdo sdílel svoji práci – intelektuální vlastnictví – s kýmkoliv jiným. Když se mi v krátké době podařilo pořídit si svůj vlastní nový počítač, požádal jsem kamaráda o pomoc nainstalovali jsme můj první Linux.

Od té doby jsem se stal uživatelem open source a free software a postupem času i jeho spolutvůrcem. Naučil jsem se, jak funguje komunita uživatelů a přispěvatelů do open source software, pochopil jsem některé jemné rozdíly mezi svobodným software a otevřeným software a tak dále. A dnes chápu, že open source, nebo obecně tak zvaný crowd source – sdílený přístup – není žádná anomálie.

Většina z vás už určitě slyšela o tom, že hnutí free software založil Richard Matthew Stallman v 80-tých letech minulého století prací na operačním systému GNU. Definoval tři základní svobody nebo práva, které by měl každý uživatel

software mít: právo vidět zdrojový kód, právo sdílet zdrojový kód a právo měnit zdrojový kód a dále ho sdílet.

V systém se stal úspěšný také díky jádru operačního systému Linux. Dnes je Linux nejpoužívanějším serverovým operačním systémem, používá se v mobilních zařízeních, na super počítačích. Proč se tak stalo? Protože je to otevřený systém – systém, ve kterém ostatní sdílí svůj kód – svoje know-how – svoje intelektuální vlastnictví s ostatními.

Ale koncept sdílení není přece vynalezen v 80-tých letech. Západní věda tak jak ji chápeme je snad od dob antiky postavena na stejných principech: publikuji výsledek způsobem, že je opakovatelný, očekávám, že někdo jiný mou práci zopakuje, přezkouší její platnost, případně navrhne některé změny, a výsledek opět publikuje. Vědecká komunita díky sdílení informací posouvá hranice našeho poznání dále, stejně jako komunita vývojářská posouvá hranice software.

Open source software není jediný příklad pro fungující intelektuální spoluvlastnictví. Nikdo snad nepochybuje, že wikipedia je spolehlivý informační zdroj a přitom na její obsah nemá nikdo monopol, neexistuje globální autorita, která by strážila věcný obsah Wikipedie. OpenStreetMap je podle mě další úspěšný projekt. Jako geoprostoroví profesionálové můžete zpochybňovat polohovou přesnost dat, jejich faktickou správnost nebo aktuálnost, ale nemůžete zpochybnit, že globálně vzato je to ucelený dataset který nemá v prorietárním světe obdoby.

Všechny tyto příklady mají díky svým licencím společné tři základní vlastnosti, které vyzývají další a další uživatele aby se přidali: vidět obsah, sdílet obsah a provádět změny v obsahu a sdílet tyto změny dál.

Je jen málo oblastí geoinformatiky, které by nebyly pokryty kvalitním open source software – i když připouštím, že jsou. Stále vznikají nové programy pro další oblasti, staré ozkoušené projekty jsou ale stále zde, stabilní, s rozšířenou vývojářskou a uživatelskou komunitou, stále dostávají nové funkce, ty staré jsou oprašovány a udržovány.

Dovolte mi jako příklad zmínit několik z nich:

GRASS GIS: – Desktopový GIS, vyvíjený původně americkou armádou od roku 1982, který obsahuje množství kódu českých a slovenských vývojářů.

QGIS: , který je sice mladší než GRASS, ale na Slovensku je opět velice populární. Daří se mu stále více nahrazovat co do vzhledu a funkcí komerčně rozšířený ArcGIS.

PostGIS: je prostorové rozšíření databáze PostgreSQL.

MapServer a GeoServer: jsou zástupci serverových programů, které po světě vydávají mapy a jsou nasazováni i všude tam, kde je potřeba řešit iniciativu INSPIRE.

PyCSW: je serverová aplikace shromažďující a vydávající metadata.

A mnoho dalších.

Kdo jste ještě nebyli na žádné konferenci zabývající se obecně geo* open source software, doporučuji vám navštívit některou z konferencí FOSS4G, ať už letos ve Spojených státech globální, nebo její evropskou odnož v Brémách. Máte také možnost nahlídnout přímo do vývojářské kuchyně na některém z code sprints, naposledy ve Vídni, další možnost budete mít v italském Bozlanu.

Z toho všeho co říkám si možná říkáte: proč to tedy nikdo nepoužívá, když je to tak skvělý produkt? Proč není open source software dávno rozšířenější, než jak to vypadá?

Důvody pro tento stav – a je jedno jestli je to objektivní fakt nebo subjektivní pocit – jsou samozřejmě mnohé a jak už to bývá, mají povahu vnitřní i vnější.

Jako jeden z prvních důvodů (bez nároku na prvenství z hlediska významnosti) uvedu to, že člověk používá to, co zná. Věřím, že obsahem výuky na školách ve všech oborech a stupních by měly být především obecně platné principy, až od nich odvozené konkrétní situace. Učíme obecně Archimedův zákon, a teprve následně několik jeho praktických aplikací, jako že ve vodě je slon lehčí nebo spolu s inženýry španělského námořnictva, že ponorka, má-li se vynořit, musí mít především v součtu menší objemovou hustotu, než voda. Proč se ale tento princip uplatňuje při výuce software jen velice zřídka? Proč výuka počítačů obecně a GIS konkrétně se téměř bez výjimky provádí na konkrétním jednom software?

Dalším důvodem je neexistence jednotného telefonního čísla – chybí marketingové oddělení open source GIS, není tu někdo, kdo by zákazníkům vysvětlil proč je konkrétní produkt ten nejlepší (v absolutním i relativním významu). U open source software se předpokládá, že jste dospělí, že jste nebo se chcete stát experty a proto na to, co je pro váš případ to nejlepší si přijdete sami. Open source je náročný na lidské zdroje, pokud nemáte u sebe někoho dalšího, jste v tom tak trochu sami, „jenom“ s podporou komunity. To může hodně lidí odradit, ale bylo mnohokrát ilustrováno, že podpora komunity funguje, první odpověď v mailing listech bývá u větších projektů v řádu minut.

Dalším důvodem bude roztříštěná nabídka open source software. Jak jsem řekl dříve, nemáte se moc koho zeptat a ještě ke všemu si musíte vybírat z množství variant – sám s tím mám často problém. Provozní výzkum, porovnávání různých programů a sledování nových je denní chléb. Mám vzít TileCache, MapStach, GeoServer cache, MapCache? A co je to ten Mapnik? A jaký je rozdíl mezi Leaflet a OpenLayers? Na tyto otázky získáte nejlépe odpověď tak, že budete číst nebo se zeptáte někoho, kdo to ví, ale jak jsem již řekl – ve vašem okolí je většinou problém najít někoho kdo by to věděl. Nebo snad ne?

Dalším důvodem je nejednotný systém návazného vzdělávání. Existují spíše jednotlivci, nabízející školení. Školení jsou ale nejednotná, různá obsahem i kvalitou.

Důvodů, proč open source software není úspěšnější – myšleno absolutně – je ještě celá další řada. Některé z nich a některé z již zmíněných se bude snažit zaplnit na nejen naší geo- scéně projekt GISMentors.

Na Slovensku už jsou firmy, nabízející, podle toho, co jsem mohl vidět, open source řešení na profesionální úrovni, včetně možnosti, jak říkávám „sáhnout do stroje“ a opravit chybu nebo přidat funkci. Myslím, že i vzhledem k tomu kolik slovenských vývojářů se pohybuje na open source GIS scéně, má Slovensko světlou geo-budoucnost.

Kontaktní adresa

Ing. Jáchym Čepický, Česká republika,
E-mailová adresa: jachym@les-ejk.cz, http://les-ejk.cz,http://gismentors.eu

6. ročník Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf
OSSConf 2014: 11–16

SKÚSENOSTI Z NASADENIA FOSS V STREDNE VEĽKOM VÝROBNOM PODNIKU

PETER MRÁZ (SK)

Abstrakt. Keď hľadáme slobodný softvér, ktorý je možné použiť vo firme. Na rôznych stránkach sa dozvieme o rôznych alternatívach ku komerčným programom. Väčšinou sa dozvieme, že miesto programu Internet Explorer môžeme použiť Firefox, miesto programu Outlook, program Thunderbird a miesto MS Office program Libre Office. Zrejme sa tiež dozvieme o tom, že je možné nahradiť aj operačný systém, používateľsky prívetivými systémami ako Ubuntu, Mint, Fedora a podobne. Aké programy sú potrebné v stredne veľkej výrobnej spoločnosti na slovensku. Vystačí si s uvedenými programami a sú adekvátnou náhradou za komerčné programy? Akým spôsobom sa dajú nasadiť?

EXPERIENCE OF FOSS DEPLOYMENT IN MID-SIZED MANUFACTURING COMPANY

Abstract. Author is unable to translate it.

Úvod

Pri hľadaní alternatívnych riešení určite narazíme na rôzne články o tom, ako sa dá komerčný softvér plnohodnotne nahradiť otvoreným softvérom. Narazíme pri tom na takéto tabuľky:

- MS Office → OpenOffice.org alebo LibreOffice,
- Internet Explorer → Firefox alebo Chromium,
- Outlook → Thunderbird + Lightning,
- Photoshop → Gimp,
- CorelDraw → Inkscape,
- Autocad → LibreCad alebo Qcad,
- Nero → InfraRecorder,
- Adobe InDesign → Scribus,

  ...

1. Windows vs. Linux

Na diskusných fórach často vznikajú nekonečné diskusie o tom, že Linux plnohodnotne dokáže nahradiť Windows.

Ak si položíme otázku či je Linux lepší ako Windows a naozaj si dáme tú námahu, že si ho doma nainštalujeme a budeme ho používať, zistíme, že v domácom prostredí dokážeme Linux používať namiesto komerčného operačného systému veľmi ľahko.

Na Linuxe oceníme hlavne:

- rýchlu a jednoduchú inštaláciu,
- inštaláciu softvéru bez neustáleho reštartovania počítača a potvrdzovania súhlasu s licenciou,
- fungovanie periférií ihneď po pripojení – Linux má v súčasnosti najväčšiu databázu podporovaného hardvéru zo všetkých operačných systémov,
- bezpečnosť bez potreby žrúta systémových prostriedkov – antivírového programu,
- rozmanitosť použitia – žiadne členenie, home, profesional, ultimate, server, standard,
- viac ako 30 000 dostupných softvérových programov nachádzajúcich sa na jednom mieste v softvérovom úložisku,
- dostupnosť bezplatnej dokumentácie a diskusných fór,
- možnosť upravovať zdrojový kód programov.

1.1. Keď je Linux taký dobrý, prečo ho všetci nepoužívame?

To je dobrá otázka. Vezmime si systém Android, ide o systém na báze Linuxu, ktorý sa masovo rozšíril v mobiloch a tabletoch. Tu je naozaj jednoznačne viditeľné, že slobodný softvér plnohodnotne dokáže nahradiť komerčné operačné systémy.

2. Použite slobodného softvéru vo firemnom prostredí

Ak si prečítame správy na Linuxových portáloch, dozvieme sa o množstve veľkých spoločností, mestských úradov, samospráv, ba dokonca štátoch, ktoré prešli na slobodný softvér a ušetrili značné prostriedky za softvérové licencie.

Zjavne je teda prechod na slobodný softvér možný aj v spoločnostiach. Keď dokážu používať otvorený softvér na Islande, prečo by sa nemal dať používať v malej firme na Slovensku.

3. Problémy použitia slobodného softvéru v malej firme na Slovensku

Povedzme, že sme firma do 200 zamestnancov na Slovensku a chceme začať používať slobodný softvér. Je mnoho problémov, na ktoré narazíme.

3.1. Bariéra zo strany štátu

Problém 1: Pri komunikácii so štátnou správou je potrebné použiť zaručený elektronický podpis, ten je však zatiaľ možné použiť iba v systéme Windows.

Problém 2: Niektoré stránky vyžadujú použitie doplnkov ActiveX, Silverlight a podobne. Typickým príkladom je zobrazenie máp na Katastrálnom portále.

Problém 3: Ako výrobný podnik musíme z času na čas vyplňovať rôzne formuláre vytvorené v MS Office obsahujúce makrá. Ak tvorcu požiadame o dokument v `pdf`, tak nám ho síce ochotne pošlú, ale pôjde o `pdf`, do ktorého sa nebude dať nič zapísať.

3.2. Bariéra zo strany ostatných spoločností

Pre ostatné spoločnosti platia tiež predchádzajúce body.

Problém 4: V prípade, že ste veľká spoločnosť, ľahšie prehovoríte svojich partnerov, aby začali používať slobodný softvér. Ak ste však malá spoločnosť, musíte sa prispôsobiť, a keď partner povie, že chce dokument vo formáte `docx` vyrobený v MS Worde a nie v LibreOffice, lebo ten sa mu zle načítava, tak to tak musí byť.

Problém 5: Výrobcovia softvéru sú spolčení s Microsoftom. Ak od svojho dodávateľa softvéru požadujete, aby vám dodal softvér, ktorý beží v Linuxe, používa mySQL alebo Postrgres alebo dorobil export do LibreOffice cez OLE Automation, nepochodíte, lebo drvivá väčšina spoločností na Slovensku je zároveň partnerom Microsoftu. Na vývoj produktov používa `.NET` nástroje a Java je podľa nich len ostrov v Indonézii. Takže, ak si urobíte prieskum, zistíte, že na Slovensku neexistuje informačný systém bežiaci v Linuxe, ktorý by spĺňal vaše požiadavky.

3.3. Bariéra zo strany vedenia a zamestnancov

Problém 6: Nebudeme dokumenty posielať vo formáte `ODS`, aby sme nevyzerali tak, že si nemôžeme dovoliť kúpiť MS Office.

Problém 7: Prečo ten LibreOffice ukazuje, že tá čiara je prerušovaná, keď MS Office to neukazoval? Po podrobnom skúmaní zistíte, že aj v MS Office je čiara prerušovaná, len z nejakých príčin sa pri malom priblížení javí ako spojitá. Takže je zjavné, že tabuľku používateľ urobil zle. Použil viacero hrúbok čiar a navyše niektoré ani nie sú spojité. Tabuľku však ako šablónu mnohokrát použil, preto, ak od neho chceme, aby si to teraz všade opravil, vraždí nás pohľadom, lebo MS Office mu to vytlačil tak, ako potrebuje. Takéto drobné odlišnosti vo fungovaní programov v domácom prostredí zrejme prekážať nebudú, no vo firemnom prostredí to predstavuje veľký problém.

4. Riešenie problémov

Veľké spoločnosti majú k dispozícii nástroj menom Virtualizácia. Vo veľkej spoločnosti nie je problém virtualizovanie problémových aplikácií, ktoré si používatelia spustia v linuxovom prostredí. Majú tiež moc vyjednať spôsob komunikácie s obchodnými partnermi, ktorí sa ochotnejšie prispôsobia.

Malá spoločnosť je odkázaná na aplikovanie homogénneho Windows prostredia, v najlepšom prípade použije heterogénne prostredie, v ktorom kombinuje komerčný softvér so slobodným.

5. Prípadová štúdia

Uvažujme výrobnú spoločnosť, ktorá používa MS Windows XP, MS Office 2003, informačný systém, ktorého klient beží len na MS Windows a používa databázu MS SQL, dochádzkový systém bežiaci rovnako len v MS Windows nad MS SQL, a ktorá potrebuje vyriešiť problém ukončenia podpory Windows XP a Windows 2003 a Office 2003 v apríli 2014.

Cieľom je zabezpečiť prechod na IT infraštruktúru využívajúcu slobodný softvér. Najväčším problémom, s ktorým sa bude musieť spoločnosť vysporiadať je, čo s informačným systémom. Informačný systém môžeme vymeniť za iný, taký, čo je slobodný alebo taký, čo dokáže bežať v otvorenom systéme.

Jediný otvorený slobodný systém, ktorý som na slovenskom trhu našiel je Adampiere. Ako sa však ukázalo, spoločnosť, ktorá sa tvári, že systém podporuje, pozostáva z jedného programátora a občasnej študentskej výpomoci. Čo je pre nás nevyhovujúce, pretože potrebujeme operačne riešiť dovývoj v rôznych oblastiach a tu hrozí, že to nebude možné.

Druhá možnosť je zadovážiť si prorpietárny softvér schopný fungovať slobodnom prostredí Linuxu a databázami MySQL či PostgreSQL. No tu je výber tiež obmedzený. Jediný softvér, ktorý sa mi podarilo nájsť, nemal modul pre technické plánovanie výroby.

Tretia možnosť je prechod na proprietárny softvér bežiaci na Windows serveri, ktorý je však schopný bežať cez webový prehliadač. Tu sme však narazili na problém s cenou. Náklady na výmenu systému sú oveľa väčšie ako náklady na zakúpenie nových operačných systémov Windows. Navyše by výmena po funkčnej stránke spoločnosť nikam neposunula.

Zostala nám teda možnosť pokúsiť sa spustiť súčasný systém v slobodnom prostredí Linuxu. Informačný systém je možné spustiť cez Wine, Hurááá!!! No bohužiaľ nedá sa myšou klikať na ikony v oknách typu MDI child.

Jediným riešením nakoniec zostáva vzdialené spustenie aplikácie zo systému Windows prostredníctvom RemteApp. Toto riešenie ale vyžaduje Windows Server alebo riešenie formou služby SaaS. Po porovnaní ponúk od spoločností, ktoré takúto službu uvádzajú s cenou nového servera vrátane 6-ročnej prevádzky, sme

zistili, že vlastný Windows server je pre nás o dve tretiny lacnejší ako platba za službu za rovnaké obdobie.

Po preštudovaní licenčnej politiky spoločnosti Microsoft som dospel k poznaniu, že mi nestačia klientské licencie na báze strojov alebo užívateľov. V prípade použitia terminálového prístupu k serveru potrebujeme VDA licenciu, ktorá umožňuje pripojenie staníc bez Windowsu k virtualizovanej aplikácii cez RemoteApp. Táto licenicia stojí viac ako OEM licencia Windows na pracovných staniciach. Začať používať Linux v našej spoločnosti by teda vďaka tzv. Vendor lockingu bolo nákladnejšie ako nakúpiť nové stanice s OEM verziami operačného systému Windows.

Slobodný softvér však našťastie funguje aj v systéme Windows. Naša spoločnosť začala používať Firefox miesto prehliadača IE, a používatelia si ho chvália a nemajú žiadny problém. Kvôli spomínaným problémom s ActiveX prvkami však sú niekedy nútení použiť aj IE.

Miesto poštového programu Thuderbird s rozšíreniami Lighning, kvôli kalendáru, Printing tools kvôli rozlíšeniu používateľa pri tlači mailov na spoločnej tlačiarni, LookOut kvôli mailom odoslaných vo formáte TNEF z MS Outlooku a Right encoding, kvôli možnosti prepnúť kódovanie pri zle označených mailoch.

Program Thunderbird funguje bezchybne v prípade, že používame predvolené nastavenia. Pri prispôsobovaných nastaveniach Thunderbird občas z neznámeho dôvodu prejde do režimu offline alebo pri odosielaní mailu zamrzne a po zatvorení okna zostane visieť v pamäti. Tieto problémy sú málo časté, preto je veľmi obtiažne odhaliť, čo ich spôsobuje. Novoobjavujúcim sa problémom sú preposielané správy zapúzdrené v súbore s príponou .msg, ktorý pochádza z Outlooku 2013.

V spoločnosti používame kancelársky balík LibreOffice. Pri jeho používaní sa však stretávame s množstvom problémov. V našej spoločnosti kancelársky balík používame najmä na tvorenie podkladových materiálov pre vrcholový manažment firmy. Pri tejto činnosti tvoríme najmä tabuľky a grafy. Vzhľadom na požiadavky vrcholového manažmentu sú na tabuľky kladené rôzne požiadavky. Napr. aby boli niektoré bunky vyznačené hrubšou čiarou, iné tenšou, prípadne farebnou a podobne. Kompatibilita hrúbok čiar medzi MS Excelom a programom Calc je katastrofálna, preto väčšinu tabuliek používatelia museli prerobiť nanovo. Pracnosť pri tvorbe tabuliek v programe Calc je pritom omnoho vyššia ako v programe MS Excel.

Ďalší problém je export údajov z informačného systému. Informačný systém exportuje údaje do pekne vyzerajúcej tabuľky prostredníctvom OLE automation. Výrobca IS sa však vyhýba naimplementovaniu exportu aj do Libre Office. Z tohto dôvodu robíme export len do CSV, a potom ručne upravujeme vzhľad tabuliek.

Tretím problémom je kompatibilita výzoru dokumentov pre vrcholový manažment. Vrcholový manažment sa často zaoberá úpravou zmlúv. Kvôli vyťaženiu prácou je pre nich otravné zaoberať sa upravením "rozhádzaného"dokumentu, v ktorom časť pre podpis pretiekla na novú stranu, čísla v očíslovanom zozname

tancujú každé iným smerom a každé má inú veľkosť. Chýba text, ktorý je neviditeľný niekde za okrajom strany a podobne.

Toto sme teda riešili zakúpením niekoľkých licencií MS office pre vrcholový manažment firmy. Nanešťastie kompatibilita MS Office s formátom `ODF` je rovnako hrozná ako kompatibilita Libre Office s formátom `.doc` a `.docx` a často mi končia v počítači dokumenty na úpravu.

Spomedzi slobodného softvéru používame ešte PDF Creator na tvorbu digitálne podpísaných `pdf` dokumentov, prehrávač VLC, ktorý je naozaj výborným prehrávačom. Redakčný systém Drupal, v ktorom je vytvorená webová stránka spoločnosti, a na úpravu obrázkov, tvorbu koláží produktov využívame Gimp a Inkscape. Oba grafické programy fungujú veľmi dobre. Inkscape v systéme Windows má však problém pri exporte priehľadných objektov do `pdf`.

V budúcnosti zvažujeme nasadenie systému Alfresco na centrálny manažment dokumentov.

Záver

Ako vidieť z môjho textu najväčšou prekážkou v širšom používaní slobodného softvéru v predstavenej spoločnosti je Vendor locking zo strany dodávateľov informačných systémov, od ktorých sú spoločnosti ako my závislé. Druhým problémom je tiež, že na slovenskom trhu pôsobí veľmi málo spoločností, ktoré by vedeli operatívne riešiť problémy a reagovať na potreby vývoja funkcionalít.

Kontaktná adresa

Mgr. Peter Mráz, DRU a.s., Referát automatizovaného spracovania dát, Strážska cesta 8700/6, 960 01 Zvolen, Slovenská republika,
E-mailová adresa: `etkinator@gmail.com`

6. ročník Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf 2014: 17–22

OPEN HARDWARE — A COMPANION TO OPEN SOFTWARE

REMIGIUSZ OLEJNIK (PL)

Abstract. The article presents Open (Source) Hardware movement, as complementary to Open Source Software. The main part of the article provides an overview of the most important Open Hardware projects. It is supplemented by a list of the most commonly used Open Hardware licenses.

Key words and phrases. Open Hardware, Arduino, Open Source licenses.

Otvorený Hardvér — partner pre Otvorený Softvér

Abstrakt. Článok predstavuje Open (Source) Hardware hnutie, ako doplnok pre Open Source Software. Hlavná časť článku obsahuje prehľad najdôležitejších projektov Open Hardware. Je doplnená zoznamom najčastejšie používaných licencií vo svete Open Hardware.

Kľúčové slová. Open Hardware, Arduino, Open Source licencie.

Introduction

Open Source movement has dedicated and growing group of followers — developers and users — worldwide. The main idea of this movement, referred to the hardware aspects is the topic of this article. First part describes the essence and definition of Open Hardware. Second part of the article provides an overview of the most important Open Hardware projects. Third part is a list of the most common licenses used in the world of Open Hardware. The article concludes with a brief summary.

1. What is Open Hardware?

The **Open Hardware** (or *Open Source Hardware*) movement refers mainly to electronics or computer-related hardware, that has been developed on similar to Open Source Software basis. It is less popular than Open Source Software, mainly due to much higher requirements on knowledge and experience (than just simple programming).

As the oldest computer-related Open Hardware project we can assume the design of IBM PC XT/AT computer that allowed for “cloning” the architecture and had the biggest impact on later computing development.

Open Hardware design is open, free of charge and available to the public; it usually consists of schematics, PCB design, Bills of Material and in many cases also covers software: HDL source code, microcontrollers' firmware source code.

There are many definition of Open Hardware (OH). Two of them are recalled:

Definition 1.1. "Open Hardware is a logical evolution of the Free Software philosophy applied to physical stuff, where both code and design blueprints coexist. However no one can deny that humankind has evolved through a DIY and hack & share-it culture until patents prevented us to do so." [1]

Definition 1.2. Open Source Hardware (OSHW) Definition v1.0 [2]: "Open Source Hardware (OSHW) is a term for tangible artifacts — machines, devices, or other physical things — whose design has been released to the public in such a way that anyone can make, modify, distribute, and use those things."

Some Open Hardware projects use "official" Open Hardware logo [3], which is presented on Figure 1, however it is still optional and strongly depends on authors level of commitment to Open Hardware movement. The logo is usually placed on the PCB or chassis.

Figure 1. Open Hardware "official" logo

2. Overview of Open Hardware projects

This part is divided into three parts: Computing, Radio/Wireless and Machines that group Open Hardware projects. It is obvious that only selected and limited number of projects are presented here.

2.1. Computing

- **Arduino** [4] is a simple computing platform aimed at educational purposes. It is based on 8-bit Atmel AVR microcontroller (ATmega8, ATmega168, ATmega328, or ATmega1280) and I/O board for external connectivity. The platform can be extended by connecting external modules

(so called "shields"). The microcontroller is preloaded with bootloader thus it doesn't need external programmer and USB/serial connector is used to upload software to Arduino. Part of the Arduino project is Java-based software package (IDE) that implements the open source Processing / Wiring language. It allows for Arduino projects (called "sketches") writing, verifying, compiling and uploading. Arduino comes in many variations and originally was "born" in Italy, however at this time very popular are much cheaper Chinese exact "clones". Another clone of Arduino is Freeduino.

- **Beagle Board** [5] is a single-board computer build around low-power OMAP 3530 Texas Instruments processor that is using ARM Cortex-A8 core. It runs Ångström Linux operating system. The PCB design files are freely available, however assembling of the board is heavy due to it's complexity and usage of BGA processor.
- **OLinuXino** [6] is another single-board computer project consisting of four models: iMX233, A13, A10S and A20. First model is build of iMX233 ARM926J processor and is running Linux/Android OS. OLinuXino is designed by OLIMEX Ltd. in Bulgaria. All the PCB design files and software are available, and home-made instances of OLinuXino can be used even in commercial environment.
- **Open Graphics Project** [7] – aimed at creating of open architecture of graphics card standard, along with open source software to be run on them.
- **NetFPGA** [8] – project consisting of hardware, software and educational materials aimed at enabling research and education on network environment. It uses FPGA core so real network throughput is easily to manage, which is not possible using other approaches. The version NetFPGA-1G is based on Xilinx Virtex-II Pro 50 along with four Gigabit Ethernet RJ-45 interfaces.
- **CPUs** – there are some "open-source hardware" CPUs, typically implemented as a soft microprocessor on FPGA structure; the most important examples are:
  - **Amber** [9] – ARM-compatible 32-bit RISC processor; it implements the ARMv2 instruction set.
  - **OpenSPARC** [10] – open-source processor project including Sun Microsystems' UltraSPARC T1 and T2 multicore processor designs. UltraSPARC T1 is full 64-bit, 32-thread microprocessor, while T2 has 8 cores, 16 pipelines with 64 threads. The code is written in Verilog.
  - **OpenRISC** [12] – flagship project of the OpenCores community [11] aimed at developing of a very-high-performance open-source RISC CPU. First design is OpenRISC 1000, a family of 32 and 64-bit processors with optional floating point and vector processing support.

2.2. Radio/Wireless

- **TAPR Open High Performance Software Defined Radio** [13]
  - next generation open source software and hardware SDR project,
  - modular – composed of eight modules with common bus:
    - Atlas – backplane,
    - Ozy, Magister, Metis – communication,
    - Mercury – receiver,
    - Penelope, Pennylane – transmitter,
    - Pennywhistle, Munin – power amplifier,
    - Alex – receiving and transmitting filters,
    - Excalibur – time standard,
    - Janus – sound interface,
    - LPU – power supply,
  - the goal: high performance and modular universal SDR (Software Defined Radio) platform,
  - PCB design files and Verilog HDL software are freely available,
  - dedicated open source software also available, three operating systems supported: Windows, MacOS X, Linux.
- **N2ADR Software Definde Radio transceiver** [14]
  - small but powerful FPGA-based SDR transceiver,
  - open design of PCBs: Eagle source files,
  - open Verilog HDL source,
  - open dedicated control software – Quisk [15].
- **Openmoko** [16]
  - open source mobile phone framework,
  - Openmoko Linux operating system,
  - latest hardware version – Golden Delicious GTA04 board: ARM Cortex A8 @1 GHz, 512 MB RAM, WiFi, Bluetooth, UMTS, sensors (barometer, compass, gyroscope, accelerometer), GPS receiver, USB.
- **OpenBTS** [17]
  - open source software based GSM base station.
- **OsmoBTS** [18]
  - another open source software based GSM base station.
- **UmTRX** [19]
  - dual-channel wide-band SDR transceiver with 1GbE connection,
  - designed to be used as a transceiver for OpenBTS and OsmoBTS GSM base stations.

2.3. Machines

- **RepRap** [20]
  - universal 3D printer,

  - it can print most of its own components,
  - consists of a computer-controlled Cartesian XYZ platform made of steel rods and a thermoplastic extruder – heart of the RepRap – mounted on the platform,
  - electronics of RepRap are based on Arduino platform connected with special boards controlling stepper motors of RepRap,
  - the object are printed from ABS, Polylactic acid (PLA), Nylon (depends on the extruder), HDPE and other similar thermopolymers.
- **Lasersaur** [21]
  - laser cutter,
  - safe and highly-capable machine for makers, artist and scientist,
  - all the subsystems (mechanics, electronics, optics and laser, control software) are fully described and detailed Bill of Materials along with instructions are available on the project's website.

3. Licensing of Open Hardware projects

Many types of licenses are used in OH world. Most of them are identical to those used by Open Source Software. However there are some quite different applied only to hardware designs. Those "universal" license used by OH projects are:

- LGPL [22] used by some OpenCores [11] projects,
- Modified BSD License [23] used by some OpenCores [11] projects,
- MIT license [24] and GPL license [25] used by Open Graphics Project [7],

and specific hardware-oriented licenses are:

- TAPR Open Hardware License [26] used by all TAPR projects (and others),
- Balloon Open Hardware License [27] used by Balloon development board project [28],
- CERN Open Hardware License [30] used by Open Hardware Repository projects [31],
- Solderpad License [32], a version of the Apache License 2.0, adapted to hardware projects requirements.

Hardware licenses are generally different because they rather rely on patent law than on copyright law. Another comparison of OH licenses can be found in [33].

4. Conclusions

The article presented the idea and the most important Open (Source) Hardware projects. This movement develops dynamically, along with the increasing availability of low-cost hardware solutions and the popularity of DIY. We can hope that more new projects based on the idea of Open Hardware will appear in subsequent years.

References

[1] http://www.hfday.org/open-hardware.
[2] http://freedomdefined.org/OSHW.
[3] http://oshwlogo.com/.
[4] http://www.arduino.cc/.
[5] http://beagleboard.org/.
[6] https://www.olimex.com/Products/OLinuXino/.
[7] http://wiki.opengraphics.org/tiki-index.php.
[8] http://netfpga.org/.
[9] http://opencores.org/project,amber.
[10] http://opensparc.net/.
[11] http://opencores.org/.
[12] http://opencores.org/or1k/Main_Page.
[13] http://openhpsdr.org/.
[14] http://james.ahlstrom.name/transceiver/.
[15] http://james.ahlstrom.name/quisk/index.html.
[16] http://openmoko.org/.
[17] http://openbts.org/.
[18] http://openbsc.osmocom.org/trac/wiki/OsmoBTS.
[19] http://umtrx.org/.
[20] http://reprap.org/.
[21] http://www.lasersaur.com/.
[22] https://www.gnu.org/licenses/lgpl.html.
[23] http://opensource.org/licenses/BSD-2-Clause.
[24] http://opensource.org/licenses/MIT.
[25] http://opensource.org/licenses/gpl-license.
[26] http://www.tapr.org/ohl.html.
[27] http://www.balloonboard.org/docs/Balloon_License_0v2.pdf.
[28] http://www.balloonboard.org/.
[29] http://www.opencollector.org/hardlicense/hdpl.html.
[30] http://www.ohwr.org/documents/294.
[31] http://www.ohwr.org/.
[32] http://solderpad.org/licenses/.
[33] http://www.inmojo.com/licenses/.

Contact address

Dr inż. Remigiusz Olejnik, Department of Computer Architecture and Telecommunication, Faculty of Computer Science and Information Technology, West Pomeranian University of Technology, Szczecin, ul. Żołnierska 49, 71-210 Szczecin, Poland,
E-mail address: rolejnik@zut.edu.pl, http://rolejnik.zut.edu.pl

6. ročník Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSS Conf

OSSConf 2014: 23–30

MOŽNOSTI TVORBY DOKUMENTŮ V TEXU POMOCÍ WEBOVÉHO PROHLÍŽEČE

JAN PŘICHYSTAL (CZ)

Abstrakt. Systém TEX se v dnešní době používá pro tvorbu nejrůznějších typů dokumentů v nejrůznějších oborech lidské činnosti. Ne každý uživatel je však schopen nebo ochoten si tento systém nainstalovat a nakonfigurovat na svém vlastním počítači. Současně se navíc rozmáhá fenomén aplikací použitelných prostřednictvím webového prohlížeče. Tento přístup nabízí různé výhody a stejným směrem se snaží jít i webová aplikace TEXonWeb. Ta je určena především začínajícím uživatelům, kteří se s TeXem a jeho nadstavbami seznamují. Z tohoto důvodu aplikace nabízí různé průvodce, například pro tvorbu tabulek, nebo šablony různých běžných dokumentů či nástrojovou lištu umožňující snadný přístup k běžným značkám. Tento příspěvek popisuje aktuální možnosti TEXonWeb a nastiňuje směr, kterým se aplikace bude ubírat dále.

Klíčová slova. TEXonWeb, TEX/LATEX, HTML5, webová aplikace, cloudová úložiště.

POSSIBILITIES FOR CREATING TEX DOCUMENTS USING A WEB BROWSER

Abstract. The TEX is nowadays used for creating various types of documents in various fields of human activity. Not every user is able or willing to have this system installed and configured on his own computer. At the same time the phenomenon of applications usable through a Web browser propagates. This approach offers several advantages and the web application TEXonWeb is trying to go the same direction. It is intended primarily for novice users who are being acquainted with TEX and its friends. For this reason, the application offers various wizards such as creating tables or templates of various common document or a toolbar for easy access to common tags. This paper describes the current options of TEXonWeb and outlines the direction which the application will proceed further.

Keywords. TEXonWeb, TEX/LATEX, HTML5, web application, cloud storage.

Úvod

Podnětem pro vznik webové aplikace TEXonWeb byla především problematická instalace a konfigurace typografického systému TEX, která odrazovala začínající uživatele od práce s tímto velmi kvalitním systémem pro přípravu dokumentů. Nesčetněkrát se mi, jako vyučujícímu předmětu Zpracování textů na počítači, stalo, že mě studenti kontaktovali s dotazy na zprovoznění tohoto systému. K definitivní myšlence, že něco není v pořádku a mělo by se s tím něco dělat, mě přivedla situace, kdy si student donesl ke mě do kanceláře svůj osobní počítač (desktop v provedení tower), abych mu s instalací a konfigurací pomohl, protože

ji sám nezvládne. Tehdy ještě notebooky nebyly tak rozšířené a připojení k internetu nebylo samozřejmostí, jako dnes. Čili i přístup k informacím byl poněkud omezený. Podrobnější popis tehdejšího stavu a motivaci pro vznik TEXonWeb uvádí [1].

Zmíněný problém s instalací se projevoval pravidelně i při instalaci TEXu na učebnách na naší fakultě s každým novým semestrem. To, že by bylo možné TEX zpřístupnit pomocí webové aplikace bylo tedy jasným myšlenkovým pochodem a k důvodům se přidala i možnost přidání funkcí, které by začátečníkům (studentům) seznamování usnadnily. Protože aplikace TEXonWeb vznikla již poměrně dávno a nejedná se o komerční aplikaci, je její vývoj velmi pozvolný. V současné době již nevyhovuje požadavkům na moderní web a některé funkce nejsou dostupné ve všech aktuálně používaných webových prohlížečích, což uživatelům způsobuje problémy.

Současný stav

TEXonWeb byl od začátku vyvíjen jako webová aplikace, která má zpřístupnit typografický systém TEX a jeho nadstavbu LATEX začínajícím uživatelům. V současné době je intenzivně využíván studenty Mendelovy univerzity při výuce předmětu Zpracování textů na počítači, kde se studenti se základy tohoto systému seznamují. TEXonWeb využívají nejen pro řešení úkolů a závěrečného projektu, ale i pro tvorbu svých závěrečných bakalářských a magisterských prací. Dále je TEXonWeb využíván i některými akademickými pracovníky pro tvorbu posudků závěrečných prací. To vše umožňují nabízené šablony zjednodušující tvorbu standardních dokumentů. Mezi dalšími je k dispozici například šablona pro tvorbu životopisů, prezentací, žádostí, dopisů a dalších.

To, že je TEXonWeb zaměřen hlavně na začínající uživatele, je patrné i z nabízené nástrojové lišty, která jim umožňuje vkládat základní značky bez nutnosti jejich vyhledávání v příručce či na internetu. Pro komplikovanější struktury, jako jsou tabulky nebo seznamy, jsou k dispozici průvodci, kteří umožní prvek navrhnout interaktivním způsobem. Poměrně kladně je hodnocen právě návrhář tabulek, který studentům ekonomických oborů výrazně usnadňuje práci, protože zápis tabulek v LATEXu opravdu není nic jednoduchého a ekonomické texty se tabulkami jen hemží.

Dále je pro uživatele připraven nástroj na vložení nezlomitelných mezer za jednopísmenné předložky a spojky a také kontrola pravopisu. Tyto funkce však nejsou dostupné ve všech webových prohlížečích, čímž se dostáváme k problémům současného řešení.

Současná verze TEXonWeb je postavena na technologiích, které byly aktuální před několika lety. Nevyužívá možností HTML5 a framework jQuery, s jehož pomocí je vytvořeno uživatelské rozhraní, je ve staré verzi. Navíc z jQuery nejsou využívány všechny možnosti. Spousta funkcí na TEXonWeb je vytvořena vlastními

silami a je tak obtížně sloučitelná s novějšími verzemi frameworku. Jediné, v čem se nám daří držet krok, je verze TEXLive. Nyní běží TEXonWeb na verzi 2013 a přechod na každou další verzi je relativně snadná záležitost.

Problematickým místem je například editor. Jeho možnosti byly stále rozšiřovány tak, aby uživateli poskytoval co největší komfort při psaní zdrojového textu. To jsme se snažili zajistit vlastními funkcemi a v podstatě nevyužívali možnosti žádných Javascriptových frameworků. Editor je však z tohoto důvodu nyní plně funkční pouze ve webovém prohlížeči Mozilla Firefox. Jeden člověk starající se o vývoj této části prostě nestíhá sledovat změny ve všech hlavních prohlížečích. U ostatních prohlížečů jsme byli nuceni přistoupit k deaktivaci některých funkcí, protože prostě s danou implementací JavaScriptu nefungovaly správně. To je často pro uživatele frustrující, protože nemohou využívat svůj oblíbený webový prohlížeč. Z těchto důvodů jsme se rozhodli k masivním úpravám a v současné době přepisujeme prakticky celý kód TEXonWeb tak, aby respektoval naše požadavky a stavěl právě na oblíbených frameworcích, které nám kompatibilitu s prohlížeči (snad) zaručí.

Abychom udrželi krok s webovými kancelářskými aplikacemi, jako je například Google Drive, začlenili jsme do množiny nabízených funkcí i možnost verzování dokumentů. Uživatel tak má možnost vracet se k původní verzi před provedenými změnami, které se rozhodl nepoužít. Verzování je založeno na programu `svn` a každý uživatel má vytvořený vlastní repozitář, kde může verzovat TEXové dokumenty. Podrobně se verzování dokumentů na TEXonWeb věnuje práce [2].

Každý uživatel, který si na TEXonWeb vytvoří vlastní účet, má k dispozici i diskový prostor pro ukládání vlastních souborů. To je další výhoda, kterou webové aplikace nabízejí oproti těm lokálně instalovaným. Uživatel má vše na jednom místě, nemusí s sebou nosit všude přenosnou flash paměť a přitom má stále k dispozici svoje dokumenty. Samozřejmě pouze v případě připojení k internetu. Nicméně uživatelé jsou v dnešní době již zvyklí využívat pro ukládání svých souborů služeb velkých poskytovatelů cloudových služeb a soubory mají například na Dropboxu, Google Drive, iCloud, či dalších úložištích. Je pak pro ně nepříjemné ukládat některé soubory tam a jiné zase jinam. Proto jsme se po vzoru i jiných webových aplikací rozhodli uživatelů nabídnout možnost ukládat soubory přímo na tato úložiště. V současné době je přístupný prozatím jen Google Drive, ale protože se ukázalo, že je to správná cesta a prakticky bez vážnějších překážek, pracujeme na integraci i Dropboxu. Více informací o začlenění cloudových úložišť do TEXonWeb lze nalézt v [3].

Plánovaná rozšíření a úpravy

Rozhodli jsme se především, že již nadále nebudeme vyvíjet vlastní editor, ale využijeme nástroj od třetích stran. Konkrétně jsme se rozhodli pro editor Ace[1],

[1] `http://ace.c9.io/`.

který nabízí vedle zvýrazňování syntaxe i zobrazení párových závorek, číslování řádků, vyhledávání a nahrazování a další funkce běžné v editorech zdrojových kódů.

Mezi nové požadavky na TEXonWeb patří i možnost používání editoru na mobilních zařízeních (tablety, mobilní telefony). V současné době je pro tato zařízení uzpůsobeno velké množství aplikací, mezi kterými najdeme i různé kancelářské aplikace. Rozhodli jsme se, že nepůjdeme cestou samostatně instalované aplikace, ale využijeme možností HTML5 a upravíme design tak, aby byl použitelný i na zařízeních s dotykovým ovládáním.

Tento způsob ovládání však s sebou přináší některá úskalí, která na běžném počítači nejsou tak zřetelná. Mezi ty největší problémy patří efektivní zápis zdrojového kódu pomocí značek. Na většině klávesnic na tabletech je poměrně zdlouhavé dostat se k speciálním symbolům, jako jsou zpětné lomítko či složené závorky, a to uživatele opravdu zdržuje. Nabízí se tedy možnost poskytnout uživateli alternativní klávesnici, která tyto symboly bude obsahovat v základním rozložení, a tu si uživatel nainstaluje na svoje dotykové zařízení. Bohužel tato možnost byla ještě donedávna nerealizovatelná na zařízeních od firmy Apple. Ta neumožňují instalaci jiných klávesnic. Podle posledních zpráv by se však tato situace s příchodem nového iOS 8 mohla změnit. Další možností je nabídnout uživateli množinu speciálních symbolů formou lišty integrované do prostředí editoru. Jednak má uživatel jistotu, že bude dostupná na všech zařízeních a také si ji případně může upravit tak, aby vyhovovala přímo jemu. Vybere si pouze ty symboly, které potřebuje a uspořádá si je podle vlastní potřeby. Tuto možnost jsme se rozhodli podporovat i na TEXonWeb.

S použitím TEXonWeb na tabletech a mobilních telefonech souvisí i změna uživatelského rozhraní. To, co uživateli vyhovuje na desktopu nebo notebooku, již na dotykovém zařízení vyhovovat nemusí. Jde především o rozmístění ovládacích prvků, chování editoru zdrojového kódu při editaci, množinu nabízených funkcí, atd. V současné době probíhá vývoj v této oblasti a spolu s odborníky na uživatelská rozhraní posuzujeme a hledáme vhodné řešení, které nabídne uživateli komfortní ovládání i na mobilních zařízeních. To vyžaduje přechod na HTML5 a nejnovější verzi jQuery. Nyní jsme zrovna ve fázi přechodu, přepisujeme staré funkce a rozhraní na nové. Daří se nám odhalovat i spousty zbytečných věcí, které se nabalily během let vývoje. Optimalizace kódu přispěje k vyšší rychlosti webových aplikací. Optimalizujeme i kaskádové styly, ve kterých je také poměrně nepořádek a spousta vlastností je několikrát za sebou v kódu znovu a znovu předefinována, což nepřispívá ani k čitelnosti kódu ani k rychlosti vykreslování stránky.

Pokračovat samozřejmě hodláme i v integraci cloudových služeb. Tak, jak v současné době podporujeme ukládání souborů na Google Drive, hodláme přidat v dohledné době i Dropbox a následně, pokud bude zájem i další úložiště. Jediným problémem, který nám může bránit v realizaci je neexistující rozhraní pro přístup

k dané službě, případně jeho komplikované použití v prostředí TEXonWeb, který je psán v programovacím jazyce Perl.

V souvislosti s tím plánujeme integrovat i možnost spolupráce více uživatelů na jednom dokumentu. Chtěli bychom se přiblížit stavu, jaký je znám například z Google Drive. Uživatelé mohou v reálném čase sledovat změny, které provedli spolupracovníci, přijímat je či zamítat a přidávat k nim komentáře. Zajímavým rysem této funkce je možnost využití sdílení i ve výuce, kdy učitel zpřístupní jeden dokument všem studentům v učebně a vysvětluje konkrétní problematiku. Student vidí změny, použité značky i postup úprav přímo na svém počítači a případně si vše může snadno a rychle zkopírovat do svého dokumentu. Na první pohled se může zdát, že oproti klasické výuce s využitím tabule zde není velká výhoda. Z vlastních zkušeností však musím říct, že čitelnost na tabuli či promítacím plátně není vždy nejlepší a u TEXového kódu často záleží opravdu na drobnostech – jestli jde o klasický nebo zpětný apostrof, například.

Vedle těchto zásadních změn plánujeme i drobné úpravy, které však mohou být pro uživatele velmi zajímavé. Chceme například přímo zobrazovat informace o případných chybách při překladu v samostatném okně tak, aby uživatel nemusel vždy otevírat logový soubor, jak tomu bylo doposud. Idea je přiblížit se chování běžného editoru.

Srovnání s alternativami

V současné době je samozřejmě k dispozici více aplikací umožňujících využívat systém TEX prostřednictvím webového prohlížeče. Mezi asi ty nejrozšířenější patří shareLATEX[2] a writeLATEX[3]. Jisté pokusy v tomto směru podnikl i Google v rámci googeLabs, kde je možné narazit na LaTeXlab. Všechny tyto aplikace nabízí v podstatě to stejné a liší se v drobnostech. Pokusme se shrnout jejich základní vlastnosti a srovnat je. Zaměříme se na aplikace shareLaTeX a writeLATEX, které jsou v dnešní době jedny z nejpoužívanějších.

Obě tyto aplikace jsou zaměřeny především na uživatele z akademické oblasti. Přímo na jejich stránkách je uvedeno, že je využívají lidé z mnoha univerzit pro tvorbu vědeckých prací. To pravděpodobně určuje i vlastnosti a vzhled jejich uživatelských rozhraní. Na rozdíl od TEXonWeb se předpokládá, že je budou využívat již informovaní uživatelé a tudíž například nástrojová lišta obsahující zkratky používaných značek nebo průvodci tvorbou tabulek či seznamů, jsou zde zbytečné. Na druhou stranu jsou zde k dispozici šablony typických dokumentů, které mohou uživatelé využít. Předpokládá se zde také, že tyto aplikace budou využívat vědečtí pracovníci pro tvorbu článků a ty jsou často psány kolektivem autorů. Takže podpora spolupráce je zde na vysoké úrovni. Tyto aplikace umožňují tvorbu projektů

[2]Dostupné na `https://www.sharelatex.com/`.

[3]Dostupné na `https://www.writelatex.com/`.

a sdílení jejich částí mezi více uživatelů, spolupráci na textu i formou komentářů a sledování verzí či slučování dokumentů.

Zajímavé je srovnání toho, s čím uživatel pracuje nejvíce a co ovlivňuje i efektivitu jeho práce – editor zdrojového kódu. Je jasné, že webové editory prozatím nedosahují kvalit těch lokálně instalovaných, ale svými funkcemi se jim již dost blíží. V dnešní době je samozřejmostí zvýrazňování syntaxe a označování párových závorek. Tyto funkce nabízejí všechny tři editory. U editoru writeLaTeX mě zaujala možnost přepnout editor tak, aby se choval jako editory Emacs či vim, možnost změny barevného schématu editoru či doplňování kódu (autocomplete). Tento editor nabízí i funkci Richtext, která přepne zobrazení zdrojového kódu do formy WYSIWYG a uživatel má rovnou představu toho, jak bude vypadat výsledný text. Autoři tohoto editoru avizují i možnost použití na mobilních zařízeních. Podle mého názoru je to ale diskutabilní a to především z důvodů uvedených výše. WriteLaTeX totiž nijak neodlišuje uživatelské rozhraní na desktopu a na tabletech a například zápis značek je opravdu komplikovaný. Editory writeLaTeX a TeXonWeb nabízejí i funkci kontroly pravopisu. Editor TeXonWeb nabízí navíc průvodce (wizardy) pro tvorbu komplikovaných částí textu, jako jsou tabulky, obrázky, nebo seznamy.

Oba konkurenční editory nabízejí přímý náhled dokumentu. Jde o funkci, kdy je zdrojový kód okamžitě při každé změně rovnou překládán a zobrazován v samostatném okně. Je to jistě komfortní funkce, která má nahradit neustálé klikání na tlačítko překladu. Podle mého názoru se však hodí pouze pro kratší dokumenty. Přesvědčil jsem se o tom sám, když i relativně krátký dokument (jednotky stran) není zobrazován ihned, ale po zhruba několika vteřinové prodlevě. Zbytečně to zatěžuje přenosovou linku a odvádí pozornost. Uživatel přeci nepotřebuje v každou chvíli vidět aktuální stav dokumentu, navíc úplně celého. Pokud už je tato funkce implementována, stačilo by překládat pouze část. To však s sebou přináší další komplikace a je otázkou, zda by to vůbec přispělo ke zrychlení zobrazení.

Aplikace shareLaTeXi writeLaTeX nabízejí bezplatné používání, ale pouze do určité míry. Při bezplatném přístupu je například u shareLaTeX omezen počet spolupracovníků nebo u writeLaTeX funkce kontroly pravopisu. Tato omezení se však jistě během doby mění. Jejich aktuální přehled je dostupný na domovských stránkách aplikací.

Některá důležité vlastnosti shrnuje tabulka 1.

Dá se v podstatě říci, že nejpokročilejším editorem z hodnocených, který nabízí online možnost tvorby dokumentů pomocí systému TeX či některé z jeho nadstaveb je writeLaTeX. Nabízí více funkcí než ostatní a jeho slabinou je prakticky jen jeden podporovaný překladač, což asi ale většině uživatelů stejně nevadí. Nicméně, některé funkce a vlastnosti, které konkurenční editory nabízí jsou placené. Studenti i pracovníci Mendelovy univerzity určitě ocení lokalizované uživatelské rozhraní či specializované šablony dokumentů a další funkce, které jsou zaměřené

Tabulka 1. Důležité vlastnosti shareLATEX, writeLATEX a TEXonWeb

	shareLATEX	**writeLATEX**	**TEXonWeb**
překladače	latex, pdflatex, xelatex, lualatex	pdflatex	tex, xetex, csplain, cslatex, pdflatex, xelatex
šablony	ano	ano	ano
kvalitní editor	ne	ano	ne
přímý náhled	ano	ano	ne
zvýraznění syntaxe	ano	ano	ano
párové závorky	ano	ano	ne
nástrojová lišta	ne	ano (omezená)	ano
návrháři kódu	ne	ne	ano
kontrola pravopisu	ne	ano (placená)	ano
spolupráce	ano (placená)	ano	ne
verzování	ano (placené)	ano (placené)	ano
ukládání	Dropbox (placené)	Dropbox, GitHub, Google Drive	Google Drive
mobilní zařízení	ne	ano	ano
cena	zdarma (placené funkce)	zdarma (placené funkce)	zdarma

na začínající uživatele. Nevýhodou aplikace TEXonWeb je zastaralý design uživatelského rozhraní, který nejenže může být méně komfortní na použití, ale může i odradit případného zájemce.

Závěr

Tento článek shrnuje současný stav webové aplikace TEXonWeb a věnuje se popisu nových funkcí a vlastností, které se chystáme zpřístupnit v tomto roce. Při hledání nápadů a stanovení priorit jejich realizace vycházíme z vlastních idejí, z připomínek uživatelů, ale inspirujeme se i u konkurenčních projektů jako jsou shareLATEXa writeLATEX.

Přestože jsou pro zájemce o tvorbu kvalitních dokumentů pomocí systému TEX k dispozici různé editory nelze říci, že by další vývoj TEXonWeb ztrácel smysl. Dosažení podobné kvality, jako nabízí konkurenční editory, nebude velký problém. Srovnání s ostatními nám přináší zajímavou inspiraci, kterou můžeme dále rozvést a například v oblasti podpory mobilních zařízení dotáhnout dále.

Poděkování. Tento článek vznikl s podporou projektu IGA PEF Mendelu 7/2014 – Inovace webového rozhraní pro tvorbu dokumentů.

Reference

[1] PŘICHYSTAL, J., RYBIČKA, J.: *Webové rozhraní pro sazbu dokumentů*, Zpravodaj CSTUG. 2004. sv. 14, č. 3, s. 190–194. ISSN 1211-6661.

[2] TELENSKÝ, V., PŘICHYSTAL, J.: *Verzování dokumentů v aplikaci TEXonWeb.* In PEFnet 2012. 1. vyd. Brno: Mendel University in Brno, 2012, s. 72–73. ISBN 978-80-7375-669-7.
[3] TELENSKÝ, V., PŘICHYSTAL, J.: *Možnost propojení cloudového úložiště Google Disk s aplikací TEXonWeb.* In PEFnet 2013. 1. vyd. Brno: MENDELU Publishing centre, 2013, s. 72–77. ISBN 978-80-7375-906-3.

Kontaktní adresa

Ing. Jan Přichystal, Ph.D., Provozně ekonomická fakulta, Mendelova univerzita v Brně, Zemědělská 1, 613 00 Brno, Česká Republika,
E-mailová adresa: `jan.prichystal@mendelu.cz`

RECENZOVANÉ PRÍSPEVKY

6. ročník Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf

OSSConf 2014: 33–40

CLAROLINE – JEDNODUCHÝ E-LEARNING V PRAXI

ĽUBOŠ BALÁŽOVIČ (SK)

Abstrakt. Článok sa zameriava na praktické využitie systému Claroline na druhom stupni základnej školy. Prvá časť stručne opisuje systém Claroline, jeho históriu a vlastnosti. Ďalšia časť rozoberá funkcionalitu užitočnú pre učiteľov základných škôl. Načrtáva jednoduché pracovné postupy od vytvorenia používateľského konta až po vytvorenie nových predmetov (použitie hlavnej stránky, popis predmetu, ukladanie dokumentov do prostredia, zadávanie a vyhodnocovanie cvičení a zadaní). Posledná tretia časť pojednáva o skúsenostiach z používania od žiakov a učiteľov získaných cez dotazníkový prieskum, z ktorých vyplynulo prijatie systému aj zo strany žiakov (a ich rodičov).

Kľúčové slová. E-learning, Claroline, vzdelávanie na základných školách.

CLAROLINE – EASY E-LEARNING SYSTEM

Abstract. This paper focuses on the application of Claroline courses and learning management system in lower Secondary Education. First part is briefly describing Claroline, its history and features. Next part elaborates related features useful for teachers at Primary Schools. Workflows from teacher's view guides form User account creation, thru creating new course (using main page, course description, storing and sharing documents and links, evaluating exercises and assignments). Final third part of the paper evaluating experiences and opinions of pupils regarding usability of the Claroline LMS. Data was collected in survey (questionary in electronic form) proves the e-learning environment was gradually accepted by pupils (and later also by their parents).

Keywords. E-learning, Claroline, primary school education.

Úvod

E-learning postupne prichádza aj do škôl na Slovensku. Okrem viacerých proprietárnych riešení (napr. na slovenských školách známy Edupage[1]) sa využíva aj niekoľko systémov so slobodnou GNU GPL licenciou. Množstvo projektov pre stredné školy a tiež viaceré univerzitné projekty (projekt *Virtuálna univerzita* na Univerzite Mateja Bela v Banskej Bystrici, e-learningový portál Univerzity Komenského v Bratislave) využívajú systém Moodle[2]. Moodle však nie je jediné možné otvorené riešenie, v našom článku predstavíme v určitých podmienkach lepšie využiteľnú alternatívu – e-learningový systém Claroline.

[1] `http://edupage.sk`.
[2] `http://www.moodle.org`.

Pri testovaní systému sme skúmali nasledujúce vlastnosti, ktoré ovplyvňujú jeho použiteľnosť v praxi:

- architektúra (štruktúra) systému,
- rýchlosť vývoja (implementácia inovácií),
- prístupnosť systému,
- úroveň spracovania dokumentácie k systému.

V zmysle týchto kritérií systém porovnáme s e-learningovým systémom Moodle.

1. Základné vlastnosti systému Claroline

E-learningový systém Claroline vznikol v roku 2001 na Catholic University of Louvain v Belgicku. V súčasnosti ho využívajú vzdelávacie inštitúcie vo viac ako 100 krajinách sveta [1]. Po technickej stránke je to jednoduchá klient-server aplikácia – serverová časť je vytvorená v PHP s využitím MySQL databázy, klientská využíva takmer čisté HTML s malým množstvom Javascriptu. Celá administrácia systému po jeho nainštalovaní sa realizuje cez webové rozhranie. študent.

1.1. Architektúra systému (komplexnosť vs. jednoduchosť)

Aj pri LMS (learning management system) softvéri väčšinou platí, že s každou ďalšou verziou rastie množstvo funkcií. Na jednej strane sa tým rozširuje variabilita použitia, avšak z hľadiska praktického nasadenia na jeden konkrétny účel tento trend systém zbytočne zneprehľadňuje a komplikuje. Z pohľadu učiteľa základnej školy je dobré, ak systém plní základné funkcie, ktoré od neho požadujeme a nesnaží sa byť za každú cenu univerzálny a plne modulárny.

Systém Claroline je v svojej aktuálnej verzii oveľa jednoduchší ako systém Moodle. Obsahuje menší počet typov vzdelávacích objektov, menší počet typov otázok v testoch a celkovo je menej konfigurovateľný. Práve pre svoju jednoduchosť v porovnaní s komplexnosťou Moodle je pre podmienky nasadenia na základnej škole vhodnejší.

1.2. Rýchlosť vývoja (konzervatívnosť vs. inovatívnosť)

Ďalšou vlastnosťou, s ktorou sa stretávame v diskusiách na rôznych fórach je kritika pomalého vývoja systému Claroline – za ostatných 10 rokov sa po obsahovej stránke zmenil len nepatrne – najväčšie zmeny boli v pozadí – čistenie kódu tak, aby generoval validné HTML dokumenty a aby bol kompatibilný s vyššími verziami PHP a MySQL.

Samozrejme inovácie nemožno zastaviť – no mnohé trendy rovnako rýchlo prídu ako odídu a systém, ktorý ich verne kopíruje nemusí byť z praktického hľadiska tak dobre použiteľný ako systém, ktorý je vo vývoji konzervatívnejší. Pri rýchlo vyvíjajúcom sa systéme sú bežní užívatelia nútení neustále študovať zmeny, ktoré nastali v užívateľskom rozhraní, či správaní sa systému navonok. Konzervatívnejší

systém dovoľuje užívateľom po prvotnom zvládnutí ovládania plne sa sústrediť na vytváraný obsah – v prípade LMS systému na vzdelávacie objekty. Nové verzie Claroline vychádzajú približne raz za rok až dva roky, kým Moodle je vydávaný dvakrát ročne [2].

1.3. Prístupnosť systému

Systém je prístupný vtedy, keď medzi jeho používateľom a systémom nie sú žiadne bariéry. Bariérou môže byť komplikovaná inštalácia, úzke previazanie technológie na špecifický hardvér, či jazyk užívateľského rozhrania.

V súčasnosti je už 8. rok prevádzkovaný a pre vzdelávacie účely voľne dostupný na jednom zo serverov Katedry didaktiky prírodných vied, psychológie a pedagogiky UK Prírodovedeckej fakulty, na webovej adrese `http://www.virtual-lab.sk/claroline/` – je tak plne prístupný ľubovoľnému učiteľovi na Slovensku bez nutnosti inštalácie systému na školský server. Obdobný projekt pre systém Moodle v súčasnosti neexistuje.

V máji 2013 bola dokončená kompletná lokalizácia do slovenského jazyka. Pri preklade výrazov sme venovali zvláštnu pozornosť tomu, aby použitá terminológia bola v zhode so zaužívanými výrazmi školskej praxe. Učiteľ tak vytvára „predmety“ nie „kurzy“, žiak „neodosiela“ test ale „odovzdáva“. Prejsť všetkými úskaliami lokalizácie však presahuje tému tohto článku, hoci problematika by stála za odbornú diskusiu. Claroline je pre učiteľa a žiakov prístupný aj z hľadiska jazyka. Moodle je tiež kompletne lokalizovaný, avšak použitá terminológia je vhodná skôr pre použitie na vysokých školách.

1.4. Dokumentácia

V rokoch 2011-2013 sme vytvorili ešte v rámci projektu *Modernizácia vzdelávacieho procesu na základných a stredných školách* kompletnú videodokumentáciu. Všetky vytvorené videonávody sú prístupné cez prezentáciu na `http://prezi.com/vybekdqzxjzd/e-learnigovy-system-claroline/`. Všetky postupy k jednotlivým činnostiam majú nahovorený komentár v slovenskom jazyku.

2. Claroline v práci učiteľa

Efektívne a zmysluplné využívanie digitálnych technológií v procese učenia a učenia sa je podmienené záujmom učiteľa meniť zaužívané (štandardné) formy prípravy, riadenia a organizácie vyučovacej hodiny.

Učiteľ môže e-learningové prostredie Claroline (ale aj väčšinu ďalších) využiť najme v týchto oblastiach:

- zverejnenie základných informácií o predmete, ako sú obsahová náplň predmetu, hodnotenie, odporúčaná (rozširujúca) literatúra,
- zverejňovanie materiálov pre žiakov/študentov,
- interaktívne pracovné zošity,

- testy pre žiakov,
- komunikácia so žiakmi,
- kooperatívne vyučovanie (tvorba a zdieľanie dokumentov),
- zadania a domáce úlohy.

V nasledujúcej časti predstavíme tieto oblasti podrobnejšie.

2.1. Hlavná stránka

Každý používateľ systému ako učiteľ, tak aj žiak musí mať vytvorené užívateľské konto. Žiacke kontá možno vytvárať aj automatizovane. Učiteľské konto má oproti žiackemu možnosť vytvárať nové predmety.

Po vytvorení konta pre učiteľa a žiakov nasleduje vytvorenie hlavnej stránky predmetu a doplnenie základných informácií.

Základom e-learningového prostredia je úvodná stránka predmetu (obr. 1), ktorá podáva základné informácie o predmete, jeho obsahu a aktivitách. Z nej je prostredníctvom panela v ľavej časti možný prístup k výukovým materiálom, testom, zadaniam a ďalším nástrojom, ktoré tvoria komunikačné rozhranie medzi žiakmi a učiteľmi (prípadne aj rodičmi). Hlavná stránka predmetu sa skladá z viacerých častí, pričom užívateľ môže priamo ovplyvniť zoznam nástrojov (modulov), v ľavej časti aj hlavnú plochu.

Hlavná plocha predmetu zvyčajne tvorí motivačnú časť projektu. Je vhodné umiestniť naň motivačný obrázok a pár informácií, ktoré môžu byť dôležité pri každom prístupe na stránku predmetu (napríklad kontakt na vyučujúceho, informácia o tom, čo všetko je súčasťou elektronickej verzie predmetu, atď.). Práca s vstavaným editorom obsahu je jednoduchá a pripomína prácu s bežným texovým procesrorom ako je MS Word alebo OpenOffice Writer. Okrem rôznych písiem, štýlov a farieb, vkladanie obrázkov a internetových odkazov umožňuje aj vkladanie videí (zo služieb Youtube[3], Vimeo[4] a iných).

Panel nástrojov v ľavej časti možno upraviť kliknutím na položku vypnutím nepotrebných modulov (cez položku „Upraviť zoznam nástrojov“), čím sa stránka, hlavne pre začínajúcich používateľov sprehľadní. Odporúčame vypnúť všetky moduly s výnimkou tých, ktoré plánujete skutočne využívať (základná užitočná zostava pozostáva z modulov *Popis predmetu*, *Cvičenia*, *Dokumenty a odkazy* a *Zadania*.

2.2. Popis predmetu

V module *Popis predmetu* môžeme uložiť základné informácie o predmete ako je spôsob hodnotenia, ciele predmetu, náplň jednotlivých hodín. Ideálne je si do tejto časti uložiť aj tematické výchovnovzdelávacie plány (obsahové a výkonové štandardy) a výňatok školského vzdelávacieho programu pre náš predmet. Claroline

[3] `http://www.youtube.com`.

[4] `http://www.vimeo.com`.

Obr. 1. Titulná stránka predmetu v systéme Claroline

nám umožňuje vypnúť viditeľnosť vybraných informácií a tak tieto dokumenty môžeme nastaviť tak, aby ich videl len učiteľ. Takýmto spôsobom dokáže učiteľ sústrediť všetky dôležité informácie o predmete na jedno miesto, ktoré bude mať kedykoľvek prístupné z ktoréhokoľvek počítača s pripojením na internet.

2.3. Dokumenty a odkazy

Modul *Dokumenty a odkazy* slúži ako virtuálny disk, na ktorý možno uložiť akékoľvek súbory. Tie budú prístupné žiakom, pokiaľ ich „neznevideľníme“. V praxi sa ukázalo nahratie rôznych obrázkov, ktoré boli využité v rámci expozičnej alebo motivačnej fázy hodiny, ako veľmi užitočné a praktické. Modul umožňuje tiež vytváranie internetových odkazov. Tak boli na stránku predmetu (v časti „Dokumenty a odkazy“) pridávané priebežne aj odkazy na zaujímavé stránky rozširujúce základné učivo, prípadne použiteľné ako zdroj informácií pri tvorbe žiackych projektov.

2.4. Cvičenia

Modul *Cvičenia* nám umožňuje vytvárať automaticky vyhodnocované interaktívne cvičenia.

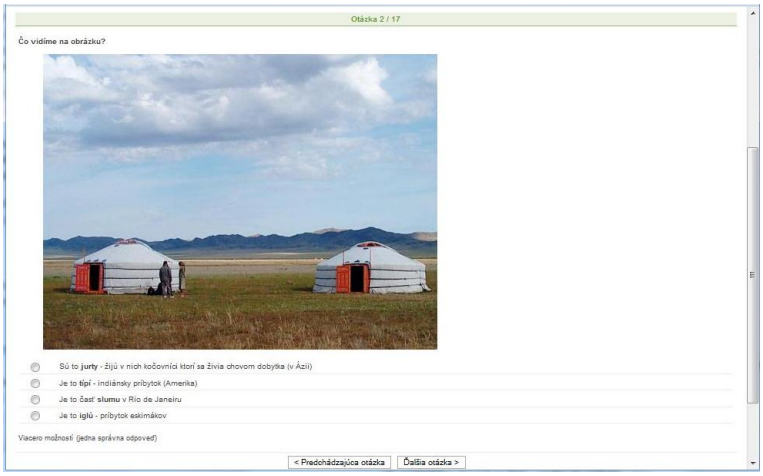
Každé cvičenie môže obsahovať ľubovoľný počet otázok (t.j. jedno cvičenie zodpovedá jednému testu). Claroline rozlišuje 5 rôznych formátov otázok. Ide o otázky:

- s výberom jednej správnej odpovede,
- otázky v ktorých je možné vybrať súčasne viacero odpovedí,
- otázky pri ktorých sú na výber len odpovede áno a nie (pravda – nepravda),

- otázka typu „doplňovačka“
- a otázka typu „priraďovačka“.

Každej vytvorenej otázke je vhodné určiť „Váhu“ v podobe bodového hodnotenia, a aby systém vedel odpovede správne vyhodnotiť zaškrtnúť správnu (očakávanú) odpoveď.

Otázky (alebo odpovede) možno obohatiť obrázkami (fotografie, tabuľky, mapy, diagramy, kartogramy, kartodiagramy a pod. – obr. 3).

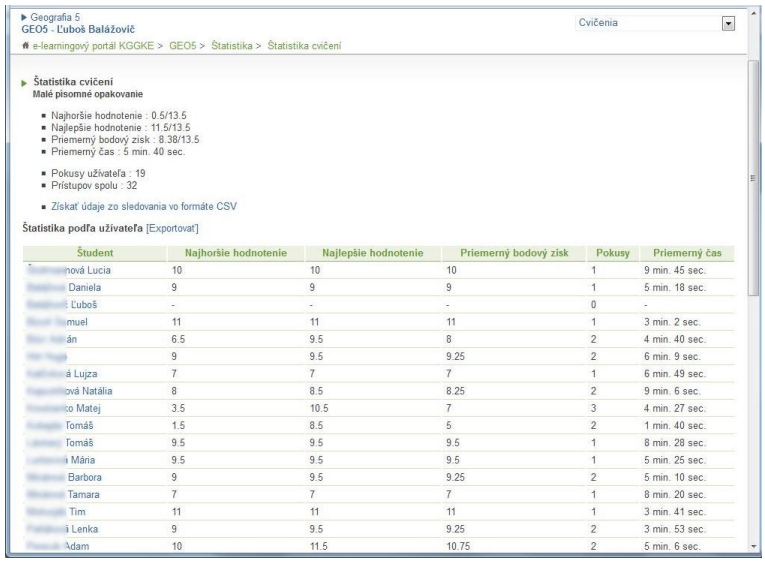


Obr. 2. Príklad testovej otázky s využitím obrázku

Nemenej dôležitou úlohou po vytvorení testu je jeho vyhodnotenie. Po tom čo žiaci test vyplnia, je automaticky vyhodnotený. Žiak má tak okamžitú spätnú väzbu a vie presne, ktoré otázky nevypracoval správne. Prístup k výsledkom má aj učiteľ, ktorý na rozdiel od jednotlivých žiakov vidí po kliknutí na ikonu „štatistika“ výsledky všetkých žiakov (v prípade opakovaných pokusov aj najlepšie, najhoršie a priemerné hodnotenie – obr. 3). Štatistika neobsahuje len vyhodnotenie z pohľadu žiakov, ale aj z pohľadu úspešnosti riešenia otázok. Učiteľ tým získa výborný prehľad, ktoré otázky robili žiakom najväčšie problémy a ktoré naopak patrili medzi tie ľahšie (resp. mali najvyššiu úspešnosť).

2.5. Zadania

Modul zadania je vhodný predovšetkým na zadávanie dlhodobejších úloh (napr. žiackych projektov). Žiaci odovzdávajú vypracovanú úlohu priamo cez prostredie Claroline, podobne učiteľ udeľuje hodnotenie, s prípadným komentárom taktiež cez tento systém. Pri zadaní je možné špecifikovať termín odovzdania a povinnú časť (napríklad súbor). Teoreticky je možné takýmto spôsobom zadávať žiakom akúkoľvek domácu úlohu.



Študent	Najhoršie hodnotenie	Najlepšie hodnotenie	Priemerný bodový zisk	Pokusy	Priemerný čas
nová Lucia	10	10	10	1	9 min. 45 sec.
Daniela	9	9	9	1	5 min. 18 sec.
Ľuboš	-	-	-	0	-
muel	11	11	11	1	3 min. 2 sec.
án	6.5	9.5	8	2	4 min. 40 sec.
	9	9.5	9.25	2	6 min. 9 sec.
á Lujza	7	7	7	1	6 min. 49 sec.
ová Natália	8	8.5	8.25	2	9 min. 6 sec.
o Matej	3.5	10.5	7	3	4 min. 27 sec.
Tomáš	1.5	8.5	5	2	1 min. 40 sec.
Tomáš	9.5	9.5	9.5	1	8 min. 28 sec.
Mária	9.5	9.5	9.5	1	5 min. 25 sec.
Barbora	9	9.5	9.25	2	5 min. 10 sec.
Tamara	7	7	7	1	8 min. 20 sec.
Tim	11	11	11	1	3 min. 41 sec.
á Lenka	9	9.5	9.25	2	3 min. 53 sec.
Adam	10	11.5	10.75	2	5 min. 6 sec.

Obr. 3. Hodnotenie cvičení (testov) v systéme Claroline

Prakticky sa však ukázalo efektívnejšie využívať tento modul len v prípade dlhodobých úloh určených napríklad na celý polrok.

3. Pilotné overenie v praxi

Aby sme zistili názory žiakov na využiteľnosť LMS Claroline v nižšom sekundárnom vzdelávaní, systém bol počas jedného roka pilotne nasadený na základnej škole v Banskej Bystrici v predmetoch Geografia a Informatika (5. a 6.ročník) v školskom roku 2011/2012. Na e-learningové prostredie si žiaci (a neskôr aj rodičia) postupne zvykli a pri záverečnom opakovaní si vyslovene žiadali test administrovaný prostredníctvom Claroline.

Výskum bol realizovaný prostredníctvom dotazníka vlastnej konštrukcie a metodiku a podrobné vyhodnotenie možno nájsť v článku [3].

Zúčastnilo sa ho 31 žiakov, čo tvorí 88 % všetkých, ktorí absolvovali vyučovanie s podporou systému Claroline (1 žiak chýbal a traja žiaci prestúpili na inú školu). Prevažovali žiaci siedmeho ročníka 65 % a v oboch triedach výraznejšie dievčatá (tvorili až 74 % respondentov).

Veľkú výpovednú hodnotu majú aj niektoré postrehy žiakov získané z otvorených odpovedí, v ktorých ocenili hlavne prehľadnosť systému a okamžitú spätnú väzbu. Pozitívne vnímali aj on-line komentáre učiteľa k žiackym projektom.

Zo zistených výsledkov vyplýva, že 1/3 opýtaných žiakov sa lepšie učilo s prostredím Claroline. O niečo menšej časti sa lepšie učí bez Claroline. Pomerne

veľká časť uviedla (29 %), že im je to jedno a nevidia medzi tým zásadný rozdiel. Pri písaní testov však až 2/3 žiakov uprednostňuje elektronické testovanie pred písomným. Táto skutočnosť poukázala na pravdepodobne najväčší potenciál e-learningového prostredia, z pohľadu možností jeho aplikácie do pedagogickej praxe. Len 13 percent žiakov vnímalo ovládanie e-learningového systému ako zložité. Potvrdilo sa, že už aj žiaci 5. a 6.ročníka berú moderné technológie a ich ovládanie veľmi prirodzene a nemajú väčšie problémy pri ich používaní.

Z prieskumu tiež vyplýva, že až 1/5 všetkých žiakov používala Claroline aj pri príprave na vyučovanie (t.j. dobrovoľne ako doplnkový študijný materiál učebnice).

Podobný prieskum realizovala Čipková (2013), pričom konštatovala, že „ Na základe analýzy jednotlivých položiek dotazníka prvého a druhého ročníka študentov študujúcich magisterský stupeň v odbore učiteľstvo všeobecnovzdelávacích predmetov môžeme potvrdiť, že študenti vnímali LMS Claroline ako užívateľsky príjemný a prehľadný, a preto vhodný pre integráciu do vzdelávacieho procesu“ [5].

Záver

Výsledky výskumu potvrdili, že zaradenie LMS Claroline do bežnej školskej výučby na základnej škole nerobí žiakom žiadne problémy. Jeho nasadenie do praxe nevyžaduje od školy žiadne finančné investície ani výrazne vyššiu časovú náročnosť prípravy vyučovacích hodín zo strany učiteľa. Jednoduché ovládanie, zrozumiteľnosť a prehľadnosť systému, ako aj rýchla a pohodlná tvorba interaktívnych cvičení výrazne zefektívňuje vzdelávací proces a robí ho žiakom atraktívnejší.

Literatúra

[1] CLAROLINE CONSORTIUM. 2014: *Claroline story*, `http://www.claroline.net/the-story-of-claroline/?lang=en`, cit. 2. jún 2014.
[2] Releases – MoodleDocs: `http://docs.moodle.org/dev/Releases`, cit. 3. jún 2014.
[3] KAROLČÍK, Š., BALÁŽOVIČ, Ľ.: *CLAROLINE LMS in the Lower Secondary Education*, Technology of Education [in review], 2014, Volume 22, Issue , ISSN 1338-1202
[4] BALÁŽOVIČ, Ľ.: *E-learnigový systém Claroline on Prezi*, `http://prezi.com/vybekdqzxjzd/e-learnigovy-system-claroline`, cit. 3. jún 2014.
[5] ČIPKOVÁ E., KAROLČÍK Š.: *Claroline IS as seen by future teachers*, Journal of Technology and Information Education 2/2013, Vol. 5, Iss. 2, pp. 44–51, ISSN 1803-537X.

Kontaktná adresa

Mgr. Ľuboš Balážovič, PhD., Katedra geografie, geológie a krajinnej ekológie, Univerzita Mateja Bela, Tajovského 40, 974 01 Banská Bystrica, Slovenská republika,
E-mailová adresa: `lubos.balazovic@umb.sk`, `http://www.fpv.umb.sk/lbalazovic/`

6. ročník Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf
OSSConf 2014: 41–48

DIGITÁLNY ATLAS MARBLE V ŠKOLSKEJ PRAXI

ĽUBOŠ BALÁŽOVIČ (SK) A MARTIN KRNÁČ (SK)

Abstrakt. Otvorené technológie vo vzdelávaní sa uplatňujú v mnohých štátoch sveta. V slovenských podmienkach zvlášť vo vyučovaní neboli zatiaľ celoplošne nasadené, avšak experimentálne metodiky dokazujú, že ich použitie by mohlo byť výrazným prínosom. Článok sa bližšie zaoberá problematikou edukačného softvéru Marble ako digitálneho atlasu/glóbusu a jeho využitiu vo vyučovaní geografie. V druhej časti stručne popisuje experimentálne navrhnutú metodiku na vyučovanie 14 rôznych tém s využitím Marble. Tretia časť sa venuje ich pilotnému odskúšaniu na viacerých základných a stredných školách v priebehu roka 2013, výsledky ktorého potvrdili kladnú spätnú väzbu u žiakov i pedagógov. Článok ďalej rozoberá technické obmedzenia a potenciál vyplývajúci z rozšíriteľnosti mapového obsahu o ďalšie voľne dostupné mapové podklady, či vlastné mapy a podmienky širšieho nasadenia vo vyučovaní v kontexte slovenského školstva.

Kľúčové slová. Marble, geografické vzdelávanie, školský atlas.

DIGITAL ATLAS MARBLE IN SCHOOL PRACTICE

Abstract. Open technologies in education are implemented in many countries in the world. In Slovak conditions, particularly in education have not yet been deployed but many experiments proving that their use could make a significant contribution for teaching. This paper closer deals with educational software Marble as a digital atlas/globe and its use in teaching geography. The second part briefly describes the experimental methodology to teach 14 different subjects using Marble. The third part is devoted to their pilot tested on several primary and secondary schools in the course of 2013, the results of which confirmed the positive feedback from pupils and teachers. It also discusses the technical constraints and potential extension of map content based on freely available maps or custom maps in conditions for wider deployment in teaching in Slovak education.

Keywords. Marble, geography education, school atlas.

Úvod

K tradičným didaktickým pomôckam pri vyučovaní geografie patria mapy, glóbusy a rôzny obrazový materiál (napr. diapozitívy). Každá z týchto pomôcok má už aj svoj ekvivalent z rodiny informačno-komunikačných technológií (zvlášť z geoinformačných). Ich implementácia do vyučovacieho procesu je síce žiadaná a patrí medzi priority rozvoja školstva [1], avšak jej realizácia nie je jednoduchá vzhľadom na rozmanitosť škôl, ich technické a personálne vybavenie. Druhou prekážkou plošného nasadenia je fakt, že mnohé z existujúcich technológií zatiaľ neboli overené v našej školskej praxi. Tento nedostatok sme sa pokúsili

čiastočne odstrániť v prípade digitálneho atlasu Marble a výsledky prinášame v tomto článku.

1. Mapa a glóbus v školskej geografii

Mapa ako zmenšený, skreslený a zjednodušený obraz zemského povrchu umožňuje učiteľovi veľmi jasne ukázať polohu všetkých geografických objektov a javov súvisiacich s preberanou témou [2]. Lokalizácia pohorí, riek či regiónov sa uskutočňuje prostredníctvom ich symbolického grafického vyobrazenia na mape.

Modernou alternatívou k tradičnej nástennej mape je digitálna mapa, ktorú možno premietnuť cez dataprojektor. Okrem neho je potrebný už len osobný počítač (najlepšie s internetovým pripojením). Medzi základné výhody digitálnych máp oproti klasickým patria:

- možnosť jednoducho prepínať rôzne mapy, resp. mapové pohľady (tú istú mapu v rôznych mierkach či stupňoch generalizácie);
- možnosť tvorby vlastných máp (pridaním alebo odobratím niektorých tematických vrstiev);
- možnosť prepojenia s ďalším faktografickým materiálom (tabuľky, obrázky, články, animácie).

Podobnú funkciu ako mapa plní aj glóbus. Učiteľovi je užitočnou pomôckou pri vysvetľovaní tém z planetárnej geografie (pohyby Zeme, ročné obdobia), ale aj mnohých ďalších tém z fyzickej, humánnej a regionálnej geografie.

Glóbus je definovaný ako zmenšený a zjednodušený model Zeme [2]. Modelovanie Zeme v čase informačných a komunikačných technológií pokročilo do stavu, keď nám súčasný softvér a hardvér umožňuje pohybovať sa po virtuálnej Zemi, pričom zachováva oblačnosť, jej tvar a tvary reliéfu na pevnine i v oceánoch. Umožňuje virtuálne prechádzku po zaujímavých prírodných a kultúrnohistorických zaujímavostiach. Súčasné modely obsahujú okrem 3D povrchu aj 3-rozmerne spracované budovy či iné stavby. Medzi voľne dostupné digitálne glóbusy možno zaradiť 2 aplikácie – *Marble Desktop Globe* (`http://edu.kde.org/marble`) a *Google Zem* (`http://earth.google.com`). Svojím obsahom a spracovaním na školské účely lepšie vyhovuje práve Marble, ktorý bol pôvodne koncipovaný práve ako edukačný softvér – učebná pomôcka. Za pedagogický (edukačný) softvér považujeme podľa [3] taký softvérový prostriedok, ktorý sa cieľavedome používa na podporu učenia a učenia sa.

2. Stolný glóbus Marble

Stolný glóbus Marble (obr. 1) je virtuálny glóbus a atlas sveta. Softvér Marble bol vyvíjaný v rámci balíka *KDE Edu*, ktorého hlavnou úlohou je pripraviť softvér využiteľný vo vzdelávaní. Ide o softvér šírený pod slobodnou licenciou GNU GPL, je teda voľne dostupný pre učiteľov i žiakov. Funguje pod väčšinou

známych operačných systémov (Linux/Unix, Microsoft Windows a Mac OS X) [9]. Prvá verzia softvéru Marble vyšla v novembri 2006, pričom nové verzie vychádzajú približne dvakrát do roka [4]. Softvér je lokalizovaný do slovenského jazyka, takže pri inštalácii a samotnom používaní nebráni jazyková bariéra.

Obr. 1. Používateľské prostredie aplikácie Marble

Ovládacie prvky aplikácie Marble sú rozdelené do piatich panelov: *Ovládanie*, *Vysvetlivky*, *Mapa*, *Aktuálna poloha* a *Navigácia*.

Panel *Ovládanie* slúži na zmenu mapového pohľadu (priblíženie, oddialenie, posun) a tiež na rýchle vyhľadanie ľubovoľného mapového objektu (možno napríklad vyhľadať svetadiel, pohorie, štát, mesto…).

Panel *Vysvetlivky* obsahuje všetky mapové značky použité v aktuálnom mapovom pohľade. Okrem informatívnej funkcie slúži aj na prispôsobenie mapového pohľadu (niektoré skupiny mapových prvkov je možné vypnúť).

Tretím panelom je panel *Mapa*. Tu je možné zvoliť mapové zobrazenie, ktoré bude použité pri vykresľovaní mapy. V ponuke sú tri zobrazenia: Nástenná mapa (Flat map, valcové rovnakodĺžkové), Glóbus (azimutálne ortogonálne zobrazenie) a Mercatorovo (valcové rovnakouhlé).

Tiež je možné vybrať si medzi 11 mapovými témami [7]:

- Atlas – predstavuje základnú topografickú mapu,
- OpenStreetMap – mapy získané zo servera OpenStreetMap, (`http://www.openstreetmap.org`), tie sú v súčasnosti spracované pre väčšinu obývaného prostredia, pričom najkvalitnejšie sú zmapované väčšie mestá,
- mapy júlových a decembrových zrážok a teplôt,
- mapa vytvorená z družicových snímok,
- historická mapa z r. 1690,
- jednoduchá obrysová mapa.

Rôzne mapové podklady, ktoré Marble neobsahuje je možné si stiahnuť bezplatne na internete. Aplikácia nás cez položku Súbor ⇒ Prevziať mapy navedie na internetovú lokalitu, kde sa nachádzajú geografické mapy Zeme, historické mapy

od roku 1502 po rok 1866, mapy Mesiaca, mapy niektorých planét a mesiacov slnečnej sústavy.

Mapové témy, ktoré sa nenachádzajú na internetovej stránke alebo v samotnom programe a sú potrebné pri vyučovacom procese si môžeme vytvoriť aj sami cez položku Súbor ⇒ Vytvoriť novú mapu, ďalej nás bude navigovať sprievodca pre vytváranie máp, ktorý upozorní na všetky potrebné kroky pri vytváraní mapovej témy. Je nutné podotknúť, že pri vytváraní vlastných máp, aplikácia dokáže spracovať iba taký mapový podklad, ktorý má aspoň základnú sieť poludníkov a rovnobežiek. Mapu bez geografickej siete program ju nevytvorí.

Medzi ďalšie funkcie Marble možno zaradiť zobrazovanie skutočnej oblačnosti (aktualizované každé tri hodiny) a aktuálne oslnenie povrchu (simuláciu oslnenia je možné zobraziť aj pre konkrétny dátum a čas).

Niektoré miesta ponúkajú možnosť prepojenia s Wikipediou. Po kliknutí na takéto miesto je možné dozvedieť sa informácie, ktoré Wikipedia ponúka. Marble umožňuje pozorovať prúdenie vzduchu v atmosfére.

Pri pripojení počítača k internetovej sieti, program ponúka aktuálne informácie ako sú zemetrasenia, počasie, fotografie, poloha satelitov (možno ich zapnúť cez Zobraziť ⇒ Online služby. Je to vhodný doplnok využiteľný napríklad pri témach o vulkanickej činnosti a pohyboch zeme, kde žiaci vidia popri vyučovaní v aktuálnom čase na ktorých miestach na zemi boli zaznamenané otrasy a s akou intenzitou.

Aplikácia je veľmi jednoduchá aj pre nenáročného používateľa, je plne prispôsobená aj potrebám študentov bez nutnosti zvládnutia špeciálnych zručností.

3. Marble ako školský atlas

So softvérom Marble sa slovenskí učitelia mohli zoznámiť v rámci národného programu *Modernizácia vzdelávania na základných a stredných školách*, ktorý prebiehal v rokoch 2008–2013, kde súčasťou kurikula 3. vzdelávacieho modulu pre učiteľov geografie boli aj tipy na využitie Marble vo vyučovaní. Systematické overovanie v praxi však bolo realizované až v práci Martina Krnáča v rokoch 2013–2014 [6].

3.1. Príprava a priebeh hodín s programom Marble

Hlavným cieľom pilotného overenia Marble v školskej praxi bolo navrhnutie a odskúšanie metodiky práce s digitálnymi mapami prostredníctvom softvéru Marble. V praxi sme vyskúšali náročnosť práce s aplikáciou, vytváranie máp, využitie vo vyučovacom procese. Prvým krokom bol návrh prípravy na hodiny s využitím Marble (obsahuje vzdelávacie ciele a priebeh v jednotlivých fázach hodiny). Tie boli následne testované v praxi. Testovanie bolo realizované na rôznych školách v rôznych ročníkoch a v dvoch rôznych predmetoch (geografia a dejepis). Prehľad tém, škôl a ročníkov sa nachádza v tabuľke 1.

Názov školy	Téma hodiny	Ročník
Ev. gymnázium BB	Typy krajín Európy	8.
Ev. gymnázium BB	Obyvateľstvo a sídla Európy	8.
Ev. gymnázium BB	Oblasti a štáty Európy	8.
ZŠ Trieda SNP BB	Bratislavský kraj	9.
ZŠ Trieda SNP BB	Trnavský kraj	9.
ZŠ Trieda SNP BB	Severná Európa – úvod	7.
ZŠ Trieda SNP BB	Severná Európa – Dánsko	7.
GAS Krupina	Severná Amerika – prírodné podmienky	2.
GAS Krupina	Severná Amerika – Obyvateľstvo a sídla	2.
GAS Krupina	Severná Európa – komplexná charakteristika	Septima
GAS Krupina	Severoslovenský región	3.
OA Krupina	Austrália – hospodárska charakteristika	1.
ZŠ Trieda SNP BB	Zámorské objavy[1]	7.

Tabuľka 1. Prehľad tém, ročníkov a škôl, na ktorých sa testoval Marble

Marble preberá hlavne funkciu tradičnej nástennej mapy, avšak s dynamicky meniacim sa obsahom (či už zmenou mapovej témy, alebo mapovej mierky). V jednotlivých fázach vyučovania bol digitálny atlas Marble použitý nasledovne:

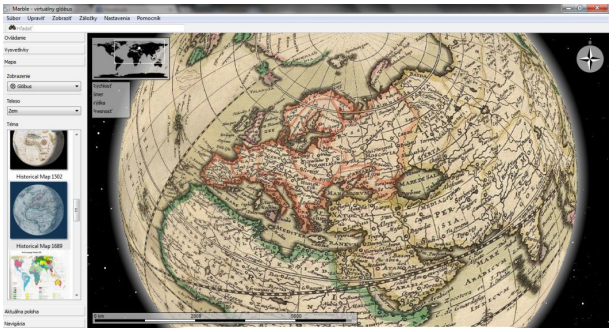
- v motivačnej fáze sme sa snažili upútať pozornosť žiakov niečím novým čo nevideli, novým pohľadom na mapu, ktorá ich mala motivovať pri osvojovaní učiva;
- v expozičnej etape sme využívali všetky dostupné mapy k danému učivu, vysvetlivky, možnosti, ktoré daná aplikácia ponúka na oboznámenie žiakov s novým učivom. Okrem softvéru Marble sme využívali v tejto časti aj pracovný list alebo prezentáciu vytvorenú v PowerPointe, v ktorej sme poskytli žiakom poznámky a doplňujúce obrázky k danému učivu;
- fixačná etapa prebiehala reproduktívnou metódou rozhovoru, pri ktorej sme zisťovali, koľko si žiaci dokázali zapamätať z hodiny;
- diagnostická etapa bola zameraná na praktickú časť, pri ktorej žiaci pracovali samostatne s programom Marble, mali za úlohu ukázať rozličné povrchové celky, moria, nížiny, mestá..., ktoré si mali osvojiť na vyučovacej hodine;
- aplikačná časť prebiehala formou domácej úlohy, alebo úloh v pracovnom zošite, prípadne iných úloh – vzhľadom na licenciu softvéru Marble mohli žiaci využívať túto pomôcku aj doma.

[1] Téma predmetu *Dejepis*.

Okrem návrhu priebehu hodiny sme sa snažili navrhnúť vhodné mapové podklady na jednotlivé hodiny vrátane postupu výroby vlastných tematických mapových podkladov. Pri výrobe vlastných podkladov na vyučovanie upozorniť na problémy vzniknuté s výberom a zadávaním samotných máp do programu a jeho možnosti využitia.

4. Výsledky

Digitálny atlas Marble bol odskúšaný v podmienkach reálneho vyučovania a z pohľadu učiteľa možno konštatovať jednoznačný prínos technológie do procesu výučby. Okrem predmetu geografia poslúžil ako vhodná pomôcka aj pri motivačnej a expozičnej časti vo vyučovaní predmetu dejepis v téme „Zámorské objavy“.



Obr. 2. Historická mapa z roku 1690 – mapová téma použitá pri vyučovaní témy *Zámorské objavy*

Menšie nedostatky sme zistili v obsahu mapových tém softvéru Marble pri výučbe a charakteristike územia na úrovni krajov a menších štátov, kde mapové témy neboli dosť podrobné, resp. dosť prehľadné (mapová téma OpenStreetMap), prípadne sme nevedeli vytvoriť mapu, ktorá by dané menšie územie zobrazovala. Naopak softvér Marble sa ukázal byť vhodným edukačným doplnkom pri plošne veľkých územiach z regionálnej a fyzickej geografie, klimageografii, biogeografii, kde existujú veľké dostupnosti mapových podkladov a nebol problém ani pri výrobe vlastných digitálnych mapových diel.

Celkove sa nám podarilo vyrobiť 6 vlastných mapových tém s názvami „Bioms“ – biómy sveta, „Economy“ – ekonomická vyspelosť sveta, „Industry“ – mapa priemyslu sveta, „Languages“ – mapa svetových jazykov, „Religion“ mapa náboženských skupín a „Ocean currents“, mapu morských prúdov. Všetky tieto mapové témy sme aj úspešne odskúšali vo vyučovacom procese. K týmto mapovým podkladom sme si „stiahli“ aj voľne dostupné mapové témy zo stránky Marble.

Obr. 3. Nová mapová téma *Typy krajín* vytvorená z voľne dostupnej mapy

4.1. Výsledky dotazníkového prieskumu

Aby naše odskúšanie softvéru Marble nemalo len rýdzo subjektívny ráz, doplnili sme náš didaktický výskum o dotazník, v ktorom mohli žiaci vyjadriť svoj postoj k priebehu hodín s programom Marble a zavádzaním digitálnych technológií do vyučovania. Dotazníkový prieskum sa uskutočnil pred koncom hodiny po ukončení všetkých aktivít žiakov a nezaberal viacej ako 5 minút. Pri hodnotení „ako známkou v škole“ 85 % respondentov hodnotilo prácu s Marble na hodine známkou „1”, zvyšných 15 percent udelilo známku „2“. Dotazník vyznel jasne v prospech programu Marble aj pri ostatných otázkach, avšak vzhľadom na počet vybraných respondentov (veľkosť vzorky štatistického súboru) môže brať jeho výsledky len informatívne.

Záver

Dnes je už zjavné, že zavádzanie digitalizácie do vyučovacieho procesu je veľkým krokom vpred. Globálna éra digitálnych médií sa stále posúva dopredu, preto by sa aj tieto technológie mali dostávať čoraz viac do vyučovacieho procesu a priniesť tak žiakom nový pohľad na možnosti vyučovania. Komplexne by sme tento jav mohli pomenovať aj ako nevyhnutnosť zavádzania do procesu výučby najmodernejšie technológie za účelom zefektívniť tento proces.

Treba však pripomenúť, že nielen žiak a edukačný softvér sú vstupnou bránou do sveta digitálnych technológií, ale je to hlavne učiteľ. Teda tento problém nevykresľovať ako technický, ale aj pedagogický. Pre úspešné zavádzanie digitálnej technológie do školskej praxe je potrebné vychovávať nielen vzdelanú mládež, ale aj digitálne gramotnú učiteľskú obec, ktorá sa dokáže uplatniť v modernej digitálnej spoločnosti.

Pri mnohých povinnostiach, ktoré sa dnes kladú na profesiu učiteľa, je táto skutočnosť pomerne rozsiahla a časovo náročná. Jej úspešné zvládnutie a zrealizovanie na našich školách vytvorí stav, kde budú moderné digitálne technológie každodennou záležitosťou vo vyučovacom procese.

Literatúra

[1] ŠIMÁŠEK, Ľ. et al: *DIGIPEDIA 2020 – Koncepcia informatizácie rezortu školstva s výhľadom do roku 2020*, Ministerstvo školstva, vedy a výskumu a športu Slovenskej republiky, 2013.

[2] BALÁŽOVIČ, Ľ.: *Všeobecné kartografické minimum a jeho postavenie v školskom systéme Slovenska*, Kartografické listy 1/2014 (v tlači), 2014.

[3] BRESTENSKÁ, B. et. al.: *Premena školy s využitím informačných a komunikačných technológií*, prvé vydanie, Košice, Elfa, s.r.o., 2010.

[4] *Marble – find your way and explore the world*, cit. 30. mája 2014 [dostupné online na `http://marble.kde.org/changelog.php`].

[5] RAHN, T., NIENHÜSER, D.: *The Marble Handbook*, 2013 [dostupné online na `http://docs.kde.org/development/en/kdeedu/marble/`].

[6] KRNÁČ, M.: *Návrh a overenie metodiky tvorby didaktických projektov z geografie s využitím softvéru Marble*, diplomová práca, Banská Bystrica UMB, 2014, 72 s.

[7] BALÁŽOVIČ, Ľ., ŠAMAJOVÁ, J.: *Geoinformačné technológie a ich miesto vo vyučovaní geografie*, Geografia č. 4, roč. 17, 2009, Geoservis, Bratislava.

[8] KUBALIAKOVÁ, K., BALÁŽOVIČ Ľ. et al.: *Využitie informačných a komunikačných technológií v predmete Geografia pre základné školy*, Košice, Elfa s.r.o, 2010, 279 s., ISBN 978-80-8086-155-1.

[9] MÁZOROVÁ, H., BALÁŽOVIČ, Ľ. et al.: *Využitie informačných a komunikačných technológií v predmete Geografia pre stredné školy*, Košice, Elfa s.r.o, 2010.

Kontaktné adresy

Mgr. Ľuboš Balážovič, PhD., Katedra geografie, geológie a krajinnej ekológie, Univerzita Mateja Bela, Tajovského 40, 974 01 Banská Bystrica, Slovenská republika,
E-mailová adresa: `lubos.balazovic@umb.sk`, `http://www.fpv.umb.sk/lbalazovic/`

Mgr. Martin Krnáč, Katedra geografie, geológie a krajinnej ekológie, Univerzita Mateja Bela, Tajovského 40, 974 01 Banská Bystrica, Slovenská republika,
E-mailová adresa: `martin.krnac@studenti.umb.sk`

6. ročník Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf
OSSConf 2014: 49–58

LaTeX A TABUĽKY

RUDOLF BLAŠKO (SK)

Abstrakt. Tabuľky tvoria významnú časť pri písaní dokumentov. Ich tvorba nie je vo všeobecnosti jednoduchá a vytvoriť kvalitnú, vizuálne peknú tabuľku, ktorá vyhovuje nielen pravopisným a typografickým normám, ale aj požiadavkám autora, môže byť problémom. V prvej časti príspevku sú predstavené niektoré on-line generátory/editory na tvorbu LaTeX-ových tabuliek. Ďalšie dve časti sa stručne zaoberajú tvorbou zložitejších tabuliek a tabuliek presahujúcich viacero strán.

Kľúčové slová. LaTeX, tabuľka, on-line, www.

LaTeX AND TABLES

Abstract. Tables are an important part when writing documents. Their formation is generally not easy. Create high-quality, visually nice table that suits not only spelling and typographical standards, as well as the requirements of the author can be a serious problem. The first part of the paper presents some online generators/editors for creating LaTeX-voltage tables. The next two sections briefly deal with the creation of complex tables and as well tables that exceed multiple pages.

Keywords. LaTeX, table, on-line, www.

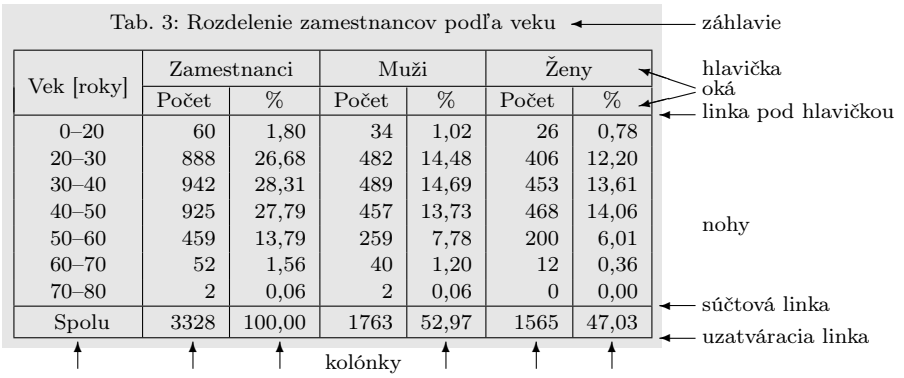
Úvod

Tabuľky tvoria významnú časť nielen odborných a vedeckých publikácií. Vyskytujú sa v knihách aj časopisoch, môžu byť ako samostatné tlačivá. Môžeme k nim priradiť aj poradovo usporiadané texty oddelené linkami. Môžu byť zaradené priamo v texte, alebo môžu byť umiestnené ako tzv. plávajúce objekty na inom vhodnom mieste. Ako plávajúce objekty sú číslované v záhlaví nad tabuľkou, aby sa na ne mohlo odvolávať. Tabuľky rozdeľujeme na:

- **Textové tabuľky** – väčšinu tabuľky tvorí text, napr. hospodárske tabuľky, cestovné poriadky.
- **Číselné tabuľky** – obsahujú číselné údaje poradovo usporiadané do prehľadných skupín, napr. matematické, fyzikálne a chemické tabuľky.
- **Linkové tabuľky** – tabuľky určené výhradne na vpisovanie údajov rukou, strojom alebo dotláčaním, napr. dotazníky.
- **Knižné tabuľky** – tabuľky sádzané výlučne z jemných liniek, ktoré sa prispôsobujú celkovej úprave knihy alebo časopisu, veľkosti sadzobného obrazca a písma základného textu.

Tabuľku môžeme rozdeliť na niekoľko častí (obr. 1):

- **Záhlavie** – priestor nad hlavičkou a využíva sa na číslovanie a označenie názvu tabuľky, napr. `Tab. 1.1: Grécka abeceda`.
- **Hlavička** – základ tabuľky, určuje jednotlivé kolónky tabuľky.
- **Oká** – jednotlivé časti hlavičky oddelené zvislými a vodorovnými linkami.
- **Linka pod hlavičkou** – vodorovná linka, ktorá uzatvára hlavičku a oddeľuje dolnú časť tabuľky.
- **Nohy** – časť pod hlavičkou, zvislé stĺpce, delia sa na jednotlivé **bunky**.
- **Kolónky** – dolné časti tabuľky (nôh) rozdelené zvislými linkami.
- **Súčtová linka** – oddeľuje priebežné číselné hodnoty od ich súčtov, pod ňou sa sádzu súčty týchto hodnôt.
- **Vodorovná uzatváracia linka** – uzatvára tabuľku v dolnej časti.



Obr. 1. Opis tabuľky

Podľa úpravy sa tabuľky delia na **otvorené**, **previsnuté** a **uzavreté**. Každá z týchto tabuliek má svoje pravidlá pri sádzaní, ktoré sa nemenia, aj keď majú rôznu úpravu a obraz použitých liniek je rozdielny. Jemné linky sa v tabuľke obvykle sádzu s hrúbkou 0,3 pt, polotučné linky s hrúbkou 0,8 pt a tučné linky s hrúbkou 1,2 pt.

- **Otvorené tabuľky:** tabuľka nemá zvislé obvodové linky, môže byť uzavretá (ale nemusí byť) v hlavičke, nemusí mať ani vodorovnú uzatváraciu linku v nohách, väčšinou priamo nadväzuje na poradovú sadzbu, textové stĺpce môžu byť (ale tiež nemusia byť) navzájom oddelené zvislými linkami.
- **Previsnuté tabuľky:** základná linka tabuľky sa sádze z polotučnej alebo tučnej šírky a prečnieva šírku tabuľky, dĺžka previsu je minimálne 6 pt, tabuľky previsnuté iba do jednej strany sa nazývajú **previsnuté do šibeničky**, používajú sa na veľké tabuľky rozdelené lomom papiera a nadväzujúce na seba.
- **Uzavreté tabuľky:** sú uzavreté zo všetkých strán linkami, linky na seba presne nadväzujú v rohoch.

Ako sme už spomínali, tabuľky tvoria významnú časť dokumentov. Ich tvorba vo všeobecnosti nie je jednoduchá. V textových editoroch typu WYSIWYG (editory OpenOffice.org, MS Office ap.) sa tabuľky vytvárajú pomocou klikania myšou. Tento proces je síce únavný a otravný, ale prináša pomerne jednoducho výsledok. Horšie je to s tvorbou tabuliek v LaTeX-u – tu síce dokážeme vytvoriť oveľa krajšie a zložitejšie tabuľky, ale potrebujeme na to určitú dávku predstavivosti a LaTeX-ovej skúsenosti. Začiatočník, sám bez podpory, väčšinou nevytvorí zložitejšie tabuľky, ale na druhej strane, dokáže s minimálnym úsilím modifikovať a napĺňať údajmi už existujúce tabuľky. Toto je problém predchádzajúcich klikacích editorov. Zmeniť nejaké čiastkové nastavenie (napr. veľkosť písma, farbu bunky, zarovnanie ap.) nie je problém, ak máme dostatok trpezlivosti a času na to, aby sme každú menenú časť tabuľky myšou označili a následným klikacím procesom zmenili jej stav.

V ďalšej časti sa venujeme tvorbe tabuliek v LaTeX-u, ale nie jej základom (viď napr. [1, 3]). V prvej kapitole sú predstavené niektoré on-line nástroje na tvorbu LaTeX-ových tabuliek. Ďalšie dve časti sa zaoberajú tvorbou zložitejších tabuliek a tabuliek presahujúcich viacero strán.

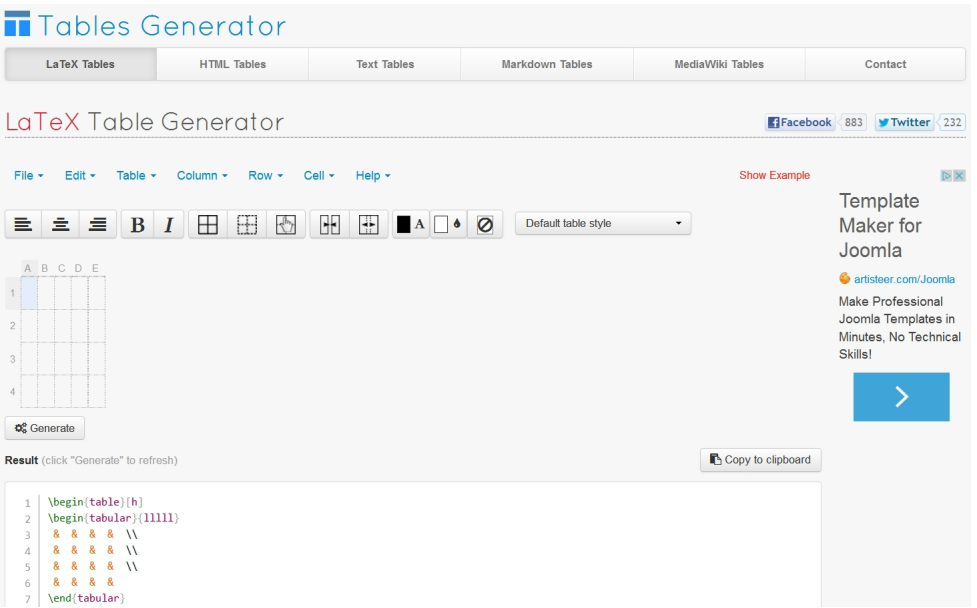
1. On-line tabuľkové editory LaTeX-ových tabuliek

Na webe existuje niekoľko editorov/generátorov tabuliek, ktoré ponúkajú interaktívnu možnosť na vytváranie LaTeX-ových tabuliek. Práca s väčšinou z nich je problematická. Najprv spomenieme `Table editor`, ktorý nájdeme na webovej stránke `http://truben.no/latex/table/`. Tento prostriedok je nepohodlný a zle sa nastavuje.

Ďalšiu, o niečo lepšiu možnosť na vytváranie LaTeX-ových tabuliek ponúka www stránka `http://authortools.aas.org/LATEX/make-latex.html` a na nej umiestnený `The AAS Journal's LaTeX/AASTeX table creator`. Práca v ňom je dosť nepraktická a neintuitívna. Generátor síce ponúka rôzne možnosti, ale treba sa preklikať cez niekoľko webových stránok, kým dostaneme LaTeX-ový zdrojový súbor. Veľkú väčšinu tohto zdrojového súboru tvoria komentáre a hlavný text je v nich utopený.

Na záver predstavíme vcelku uspokojujúci tabuľkový on-line editor „Tables Generator“, ktorý sa nachádza na adrese `http://www.tablesgenerator.com/`. Tomuto editoru sa potešia hlavne zástancovia klikacích WYSIWYG editorov, t. j. tí, čo píšu v OpenOffice.org alebo MS Office. V uvedených editoroch sa tabuľky tvoria jednoduchými kliknutiami myšou a všetky atribúty sa tam nastavujú cez rôzne (a hlavne na rôznych úrovniach skryté) úrovne menu. Skúsený užívateľ sa takejto možnosti poteší, pretože mu ušetrí mnoho práce a vygeneruje mu jednoduchý a prehľadný zdrojový kód (vrátane nevyhnutných balíčkov), ktorý môže ďalej podľa potreby upraviť.

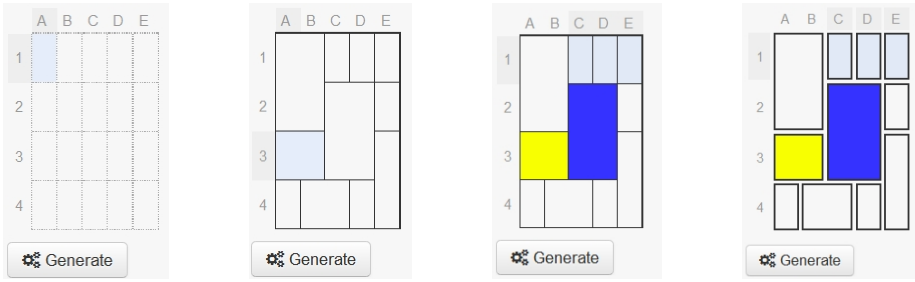
Tento generátor tabuliek ponúka okrem možnosti LaTeX-ového výstupu aj výstupy v tvare HTML, jednoduchých tabuliek v textovom súbore (Text Tables, resp. Markdown Tables) a webových tabuliek MediaWiki Tables. Práca s editorom je pohodlná, všetko sa ovláda výberom a označením vhodných buniek, prípadne celej tabuľky myšou a následným výberom požadovanej vlastnosti z jednoduchého menu (obr. 2). Uvedené menu dovoľuje nastaviť niekoľko parametrov tabuľky, ako sú zarovnanie textu v bunke k jednotlivým krajom/centrovanie, tučné písmo, kurzíva, rôzne orámovania, farby textu a farby pozadia. Ďalej môžeme jednoducho zlučovať do jednej alebo rozdeľovať označené bunky na pôvodné. V LaTeX-u sa k tomuto účelu používajú príkazy `\multicolumn` a `\multirow`, ktorým sa budeme venovať neskôr.



Obr. 2. Náhľad na stránku `http://www.tablesgenerator.com/` s generátorom tabuliek „Tables Generator“

Niektoré možnosti pri vytváraní tabuľky sú znázornené na obr. 3. Kvôli zjednodušeniu je na obrázku znázornená iba tabuľka, ako ju vidí užívateľ, keď ju vytvára. Úplne napravo je znázornená dekomponovaná tabuľka pri aktivácii voľby „ruka v rámčeku“ (viď obr. 2), kde vidíme, ako sú zoskupené a orámované jednotlivé bunky tabuľky.

Po navrhnutí tabuľky vygenerujeme jej LaTeX-ový zdrojový kód jednoducho stlačením tlačidla Generate. Ukážka výsledného kódu je na obr. 4. Ako vidíme, text je úhľadný, zoradený podľa stĺpcov a farebne odlíšený. Ak je nutné načítať nejaký balíček navyše, je na začiatku zakomentovaný text s poznámkou, čo



Obr. 3. Príklad postupného vytvárania tabuľky

treba urobiť. V našom prípade treba do preambuly LaTeX-ového súboru vložiť riadky `\usepackage{multirow}`, `\usepackage[table,xcdraw]{xcolor}`. Posledný zakomentovaný riadok uvádza, aké príkazy treba aplikovať pre triedu `Beamer`. Tabuľku si môžeme uložiť alebo text jednoducho skopírovať do svojho súboru so zdrojovým textom.

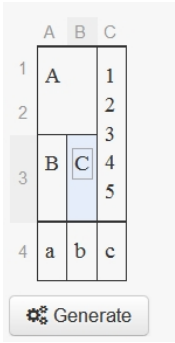
Obr. 4. Vygenerovaný LaTeX-ový zdrojový kód tabuľky

Je samozrejmé, že tabuľku môžeme vypĺňať priamo na webovej stránke, len si musíme dávať pozor na počet riadkov, aby korešpondoval s číselnou hodnotou určujúcou počet riadkov tabuľky, ktorá je vľavo od vytváranej tabuľky. V opačnom prípade sa nám môže stať, že vytvoríme podobný nezmysel ako je znázornený na obr. 5. V tomto prípade je tabuľka navrhnutá na 4 riadky, ale v pravom stĺpci C sme použili päť riadkov – aj keď by mal byť rozdelený iba na tri riadky. Spolu je teda umiestnených päť riadkov na plánované tri riadky. Dokazuje to príkaz `\multirow` a v ňom vložená tabuľka s piatimi riadkami:

```
\multirow{3}{*}{
        \begin{tabular}[c]{@{}l@{}}1\\ 2\\ 3\\ 4\\ 5\end{tabular}}
```

Pri tvorbe uvedenej tabuľky bolo v stĺpci C použité medzi vkladaním čísel 1 až 5 tlačidlo ENTER. Výsledok snaženia takto vytvoreného návrhu je na obr. 5 vpravo. Ak používame v slovenčine alebo češtine babelizovaný LaTeX, musíme hneď za príkaz `\begin{document}` vložiť príkaz `\shorthandoff{-}`, aby nám neexpandovala pomlčka. Toto je dôležité, aby sme mohli používať príkaz `\cline`.

Editor ponúka možnosť uložiť tabuľku ako plávajúci objekt, t. j. zapúzdriť do prostredia `table` a popis tabuľky `\caption` umiestniť pod alebo nad vlastnú tabuľku, prípadne horizontálne centrovať tabuľku. Tieto voľby sa nachádzajú v tlačidle Extra options....




```
\begin{tabular}{|l|l|l|}
\hline
\multicolumn{2}{|l|}{\multirow{2}{*}{A}} &
 \multirow{3}{*}{\begin{tabular}[c]{@{}l@{}}
 1\\ 2\\ 3\\ 4\\ 5\end{tabular}} \\
\multicolumn{2}{|l|}{}   &    \\ \cline{1-2}
B                   & C  &    \\ \hline
a                   & b  & c  \\ \hline
\end{tabular}
```

A		1 2
B	C	3
a	b	4 5

Obr. 5. Príklad nesprávneho tvorenia tabuľky (vľavo), vygenerovaný zdrojový kód (stred), spravodlivý výsledok tohto snaženia (vpravo)

2. Použitie príkazov `\multicolumn` a `\multirow`

Jednotlivé bunky tabuľky môžeme zlučovať pomocou príkazov `\multicolumn` a `\multirow`. Príkaz `\multicolumn` je súčasťou LaTeX-u, t. j. nemusíme načítavať žiadny balíček a používa sa v tvare `\multicolumn{číslo}{typ stĺpca}{text}`.

Parameter `číslo` určuje počet stĺpcov, ktoré chceme v danom riadku spojiť. Tento počet stĺpcov sa tvári ako jeden stĺpec tabuľky. Za príkazom `\multicolumn` musí nasledovať oddeľovač stĺpcov `&` alebo oddeľovač riadkov `\\`. To znamená, že parameter `číslo` nahrádza o jeden menej znakov `&`, ako je jeho hodnota.

Parameter `typ stĺpca` má identický význam ako pri definícii tabuľky (napr. `l`, `|p{7em}|` alebo `|@{\enspace}r@{ = }`). Posledný parameter `text` predstavuje vlastný text zlúčeného stĺpca a musí spĺňať rovnaké podmienky ako stĺpce rovnakého typu v tabuľke. Príklad použitia vidíme napríklad na obr. 5.

Spájať riadky v rámci jedného stĺpca môžeme pomocou príkazu `\multirow` z balíčka `multirow`. To znamená, že v preambule musí byť `\usepackage{multirow}`. Uvedený príkaz sa používa v tvare:

```
\multirow{číslo}[bigstruts]{šírka stĺpca}[fixup]{text}
```

Parameter `číslo` určuje počet riadkov, ktoré chceme v danom stĺpci spojiť. Kladná hodnota určuje počet nasledujúcich spájaných riadkov, pričom sa na príslušné stĺpcové miesto v týchto riadkoch nič nepíše. V opačnom prípade by sa tieto texty prekrývali. Ak je toto číslo záporné, postup je opačný a najskôr sa vynechá text predchádzajúcich riadkov v príslušnom stĺpci.

Z uvedeného vyplýva, že nasledujúce konštrukcie majú rovnaký význam. Výsledok je zobrazený vpravo.

```
\begin{tabular}{|l|l|}
\hline
aa & \\ \cline{1-1}
cc & \\ \hline
dd & \multirow{-3}*{bb}\\
\hline
\end{tabular}
```

```
\begin{tabular}{|l|l|}
\hline
aa & \multirow{3}*{bb}\\
\cline{1-1}
cc & \\ \hline
dd & \\ \hline
\end{tabular}
```

aa	
cc	bb
dd	

Voliteľný parameter `bigstruts` sa používa v spojení s príkazom `\bigstrut`. Jeho nastavenie a použitie nájde hĺbavý čitateľ v textovej dokumentácii balíčka `multirow`, t. j. stačí zadať na príkazovej úrovni (ako v linuxe, tak aj v Microsoft Windows) `texdoc multirow`. Parameter `šírka stĺpca` sa vyjadruje ako číslo s dĺžkovou jednotkou alebo ako znak `*`, ktorý indikuje prirodzenú potrebnú šírku stĺpca. Druhý voliteľný parameter `fixup` sa tiež vyjadruje ako číslo s dĺžkovou jednotkou a určuje vertikálne posunutie textu nahor (kladná hodnota) alebo nadol (záporná hodnota). Posledný parameter `text` predstavuje vlastný text.

Príkazy `\multirow` a `\multicolumn` môžeme vkladať navzájom do seba (napr. `\multicolumn{2}{|l|}{\multirow{2}{*}{A}}` v nasledujúcej tabuľke). Na záver tejto časti uvedieme jednu z možností ako opraviť tabuľku z obrázku 5 tak, aby sa neprekrývali jednotlivé bunky.

```
\begin{tabular}{|l|l|l|} \hline
\multicolumn{2}{|l|}{\multirow{2}{*}{A}} &
 \multirow{3}{*}{\begin{tabular}[c]{@{}l@{}}
 1\\ 2\\ 3\end{tabular}} \\
\multicolumn{2}{|l|}{}   &    \\ \cline{1-2}
B                  & C  &    \\ \hline
a                  & b  & c  \\ \hline
\end{tabular}
```

A		1 2
B	C	3
a	b	c

3. Balíček `longtable` a tabuľky na viacero strán

Ak máme tabuľku, ktorej rozsah prekračuje niekoľko strán, máme dve možnosti. Jedna je ručne tabuľku rozdeliť na potrebný počet samostatných tabuliek a tie vysádzať na jednotlivé strany. Druhá, jednoduchšia, možnosť je použitie balíčka `longtable`. Aby sme mohli tento balíček používať, musíme do preambuly dokumentu vložiť riadok `\usepackage{longtable}`.

Použitie balíčka `longtable` je jednoduché, ale aby tabuľky vyzerali prirodzene, musí sa zdrojový dokument väčšinou preložiť niekoľkokrát (niekedy až 4 razy).

Pri vytváraní „dlhých tabuliek“ pomocou `longtable` môžeme používať všetky príkazy ako pri normálnych tabuľkách (vrátane farieb, `\multirow`, `\multicolumn`, popisu `caption` ap.). Štruktúru zdrojového kódu takýchto tabuliek ilustrujú nasledujúce príkazy. Medzi jednotlivými skupinami uvedených príkazov by mal byť vynechaný riadok.

```
\begin{longtable}{|c|l|r|}
\caption[Krátke záhlavie]{Záhlavie tabuľky} \label{navestie} \\
```

Príkaz `caption` je nepovinný. Ak sa nepoužije, tabuľka nebude mať záhlavie. Nepovinný parameter `Krátke záhlavie` sa sádze do zoznamu tabuliek.

```
\hline Hlavička & prvej & podtabuľky \\ \hline
\endfirsthead
```

Táto skupina príkazov určuje hlavičku tabuľky na prvej strane, t. j. prvej podtabuľky. Táto hlavička je ukončená príkazom `\endfirsthead`. Ak sa vynechá, použije sa alternatívna hlavička tabuľky, ktorá sa sádze (pokiaľ tam je) ako hlavička ďalších podtabuliek. Alternatívna hlavička je ukončená príkazom `\endhead`.

```
\multicolumn{3}{c}%
{{\tablename\ \thetable{} -- pokračovanie}} \\ %alternatívne záhlavie
\hline Hlavička & ďalších & podtabuliek \\ \hline
\endhead
```

Pätku – text za jednotlivými podtabuľkami na konci strán môžeme definovať nasledujúcimi nepovinnými príkazmi:

```
\hline \multicolumn{3}{|r|}{{pokračovanie}} \\ \hline
\endfoot
```

Pätka je ukončená príkazom `\endfoot`. Ukončenie tabuľky môžeme predefinovať nasledujúcimi nepovinnými príkazmi.

```
\hline
\endlastfoot
```

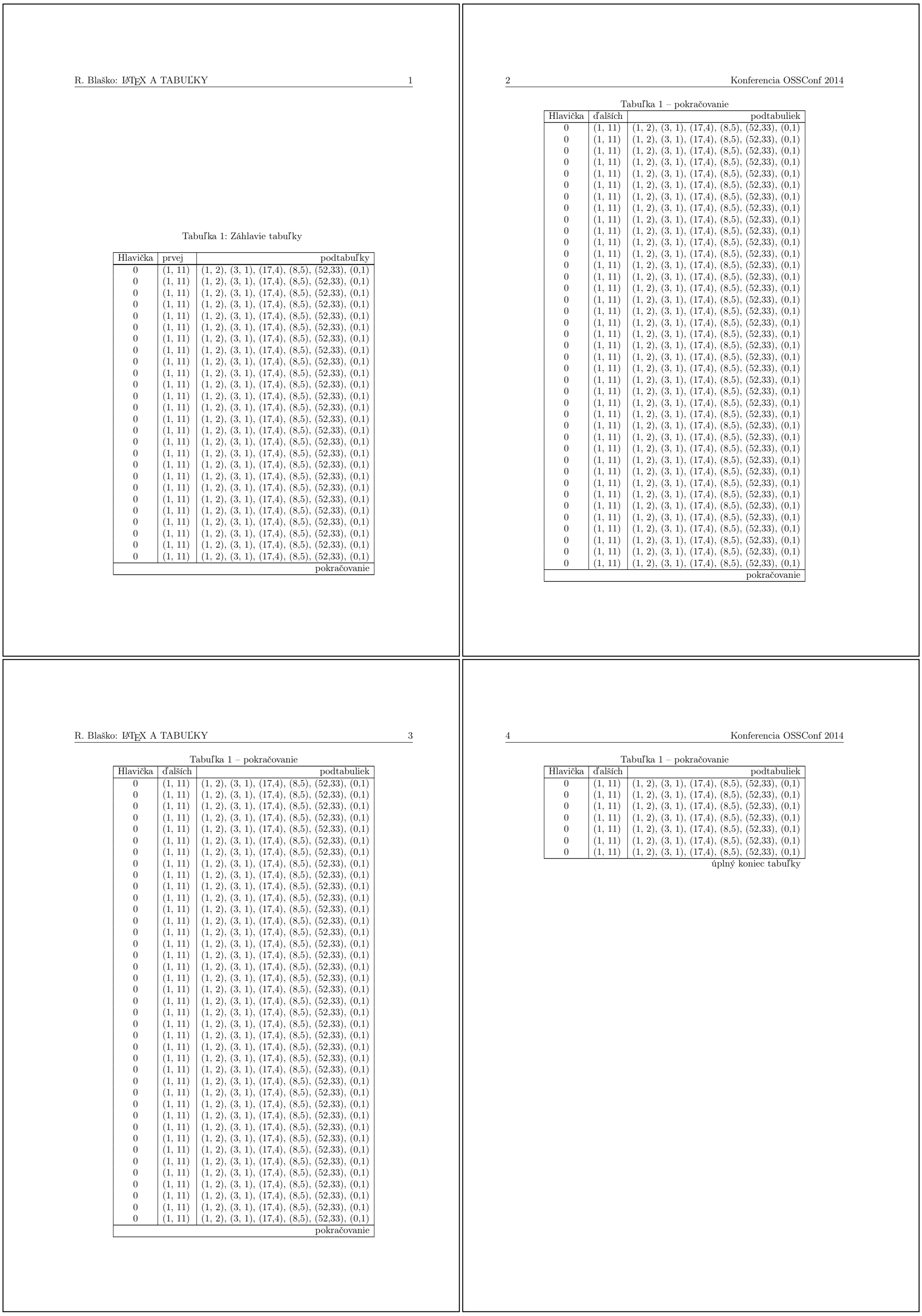
Ak tam chceme použiť aj nejaký text môžeme napríklad použiť:

```
\hline \multicolumn{3}{r}{úplný koniec tabuľky}
\endlastfoot
```

Potom nasledujú vlastné údaje tabuľky, ktoré musia byť ukončené príkazom `\end{longtable}`. Tieto údaje by mali byť od hlavičiek a pätiek oddelené prázdnym riadkom.

```
\hline
0 & (1, 11) & (1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1) \\
 ... údaje tabuľky ...
0 & (1, 11) & (1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1) \\
\end{longtable}
```

Tabuľka vysádzaná podľa predchádzajúceho vzoru je na obr. 6.



R. Blaško: LaTeX A TABUĽKY 1

Tabuľka 1: Záhlavie tabuľky

Hlavička	prvej	podtabuľky
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
		pokračovanie

2 Konferencia OSSConf 2014

Tabuľka 1 – pokračovanie

Hlavička	ďalších	podtabuliek
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
		pokračovanie

R. Blaško: LaTeX A TABUĽKY 3

Tabuľka 1 – pokračovanie

Hlavička	ďalších	podtabuliek
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
		pokračovanie

4 Konferencia OSSConf 2014

Tabuľka 1 – pokračovanie

Hlavička	ďalších	podtabuliek
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)
0	(1, 11)	(1, 2), (3, 1), (17,4), (8,5), (52,33), (0,1)

úplný koniec tabuľky

Obr. 6. tabuľka vysádzaná pomocou `longtable`

Záver

Tento príspevok sa zaoberá tvorbou tabuliek v LaTeX-u. Nie je určený začiatočníkom, predpokladá určité (nie veľmi pokročilé) vedomosti a schopnosti s prácou v LaTeX-u. Úvod je venovaný tabuľkám všeobecne, v prvej časti sú spomenuté niektoré webové on-line nástroje na tvorbu tabuliek. Druhá časť sa zaoberá tvorbou tabuliek s využitím `\multicolumn` a `\multirow` a posledná časť je venovaná „dlhým tabuľkám“ vytvoreným pomocou balíčka `longtable`.

Poďakovanie. Tento príspevok bol podporený grantom slovenskej kultúrno-edukačnej agentúry KEGA č. 011ŽU-4/2014 „Experimentálna matematika – zviditeľnenie neviditeľného“.

Literatúra

[1] BLAŠKO, R.: *LaTeX nie je farba na maľovanie, ale na písanie*, Otvorený softvér vo vzdelávaní, výskume a IT riešeniach, zborník medzinárodnej konferencie OSSConf 2011, Žilina, 6.–9. júla 2011, str. 223–235, ISBN 978-80-970457-1-5.

[2] KOZUBÍK, A.: *Prezentačné materiály v triede beamer*, Otvorený softvér vo vzdelávaní, výskume a IT riešeniach, zborník medzinárodnej konferencie OSSConf 2011, Žilina, 6.–9. júla 2011, str. 249–258, ISBN 978-80-970457-1-5.

[3] RYBIČKA, J.: *LaTeXpro začátečníky*, ISBN 80-7302-049-1, Konvoj Brno, 2003 (3. vydání).

[4] RYBIČKA, J., ČAČKOVÁ, P., PŘICHYSTAL, J.: *Průvodce tvorbou dokumentů*, ISBN 978-80-87106-43-3, 1. vyd. Bučovice: Martin Stříž, 2011.

[5] ŠÍP, R.: *Typografické minimum*, Zväz polygrafie na Slovensku, ISBN 80-967598-5-X, vyd. SOU polygrafické Bratislava 2000.

Kontaktná adresa

RNDr. Rudolf Blaško, PhD., Katedra matematických metód, Fakulta riadenia a informatiky, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovenská republika, *Aktuálna adresa*: SOIT, 010 01 Žilina, Slovenská republika,
E-mailová adresa: `beerb@frcatel.fri.uniza.sk`, `http://frcatel.fri.uniza.sk/~beerb/`

6. ročník Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf 2014

OSSConf 2014: 59–66

OPEN SOURCE AKO NÁSTROJ K ZVIDITEĽNENIU MYŠLIENKOVÝCH EXPERIMENTOV V MATEMATIKE

RUDOLF BLAŠKO (SK) A ALEŠ KOZUBÍK (SK)

Abstrakt. Medzi ľuďmi je veľmi rozšírený názor, že matematika a hlavne jej výučba je niečo „ťažké, ťažko predstaviteľné a nezrozumiteľné“. Základom poznania vo vedeckom výskume je experiment a matematika nie je výnimkou, aj keď zviditeľnenie myšlienkových experimentov nie je jednoduché a donedávna bolo aj nemysliteľné. Dobrú príležitosť nám k tomu dáva výrazný rozvoj výpočtovej techniky v posledných rokoch. Súčasné výpočtové prostriedky spolu s voľne šíriteľným softvérom poskytujú nové možnosti v experimentovaní. Namiesto drahých laboratórnych prístrojov nám postačí počítač. V príspevku predstavíme takýto vzdelávací projekt „Experimentálna matematika – zviditeľnenie neviditeľného“, ktorý získal grant KEGA.

Kľúčové slová. matematika, informatika, názornosť, Open Source, počítačové experimenty, numerické výpočty, symbolické výpočty.

OPEN SOURCE AS A TOOL FOR VISUALIZATION OF THE THOUGHT EXPERIMENTS IN MATHEMATICS

Abstract. Between people is a widespread view that mathematics and especially its teaching is “something difficult, inconceivable and incomprehensible”. The basis of knowledge in scientific research is experiment and mathematics is no exception, although visibility of the thought experiment is not easy and until recently was also unthinkable. Good opportunity gives us significant development of computer technology in recent years. The current computing resources together with free PC software provides new opportunities to experiment. Instead of expensive laboratory equipment the computer will be sufficient. In this paper we introduce such an educational project “Experimental Mathematics – how to see invisible”, which received a grant KEGA.

Keywords. Mathematics, Informatics, Ilustratively, Open Source, Computer Experiments, Numerical Computation, Symbolic Computation.

Úvod

Medzi ľuďmi je veľmi rozšírený názor, že matematika a hlavne jej výučba je niečo „ťažké, ťažko predstaviteľné a nezrozumiteľné“. Buď ju človek vie a nemá problém, alebo ju človek nevie, nerozumie jej a teda sa ju aj nikdy nenaučí. Ale to je veľký omyl, matematika je ako každá iná veda. Základom je talent a zbytok sa človek musí naučiť. To znamená, že talent nestačí, treba sa vzdelávať a mnoho sľubných mladých matematikov doplatilo na lenivosť.

Mnohým ľuďom pomáha pri osvojovaní si nových poznatkov vizuálna predstava – vie si to predstaviť. Vizualizácia v matematike bola donedávna problematická a ťažko realizovateľná. V dnešnej dobe, v dobe búrlivého rozvoja výpočtovej techniky, ako hardvéru, tak aj softvéru, sa blýska na lepšie časy a zviditeľnenie matematiky dostáva iný rozmer.

Základom poznania vo vedeckom výskume je experiment a matematika nie je výnimkou, aj keď zviditeľnenie myšlienkových experimentov nie je jednoduché ale v dnešnej dobe je možné. Ako sme už spomínali, súčasné výpočtové prostriedky spolu s voľne šíriteľným softvérom poskytujú nové možnosti v experimentovaní. Namiesto drahých laboratórnych prístrojov nám postačí počítač. Autori sa pokúsili túto víziu naplniť aj prakticky, podali vzdelávací projekt „Experimentálna matematika – zviditeľnenie neviditeľného“, ktorý získal grant KEGA.

Hlavným cieľom projektu je využívanie slobodného softvéru, žiadne drahé programy. Súčasťou projektu bude uvedenie do prevádzky multifunkčného on-line portálu, ktorý bude vybudovaný pomocou open source (voľne šíriteľných) nástrojov a bude bezplatne voľne prístupný nielen študentom a učiteľom FRI ŽU, ale každému užívateľovi. Budú na ňom implementované interaktívne aplikácie slúžiace na vizualizáciu matematických objektov a modelov, samostatnú experimentálnu činnosť (simulácia vlastných problémov, vytváranie používateľských modelov a 2D, resp. 3D vizualizácia týchto modelov), na on-line tvorbu profesionálnych výstupov vo formáte PDF súborov a interaktívnych prezentácií (taktiež vo formáte PDF).

Projekt bude postavený na základe existujúceho otvoreného softvéru (Python s rôznymi modulmi numpy, scipy, matplotlib ap., ďalej Sage, Maxima, Geogebra, Mayavi2, TeX...) pre numerické i symbolické matematické výpočty a tiež pre dvojrozmernú a trojrozmernú vizualizáciu a následné vhodné prezentovanie získaných výsledkov formou tlačených alebo elektronických publikácií (TeX, pdfLaTeX, AcroTeX, ap.). Na synergické spojenie týchto systémov do webovej aplikácie a ich užívateľsky príjemnú prezentáciu bude využitá vhodná webová platforma (Turbogears, Django, Pylons ap.) s využitím AJAX technológie a vhodných knižníc pre Javascript (napr. ExtJS).

Vytvorená aplikácia bude modulárna a voľne šíriteľná a rozšíriteľná pod licenciou GPL. V rámci projektu bude vytvorená používateľská i vývojárska dokumentácia a následne budú k jednotlivým predmetom participujúcim na tomto projekte vytvorené VŠ učebnice, ktoré budú obsahovať nielen teoretické poznatky a výklad daného predmetu, ale aj praktické príklady a cvičenia, pri riešení ktorých sa budú využívať vytvorené výsledky projektu.

Jedným z cieľom projektu je snaha poopraviť všeobecne zaužívaný názor, že matematika a jej aplikovanie v ostatných vedných disciplínach, v praxi, a v neposlednom rade aj v bežnom živote, musí byť náročné a zložité a ukázať, že to naopak môže byť jednoduché a zaujimavé. K tomuto účelu chceme vybudovať a

prevádzkovať výučbový on-line portál, na ktorom by si mohol každý vyskúšať, namodelovať, vypočítať a v neposlednej miere aj zoradiť do prehľadných výstupov (tabuľky, grafy, formuláre, dotazníky...) a následne vytvoriť kvalitný dokument vhodný do tlače, prípadne interaktívny dokument vhodný na on-line prezentovanie. Budeme pri tom využívať otvorené softvérové nástroje (vlastné a aj voľne prístupné) a spolupracovať s našimi partnermi na Slovensku, v Čechách a v Poľsku. V nasledujúcich častiach si popíšeme niektoré myšlienkové experimenty, s ktorými budú môcť v budúcnosti užívatelia experimentovať.

1. Experiment prvý — aritmetická postupnosť

V dnešnej dobe, keď je naše školstvo takpovediac v kríze sa študenti na väčšine stredných škôl stretnú iba s aritmetickou a geometrickou postupnosťou. Pozrime sa bližšie na aritmetickú postupnosť a na jej prepojenie na teóriu grafov alebo na kombinatoriku.

Aritmetická postupnosť je asi prvá postupnosť čísel, ktorú si človek všimol. Je to postupnosť, v ktorej je rozdiel dvoch susedných členov konštantný (diferencia). Súčet prvých n-členov aritmetickej postupnosti je známy

$$s_n = \frac{n(a_1 + a_n)}{2}.$$

Ale ako sme sa k nemu dostali. Určite si ho naši predkovia nevymysleli, ale dospeli k nemu po množstvách rôznych viac-menej predstaviteľných pokusoch. Najjednoduchšie je aritmetickú postupnosť napísať do dvoch riadkov. Do prvého riadku najmenšieho člena po najväčší a do druhého riadku os najväčšieho po najmenší a po stĺpcoch sčítať:

$$\begin{array}{ccccccccccc} 1 & + & 2 & + & 3 & + & \cdots & + & n & = & s_n \\ n & + & n{-}1 & + & n{-}2 & + & \cdots & + & 1 & = & s_n \\ \hline (n{+}1) & + & (n{+}1) & + & (n{+}1) & + & \cdots & + & (n{+}1) & = & n(n{+}1). \end{array}$$

Ako vidíme súčet dvoch postupností $2s_n = n(n{+}1)$, t. j. $s_n = \frac{n(n+1)}{2}$.

Samozrejme si môžeme tento výsledok overiť aj matematickou indukciou, ale predchádzajúci experiment má váhu dôkazu, takže v tomto kontexte je tento postup zbytočný.

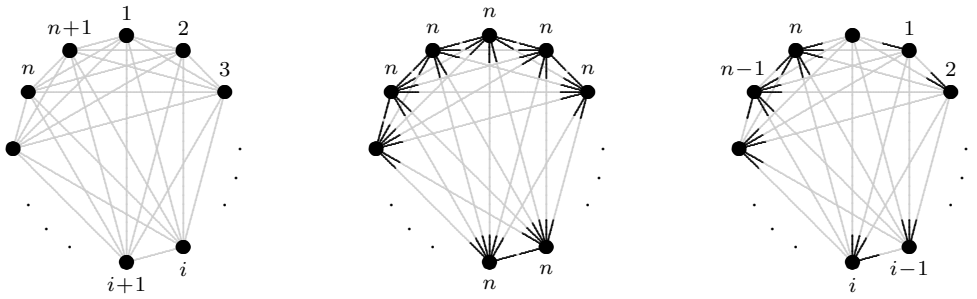
Predstavme si pomaturitné stretnutie po desiatich rokoch. Najprv je v miestnosti stretnutia organizátor sám, potom príde prvý spolužiak a podajú si ruky (t. j. 1-krát), potom príde druhý spolužiak a opäť si podajú ruky (t. j. 2-krát – nový spolužiak podal ruku tým v miestnosti, alebo naopak študenti v miestnosti podali ruku novému spolužiakovi) a takto sa to opakuje aspoň tridsať-krát (kvôli jednoznačnosti 30 študentov). Otázka je: „Koľkokrát si podali ruky?“.

Z uvedenej formulácie sa nám to javí ako kombinatorický problém. Je 30 študentov a podaní rúk bude $\binom{30}{2}$. Ale keď sa nato pozeráme jednotlivo je to súčet aritmetickej postupnosti $1 + \cdots + 29 = \frac{29(1+29)}{2}$, t. j. $\binom{30}{2} = \frac{29(1+29)}{2}$.

Teraz sa na to pozrime ako problém teórie grafov. Zoraďme študentov do kruhu a každého z nich označme vrcholom s poradovým číslom (obr. 1 vľavo). V tomto prípade môžeme podanie ruky medzi jednotlivými spolužiakmi reprezentovať hranou grafu. Otázka súčtu podania rúk je ekvivalentná súčtu všetkých hrán a keďže uvedený graf je úplný, predstavuje súčet všetkých hrán úplného grafu.

Pre zjednodušenie sme označili počet vrcholov $n+1$ (namiesto n). Poďme počítať celkový počet hrán (ako uvidíme súčet aritmetickej postupnosti). Na obr. 1 v strede vidíme, že z každého vrchola (je ich $n+1$) vychádza n hrán, t. j. celkovo $n(n+1)$ hrán, ale keďže každá je započítaná dva razy, ich celkový súčet je $\frac{n(n+1)}{2}$.

Na druhej strane (obr. 1 vpravo) z vrchola $n+1$ vychádza n hrán, z vrchola n vychádza ďalších $n-1$ hrán, ..., z vrchola 3 vychádzajú dve nové hrany a z vrchola 2 vychádza jedna hrana, t. j. máme aritmetickú postupnosť a hľadáme jej súčet.



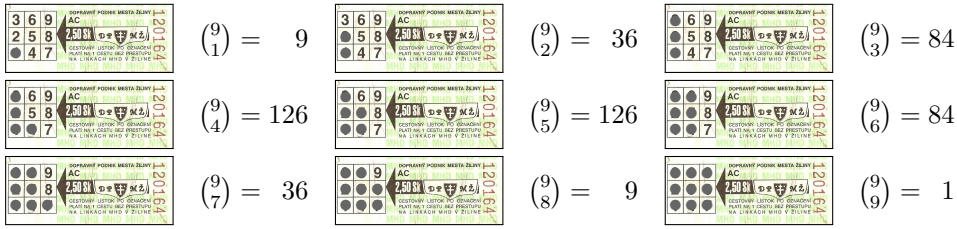
Obr. 1. Aritmetická postupnosť a teória grafov

2. Experiment druhý — cestovný lístok

V dobách nedávno minulých, keď bola úroveň počítadiel cestovných lístkov na úrovni dierných pások sa používalo k označeniu bielkeho pasažiera perforovanie papierových cestovných lístkov (obr. 2). Systém bol jednoduchý, na lístku bolo 9 čísel a tie bolo treba perforovať v mechanickom strojčeku, ktorý sa nachádzal v autobuse. Najčastejšie sa používali tri alebo štyri dierky (84 alebo 126 rôznych možností dierovania). Keďže doba už značne pokročila a dnešní mladší kolegovia si takéto lístky už nepamätajú, na obrázku 2 je dobová fotografia takéhoto lístka (foto: Peter Šimko, zdroj: `http://imhd.zoznam.sk/za/doc/sk/12545/Historia-zilinskej-MHD-od-roku-1987-do-sucasnosti.html`)

Našou úlohou je vypočítať koľko rôznych možností existuje na prederavenie nášho cestovného lístka. Môžeme sa na to pozerať z dvoch rôznych uhlov.

1. pohľad. Na každé číslo máme dve možnosti – prederavíme číslo alebo nie. Spolu je to $2 \cdot 2 \cdots 2 = 2^9$ možností, t. j. 512 možností. V reči stredoškolskej



Obr. 2. Cestovný lístok a kombinatorika

matematiky sú to variácie s opakovaním, ale priemerne vzdelaný sedliak tento názov nepotrebuje. Aby bol súčet presný, musíme odpočítať počet neperforovaných lístkov, t. j. náš výsledok je 511.

2. pohľad. Na to aby sme perforovali lístok jednou dierou (obr. 2), máme 9 možností (kombinácie prvej triedy z deviatich prvkov). Na to aby sme označili lístok dvomi dierami máme 36 možností (kombinácie druhej triedy z deviatich prvkov). Takto môžeme pokračovať až po deväť dier a v súvislosti s prvým pohľadom dostaneme vzťah:

$$\sum_{i=1}^{9} \binom{9}{i} = \binom{9}{1} + \binom{9}{2} + \cdots + \binom{9}{9} = 2^9 - 1 = 511.$$

Ale tu máme pekný príklad na aplikáciu binomickej vety:

$$\sum_{i=0}^{n} \binom{n}{i} a^i b^{n-i} = (a+b)^n.$$

Koeficienty v binomickej vete korešpondujú s členmi v Pascalovom trojuholníku (obr. 3). Ak zoberieme predchádzajúci príklad, potom $\sum_{i=0}^{9} \binom{9}{i} = 2^9$, t. j. súčet binomických čísel v riadku Pascalovho trojuholníka je rovný mocnine dva na riadok. To súhlasí s binomickou vetou:

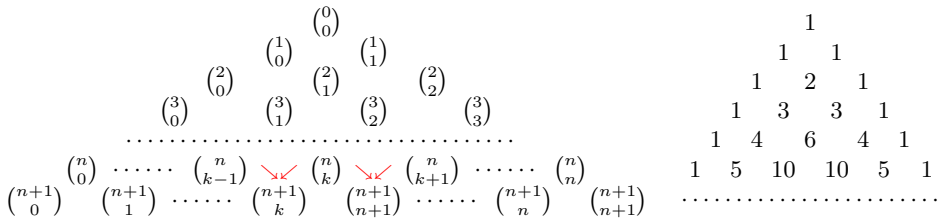
$$\sum_{i=0}^{n} \binom{n}{i} 1^i 1^{n-i} = (1+1)^n = 2^n.$$

Na druhej strane, ak uvažujeme binomické koeficienty v riadku Pascalovho trojuholníka striedavo kladné a záporné, ich súčet je nulový:

$$\sum_{i=0}^{n} \binom{n}{i} 1^i (-1)^{n-i} = (1-1)^n = 0^n.$$

Z konštrukcie Pascalovho trojuholníka vieme, že súčet člena v nižšom riadku sa rovná súčtu členov nad ním (obr. 3 svetlé šípky), t. j. platí:

$$\binom{n}{k-1} + \binom{n}{k} = \binom{n+1}{k}, \quad \text{resp. } \binom{n}{k} + \binom{n}{k+1} = \binom{n+1}{k+1}.$$



Obr. 3. Pascalov trojuholník

3. Experiment tretí — Riemannov integrál

Aj výstavbu pojmu určitý integrál môžeme chápať ako výsledok určitého myšlienkového experimentu. Na počiatku je motivácia určením veľkosti plochy ohraničenej grafom funkcie a x-ovou na určitom intervale $\langle a, b\rangle$. Aj pri intuitívnom ponímaní pojmu plochy ľahko pochopíme, že jej veľkosť môžeme približne určiť tak, že interval $\langle a, b\rangle$ rozdelíme pomocou deliacich bodov $x_0 = a, x_1, \ldots, x_n = b$ na konečný počet podintervalov $\langle x_{i-1}, x_i\rangle$. Nad každým takýmto intervalom zostrojíme obdĺžnik, ktorého výška bude rovná maximálnej resp. minimálnej hodnote danej funkcie $f(x)$ na príslušnom intervale $\langle x_{i-1}, x_i\rangle$ a sčítame plochy týchto obdĺžnikov (obr. 4). Je prirodzené očakávať, že pri jemnejšom delení, teda pri väčšom počte deliacich bodov získame presnejšiu aproximáciu skutočnej veľkosti danej plochy.

Ak označíme $m_i = \inf\{f(x), x \in \langle x_{i-1}, x_i\rangle\}$ a $M_i = \sup\{f(x), x \in \langle x_{i-1}, x_i\rangle\}$, potom môžeme definovať dolný resp. horný integrálny súčet vzťahmi (obr. 4):

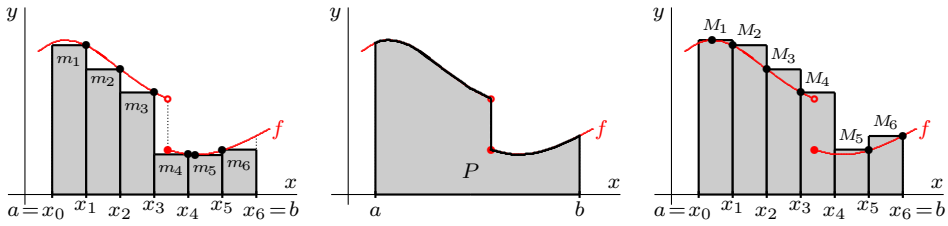
$$s = \sum_{i=1}^{n} m_i(x_i - x_{i-1}), \quad \text{resp. } S = \sum_{i=1}^{n} M_i(x_i - x_{i-1}).$$

Ak budeme zjemňovať delenie, t. j. budeme zmenšovať vzdialenosť medzi bodmi x_{i-1} a x_i, bude sa hodnota súčtov s a S približovať k hodnote integrálu $\int_a^b f(x)\,\mathrm{d}x$. Z geometrického hľadiska predstavuje určitý integrál obsah plochy určenej funkciou f a osou x, tzv. krivočiary lichobežník P (obr. 4). Zväčšovanie, resp. zmenšovanie počtu deliacich bodov môžeme jednoducho simulovať napríklad v programoch Geogebra alebo Maxima.

Vzťah medzi určitým a neurčitým integrálom vyjadruje Newton-Leibnizov vzorec (základná veta integrálneho počtu). Ak $F(x)$ je ľubovoľná primitívna funkcia k funkcii $f(x)$ na intervale $\langle a, b\rangle$, potom hodnota určitého integrálu funckie $f(x)$ na intervale $\langle a, b\rangle$, sa rovná

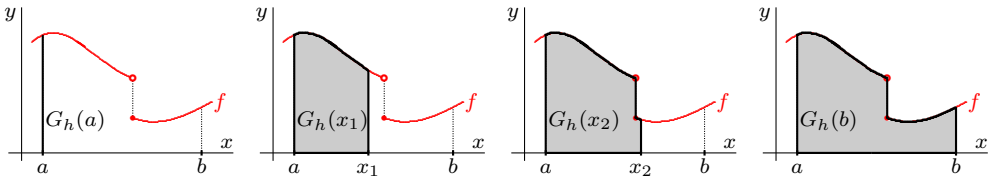
$$\int_a^b f(x)\,\mathrm{d}x = F(b) - F(a).$$

Z teórie integrálneho počtu je známe, že všetky primitívne funkcie k funkcii $f(x)$ na intervale $\langle a, b\rangle$ sú spojité a líšia sa o reálnu konštantu. Z uvedeného dôvodu



Obr. 4. Krivočiary lichobežník P určený funkciou f na intervale $\langle a;\ b\rangle$ a jeho aproximácia pomocou dolných (vľavo) a horných (vpravo) integrálnych súčtov

môžeme ilustrovať vzťah medzi primitívnou funkciou a určitým integrálom pomocou integrálu ako funkcie hornej hranice (hornej medze). Táto reprezentácia je prirodzená a veľmi dobre ilustruje vzťah medzi určitým a neurčitým integrálom (obr. 5). Aj predstava spojitosti primitívnej funkcie je prirodzená a veľmi dobre viditeľná.



Obr. 5. Integrál ako funkcia hornej hranice

Záver

V príspevku sme na niekoľkých jednoduchých príkladoch ilustrovali, ako je možné s využitím moderných softvérových technológií vizualizovať niektoré myšlienkové experimenty v matematike. V budúcnosti, v rámci projektu, chceme ponúknuť užívateľom oveľa väčšie množstvo možností na experimentovanie – a to nielen nami predpripravené experimenty, ale aj možnosť vytvárať vlastné experimenty. Aj keď v súčasnosti stojíme na prahu nového začiatku a nachádzame sa len v štádiu výberového konania na nákup potrebného servera, potrebného pre sľubovaný on-line portál, myslíme, že cesta týmto smerom je správna. Je teda čas tvorby a očakávaný zber ovocia je ďaleko. Ostáva nám teda len veriť, že na konci nášho projektu sa budeme mať s Vami o čo rozdeliť.

Poďakovanie. Tento príspevok bol podporený grantom slovenskej kultúrno-edukačnej agentúry KEGA č. 011ŽU-4/2014 „Experimentálna matematika – zviditeľnenie neviditeľného“.

Literatúra

[1] BLAŠKO, R.: *Matematická analýza 1*, Žilina, EDIS 2009, ISBN 978-80-554-0119-5.
[2] BLAŠKO, R.: *Neurčitý a určitý integrál reálnej funkcie* (učebné texty), `http://frcatel.fri.uniza.sk/users/beerb/ma1/sa2.pdf`.
[3] BLAŠKO, R.: *LaTeX nie je farba na maľovanie*, Otvorený softvér vo vzdelávaní, výskume a IT riešeniach, zborník medzinárodnej konferencie OSSConf 2010, Žilina, 1.–4. júla 2010, str. 43–52, ISBN 978-80-970457-0-8, `http://sospreskoly.org/files/OSSConf2010/ossconf2010-Blasko.pdf`.
[4] BUŠA, J.: *MAXIMA*, Slovak Open Source Initiative, FEI TU Košice, 2006, ISBN 80-8073-640-5.
[5] KAUKIČ, M.: *Open Source softvér vo výuke matematiky a výskume*, 4. Konference o matematice a fyzice na vysokých školách technických, Sborník příspěvků, Brno, Univerzita Obrany, 2005, str. 80–85, ISBN 80-85960-91-5.
[6] KAUKIČ, M.: *Funkcie komplexnej premennej s podporou Open Source softvéru*, 5. Konference o matematice a fyzice na vysokých školách technických, Sborník příspěvků, Brno, Univerzita Obrany, 2007, str. 175–184, ISBN 978-80-7231-274-0.
[7] KOZUBÍK, A.: *Využití programu MAXIMA při vyučování matematické analýzy*, 5. Konference o matematice a fyzice na vysokých školách technických, Sborník příspěvků, Brno, Univerzita Obrany, 2007, str. 175–184, ISBN 978-80-7231-274-0.

Kontaktné adresy

RNDr. Rudolf Blaško, PhD., Katedra matematických metód, Fakulta riadenia a informatiky, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovenská republika,
E-mailová adresa: `beerb@frcatel.fri.uniza.sk`, `http://frcatel.fri.uniza.sk/~beerb/`

RNDr. Aleš Kozubík, PhD., Katedra matematických metód, Fakulta riadenia a informatiky, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovenská republika,
E-mailová adresa: `alesko@frcatel.fri.uniza.sk`

6. ročník Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf
OSSConf 2014: 67–72

VÝHODY POUŽITIA DYNAMICKÉHO GRAFICKÉHO SOFTVÉRU PRI OVEROVANÍ VZŤAHOV MEDZI GEOMETRICKÝMI ÚTVARMI V ŠKOLSKEJ MATEMATIKE

ZUZANA BORČINOVÁ (SK)

Abstrakt. Článok ukazuje možnosť zefektívnenia výučby planimetrie na strednej škole s využitím dynamického grafického softvéru. Súčasťou článku sú vybrané úlohy pojednávajúce o štyroch význačných priesečníkoch v trojuholníku. Ukážky riešenia sú spracované použitím otvoreného softvéru GeoGebra.

Kľúčové slová. GeoGebra, planimetria, dynamický grafický softvér, otvorený softvér, význačné priesečníky v trojuholníku, Eulerova priamka, Feuerbachova kružnica.

BENEFITS OF USING DYNAMIC GRAPHIC SOFTWARE FOR VERIFYING RELATIONS BETWEEN GEOMETRICAL OBJECTS IN SCHOOL MATHEMATICS

Abstract. Paper describes the possibility of enhancing the effectivity of teaching planimetrics in high school using dynamic graphic software. The paper includes selected excersises focused on four significant intersections in triangle. Sample solutions are created using opensource system GeoGebra.

Keywords. GeoGebra, planimetrics, dynamic graphic software, open software, significant intersections of triangle, Euler line, Feuerbach circle.

Úvod

Na hodinách geometrie často riešime dilemu, či zaradiť úlohu, ktorá je síce zaujímavá, ale súčasne je aj časovo náročná. Navyše konštrukčné úlohy si vyžadujú vysokú presnosť pri rysovaní, čo v školskej matematike nie je jednoduché dosiahnuť. Na príčine sú nielen manuálne zručnosti žiakov, ale aj kvalita rysovacích pomôcok, ktoré používajú. Tento problém sa dá eliminovať použitím dynamického geometrického softvéru, napríklad GeoGebra.

Nástroje geometrického softvéru poskytujú možnosť automatickej konštrukcie základných útvarov (stred úsečky, os úsečky, kolmica a pod.), čím sa práca zefektívni a žiaci sa môžu sústrediť na to, čo je v riešení danej konštrukčnej úlohy podstatné. Okrem toho je výsledok omnoho prehľadnejší, než pri podrobnej konštrukcii so všetkými pomocnými čiarami. Za ďalší bonus možno považovať fakt, že program GeoGebra je bezplatný, takže ho môžu žiaci používať aj pri domácej príprave, či už pri opakovaní (prehrávaním vyriešených úloh) alebo pri riešení domácej úlohy.

1. Úlohy o štyroch význačných priesečníkoch v trojuholníku

Jednou zo základných tém planimetrie, ktoré sa vyučujú na strednej škole je úloha o štyroch význačných priesečníkoch v trojuholníku [3], za ktoré sa považujú stred kružnice opísanej trojuholníku, stred kružnice vpísanej do trojuholníka, priesečník výšok a ťažisko. V nasledujúcich ukážkach sa zameriame na overovanie ich vlastností pomocou programu GeoGebra.

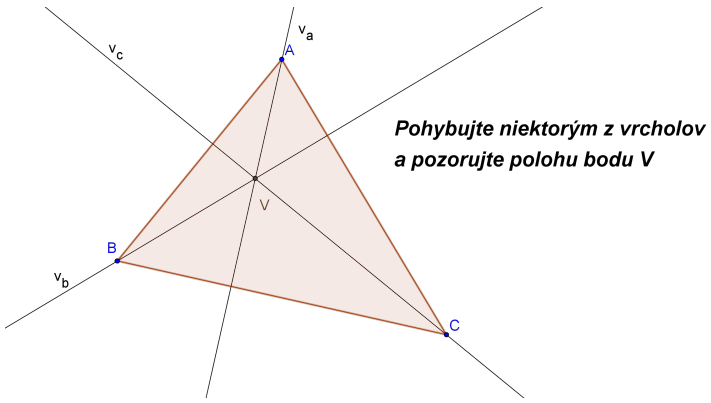
1.1. Úloha 1.

V prvej časti overíme základné tvrdenie, že výšky ľubovoľného trojuholníka sa pretínajú v jednom bode, ktorý sa tiež niekedy nazýva **ortocentrum**.

Zadanie:
Overte, že v každom trojuholníku sa všetky tri výšky pretínajú v jednom bode.

Riešenie:
Zostrojíme trojuholník ABC, ktorý nie je pravouhlý ani rovnostranný a v ňom výšku na stranu a (priamka v_a), výšku na stranu b (priamka v_b). Vyznačíme priesečník V výšok v_a a v_b. Potom zostrojíme výšku na stranu c (priamka v_c) a ukážeme, že aj táto výška prechádza bodom V (obr. 1).

Pohybovaním niektorého z vrcholov trojuholníka budeme meniť tvar trojuholníka a pritom budeme pozorovať polohu bodu V. Bod V sa bude pohybovať len po priamke v_c. Takto overíme, že bod V je spoločným priesečníkom všetkých troch výšok v trojuholníku, bez ohľadu na to, aký je to trojuholník.



Obr. 1. Ukážka z programu GeoGebra, ktorá názorne ilustruje prienik výšok trojuholníka v jedinom bode

Analogickým spôsobom môžeme overiť aj ostatné tvrdenia o význačných priesečníkoch v trojuholníku:

- Všetky osi strán trojuholníka sa pretínajú v jednom bode (**stred kružnice opísanej trojuholníku**).

- Všetky osi vnútorných uhlov trojuholníka sa pretínajú v jednom bode (**stred kružnice vpísanej trojuholníku**).
- Všetky ťažnice trojuholníka sa pretínajú v jednom bode (**ťažisko**).

Situáciu v špeciálnych prípadoch – v pravouhlom, rovnostrannom alebo rovnoramennom trojuholníku – môžu žiaci preskúmať v rámci domácej úlohy.

V ďalších ukážkach overíme dve zaujímavé tvrdenia o týchto význačných priesečníkoch v trojuholníku, ktoré súvisia s pojmami Eulerova priamka a Feuerbachova kružnica.

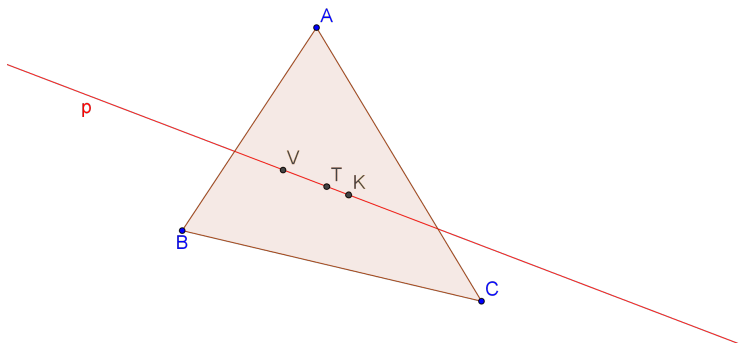
1.2. Úloha 2.

Tvrdenie, ktorého platnosť experimentálne overíme v nasledujúcej úlohe, prvýkrát dokázal švajčiarsky matematik Leonhard Euler. Hovorí, že v každom trojuholníku priesečník výšok, ťažisko a stred opísanej kružnice ležia na jednej priamke. Táto priamka sa nazýva **Eulerova priamka trojuholníka**.

Zadanie:
Overte, že v každom trojuholníku priesečník výšok V, ťažisko T a stred K opísanej kružnice ležia na jednej priamke.

Riešenie: Zostrojíme všeobecný trojuholník ABC a v ňom postupne skonštruujeme priesečník výšok V, ťažisko T a stred K opísanej kružnice. Kvôli prehľadnosti obrázka pomocné čiary skryjeme. Bodmi V a T preložíme priamku p a ukážeme, že aj bod K leží na tejto priamke (obr. 2).

Potom pohybujeme niektorým z vrcholov trojuholníka a pozorujeme polohu bodu K vzhľadom na priamku p. Bod K sa bude pohybovať len po priamke p. Tým sa presvedčíme, že tieto tri body vždy ležia na jednej priamke.



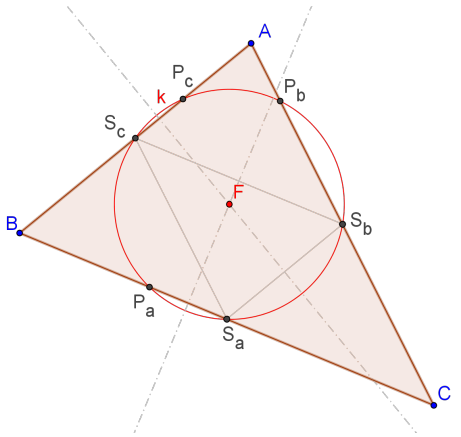
Obr. 2. Eulerova priamka

1.3. Úloha 3.

V ďalších dvoch úlohách sa presvedčíme, že stredy strán a päty výšok trojuholníka ležia na jednej kružnici, ktorej stred leží v strede úsečky určenej priesečníkom výšok a stredom kružnice opísanej trojuholníku.

Zadanie:
Zostrojte kružnicu k, ktorá prechádza stredmi strán trojuholníka ABC. Overte, že na tejto kružnici ležia aj päty výšok trojuholníka ABC.

Riešenie: Zostrojíme všeobecný trojuholník ABC a v ňom postupne vyznačíme stredy strán (body S_a, S_b, S_c) a päty výšok (body P_a, P_b, P_c). Kvôli prehľadnosti obrázka pomocné čiary skryjeme. Skonštruujeme kružnicu k, ktorá prechádza bodmi S_a, S_b a S_c. Kružnica k je kružnicou opísanou trojuholníku $S_aS_bS_c$, t. j. jej stred F je priesečník osí strán trojuholníka $S_aS_bS_c$. Ukážeme, že aj body P_a, P_b a P_c ležia na kružnici k, a to aj vtedy, ak zmeníme tvar trojuholníka (obr. 3).

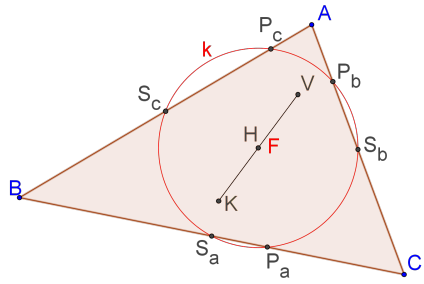


Obr. 3. Stredy strán a päty výšok v trojuholníku ležia na jednej kružnici

1.4. Úloha 4.

Zadanie:
Overte, že stredom kružnice k, ktorá prechádza stredmi strán trojuholníka, je stred úsečky určenej priesečníkom výšok a stredom kružnice opísanej trojuholníku.

Riešenie: V konštrukcii z predchádzajúcej úlohy skryjeme pomocné čiary a zobrazíme priesečník výšok (bod V) a stred kružnice opísanej trojuholníku ABC (bod K). Zostrojíme úsečku VK a vyznačíme jej stred H. Overíme, že stred kružnice k (bod F) je s ním totožný (obr. 4). Opäť budeme meniť tvar trojuholníka a pritom sledovať, že vzájomná poloha bodov F a H sa nezmení.



Obr. 4. Stred kružnice k leží v strede úsečky VK

1.5. Úloha 5.

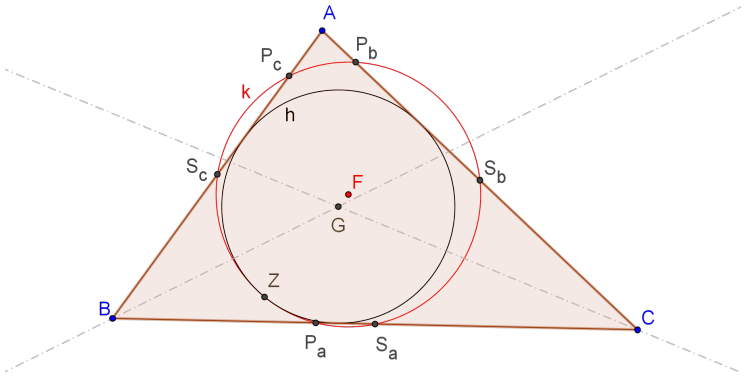
Kružnica k sa nazýva **Feuerbachova kružnica** podľa nemeckého matematika Karla Feuerbacha, ktorý dokázal, že táto kružnica sa zvnútra dotýka kružnice vpísanej trojuholníku. V tejto úlohe overíme toto tvrdenie.

Zadanie:
Overte, že kružnica k sa zvnútra dotýka kružnice h vpísanej trojuholníku ABC.

Riešenie: Najprv zostrojíme kružnicu h vpísanú trojuholníku ABC (jej stred G je priesečníkom osí vnútorných uhlov trojuholníka ABC) a skryjeme pomocné čiary. Vyznačíme priesečník kružnice k a kružnice h (bod Z). Ukážeme, že aj pri zmene trojuholníka, majú kružnice k a h práve jeden spoločný bod, čiže sa vzájomne dotýkajú (obr. 5).

Záver

„Kto chce pochopiť základy, mal by najprv rysovať sám. Skúsenosti a porozumenie získané vlastnou manipuláciou s kružidlom, pravítkom a uhlomerom nemôže v tejto prvej etape nahradiť žiadny počítačový program. [3]“ V článku sme chceli ukázať, že v ďalšej etape, keď majú žiaci objavovať vlastnosti a vzťahy medzi geometrickými objektmi a získané poznatky využívať pri riešení geometrických konštrukčných úloh, môže byť geometrický softvér vhodným prostriedkom na zefektívnenie výučby planimetrie. Presnosť počítačovej konštrukcie umožňuje ľahko demonštrovať a overovať vlastnosti útvarov a vďaka dynamickosti a interaktívnosti softvéru je možné preskúmať väčšie množstvo rôznych prípadov, než pri ručnom rysovaní.



Obr. 5. Feuerbachova kružnica sa zvnútra dotýka kružnice vpísanej trojuholníku

Poďakovanie. Tento príspevok bol podporený grantom slovenskej kultúrno-edukačnej agentúry KEGA č. 011ŽU-4/2014 „Experimentálna matematika – zviditeľnenie neviditeľného“.

Literatúra

[1] HOHENWARTER, M., PREINER, J.: *GeoGebra nápoveda 3.0*, dokument, 2007, `http://www.geogebra.org/help/docucz/`,.
[2] JANČAŘÍK, A.: *Geometrie, GeoGebra a potreba presnosti* dokument, 2012, `http://clanky.rvp.cz/clanek/k/z/14871/GEOMETRIE-GEOGEBRA-A-...3/`.
[3] KUBÁČEK, Z.: *Matematika pre 3. ročník gymnázia a 7. ročník gymnázia s osemročným štúdiom*, 2.časť, Bratislava, SPN, 2013, ISBN 978-80-10-02289-2.
[4] STODOLOVÁ, K.: *GeoGebra* seminární práce, květen 2011, `http://www.karlin.mff.cuni.cz/katedry/kdm/diplomky/kristyna_stodolova.sem/geogebra.html`.
[5] GeoGebra, `http://www.geogebra.org`.

Kontaktná adresa

RNDr. Zuzana Borčinová, Katedra matematických metód, Fakulta riadenia a informatiky, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovenská republika,
E-mailová adresa: `borcinova@frcatel.fri.uniza.sk`

6. ročník Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf
OSSConf 2014: 73–78

OTEVŘENÝ (?) INFORMAČNÍ SYSTÉM PRO ZÁKLADNÍ ŠKOLU

PETRA ČAČKOVÁ (CZ)

Abstrakt. Tento článek se zabývá výběrem informačního systému pro (nejen) základní školu. Škola má specifickou organizační strukturu – jde o soukromý subjekt zahrnující základní i mateřskou školu a též tzv. studijní centrum, které organizuje doučování. Běžně dostupné školní informační systémy nenaplňují všechny požadavky z této struktury vyplývající, škola proto hledá přizpůsobitelný informační systém. Dále je třeba optimalizovat finanční náklady. Obojí hovoří pro využití otevřeného softwaru. Software splňující požadavky však není k dispozici, proto bude třeba hledat jiné vhodné řešení.

Klíčová slova. Informační systém, otevřený software, základní škola.

OPEN-SOURCE (?) PRIMARY SCHOOL INFORMATION SYSTEM

Abstract. This article deals with the selection of information system for a (not only) primary school. The school has a specific organizational structure – it is a private entity comprising a primary school and a kindergarten, as well as a study center which organizes remedial classes. Commonly available school information systems do not meet all the requirements of this structure, the school is therefore looking for a customizable information system. It is also necessary to optimize the costs. Both factors lead to the use of open source software. However, software that meets the requirements is not available, so other suitable solution will be sought.

Keywords. Information system, Open-Source software, Primary school.

Úvod

Tento článek se zabývá výběrem informačního systému pro (nejen) základní školu. To „nejen“ je důležité – školních informačních systémů totiž existuje celá řada (viz dále), ale tato škola je specifická a má tedy specifické požadavky. Budeme pro ni hledat informační systém, který bude respektovat její organizační strukturu. Dále je třeba optimalizovat finanční náklady. Obojí hovoří pro využití otevřeného softwaru.

1. Představení školy

Základní škola a mateřská škola Basic, o. p. s. je soukromá brněnská škola. Má specifickou organizační strukturu, zahrnuje:

- základní školu – devět ročníků, ale s nízkým počtem žáků ve třídách (max. 12 žáků; v současné době celkem 42 žáků),
- mateřskou školu – nyní max. 20 dětí,
- studijní centrum, které organizuje doučování pro žáky z řad široké veřejnosti (kapacita téměř neomezená).

Všechny tyto organizační součásti sídlí v jedné budově a také zaměstnanci často pracují ve více než jedné organizaci, i veškerá evidence musí proto být provázaná. Klasický informační systém pro školy tedy nestačí, potřebujeme pokrýt i další součásti školy. Naopak vzhledem k velikosti školy není nutná rozsáhlá elektronizace kde čeho (třídnice, žákovská, rozvrh).

Vybavenost školy výpočetní technikou není velká – jedna malá (mobilní) učebna výpočetní techniky, počítač v každé třídě, dva počítače pro administrativu – vše starší stroje s OS Windows XP. Projektory, interaktivní tabule či tablety nejsou a prozatím se neplánují (dáno nízkým počtem žáků i finanční náročností). Rovněž informační systém se zatím nepoužívá žádný (administrativa se řeší v tabulkovém procesoru), ale s rostoucím počtem žáků se administrativní zátěž zvyšuje a informační systém by se velmi hodil.

Škola tak má v zásadě tři možnosti:

1. nechat si IS vyvinout na zakázku,
2. vybrat některý z existujících informačních systémů:
   (a) dobrých a profesionálně řešených, ale uzavřených a komerčních,
   (b) jednodušších, ale možná dostačujících, nejlépe otevřených a finančně nenáročných.

Možnost 1 by byla kvůli specifické organizační struktuře a speciálním požadavkům nejlepší, software by realizovala spřátelená firma (obeznámená s těmito požadavky) za výhodnou cenu. I tak představuje vývoj vlastního IS nemalou investici. Možnost 2 znamená hledání co nejlepší shody s požadavky při co nejmenší finanční zátěži, budeme tedy upřednostňovat možnost 2b.

1.1. Požadavky na informační systém

Před výběrem informačního systému je třeba stanovit kritéria pro výběr. Přehled kritérií, získaných z diskuzí s učiteli a řediteli škol, přináší např. [1]. Uvádí zde např. rozšířenost IS, zázemí a renomé výrobce, podporu uživatelů, možnost IS vyzkoušet, mít známého/kolegu k dispozici (v roli konzultanta), komplexnost, rozšiřitelnost, přístup k datům přes internet, aktualizace, cenu.

Tato kritéria mohou mít různou váhu podle okolností. Naše škola se nachází v situaci, kdy IS ještě nutně nepotřebuje, ani ji nic „netlačí“. Hlavním kritériem bude cena, dále přizpůsobitelnost/modularita (vzhledem k výše zmíněné specifické

organizační struktuře) a podpora ze strany dodavatele (aktualizace, možnost modifikace a nastavení dle potřeb). Z hlediska softwaru: nejvhodnější bude nástroj dostupný on-line, desktopová instalace bude spíše nevhodná. Z hlediska hardwaru: škola bude preferovat řešení, při kterém nebude potřebovat vlastní server.

2. Hledáme informační systém

2.1. Hotové školní informační systémy

Nejprve se podívejme na možnosti nejrozšířenějších školních informačních systémů.

Snad nejobvyklejším systémem je IS *Bakaláři*. Má na školách již dlouhou tradici. Je vhodný pro různé typy škol, má mnoho potřebných funkcionalit. K dispozici je i shareware; doplňky existujících modulů od dalších autorů. IS Bakaláři je třeba instalovat na server i na klientské stanice (tyto vyžadují OS Windows). Cena licence pro školu s 200 žáky je 2 500 Kč (čistě pro ZŠ by bylo možné obstarat tzv. malou evidenci omezenou na 100 žáků za 1 600 Kč).

Dalším často používaným systémem je *SAS* [3]. Aplikace je desktopová, se složitou licenční politikou. Cena je cca 10 000 Kč za rok pro ZŠ a 2 700 Kč pro MŠ, proto se tímto systémem nebudeme dále zabývat.

Škola OnLine [4] je už podle názvu webový informační systém. Poskytuje všechny obvyklé funkce požadované na ZŠ. Cena se odvíjí podle počtu žáků a požadovaných modulů, pro školu se 120 žáky vyjde na 3 500 Kč za rok.

Na podobném principu je založena *iŠkola* [5]. Výhodou je, že není určena jen pro školy (základní, střední. . .), ale i další instituce zaměřené na vzdělávání, takže by snad mohla být vhodná i pro mateřskou školu a studijní centrum. Cena za roční licenci pro 100 žáků je 2 400 Kč a obsahuje všechny nabízené moduly (i ty, které nebudou využívány).

IS *Edookit* [6] je moderní online systém, který opět obsahuje potřebné funkce pro školu. Kromě toho zdůrazňuje rychlost, zabezpečení a orientaci na mobilní dotyková zařízení. Snad jako jediný nabízí i variantu přímo pro mateřské školy [7]. Cena se skládá z registračního poplatku a měsíčního paušálu. Náklady pro ZŠ přepočítané na rok provozu tak dosahují částky přes 10 000 Kč, u MŠ okolo 4 500 Kč.

Ještě je třeba zmínit se o systému *E-třídnice* [8], což je jednoduchý IS skládající se z modulů Elektronická třídní kniha, Elektronická žákovská knížka a Elektronický deník praxe. Toto zaměření nekoresponduje s potřebami školy.

Všechny zmíněné systémy jsou však k dispozici jen jako komerční software, patřičně placený, uzavřený a bez možnosti změn a přizpůsobení organizační struktuře (ZŠ, MŠ a studijní centrum). Navíc nabízejí – vzhledem k potřebám této školy – mnoho možností, které nejsou ani tolik vyžadovány. Všechny tyto softwary proto zamítneme.

Dalším řešením by mohl být informační systém z mimoškolní oblasti – zde se nabízí např. *Helios Fenix* [9], jenž by byl použitelný pro školu, i když je určen hlavně pro státní správu (územní subsystém, agendy týkající se přestupků apod.). Ceník na webu není, ani cenovou náročnost nelze odhadnout. Jedná se ovšem opět o komerční řešení a byť by pro neziskové organizace byla cena možná nižší, stále je to uzavřený produkt. Z těchto dvou důvodů nebudou dále zkoumány informační systémy určené primárně pro podniky, kterých je velmi mnoho (výběr umožňuje například konfigurátor [10]).

Pojďme nyní do oblasti otevřeného softwaru. Aplikací, i vhodných pro školy, je mnoho (přehled uvádí např. [11]). Informační systém však mezi nimi těžko hledat, natož školní. Mnohem povzbudivější výsledky však přineslo lovení v zahraničních vodách.

OpenSIS [12] je open-source školní informační systém, přizpůsobitelný, náklady jsou nízké (za technickou podporu). Jako otevřený software by byl přizpůsobitelný (a přeložitelný; je v angličtině). Dále by bylo třeba vyřešit odesílání dat školní matriky, což v zámořském systému pochopitelně není. Použití tohoto systému by bylo schůdné, jen pokud by se našel někdo, kdo by systém celý přeložil a významně přizpůsobil – z hlediska času i nákladů se dostáváme blízko k variantě vývoje vlastního informačního systému.

SchoolTool [13] je open-source webový informační systém pro školy v rozvíjejícím se světě, se silnou podporou pro překlady a lokalizaci (do češtiny a slovenštiny je nyní přeložena zhruba polovina). Je dostupný pouze pro Ubuntu (pro Windows sice také, ale s možností vzniku chyb, což je pro školní IS nepřijatelné). Systém s výbornou myšlenkou a šikovnými funkcemi by však bylo nutné pro potřeby školy upravit, čímž se dostáváme k problémům popsaným výše.

2.2. Vlastní systém?

Dalším možným řešením je vývoj vlastního systému. Tímto směrem se situace ubírá, pokud se zjistí, že existující školní IS jsou pro školu moc drahé/zbytečné/-nepoužitelné nebo jinak nevhodné.

Tvorba školního IS je také tématem různých kvalifikačních prací (např. [14]), vděčným námětem jsou i mateřské školy (např. [15, 16]), na něž se komerční výrobci téměř nesoustředí. Tyto systémy jsou však výstupem kvalifikačních prací a pokud byly vůbec dovedeny do fáze realizace, pak často pouze pilotně, pro jedinou školu a také často bez vize systém udržovat, podporovat a vyvíjet.

Závěr

Optimální řešení nebylo nalezeno[1]. Školní informační systémy jsou buďto:

(1) komerční (drahé) a těžko přizpůsobitelné (obyčejně jen volbou modulů),

[1]Pokud kdokoli, kdo čte tento text, našel šikovné řešení, ráda o něm uslyším.

nebo:

(2) otevřené, cizojazyčné; sice přizpůsobitelné, ale těžko (kdo bude zdlouhavě studovat a poté upravovat zdrojové kódy?), a protože ani programátor zadarmo nehrabe, tak také drahé.

Tedy docházím k rovnici (1) = (2). Problém volby informačního systému se pak dá vyřešit dočasnou variantou spočívající ve využití databázového systému (Base) z některého otevřeného kancelářského balíku, pomocí nějž se vybuduje základní evidence. Další možností je přesvědčit softwarovou firmu, aby systém vyvinula na míru, ale jako open-source, který by pak mohla využívat nejen studijní centra po celé ČR a SR, ale i další vzdělávací instituce (které si jej přizpůsobí). Závěrem také konstatuji, že nabídka informačních systémů v oblasti open-source softwaru je nízká (nulová?). Je na čase to změnit?

Reference

[1] Neumajer, O.: *Školní informační systémy*, 2010, [cit. 2014-06-02], `http://clanky.rvp.cz/clanek/c/Z/8019/skolni-informacni-systemy.html/`.

[2] *Bakaláři: O programech*, 2014, [cit. 2014-06-02], `http://www.bakalari.cz/programy.aspx`.

[3] MP-Soft, a. s.: *Systém agend pro školy SAS*, 2014, [cit. 2014-06-02], `http://www.mp-soft.cz/index.php?id=sas%2Fcast0&mf=5002048`.

[4] ŠKOLA ONLINE, a. s.: *Funkce systému pro učitele*, 2014, [cit. 2014-06-02], `http://www.skolaonline.cz/Ucitel/Funkce.aspx`.

[5] Computer Media, s. r. o.: *iŠkola.cz – online školní informační systém*, 2014, [cit. 2014-06-02], `https://www.iskola.cz/`.

[6] Edookit: *Funkce informačního systému pro základní školy*, 2014, [cit. 2014-06-02], `http://www.edookit.cz/zakladni-skoly-funkce`.

[7] Edookit: *Funkce informačního systému pro mateřské školy*, 2014, [cit. 2014-06-02], `http://www.edookit.cz/materske-skoly-funkce`.

[8] just4web.cz, s. r. o.: *E-třídnice : Informační systém pro školy*, 2014, [cit. 2014-06-02], `http://www.etridnice.cz/`.

[9] Asseco Solutions, a. s.: *HELIOS Fenix*, 2014, [cit. 2014-06-03], `http://www.helios.eu/cz/produkty/helios-fenix.html`.

[10] Beran, J.: *Ekonomické softwary*, 2014, [cit. 2014-06-02], `http://www.ekonomicke-softwary.cz/cz/prehled`.

[11] Liberix, o. p. s.: *Zdarma: Software pro Linux i Windows*, 2014, [cit. 2014-06-03], `http://liberix.cz/svobodny-software/doporuceny-software/`.

[12] Open Solutions for Education, Inc.: *OpenSIS: Core Features*, 2013, [cit. 2014-06-03], `http://www.opensis.com/features.php`.

[13] The Shuttleworth Foundation: *SchoolTool: the Global Student Information System*, 2014, [cit. 2014-06-03], `http://www.schooltool.org/`.

[14] Hrnčíř, M.: *Informační systém školy*, bakalářská práce, Masarykova univerzita, Pedagogická fakulta, Brno, 2013, `http://is.muni.cz/th/255550/pedf_b/hrncir_bc_final.pdf`.

[15] Bříza, T.: *Informační systém pro mateřskou školu*, bakalářská práce, Fakulta elektrotechniky a informatiky Univerzity Pardubice, Pardubice, 2010, `http://hdl.handle.net/10195/36428`.

[16] Koudelka, J.: *Publikační systém mateřské školy*, bakalářská práce, České vysoké učení technické v Praze, Fakulta elektrotechnická, Praha, 2009, https://dip.felk.cvut.cz/browse/pdfcache/koudej1_2009bach.pdf.

Kontaktní adresa

Ing. Petra Čačková, Ph.D., Provozně ekonomická fakulta, Mendelova univerzita v Brně, Zemědělská 1, 613 00 Brno, Česká republika,
E-mailová adresa: petra.cackova@mendelu.cz

6. ročník
Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf

OSSConf 2014: 79–86

ZROVNOMERŇOVANIE MATICOVÝCH ROZVRHOV V PYTHONE

ROMAN HAJTMANEK (SK)

Abstrakt. Problematika zrovnomerňovania maticových rozvrhov sa v praxi objavuje všade tam, kde potrebujeme rozdeliť výkon v čase čo najrovnomernejšie medzi výkonovými prvkami. Príspevok sa zaoberá implementáciou algoritmu zrovnomerňovania maticových rozvrhov pomocou algoritmu zrovnomerňovania dvojíc riadkov s najväčším rozdielom výkonov. Tento algoritmus vychádza z idey riadkovej dekompozície maticového rozvrhu, pričom využíva Karmarkar & Karp diferenčný algoritmus riešiaci problém rozdelenia multimnožiny čísiel na dve multimnožiny čísiel s rovnakým súčtom (Number Partition Problem). Na implementáciu a otestovanie algoritmu bol vybraný programovací jazyk Python.

Kľúčové slová. Rovnomerné rozvrhy, permutačná matica, diferenčný algoritmus, Python.

REGULARIZING OF MATRIX SCHEDULES IN PYTHON

Abstract. A regularizing of matrix schedules problem is appearing in practice everywhere, where we need to divide performance in time between performance elements most equally. This article deals with implementation of regularizing matrix schedules algorithm by algorithm of regularizing row pairs with biggest difference of performance. This algorithm is derived from idea of row decomposition of matrix schedule, while using Karmarkar & Karp differencing algorithm, which solves problem of dividing a multiset in two multisets with equal row sums (Number Partition Problem). To implement and to test the algorithm, the Python language was chosen.

Keywords. Regular schedules, Permutation matrix, Differencing algorithm, Python.

Úvod

Problém rovnomerných rozvrhov bol motivovaný tvorbou rozvrhu dopredu určených rôznych jázd, ktoré musí realizovať m rovnakých vozidiel. Ak by sa majiteľ podniku nezaoberal otázkou celkového výkonu každého vozidla a priraďoval by im jazdy povedzme náhodne, tak by každé vozidlo malo na konci účtovného obdobia iný úhrnný výkon. Keďže od tohto výkonu sa odvíja plat vodičov jednotlivých vozidiel, došlo by pri výplatách k nespokojnosti vodičov s menšími úhrnnými výkonmi. Preto sa zamestnávateľ snaží vytvoriť taký rozvrh jázd, aby výsledné výkony vozidiel a tým aj platy vodičov vykazovali čo možno najmenšie vzájomné rozdiely.

1. Modelovanie rovnomerných rozvrhov

Ako bolo ukázané v [1], dá sa problém rovnomerných rozvrhov formulovať ako problém permutácie matíc. Rovnomerný rozvrh predstavuje matica A typu $m \times n$, pričom riadky tejto matice prislúchajú jednotlivým vozidlám. Stĺpce potom predstavujú časové úseky v rámci účtovnej periódy, napríklad pracovné dni v mesiaci. Vektor S riadkových súčtov matice A potom predstavuje úhrnné výkony jednotlivých vozidiel. Nerovnomernosť vektora S sa posudzuje pomocou nejakých funkcií $f(S)$,

Pri tvorbe rovnomerného rozvrhu hľadáme takú permutačnú maticu P, ktorá permutuje stĺpce matice tak, aby hodnota vybranej miery nerovnomernosti $f(S)$ bola minimálna. Táto úloha je vo všeobecnosti NP-ťažká s časovou náročnosťou $O((m!)^{(n-1)})$. Peško v [2] ukázal, že triviálnou je táto úloha len pre dvojstpcové matice A, kedy stačí obidva stĺpce matice usporiadať v navzájom opačnom poradí (jeden vzostupne a druhý zostupne), čím dosahujeme optimum, t. j. minimálnu hodnotu miery nerovnomernosti vektora S.

Na tomto základe zostavili Peško a Černý [3] pravdepodobnostný algoritmus zrovnomerňovania rozvrhov. Tento algoritmus najprv stĺpce matice A náhodne rozdelí do dvoch podmnožín, pre každú podmnožinu vytvorí súčtový vektor a obe podmnožiny zoradí podľa ich súčtových vektorov v navzájom opačnom poradí. Tento postup existujúcu nerovnomernosť môže zlepšiť, ale nemôže zhoršiť. Opakovaním tejto heuristiky aproximujeme optimálne riešenie. Z pravdepodobnostného charakteru tohoto algoritmu však vyplýva, že je ťažké určiť, ako blízko pri optimálnom riešení sa nachádzame. V súčasnosti ešte nie sme schopní určiť aproximačný koeficient heuristiky.

Pri riešení rovnomerných rozvrhov matice A sme sa preto pokúsili zrovnomerňovanie rozvrhu riešiť nie rozdeľovaním stĺpcov, ale približovaním riadkových súčtov. Vychádza sa zo skutočnosti, že keď usporiadame všetky stĺpce matice A v zhodnom poradí, dostaneme z hľadiska rovnomernosti najhorší prípad. Keď však budeme riešiť výmenu prvkov len medzi dvomi riadkami takto usporiadaného rozvrhu, najväčšie zmeny nerovnomernosti, získame pri výmene prvkov tých dvoch riadkov, ktoré majú najväčší rozdiel ich súčtov. Keď teda zoberieme dva najviac vzdialené riadky a tieto vzájomnou výmenou prvkov zrovnomerníme (priblížime), tak môžeme zobrať ďalšie dva riadky, ktoré majú najväčí rozdiel súčtov a postupovať dovtedy, kým je možné takýmto spôsobom dosahovať zlepšenie.

Zrovnomernenie dvoch riadkov je tiež úloha NP-ťažká. Odčítaním jedného riadku od oboch riadkov sa však prevedie na jednu z pseudopolynomiálne riešiteľných NP-ťažkých úloh, na rozklad multimnožiny (Parttition problem). Existuje niekoľko algoritmov, ktoré viacmenej úspešne riešia tento problém. Najlepším je diferenčný algoritmus, ktorého autormi sú Narendra Karmarkar a Richard Karp. Tento algoritmus je založený na pažravej stratégii a nie vždy nájde optimálne riešenie. Bol však preň určený aproximačný koeficient, čo by sme radi využili

pri určení aproximačného koeficientu tvorby rovnomerných rozvrhov. Tento algoritmus má grafovú interpretáciu, ktorá je veľmi názorná. Na jej základe sme implementovali algoritmus v jazyku Python.

2. Diferenčný KK algoritmus

Multimnožinou ďalej budeme rozumieť taký súbor prvkov, v ktorom sa niektoré prvky môžu opakovať viac krát. Diferenčný algoritmus Karmatkara a Karpa [4] (dalej len KK algoritmus) hľadá taký rozklad multimnožiny M na dve podmultimnožiny, že súčty prvkov podmultimnožín majú čo najmenší rozdiel. Hlavná idea algoritmu je založená na skutočnosti, že ak vyberieme 2 prvky z pôvodnej multimnožiny a pridáme ich do cieľových multimnožín, rozdiel súčtov cieľových multimnožín sa zmení tak, ako keby sme do jednej z nich pridali prvok rovný rozdielu oboch prvkov.

Algoritmus preto najprv nahrádza dvojice prvkov pôvodnej multimnožiny ich rozdielmi, postupne od najväčších, až napokon zostane v multimnožine jediný prvok, ktorý predstavuje konečný rozdiel, ktorý týmto postupom dosiahneme. V ďalšom kroku algoritmus rekonštruuje pôvodné dvojice prvky multimnožiny M a rozďeľuje ich do cieľových multimnožín tak, aby rozdiel prislušnej dvojice prvkov zmenšil celkový rozdiel medzi súčtami cieľových multimnožín.

Pri grafovej interpretácii algoritmu budú jednotlivé prvky pôvodnej multimnožiny predstavovať ohodnotené vrcholy grafu. V tomto grafe potom vytvárame hrany medzi dvojicami vrcholov postupne od vrcholov s najväčším ohodnotením.

Práve spojeným vrcholom zmeníme ohodnotenie nasledovne: Vrcholu s väčším ohodnotením priradíme rozdiel ich ohodnotení a vrcholu s menším ohodnotením priradíme hodnotu $-\infty$.

Takto postupujeme, pokiaľ máme aspoň jednu nezáporne ohodnotenú dvojicu vrcholov. Vznikne strom, v ktorom ofarbíme vrcholy farbami $\{0, 1\}$ tak, aby žiadna hrana nemala obidva incidentné vrcholy ofarbené rovnakou farbou. Keďže jednotlivé vrcholy predstavujú pôvodnú multimnožinu M, farbenie vrcholov ju rozdelilo na dve výsledné podmultimnožiny.

Pre väčšiu zrozumiteľnosť uvádzame algoritmus v pseudojazyku:

```
VSTUP:  Multimnozina M = {m_1,m_2,...,m_n}, m_i > 0

VYSTUP: Rozklad M do dvoch multimnozin A a B
        s minimalnym rozdielom suctov

KROK1:  Tvorba vrcholovo ohodnoteneho stromu G=(V,H),
        kde V~= {1,2,...,n} a m_i je dane ohodnotenie vrcholu i.

        Mnozinu hran vytvorime takto: Polozime H = {}, L(i) = m_i
        while (existuje aspon jedna dvojica vrcholov
```

```
                s nezapornymi ohodnoteniami vrcholov L()) {
            Ak u,v je dvojica s najvacsimi ohodnoteniami
               L(u) >= L(v) potom pridame {u,v} do H a
               polozime L(u) = L(u) - L(v), L(v) = -oo
        }

KROK2:  Ofarbime vrcholy stromu G farbami 0 a 1 t.j. F: V -> {0,1}
        F(i) != F(j) pre {i,j} in H.

KROK3:  A = {m_i: i in V for F(i)=0}
        B = {m_i: i in V for F(i)=1}
```

3. Ilustračný príklad

Uvažujme pre jednoduchosť výkladu dvojriadkovú maticu

$$A = \begin{pmatrix} 11 & 10 & 8 & 6 & 4 & 5 \\ 2 & 3 & 3 & 1 & 1 & 4 \end{pmatrix},$$

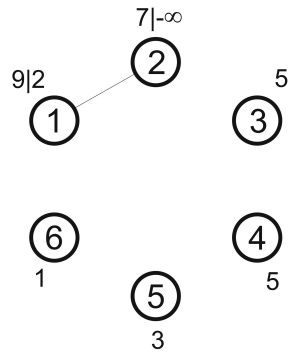
ktorej zodpovedá multimnožina $M = \{9, 7, 5, 5, 3, 1\}$.

Najskôr vytvoríme graf, v ktorom sú vrcholy ohodnotené prvkami multimnožiny M pomocou vrcholovo ohodnoteného stromu takto:

Nájdeme dva vrcholy s najväčším ohodnotením a spojíme ich novou hranou:

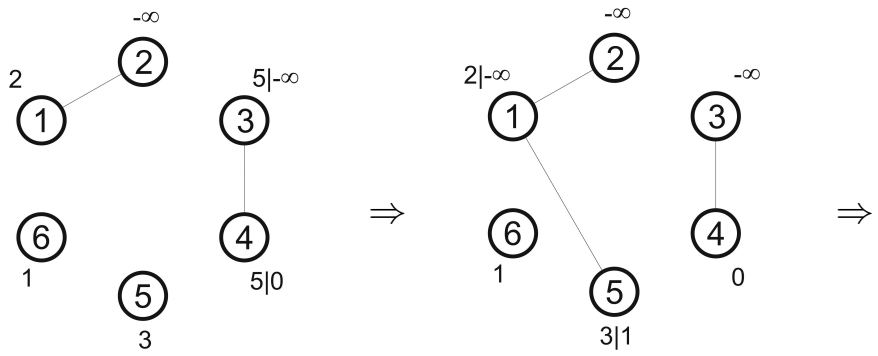
$$V_1 = 9, \quad V_2 = 7, \quad H = \{(V_1, V_2)\}.$$

Ohodnotenie týchto dvoch vrcholov zmeníme tak, že vrcholu s väčším ohodnotením priradíme rozdiel ohodnotení obidvoch vrcholov a vrcholu s menším ohodnotením priradíme $-\infty$:

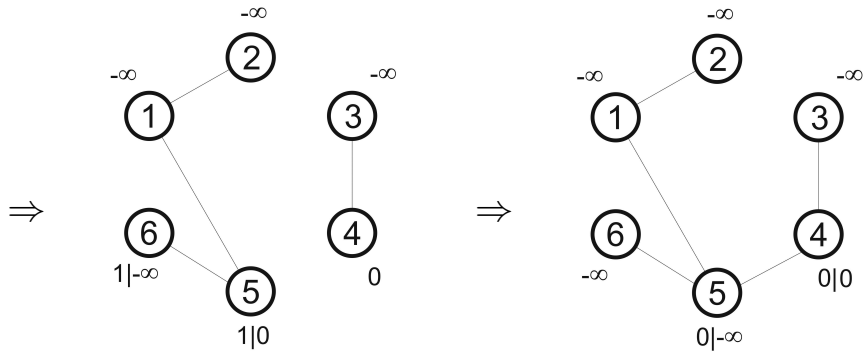


$$V_1 = 2, \quad V_2 = -\infty$$

Tento postup opakujeme dovtedy, pokiaľ máme aspoň dva nezáporne ohodnotené vrcholy:

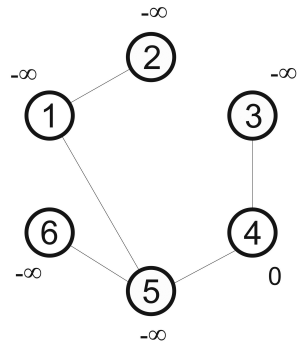


$V_4 = 0, \quad V_3 = -\infty$ $\qquad$ $V_5 = 1, \quad V_1 = -\infty$

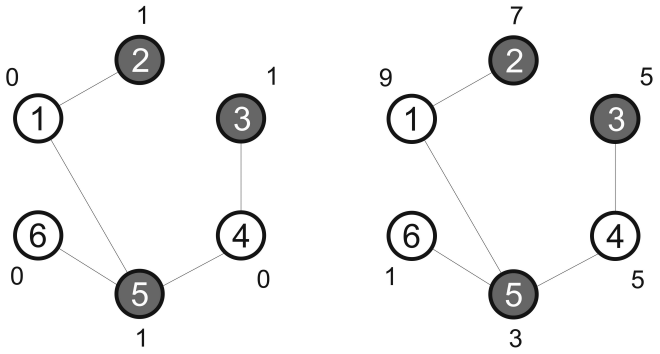


$V_1 = 0, \quad V_6 = -\infty$ $\qquad$ $V_4 = 0, \quad V_5 = -\infty$

Nakoniec sme dostali takýto strom:



Ofarbíme vrcholy farbami $\{0, 1\}$ a získame výsledné rozdelenie do dvoch multimnožín $M_1 = \{3, 5, 7\}$ a $M_2 = \{1, 4, 6\}$:



Tomuto rozdeleniu zodpovedá optimálne permutovaná matica:

$$A = \begin{pmatrix} 11 & 3 & 8 & 1 & 1 & 5 \\ 2 & 10 & 3 & 6 & 4 & 4 \end{pmatrix},$$

s mierou nerovnomernosti $f((25, 25)) = 0$.

4. Implementácia KK algoritmu v Pythone

Pri implementácii KK algoritmu v Pythone sme s výhodou využili knižnicu NetvorX [6] na pohodlnú manipuláciu s vrcholmi a hranami vytváraného stromu. Výsledná implementácia je uvedená v nasledovnom výpise zdrojového kódu:

```
    #from numpy.random import random_integers
from copy import copy
import numpy as np
import networkx as nx

def KK(M):
  N = copy(M)
  G = nx.Graph()
  G.add_nodes_from(range(len(N)))
  v~= 0
  for k~in N:
    G.node[v]['w'] = k; v~= v~+ 1
### Create edges of tree G with two largest labels of nodes
  while len(N) > 1:
    a = max(N); N.remove(a); b = max(N)
    N.remove(b); c = a - b; N.append(c)
    ua = [u~for u~in G.nodes() if G.node[u]['w']==a][0]
    ub = [u~for u~in G.nodes() if G.node[u]['w']==b and u!=ua][0]
```

```
    G.add_edge(ua,ub); G.node[ua]['w'] = c; G.node[ub]['w'] = -1e7
### Coloring of tree, color = 0 for patition A, = 1 for patition B
  AB = [[],[]]; V~= [ua]; G.node[ua]['w'] = 0; flag = 0
  while V~!= []:
    AB[flag] = AB[flag] + V; W = []
    for v~in V:
      W = W + G.neighbors(v)
        for w in G.neighbors(v):
          G.node[w]['w'] = 1-flag
          G.remove_edge(v,w)
    flag = 1-flag; V~= copy(W)
  return c, [M[k] for k~in AB[0]], [M[k] for k~in AB[1]]
```

Záver

Algoritmus v jeho grafovej forme sme implementovali v jazyku Python, ktorého interaktívny charakter a flexibilná syntax ho predurčuje používať ako prostriedok na komfortné prototypovanie problémov. K tomu nezmernou mierou prispieva aj bohatosť jeho knižníc (napr. NetworkX), ktoré obsahujú funkcie riešiace široké spektrum problémov z matematiky a technických vied. Vďaka tomu je možné zamerať sa len na riešenie vlastného problému a nerozptyľovať sa programovaním pomocných všeobecných algoritmov, ktoré už boli raz dobre naprogramované. Jeho rozšíreniu prispieva samozrejme aj to, že je šírený pod licenciou Open Source.

Poďakovanie. Tento príspevok bol podporený grantom slovenskej kultúrno-edukačnej agentúry KEGA č. 011ŽU-4/2014 „Experimentálna matematika – zviditeľnenie neviditeľného“.

Literatúra

[1] HAJTMANEK, R.: *Heuristické algoritmy pre maticové rovnomerné rozvrhy*, Otvorený softvér vo vzdelávaní, výskume a IT riešeniach, zborník medzinárodnej konferencie OSSConf 2013, Žilina, 2.–4. júla 2013, str. 33–38, ISBN 978-80-970457-3-9.

[2] PEŠKO, Š.: *Optimalizácia NP-ťažkých dopravných rozvrhov*, Habilitačná práca, ŽU Žilina, str. 120, (2002), `http://frcatel.fri.uniza.sk/users/pesko/publ/hab.pdf`.

[3] PEŠKO, Š., Černý, J.: *Uniform Splitting in Managerial Decision Making*, E+M, Economics and Management IX, vol. 4, (2006), Technical university of Liberec, pp. 67–71.

[4] MICHIELS, W., KORST J., AARTS E., VAN LEEUWEN J.: *Performance Ratios for the Karmarkar-Karp Differencing Method*, Proceedings of the 20th Annual Symposium on Theoretical Aspects of Computer Science, London, UK, 2003, pp. 583–595, ISBN 3-540-00623-0, `http://citeseer.uark.edu:8080/citeseerx/viewdoc/download;jsessionid=BAA0912834DE60577B71EC073914D2B9?doi=10.1.1.107.1332&rep=rep1&type=pdf`.

[5] KORF, R. E.: *From Approximate to Optimal Solutions: A Case Study of Number Partitioning*, Proceedings of the 14th international joint conference on Artificial intelligence, Volume 1, ISBN 1-55860-363-8.

[6] HAGBERG, A. A., SCHULT, D. A., SWART, P. J.: *Exploring network structure, dynamics, and function using NetworkX*, Proceedings of the 7th Python in Science Conference (SciPy2008), Gäel Varoquaux, Travis Vaught, and Jarrod Millman (Eds), (Pasadena, CA USA), pp. 11–15, Aug 2008.

Kontaktná adresa

Ing. Roman Hajtmanek, Katedra matematických metód, Fakulta riadenia a informatiky, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovenská republika, *Aktuálna adresa*: SOIT, 010 01 Žilina, Slovenská republika,

E-mailová adresa: `roman.hajtmanek@fri.uniza.sk`

6. ročník
Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf

OSSConf 2014: 87–92

UDALOSŤAMI RIADENÉ PROGRAMOVANIE V OS SUZUHA

MICHAL CHOVANEC (SK)

Abstrakt. Článok popisuje implementáciu podsystému správ v OS Suzuha. Technika programovania pomocou správ - udalostí umožňuje efektívne využívať procesorový čas nezávisle od použitého hardvéru. Udalosti zovšeobecňujú hardvérové prerušenia a komunikáciu pomocou správ.

Kľúčové slová. operačný systém, systém reálneho času, udalosťami riadené programovanie, arm cortex.

EVENT DRIVEN PROGRAMMING IN SUZUHA OS

Abstract. This paper describes the implementation of the message subsystem in OS Suzuha. Techniques of programming by messages - events make efficient use of processor time independently of the hardware. Events generalize hardware interrupts and communicate via messages.

Keywords. operating system, real time system, event driven programming, arm cortex.

Úvod

Riadenie procesu pomocou mikrokontroléra vyžaduje včasnú reakciu na zmenu vstupných veličín. V súčasnosti má takmer každý riadiaci systém viac vstupov i výstupov. Dáta z jednotlivých vstupov prichádzajú v rôznych časových intervaloch (periodicky i aperiodicky) a podobne aj výstupy vyžadujú rôzne časovanie.

Jedným z riešení ako spracovať takéto informácie je implementácia udalosťami riadeného programovania. Mikrokontroléry k tomu poskytujú rozsiahly systém prerušení. Táto časť je však hardvérovo závislá a pri migrácii medzi rôznymi typmi mikrokontrolérov predstavuje často rozsiahle zmeny v programe.

Problém je možné riešiť zavedením podsystému správ ako súčasť operačného systému. Toto riešenie zovšeobecní riadenie prerušeniami a medzivláknovú komunikáciu do jednotného udalostne riadeného systému.

Systém správ teda poskytuje vhodný prostriedok abstrakcie od hardvéru.

1. Popis OS Suzuha

Operačný systém Suzuha je open source systém reálneho času určený pre mikrokontroléry ARM Cortex. V súčasnosti bol systém testovaný na troch rôznych architektúrach:

- ARM Cortex M0: mikrokontrolér KL25Z128.
- ARM Cortex M3: mikrokontrolér STM32F100.
- ARM Cortex M4: mikrokontrolér TM4C123G.

Prípadná prenositeľnosť systému medzi ďalšími radami Cortex (Mx, Rx, Ax) je bezproblémová.

Systém je písaný striktne modulárne so zreteľom na jednoduchosť. Nesnaží sa zavádzať revolučné riešenia.

Systém je rozdelený do niekoľkých vrstiev

- Kernel: jadro systému.
- Systémové knižnice: ovládače hardvéru, mutexy, systém správ a štandardný výstup.
- Aplikačné knižnice: knižnice závislé od aplikácie, tvorí ich používateľ.
- Aplikácia: vlastná používateľská aplikácia.

OS Suzuha je možné stiahnúť pod GNU GPL licenciou z [1]. Pre ďalší vývoj je vhodné začať jednoduchým jadrom Cortex M0, jeho popis je možné nájsť v [2].

2. Systém správ

Medzivláknová komunikácia pomocou správ je založená na modeli klient-server.

Server sa registruje v systéme volaním:

```
msg_register(MSG_SERVER);
```

kde MSG_SERVER je jedinečné 32bitové číslo identifikujúce server. Tento identifikátor je nevyhnutný pre pripojenie sa k serveru.

Vlákno servera po registrovaní obvykle čaká na príjem správy:

```
while (1)
{
  struct  sMsg msg;
  msg_get(&msg);
  ...
}
```

Pri čakaní na správu sa vlákno prepne do stavu WAITING a jadro systému s nim pri plánovaní ďalšieho procesu nepočíta. Spiace vlákna sú jadrom ignorované, šetrí sa tak výpočtový výkon a spotreba.

Vlákno klienta posiela správy volaním

```
msg_raise(&msg);
```

Naplnenie štruktúry sMsg a odoslanie je implementované nasledovne:

```
while (1)
{
  ...

  struct sMsg msg;
```

```
  msg.destination = MSG_SERVER; /* ID servera = prijemca */
  msg.source = KLIENT_ID; /* ID odosielatela */
  msg.size = 4;     /* velkost spravy v bytoch */
  msg.data = data;  /* vlastna sprava */
  msg_raise(&msg);  /* poslanie spravy */
}
```

Naplnením štrutúry struct sMsg adresou servera je zabezpečené doručenie správy k serveru. Funkcia msg_raise(struct sMsg *msg) čaká na prijatie správy serverom, je tak zabezpečené kompletné prijatie správy. Zo strany servera môže byť zaslaná odpoveď. Vlákno klienta však môže predčasne ukončiť činnosť, alebo môže dôjsť k jej zlyhaniu. K dispozícii je preto funkcia msg_raise_async(struct sMsg *msg), ktorá nečaká na príjem - server preto nezostane čakať a môže obsluhovať ďalšie správy vo fronte.

3. Implementácia systému správ

Vlastná implementácia systému správ je písaná s ohľadom na dostupné technické prostriedky. Limitujúcim faktorom je najmä pamäť RAM. Spomenuté mikrokontroléry majú od 4KB po 32KB RAM pamäte.

V systéme je preto implementovaná len jeden front správ, spoločná pre všetky vlákna. Vzhľadom na modularitu však nie je problém nahradiť doterajšie riešenie viacfrontovým systémom, známym z operačných systémov pre PC. Štruktúra správy je nasledovná:

```
struct sMsg
{
  u32 destination, source;
  u32 data;
  u32 size;
};
```

Položky *destination* a *source* predstavujú ID cieľa a zdroja. Položka *data* predstavuje 32 bitové číslo, nesúce vlastnú správu. Premenné sú typu $u32$ – obvykle odvodený od *unsigned int*. Podľa potreby je možné pretypovať ukazovateľ na pole dát a prenášať tak ľubovoľný objem dát. Nevyhnutné je potom správne nastaviť položku *size*:

```
data = (u32)bytes_array;
size = sizeof(bytes_array)/sizeof(u32);
```

Týmto spôsobom je prenášaný len ukazovateľ do zdieľanej pamäte. Je preto nevyhnutné aby odosielateľ nemenil obsah dát po odoslaní kým nedostane potvrdenie zo strany servera.

Pre efektívne využitie systému správ viacerými servermi je implementovaný front:

```
volatile struct sMsg __msg__[MSG_FIFO_SIZE];
```

Pre odosielanie správy je najdôležitejšia funkcia

`u32 msg_raise_async(struct sMsg *msg)`

Je nevyhnutné aby táto funkcia prebehla atomicky – nielen na logickej úrovni, ale aby skutočne počas jej vykonávania nedošlo k prepnutiu kontextu. V opačnom prípade môže dôjsť k predčasnému ukončeniu niektorého vlákna alebo zaplneniu frontu správ.

Najprv je nevyhnutné overiť, či existuje príjemca správy. Ten je overený prehľadaním dostupných identifikátorov v poli *__msg_names__:

```
i = 0;
while ((i < THREADS_MAX_COUNT) && (msg->destination != __msg_names__[i]))
    i++;
```

Po nájdení a overení príjemcu dôjde k samotnému zaradeniu správy do frontu:

```
/*najdenie volnej polozky vo fronte*/
for (i = 0; i < MSG_FIFO_SIZE; i++)
  if (__msg__[i].source == MSG_NULL)
  {
    /*vlastny prenos spravy*/
    __msg__[i].source = msg->source;
    __msg__[i].destination = msg->destination;
    __msg__[i].data = msg->data;
    __msg__[i].size = msg->size;

    /*navrat z funkcie*/
    sched_on();
    wake_up_threads();
    return MSG_SUCESS;
  }
```

Veľkosť frontu správ __msg__ je volená pri kompilácii systému. Po nájdení prvej voľnej položky je vykonaný proces odoslania správy. Samotný návrat z funkcie vyžaduje ešte dva kroky:

sched_on(): odomknutie plánovača - povolenie prerušení,

wake_up_threads(): zobudenie spiacich vlákien.

Funkcia však môže byť predčasné ukončená a k doručeniu správy nedôjde. Najčastejšie z dôvodu neexistujúceho príjemcu alebo preplnenia frontu správ. Z toho dôvodu je implementovaná funkcia, ktorá garantuje zaradenie správy do frontu:

`u32 msg_raise(struct sMsg *msg).`

Funkcia čaká na uvoľnenie miesta vo fronte. K predčasnému ukončeniu dôjde len ak príjemca správy neexistuje. Samotný proces čakania je riešený vynútením prepnutia kontextu na ďalšie vlákno - nie je opodstatnené cyklicky testovať hodnotu ktorá sa môže zmeniť len v ďalšom vlákne:

```
/*skusi poslat spravu*/
msg_res = msg_raise_async(msg);
```

```
  /*prijemca neexistuje*/
  if (msg_res == MSG_NO_REGISTED)
    return msg_res;

  /*front plny, vynutenie prepnutia kontextu*/
  if (msg_res != MSG_SUCESS)
    sched_next();
```

Funkcia pre príjem správy:

```
void msg_get(struct sMsg *msg)
```

Čakanie na príjem správy prebieha v cyklickom prehľadávaní frontu správ. Po nájdení položky patriacej aktuálnemu vláknu sa vykoná samotné prebratie správy:

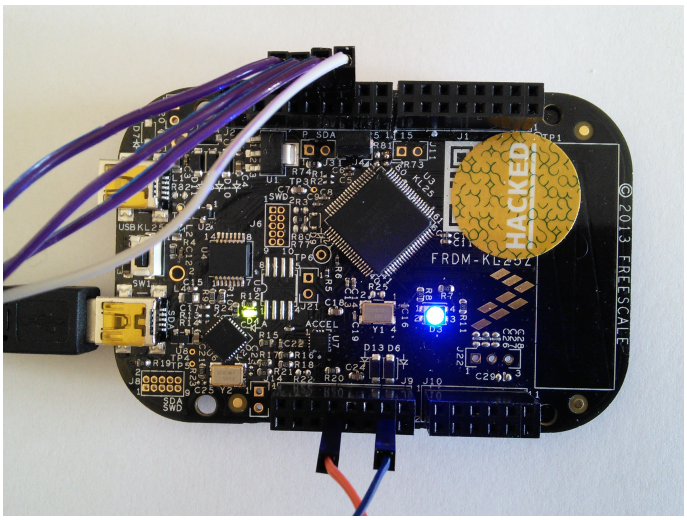
```
if ( __msg__[i].destination == __msg_names__[tid] )
{
  /*prenos spravy z frontu na vystup*/
  msg->source = __msg__[i].source;
  msg->destination = __msg__[i].destination;
  msg->data   = __msg__[i].data;
  msg->size   = __msg__[i].size;

  /*vycistenie polozky vo fronte*/
  __msg__[i].source      = MSG_NULL;
  __msg__[i].destination = MSG_NULL;
  __msg__[i].data        = MSG_NULL;
  __msg__[i].size        = MSG_NULL;

  sched_on();
  return;
}
```

Funkcia preberie správu z frontu a uvoľní miesto pre ďalšiu správu. Funkcia zistí meno aktuálneho vlákna volaniami:

```
u32 tid = get_thread_id();
destination_name = __msg_names__[tid];
```

kde get_thread_id() je volanie jadra systému na získanie ID vlákna. Identifikátor *tid* sa preloží volaním destination_name = __msg_names__[tid]; na meno príjemcu správy. Tento zdvojený prístup je zvolený z toho dôvodu, že užívateľ vo všeobecnosti nemá kontrolu nad tým, aké *tid* dostane vlákno.

Čakanie na príjem správy je zefektívnené podobne ako jej odoslanie: po jednom cykle prehľadávania frontu správ je vynútené prepnutie kontextu.



Obr. 1. Testovacia doska s mikrokontrolérom Kinetis Cortex M0+ [3]

Záver

Model systému na architektúre klient-server predstavuje robustné a modulárne riešenie pre spracovanie dát v procese riadenia. Systém tak dokáže pracovať aj s výpadkom niekoľkých zdrojov dát.

Kritickou časťou systému správ je požiadavka na stabilitu serverovej časti. Ďalšie smerovanie vývoja preto môže byť návrh decentralizovaného prístupu a možnosť posielať správy nielen v rámci jedného mikrokontroléra, ale pomocou komunikačného rozhrania (RF, IRDA, SPI...) posielať správy vzdialenému mikrokontroléru.

Literatúra

[1] Zdrojové kódy OS suzuha: `http://sourceforge.net/projects/suzuhaos/files`
[2] Popis jadra Cortex M0: `http://infocenter.arm.com/help/topic/com.arm.doc.dui0497a/DUI0497A_cortex_m0_r0p0_generic_ug.pdf`
[3] Testovacia doska: `http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=FRDM-KL25Z`

Kontaktná adresa

Ing. Michal Chovanec, Katedra technickej kybernetiky, Fakulta riadenia a informatiky, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovenská republika,
E-mailová adresa: `michal.nand@gmail.com`

6. ročník Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf 2014: 93–100

USING QUINE-MCCLUSKEY ALGORITHM IN RELIABILITY ANALYSIS

PETER KAPUSTA (SK) AND MIROSLAV KVAŠŠAY (SK)

Abstract. There exist two different types of systems in reliability analysis – coherent and noncoherent. In a coherent system, there exists no situation in which failure of any system component results system repair. A noncoherent system does not meet this assumption and, therefore, its analysis is more complicated than in the case of a coherent system. One of the principal problems in reliability analysis is identification of minimal scenarios that are needed to ensure system work or minimal scenarios that results system failure. There exist some algorithms that can be used for this purpose in the case of coherent systems. However, those algorithms cannot be used for noncoherent systems. One of the algorithms that can be used in the analysis of noncoherent systems is Quine-McCluskey algorithm that was originally developed for minimization of Boolean functions.

Key words and phrases. Structure function, Quine-McCluskey algorithm, OpenMP.

POUŽITIE QUINE-MCCLUSKEYHO ALGORITMU V TEÓRII SPOĽAHLIVOSTI

Abstrakt. V teórii spoľahlivosti rozlišujeme dva typy systémov – koherentné a nekoherentné. O koherentnom systéme hovoríme v prípade, že zlyhanie ľubovoľného komponentu systému nikdy nemôže viesť k oprave, resp. zlepšeniu činnosti systému. Nekoherentný systém je charakteristický tým, že v ňom môžu existovať situácie, kedy porucha komponentu zapríčiní zlepšenie činnosti systému. Jednou zo základných úloh teórie spoľahlivosti je identifikácia minimálnych scenárov, ktoré za každých okolností zaistia buď správnu činnosť systému, alebo, naopak, zapríčinia jeho zlyhanie. Pre ich identifikáciu bolo navrhnutých niekoľko postupov. Väčšina z nich však predpokladá, že systém je koherentný. Jedným z algoritmov, ktorý môže byť aplikovaný aj na nekoherentné systémy, je Quine-McCluskeyho algoritmus, ktorý bol pôvodne navrhnutý pre minimalizáciu booleovských funkcií.

Kľúčové slová. Štrukturálna funkcia, Quine-McCluskeyho algoritmus, OpenMP.

Introduction

In reliability analysis, we assume that a system and all its components can be only in one of two possible states - functional (represented by 1) and failed (presented as 0). The correlation between system state and states of individual system components is defined by the structure function [1], [2]:

$$\phi(\boldsymbol{x}) = \phi(x_1, x_2, \ldots, x_n) : \{0,1\}^n \rightarrow \{0,1\}, \tag{1}$$

where

- n is a number of system components,
- x_i is a variable that represent the state of the i-th component, for $i = 1, 2, \ldots, n$ (0 – the component is failed; 1 – the component is functional),
- $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ is a vector of system states, i. e. a state vector.

There exist two different types of systems according to properties of the structure function – coherent and noncoherent. A coherent system meets the assumption that its structure function is non-decreasing. This implies that the failure of any system component can causes only system failure, i.e. in a coherent system, there is no situation in which the failure of some system component results system repair. If there is such situation, then the system is identified as noncoherent.

The structure function describes the topological properties of a system. In reliability analysis, it can be used in several ways. If we know availabilities of individual system components, then, using the structure function, we can compute system reliability characteristics, such as system availability and unavailability, mean time to failure or mean time to repair, or identify the influence of individual components on the system work. For this task, some measures have been proposed. These measures are known as importance measures [3]. The problem of these measures is that they were originally developed for coherent systems and, therefore, they do not take into account cases when system repair coincides with the failure of a system component or when the system component repair results the failure of the system. Some generalizations of them for noncoherent systems were proposed in papers [4] and [5]. Those generalizations are based on the assumption that all minimal scenarios that are needed to ensure system work or minimal scenarios that results system failure are known. These scenarios are also known as prime implicants or prime implicates. In the next parts, we focused only on the prime implicants.

1. Prime Implicants

The structure function 1 can be interpreted as a Boolean function. This fact allows us to use some approaches of Boolean algebra in reliability analysis. One of those approaches is known as minimization of Boolean functions that is focused on finding the minimal disjunctive normal form of a Boolean function.

A disjunctive normal form (also known as a sum of products) is a disjunction of conjunctive clauses. Every clause covers one or more cases in which a Boolean function has value 1 (or true in terms of Boolean logic). A disjunctive normal form covers all situations in which the considered Boolean function has value 1. There exist several types of disjunctive normal forms. For our purpose, the most important are the full disjunctive normal form and the Blake canonical form [6].

The full disjunctive normal form (also known as the minterm canonical form) is the disjunction of all 1-minterms of a Boolean function. A minterm of a Boolean function of n Boolean variables is a conjunction of all variables that can be either

in the direct or inverse form. A 1-minterm is a minterm for which the considered Boolean function has value 1. Every minterm can also be expressed in the vector form that contains value 1 at positions of the direct variables and value 0 at positions of the inverse variables.

For example, consider a simple Boolean function of 3 variables in Table 1. The 1-minterms of the function are $\overline{x_1}x_2x_3$, $x_1\overline{x_2}\overline{x_3}$, $x_1x_2\overline{x_3}$ and $x_1x_2x_3$. The vector forms of the 1-minterms are $(0,1,1)$, $(1,0,0)$, $(1,1,0)$ and $(1,1,1)$. From the 1-minterms, the full disjunctive normal form of the considered function can be derived as follows: $\overline{x_1}x_2x_3 \vee x_1\overline{x_2}\overline{x_3} \vee x_1x_2\overline{x_3} \vee x_1x_2x_3$.

Table 1. A simple Boolean function

x_1	x_2	x_3	$\phi(\boldsymbol{x})$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

The 1-minterms of a Boolean function imply that the function has value 1 in situations that are characterized by them. Therefore, the 1-minterms are often named as the implicants of the Boolean function. However, in many cases, two or more 1-minterms can be combined into one conjunction that has less than n variables and that cover two or more situations for which the Boolean function has value 1. Therefore this conjunction can also be named as the implicant of the function. Generally, an implicant of a Boolean function of n variables is a conjunction of n or less Boolean variables that can be either in direct or inverse form. A prime implicant is an implicant from which no variable can be removed without losing its status as an implicant. A disjunctive normal form that contains all prime implicants of a Boolean function is known as the Blake canonical form.

Every implicant can be expressed in the vector form. The vector form contains value 1 at positions of the direct variables, value 0 at positions of the inverse ones and symbol '$-$' at others.

For example, consider the Boolean function defined in Table 1 again. Its implicants are $\overline{x_1}x_2x_3$, $x_1\overline{x_2}\overline{x_3}$, $x_1x_2\overline{x_3}$, $x_1x_2x_3$, x_2x_3, x_1x_2 and $x_1\overline{x_3}$. The prime implicants are only the last three ones, i.e. x_2x_3, x_1x_2 and $x_1\overline{x_3}$. The prime implicants can also be expressed in the form of vectors as follows: $(-,1,1)$, $(1,1,-)$, $(1,-,0)$. The Blake canonical form of this function has the following form: $x_2x_3 \vee x_1x_2 \vee x_1\overline{x_3}$.

In terms of reliability analysis, a prime implicant corresponds to occurrence of minimal set of events that are needed for proper work of the system. For example, the Boolean function in Table 1 was used as the structure function of the liquid-level control-system considered in paper [4]. The prime implicants of this function implies that the system is working if at least the second and the third components of the system are functional or at least the first and the second components are working or at least the first component is working and the third one is failed. This system is noncoherent, because when the first component is functional and the second one is failed then the failure of the third one causes system repair. Therefore, if we want to use methods defined in papers [4] and [5] to analyze its reliability, all prime implicants should be known. For this task, the Quine-McCluskey algorithm can be used.

2. Quine-McCluskey Algorithm

The Quine-McCluskey (QM) algorithm [7], [8], is an exact method for minimization of Boolean functions. It consists of two separate steps. In the first step, all prime implicants of a Boolean function are identified. In the second step, only prime implicants that are necessary to cover the function are selected. In reliability analysis, we need to know all prime implicants, therefore, only the first phase of the QM algorithm is important. If we assume that the 1-minterms and implicants of a Boolean function are expressed in the vector form, then the first phase can be defined in the next steps:

Step 0: Take all 1−minterms of the function and identify them as unprocessed implicants.

Step 1: All unprocessed implicants contains the same count of symbol '−', i.e. k. According to number of zeroes in unprocessed implicants, split them into $n-k+1$ groups. Every implicant in the i–th group contains $n-k+i$ zeroes. The groups are indexed from 0 to $n-k$.

Step 2: For $i = 0, 1, \ldots, n-k-1$ compare every implicant from group i with every implicant from group $i+1$.

a) If two compared implicants have symbol '−' at the same positions and they differ only at one position, e.g. l, then combine them into a new implicant that has the same values as the compared ones at positions different from l and symbol '−' at l.

b) If an implicant from group i cannot be combined with anyone from group $i+1$, then it is a prime implicant.

Step 3: Implicants from group $n-k$ that were not combined with implicants from group $n-k-1$ are prime implicants.

Step 4: If new implicants were identified in step 2, then mark them as unprocessed implicants and go to step 1.

For example, consider the Boolean function defined in Table 1. Firstly, we split 1-minterms into groups according to number of zeroes in their vector form (Table 2). From Table 2, we can see that the 1-minterm from group 1 can be combined with the second 1-minterm from the group 2 in implicant $(1, -, 0)$ (the ticks in the first column). Booth 1-minterms from group 2 can be combined with the 1-minterm from group 3 into implicants $(-, 1, 1)$ and $(1, 1, -)$ (the ticks in the second and third column). So, every 1-minterm could be combined with another one from the neighboring groups, therefore no 1-minterm is a prime implicant.

Table 2. Splitting 1-minterms from Table 1 into groups

Groups	**1-minterms**	**Can be combined?**		
0	–	–		
1	$(1, 0, 0)$	✓		
2	$(0, 1, 1)$		✓	
	$(1, 1, 0)$	✓		✓
3	$(1, 1, 1)$		✓	✓

In the next phase, we split implicants obtained in the previous step into new groups according to number of zeroes (Table 3). Now, we compare the implicants from the neighboring groups. However, no implicants can be combined into new ones, therefore implicants $(1, -, 0)$, $(-, 1, 1)$ and $(1, 1, -)$ are all prime implicants of the Boolean function from Table 1.

Table 3. Splitting implicants from Table 2 into groups

Groups	**Implicant**	**Can be combined?**
0	–	–
1	$(1, -, 0)$	**x**
2	$(-, 1, 1)$	**x**
	$(1, 1, -)$	**x**

As we can see, the main part of the first phase of the QM algorithm is focused on the comparing and combining implicants from the neighboring groups. This task can be very easy implemented in parallel way. Therefore, we decided to investigate the speed-up of the first phase of the QM algorithm by using a parallel approach.

3. Experiments

Firstly, we implemented the serial version of the QM algorithm in C++ language. Then, we implemented its parallel version by using OpenMP library. For experiments, we used some functions from the standard set LGSynth91 (`http://www.cbl.ncsu.edu:16080/benchmarks/LGSynth91/`) and also some randomly

generated functions that had from 8 to 13 variables. The experiments were performed on a computer with CPU Intel Core i5-2400 (3.1 GHz, 4 cores) and 8 GB of RAM. The graphical comparison of serial and parallel implementation for functions of 8 and 13 variables is presented in Fig. 1 – 4. The tabular comparison is presented in Table 4.

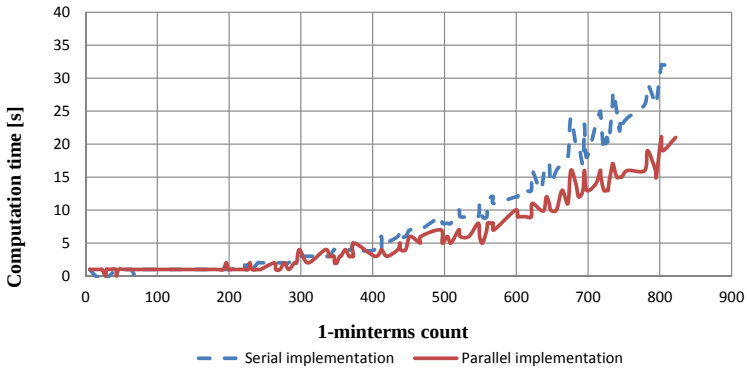


Figure 1. Computation time of the QM algorithm for Boolean functions of 10 variables depending on the number of 1-minterms

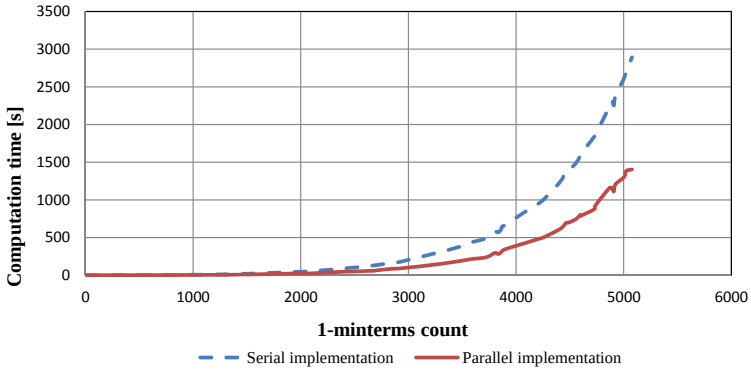


Figure 2. Computation time of the QM algorithm for Boolean functions of 13 variables depending on the number of 1-minterms

From Fig. 1 and 2, it is clear that the speed-up of the parallel version of the QM algorithm is more noticeable when the Boolean function contains more 1-minterms. Similarly, in Fig. 3 and 4, we see that the effectiveness of the parallel version go up with the increasing number of the total count of prime implicants that can be identified in a Boolean function.

The total results of the experiments are presented in Table 4. This table shows two important facts. Firstly, the impact of the parallelization is more noticeable

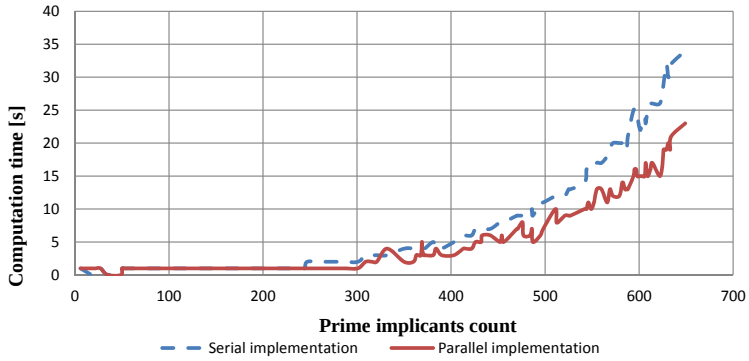


Figure 3. Computation time of the QM algorithm for Boolean functions of 10 variables depending on the number of prime implicants

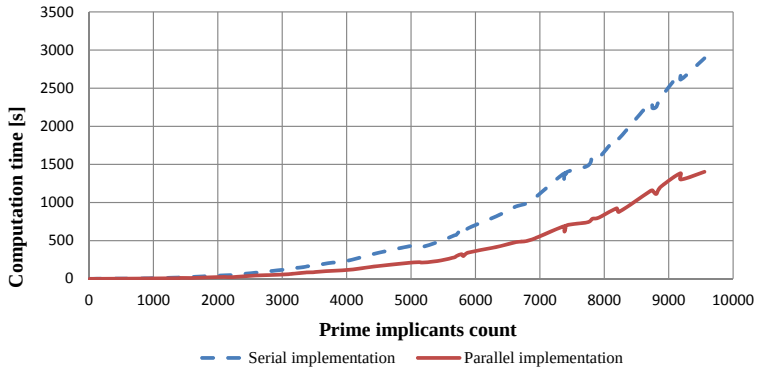


Figure 4. Computation time of the QM algorithm for Boolean functions of 13 variables depending on the number of prime implicants

for Boolean functions of more variables. Secondly, for 13 variables, the average speed-up of the QM algorithm caused by its parallelization is only about 50%, although the algorithm was running on 4 CPU cores. The reason is that we implemented concurrently only one part of the algorithm, in which implicants from the neighboring groups are compared.

4. Conclusion

One of the important tasks of reliability analysis is identification of minimal scenarios that are needed to ensure system work or minimal scenarios that results system failure. In a case of noncoherent systems, this problem can be solved using the QM algorithm. The QM algorithm is an exact method that can be used for minimization of Boolean functions. The main problem of this algorithm is its

Table 4. The average decrease of computation time depending on the number of 1-minterms and the number of prime implicants

Variables counts	Decrease of computation time depending on the number of 1-minterms [%]	prime implicants [%]
8	18.81	21.09
10	24.03	19.80
11	21.71	19.94
13	49.22	49.22

time complexity. In this paper, we have tried to increase the speed of the QM algorithm by its parallelization. We used OpenMP library for this purpose. The results of experiments showed that the parallel implementation is faster than the serial one. However, the speed-up is not as big as one could expect and, therefore, another research will be needed.

References

[1] ZAITSEVA, E. N., LEVASHENKO, V. G.: *Importance analysis by logical differential calculus*, Automation and Remote Control, February 2013, vol. 74, no. 2, pp. 171–182.

[2] ZAITSEVA, E., LEVASHENKO, V.: *Multiple-Valued Logic mathematical approaches for multi-state system reliability analysis*, Journal of Applied Logic, Special Issue 2013, vol. 11, no. 3, pp. 350–362.

[3] KUO, W., ZHU, X.: *Importance Measures in Reliability, Risk, and Optimization*, Chichester, UK, John Wiley & Sons, Ltd., 2012, ISBN 1-119-99344-5.

[4] ANDREWS, J. D., BEESON, S.: *Birnbaum's measure of component importance for non-coherent systems*, IEEE Transactions on Reliability, June 2003, vol. 52, no. 2, pp. 213–219.

[5] BEESON, S., ANDREWS, J. D.: *Importance measures for non-coherent-system analysis*, IEEE Transactions on Reliability, September 2003, vol. 52, no. 3, pp. 301–310.

[6] BROWN, F. M.: *Boolean Reasoning: The Logic of Boolean Equations*, Mineola, NY, USA, Dover Publications, Inc., 2012, ISBN 1-119-99344-5.

[7] QUINE, W. V.: *The problem of simplifying truth functions*, American Mathematical Monthly, October 1952, vol. 59, no. 8, pp. 521–531.

[8] MCCLUSKEY, E. J.: *Minimization of Boolean functions*, Bell System Technical Journal, November 1956, vol. 35, pp. 1417–1444.

Contact addresses

Peter Kapusta, Faculty of Management Science and Informatics, University of Žilina, Univerzitná 8215/1, 010 26 Žilina, Slovak Republic,
E-mail address: peterkapusta22@gmail.com

Ing. Miroslav Kvaššay, Department of Informatics, Faculty of Management Science and Informatics, University of Žilina, Univerzitná 8215/1, 010 26 Žilina, Slovak Republic,
E-mail address: Miroslav.Kvassay@fri.uniza.sk

6. ročník Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf 2014: 101–109

PRÍPRAVA ELEKTRONICKÝCH PUBLIKÁCIÍ S BALÍČKOM PDFSCREEN

ALEŠ KOZUBÍK (SK)

Abstrakt. Príspevok je venovaný príprave e-publikácií. Predstavuje sa balíček `pdfscreen`, ktorý umožňuje prípravu obrazovkovej verzie dokumentu z rovnakého zdrojového súboru. Tento balíček umožňuje prispôsobenie obsahu na ľubovoľný formát monitoru počítača alebo čítačky kníh. Taktiež sú predstavené možnosti navigačných prvkov vo výslednom `pdf` dokumente. V záverečnej časti sú uvedené niektoré dostupné nástroje pre konverziu do formátu `epub`. Ich výsledky však nie sú plne uspokojivé, ako je možné vidieť z priložených ukážok.

Kľúčové slová. LaTeX, `pdfcsreen`, elektronické publikácie.

PREPARING OF THE ELECTRONIC PUBLICATIONS USING THE PDFSCREEN PACKAGE

Abstract. This paper is devoted to the preparation of e-publications. It presents the package `pdfscreen`, which allows the preparation of the screen version of a document from the same source file.This package allows you to redesign the pdf output to any size computer monitor or e-book reader. Here is also illustrated the possibility of managing the navigation elements in the resulting `pdf` document. The final section introduces some tools available for converting into `epub` document format. But their results are not fully satisfactory, as we can see from the examples.

Keywords. LaTeX, `pdfcsreen`, e-publications.

Úvod

V každodennej realite sa stáva čoraz častejšou požiadavkou na vytvorené dokumenty a publikácie ich prevedenie do podoby, zobraziteľnej na monitore počítača, na tablete či elektronickej čítačke kníh. Na rozdiel od prednáškových slajdov, ktoré predstavujú len určité podporné podklady pre potreby prednášky a sú teda určitým stručným výťahom z textu, pri elektronickej verzii ide o zobrazenie kompletného textu. Nevystačíme teda s dokumentmi triedy `beamer`, ktoré sme si predstavili na III. ročníku konferencie v príspevku [4]. Naopak, hľadáme možnosť preformátovania, pokiaľ možno nezmeneného, zdrojového textu do rozmerov, vhodných pre zobrazenie, či už na monitore alebo v čítačke.

Pomerne jednoduchým, avšak nesmierne užitočným nástrojom, ktorý takéto preformátovanie umožňuje, je balíček `pdfscreen`. Tento balíček si môžeme predstaviť na pomerne malom priestore, nakoľko vo svojej podstate predstavuje len rozšírenie balíčka `hyperref`. Jeho autorom je Sebastian Ratz.

1. Charakteristika balíčka `pdfscreen`

Balíček `pdfscreen` je nástrojom, ktorého hlavnou úlohou je zmeniť rozmery výšky a šírky textu. Túto zmenu pri tom vykonať tak, aby bola optimálna vzhľadom na rozmery zobrazovacej jednotky. Jeho použitie deklarujeme štandardným spôsobom v preambule dokumentu. To znamená, že použijeme príkaz:

```
\usepackage[voľby]{pdfscreen}
```

Pritom je automaticky nahratý už vyššie spomenutý balíček hyperref. Ak však chceme definovať niektoré vlastné voľby balíčka `hyperref`, je potrebné tento balíček aj s požadovanými voľbami nahrať skôr, ako bude v preambule zaradené nahratie balíčka `pdfscreen`.

1.1. Voľby balíčka `pdfscreen`

Pri použití balíčka `pdfscreen` máme k dispozícii nasledujúce voliteľné parametre:

1. `screen`, výsledkom je vygenerovanie obrazovkovej verzie dokumentu,
2. `print` vygenerovanie verzie dokumentu vhodnej pre tlač,
3. `panelleft` umiestnenie navigačného panelu na ľavý okraj stránky,
4. `panelright` umiestnenie navigačného panelu na pravý okraj stránky,
5. `nopanel` potláča zobrazenie navigačného panelu,
6. `paneltoc` zobrazenie obsahu dokumentu sa stáva súčasťou navigačného panelu. V takom prípade už nepoužívame `\tableofcontents` v rámci dokumentu.
7. `sectionbreak` spôsobí zalomenie novej stránky pred každou novou sekciou,
8. `code` poskytuje príkazy, ktoré umožňujú výpis prostredia `verbatim` v podobe programového kódu,
9. je možné špecifikovať vhodný driver, ako napr. `dvips`, `dvipsone`, . . . , `vtex`. Implicitne je nastavený `pdftex`.
10. Je možné zvoliť jednu zo šiestich farebných schém – bluelace, blue, gray, orange, palegreen a chocolate. Tieto definujú farebnosť navigačného panelu a tlačítká navigácie. Implicitnou voľbou je blue.
11. Podpora cudzích jazykov. Nie všetky jazyky sú podporované, aktuálne je podporovaných len 15 európskych jazykov. Je však možné použiť ako voľbu všetky jazyky, ktoré používame s balíčkom `babel`. Ak jazyk nie je k dispozícii, použije sa implicitná voľba `english`. My máme to šťastie, že `slovak` aj

czech patria medzi podporované jazyky a je k dispozícii príslušný preklad názvov tlačítok navigácie.

12. nocfg je voľba, ktorá potláča uplatnenie konfiguračného súboru, ak si neželáme použiť jeho špecifikácie. Svoje vlastné požiadavky si totiž môžeme uložiť do konfiguračného súboru pdfscreen.cfg. Tento súbor môže obsahovať napr. vlastné preklady navigačných tlačítiek ak jazyk, v ktorom píšeme nepatrí medzi podporované, vlastné farebné schémy, názov grafického súboru s vlastným logom a iné požiadavky na layout stránky. Je teda akousi vlastnou šablónou pre dokument. Ak si ju neželáme v danom prípade použiť, vyradíme ju touto voľbou.

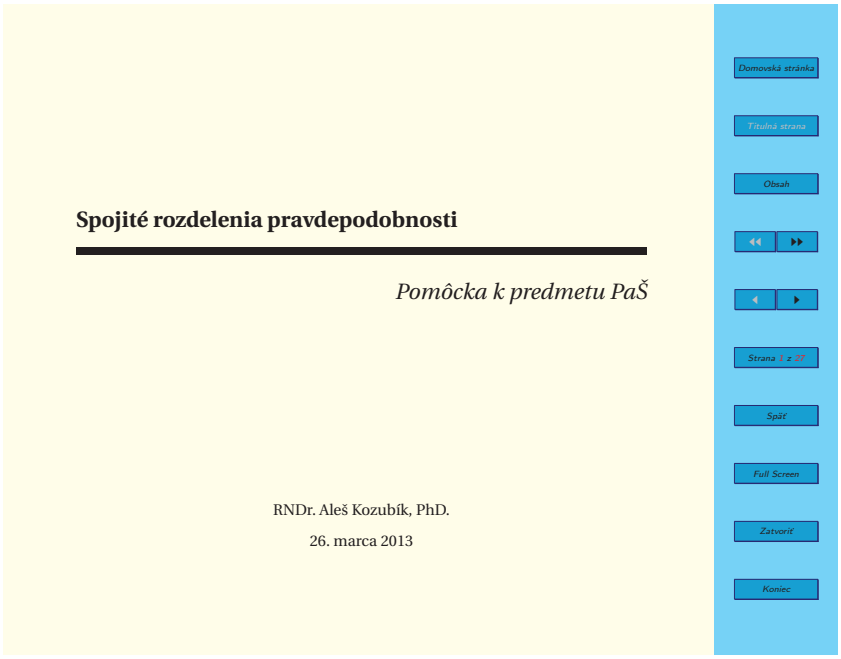
1.2. Ďalšie nastaviteľné parametre

Okrem parametrov zadávaných v rámci volieb existuje ešte niekoľko parametrov, ktoré je možné upraviť, čo prispieva ku väčšej pružnosti výstupu, vytvoreného pomocou balíčka pdfscreen. Ku týmto parametrom patrí:

- \emblema{grafický súbor} určuje názov súboru s grafickým obsahom, ktorý je použitý ako obrázok v navigačnom panele,
- \urlid{URL adresa} priraďuje URL adresu, na ktorú odkazuje tlačítko „Domovská stránka“,
- \screensize{výška,šírka} určuje rozmery obrazovky, na ktorú je prispôsobený pdf výstup. Balíček nemá žiadne implicitné nastavenie rozmerov obrazovky, preto je táto špecifikácia parametrov povinnou. Na rozmery obrazovky neexistujú žiadne obmedzenia, je ju možné zadávať v ľubovoľných jednotkách, ktoré sú dostupné v systéme LaTeX. Pre navigačný panel je implicitne nastavená šírka, ktorá predstavuje 15 % šírky obrazovky.
- \margins{ľavý}{pravý}{horný}{dolný} nastavuje okraje stránky. Poradie nastavených okrajov je zrejmé z pomenovania. Ani okraje nemajú implicitné nastavenia a ich deklarácia je v preambule povinná.

Pre prácu s balíčkom pdfscreen je nevyhnutné mať nainštalované niektoré balíčky, ktoré pdfscreen využíva. Nie je ich potrebné v preambule nahrávať, pdfscreen ich načíta automaticky. Výnimku tvorí len prípad, ak ich chceme načítať so špecifickými voľbami. V takom prípade ich však treba vložiť skôr, než dôjde k načítaniu balíčka pdfscreen. Zoznam potrebných balíčkov uvádzame tu:

hyperref.sty	comment.sty	truncate.sty
graphicx.sty	color.sty	colortbl.sty
calc.sty	amssymb.sty	amsbsy.sty
shortvrb.sty		fancybox.sty



Obr. 1. Ukážka titulnej stránky obrazovkovej verzie dokumentu vytvoreného s balíčkom `pdfscreen`. Navigačný panel umiestnený vpravo

.

2. Navigačný panel

Dizajn navigačného panelu je možné plne prispôsobiť predstavám užívateľa. Má síce svoj implicitný tvar, ktorého ukážka je na obrázku 1, avšak možno ho viacerými príkazmi modifikovať. Predovšetkým príkazom

```
\addButton{dĺžka}{text}
```

je možné vytvoriť tlačítko podľa vlastnej požiadavky. Tak napríklad príkazom

```
\Acrobatmenu{NextPage}{\addButton{1.25in}{Ďalej}}
```

vytvoríme tlačítko odkazujúce na ďalšiu stránku v tejto podobe:

Ďalej

a kliknutím na toto tlačítko sa posunieme na nasledujúcu stránku v dokumente.

Taktiež je možné vytvoriť tlačítka s obrázkom namiesto popisu, k čomu slúži príkaz `\imageButton`. Tento má tri parametre, ktoré sa zadávajú v poradí `šírka`, `výška` a `meno súboru`. Tak napríklad pomocou príkazu:

```
\href{http://www.linux.org}{\imageButton}{1.25cm}{!}{tux.png}
```

vytvoríme tlačítko s obrázkom tučniaka odkazujúce na známu stránku:

Navigačný panel je možné umiestniť na ľavú alebo pravú stranu obrazovky, prípadne je ho možné úplne potlačiť. To docielime voľbami `panelleft`, `panelright` alebo `nopanel`, pri zavádzaní balíčka `pdfscreen`. Takisto šírku navigačného panelu je možné ovplyvniť príkazom `\panelwidth=rozmer`. Implicitne je šírka navigačného panelu nastavená na hodnotu 15 % šírky obrazovky resp. na minimálnu hodnotu šírky jedného palca v prípade, že 15 % šírky obrazovky by kleslo pod túto hodnotu.

3. Ďalšie možnosti

3.1. Pozadie

Pozadie stránky na obrazovke môže byť pokryté grafickým súborom pomocou príkazu `\overlay{názov súboru}`. Taktiež je možné určiť vlastnú farbu pozadia príkazom `backgroundcolor{farba}`, kde `farba` je názov požadovanej farby preddefinovanej v balíčku `color.sty` resp. `xcolor.sty`.

Rovnako ako je možné meniť pozadie stránky na obrazovke, je možné aj pozadie navigačného panelu pokryť požadovaným grafickým súborom. k tomu slúži príkaz `\paneloverlay{názov súboru}`. Ak nešpecifikujeme grafický súbor pre prekrytie, uplatní sa farba pozadia panelu. Pre zmenu farebnosti pozadia navigačného panelu je potrebné túto farbu predefinovať v konfiguračnom súbore `pdfscreen.cfg`.

3.2. Ďalšie tlačítka

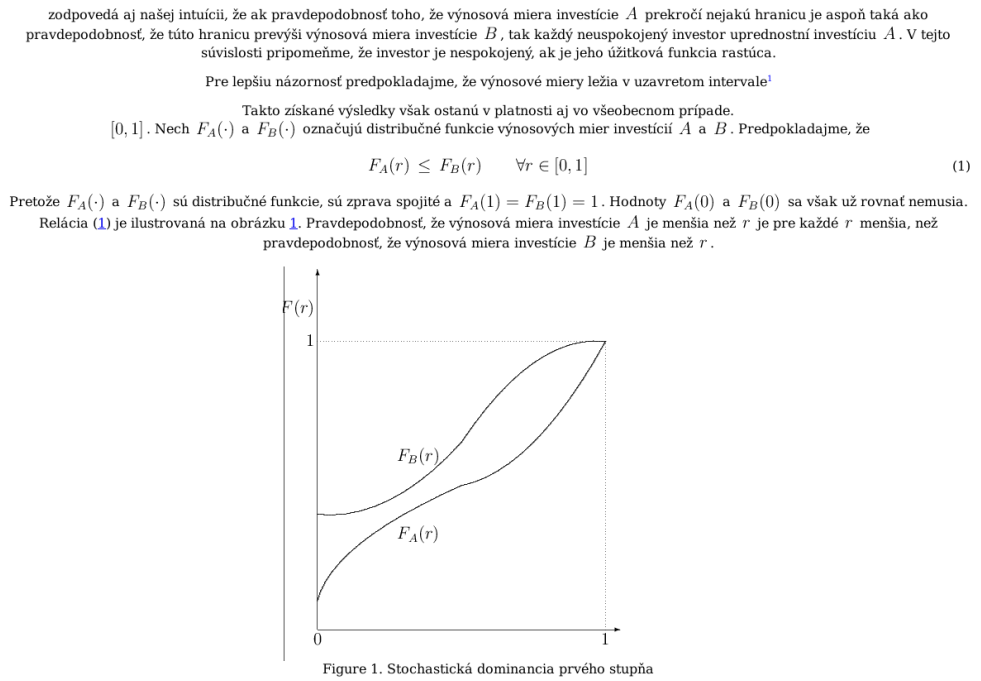
Na stránky je možné pridávať tlačítka do päty resp. záhlavia navigačného panelu. Slúžia k tomu príkazy `\bottombuttons` alebo `\topbuttons`, ktoré spôsobia zobrazenie troch linkov: prepínanie na celoobrazovkový režim zobrazenia, uzavretie daného súboru a ukončenie zobrazovacieho nástroja (obvykle Acrobat Reader).

3.3. Obsah v navigačnom paneli

Do navigačného panela je možné pridať zobrazenie obsahu dokumentu, ak pri načítaní balíčka `pdfscreen` použijeme voľbu `paneltoc`. Tento obsah je však potrebné používať s rozvahou, nakoľko hrozí možnosť vytvorenia príliš dlhého obsahu, ktorý by zaberal aj niekoľko stránok.

4. Iné alternatívy prípravy elektronických publikácií

Popri `.pdf` formáte sa s nástupom elektronických čítačiek kníh čoraz viac presadzuje aj formát `.epub`. Autor sám vyskúšal viaceré nástroje pre tvorbu takýchto dokumentov dokumentov zo zdrojového súboru LaTeX-u, avšak s málo uspokojivými výsledkami.



zodpovedá aj našej intuícii, že ak pravdepodobnosť toho, že výnosová miera investície A prekročí nejakú hranicu je aspoň taká ako pravdepodobnosť, že túto hranicu prevýši výnosová miera investície B, tak každý neuspokojený investor uprednostní investíciu A. V tejto súvislosti pripomeňme, že investor je nespokojený, ak je jeho úžitková funkcia rastúca.

Pre lepšiu názornosť predpokladajme, že výnosové miery ležia v uzavretom intervale[1]

Takto získané výsledky však ostanú v platnosti aj vo všeobecnom prípade.
$[0, 1]$. Nech $F_A(\cdot)$ a $F_B(\cdot)$ označujú distribučné funkcie výnosových mier investícií A a B. Predpokladajme, že

$$F_A(r) \leq F_B(r) \qquad \forall r \in [0, 1] \tag{1}$$

Pretože $F_A(\cdot)$ a $F_B(\cdot)$ sú distribučné funkcie, sú zprava spojité a $F_A(1) = F_B(1) = 1$. Hodnoty $F_A(0)$ a $F_B(0)$ sa však už rovnať nemusia. Relácia (1) je ilustrovaná na obrázku 1. Pravdepodobnosť, že výnosová miera investície A je menšia než r je pre každé r menšia, než pravdepodobnosť, že výnosová miera investície B je menšia než r.

Figure 1. Stochastická dominancia prvého stupňa

Obr. 2. Ukážka dokumentu po konverzii nástrojom `pandoc` do formátu `html`

Prvou kombináciou bolo použitie `pandoc`, ktorý je súčasťou distribúcie Debian Wheezy a nástroja `Calibre`. Postupuje sa pritom tak, že pomocou príkazu

```
pandoc -f latex -t html subor.tex -s -S -R --toc -o subor.html
```

skonvertujeme LaTeX-ový zdrojový text do formátu `.html`. Príslušný `html` súbor potom otvoríme pomocou `calibre`, zaradíme medzi e-knihy a skonvertujeme do formátu `epub`. Výsledky ilustrujeme na obrázkoch 2 a 3.

Ak konverzia do formátu `html` je pomerne úspešná, nemôžeme už to isté tvrdiť o konverzii do formátu `epub`. Vytvorený `html` dokument má nedostatky len pokiaľ sa týka násilného zarovnania všetkých riadkov na stred a nezvládnuté popisky plávajúcich objektov vzhľadom na použitý jazyk dokumentu. Naproti tomu, vo formáte `epub` možno pozorovať úplné „rozpadnutie“ sadzby matematiky a je v tejto podobe de facto nepoužiteľný.

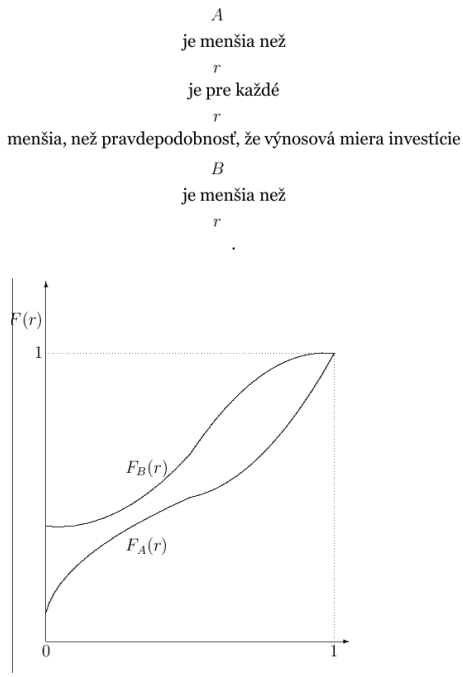


Figure 1. Stochastická dominancia prvého stupňa

Obr. 3. Ukážka dokumentu po konverzii pomocou `calibre` do formátu `epub`

Alternatívnou voľbou je použitie nástroja `text4ht` opäť v kombinácii s nástrojom `calibre`. Balíček `text4ht` je taktiež súčasťou distribúcie Debian Wheezy[1] a po jeho nainštalovaní príkazom

```
htlatex subor.tex
```

transformovať zdrojový text LaTeX-u do `html` formátu. Získaný `html` dokument potom opäť pomocou `calibre` zaradíme medzi e-knihy a vykonáme konverziu do `epub`.

Ako je vidieť z obrázka 4, konverzia v tomto prípade dopadla o poznanie lepšie, aj keď výsledok nie je ani zďaleka optimálny. Je zvládnuté označenie plávajúcich objektov v požadovanom jazyku, sadzba matematiky je zachovaná. Avšak stále pretrváva zarovnávanie na stred riadku (s výnimkou popisu obrázku, ktorý je paradoxne zarovnaný doľava), rovnica je číslovaná o riadok nižšie.

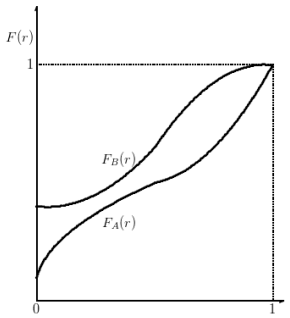
Navyše, ako dokumentuje obrázok 5, stále nie je zvládnutá sadzba tabuliek.

[1] Autor používa túto distribúciu, pre ostatné systémy odporúčam hľadať riešenie na `CTAN`.

Pre lepšiu názornosť predpokladajme, že výnosové miery ležia v uzavretom intervale[1] [0, 1]. Nech $F_A(\cdot)$ a $F_B(\cdot)$ označujú distribučné funkcie výnosových mier investícií A a B. Predpokladajme, že

$$F_A(r) \leq F_B(r) \qquad \forall r \in [0,1] \tag{1}$$

Pretože $F_A(\cdot)$ a $F_B(\cdot)$ sú distribučné funkcie, sú zprava spojité a $F_A(1) = F_B(1) = 1$. Hodnoty $F_A(0)$ a $F_B(0)$ sa však už rovnať nemusia. Relácia (1) je ilustrovaná na obrázku 1. Pravdepodobnosť, že výnosová miera investície A je menšia než r je pre každé r menšia, než pravdepodobnosť, že výnosová miera investície B je menšia než r.



Obr. 1: Stochastická dominancia prvého stupňa

Obr. 4. Ukážka dokumentu po konverzii pomocou `text4ht` a `calibre` do formátu `epub`

Príklad Predpokladajme, že výnosové miery investícií A a B môžu nadobúdať len tri hodnoty:0, $\frac{1}{2}$ a 1, s kumulovanými pravdepodobnosťami popísanými v tabuľke 1. Ľahko sa overí, že podmienka (1) je splnená.

r	0	$\frac{1}{2}$	1
$F_A(r)$	$\frac{1}{3}$	$\frac{2}{3}$	1
$F_B(r)$	$\frac{1}{2}$	$\frac{5}{6}$	1

Tabuľka 1: Príklad stochastickej dominancie prvého stupňa

Obr. 5. Ukážka dokumentu obsahujúceho tabuľku po konverzii pomocou `text4ht` a `calibre` do formátu `epub`

Záver

V príspevku sme si predstavili niekoľko možností, ako zo zdrojového súboru pre LaTeX vytvoriť elektronickú publikáciu. Ako je vidieť z ukážok, stále je najlepšou voľbou použitie balíčka `pdfscreen`, aj keď je už pomerne staršieho dáta a zdá sa,

aj jeho vývoj je zastavený. Novšie nástroje určené pre konverziu do formátu epub ešte majú pred sebou dlhú cestu, kým dosiahnu dokonalosť.

Poďakovanie. Tento príspevok bol podporený grantom slovenskej kultúrno-edukačnej agentúry KEGA č. 011ŽU-4/2014 „Experimentálna matematika – zviditeľnenie neviditeľného“.

Literatúra

[1] BLAŠKO, R.: *LaTeX nie je farba na maľovanie*, Otvorený softvér vo vzdelávaní, výskume a IT riešeniach, zborník medzinárodnej konferencie OSSConf 2010, Žilina, 1.–4. júla 2010, str. 43–52, ISBN 978-80-970457-0-8, http://sospreskoly.org/files/OSSConf2010/ossconf2010-Blasko.pdf.

[2] BLAŠKO, R.: *LaTeX nie je farba na maľovanie, ale na písanie*, Zborník príspevkov z medzinárodnej konferencie OSSConf 2011, Žilina, 1.–4. júla 2011, str. 249–258, ISBN 978-80-970457-1-5.

[3] KOPKA, H. – DALY, P. W.: *LaTeX – Podrobný průvodce*, Brno, Computer Press, 2004, ISBN 80-722-6973-9.

[4] KOZUBÍK, A.: *Prezentačné materiály v triede Bemer*, Zborník príspevkov z medzinárodnej konferencie OSSConf 2011, Žilina, 1.–4. júla 2011, str. 223–235, ISBN 978-80-970457-1-5.

[5] RYBIČKA, J.: *LaTeX pro začátečníky*, Brno, KONVOJ 2003, ISBN 80-7302-049-1.

[6] RYBIČKA, J., ČAČKOVÁ, P., PŘICHYSTAL J.: *Průvodce tvorbou dokumentů*, Bučovice, Nakladatelství Martin Stříž 2011, ISBN 978-80-87106-43-3.

[7] RADHAKRISNAN, C.v.: *Manual pdfscreen*. http://www.ctan.org/tex-archive/macros/latex/contrib/pdfscreen

[8] STŘÍŽ, P.: *Sazba v TeXu a kresba v METAPOSTu*, Bučovice, Nakladatelství Martin Stříž 2011, ISBN 978-80-87106-51-8.

Kontaktná adresa

RNDr. Aleš Kozubík, PhD., Katedra matematických metód, Fakulta riadenia a informatiky, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovenská republika, *Aktuálna adresa*: SOIT, 010 01 Žilina, Slovenská republika,

E-mailová adresa: alesko@frcatel.fri.uniza.sk

6. ročník Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf

OSSConf 2014: 111–120

MIESTO A ÚLOHA TYPOGRAFIE VO VZDELÁVANÍ

ALEŠ KOZUBÍK (SK) A RUDOLF BLAŠKO (SK)

Abstrakt. V príspevku autori predstavujú nový predmet, ktorý je venovaný problematike sadzby textu. V prvej časti predstavujú obsahovú náplň nového predmetu vyučovaného na Fakulte riadenia informatiky Žilinskej Univerzity. V druhej časti na niekoľkých ukážkach ilustrujú skutočnosť, že správna sadzba nie je vecou intuície a je potrebné sa jej systematicky venovať. Typografické minimum by sa tak malo stať organickou súčasťou gramotnosti.

Kľúčové slová. Sadzba textu, LaTeX, textový procesor, spracovanie dokumentov, výučba.

THE PLACE AND ROLE OF THE TYPOGRAPHY IN EDUCATION

Abstract. In this paper authors present a new object which deals with problems of typesetting. In the first part they present the content part of the new subject taught at the Faculty of Management Science and Informatics of the University of Žilina. In the second part they illustrate in several demonstrations the fact that the correct typesetting is not a matter of intuition and we should be systematically care of it. Therefore, the typographic minimum should become an organic part of literacy.

Keywords. Typesetting, LaTeX, text processor, document processing, education.

Úvod

Titul tohto príspevku v sebe vlastne implicitne skrýva odpoveď autorov na nevyslovenú otázku, či je alebo nie je potrebné žiakov resp. študentov učiť, ako má vyzerať správne počítačovo spracovaný dokument. Naše skúsenosti totiž ukazujú, že často sa uspokojujeme s konštatovaním, že každý už dnes ovláda nejaký ten kancelársky balík. A nech už si vyberie akýkoľvek z nich, či otvorený alebo nie, nejakú prácu predsa nejako napíše. Jej vzhľad potom ponúkanými nástrojmi nejako doladí. Veď celú úpravu a formátovanie dokumentu vykoná tento kancelársky balík za nás. Výsledkom takéhoto prístupu potom býva, že sa študent zameria na neprebernú plejádu rôznych použitých písiem, veľké množstvo rôznych typov zvýraznení, vlastné formátovanie nadpisov kapitol či odsekov neudržiac pri tom jednotný štýl v celej práci. Výsledkom potom je dokument, ktorý je neprehľadný a obtiažne čitateľný.

Existuje však aj protinázor, že správnej úprave dokumentu sa treba, tak ako všetkému, učiť. Vytvoriť skutočne kvalitný, prehľadný a dobre čitateľný dokument len na základe intuície je prakticky nemožné. Je ku tomu totiž potrebný súlad viacerých faktorov, ako sú dobrá kompozícia a vhodný sadzbový obrazec, vhodný

typ písma, vhodné riadkovanie, vhodná forma zvýrazňovania textu, nastavenie hlavičiek a pätičiek stránky, rozmiestnenie obrázkov a tabuliek atď. Všetky tieto atribúty kvalitnej sadzby textu patrili v minulosti do výbavy skúsených sadzačov, kým autori sa zameriavali predovšetkým na obsahovú stránku textu.

V dnešnej dobe sa práca posúva do nových dimenzií, kedy sa autor stáva súčasne aj sadzačom svojho vlastného textu. Tento neraz potrebujú prezentovať nielen v tlačenej podobe, ale aj v podobe elektronických výstupov ako sú napríklad webové stránky alebo prezentácie. Ako ukazujú skúsenosti, bez podrobnejšieho oboznámenia sa so zásadami typografie k vytvoreniu kvalitného dokumentu nestačí iba zvládnutie kancelárskych programov. Ako si ukážeme na niektorých ukážkach, aj skutočne vysoko kvalitné monografie či učebnice môžu byť úplne znehodnotené zlým typografickým spracovaním.

Tieto skúsenosti nás viedli k príprave nového predmetu, ktorý je od tohto akademického roku vyučovaný pod názvom „Elektronické spracovanie a úprava dokumentov“. V tomto príspevku teda hovoríme najmä o obsahovej náplni tohto nového predmetu, avšak mnohé ostáva v platnosti aj v širšom kontexte všeobecného povedomia o tvorbe dokumentov.

1. Elektronické spracovanie a prezentácia dokumentov

Elektronické spracovanie a prezentácia dokumentov je názvom predmetu, ktorý sme premiérovo vyučovali v tomto akademickom roku. Uvedený predmet si nevyžaduje žiadne prerekvizity a môže si ho tak zapísať ľubovoľný študent fakulty, či bakalárskeho alebo inžinierskeho štúdia. Predmet je rovnako otvorený pre všetky študijné odbory, teda Manažment, Informatika aj Počítačové inžinierstvo. Aj pri svojej premiére si predmet vybrali študenti z viacerých odborov jak z bakalárskeho, tak aj z inžinierskeho štúdia.

Pri koncipovaní predmetu sme sa nechali inšpirovať najmä Mendelovou univerzitou v Brne, kde sa podobný predmet už niekoľko rokov vyučuje. Rozsah predmetu je v podobe dvoch hodín prednášky a dvoch hodín laboratórneho cvičenia v počítačovom laboratóriu. Predmet si kladie za cieľ poskytnúť študentom základné poznatky z typografie a tvorby a koncepcie dokumentov v papierovej aj elektronickej podobe. Do budúcna sa v súvislosti s novou akreditáciou počíta so zaradením tohto predmetu ako povinne voliteľného do druhého ročníka bakalárskeho štúdia.

1.1. Obsah a ciele predmetu

Obsahovú náplň predmetu je možné charakterizovať v podobe veľmi skrátenej osnovy ako: typografia a jej základy, základy práce v LaTeX-u, MS Worde a Open resp. LibreOffice.org, tvorba dokumentov s pomocou uvedených nástrojov, prezentácie výsledkov s využitím nástrojov `beamer`, MS powerpoint, Impress.

Tomu zodpovedajú aj štyri základné tematické oblasti výučby: základy typografie, tvorba dokumentov s pomocou kancelárskych balíkov, práca v typografickom systéme TEX resp LATEX a prezentácia výsledkov.

Základné ciele, ktoré si kladieme pri tomto predmete, možno zhrnúť v podobe základných vedomostí a zručností, ktoré by mal študent po jeho absolvovaní získať. Je ich možné zhrnúť do nasledujúcich bodov:

- mať základné vedomosti z typografie a polygrafie, poznať pojmy ako napr. sadzobný obrazec, optický stred stránky, hladká a zmiešaná sadzba, poznať náležitosti jednotlivých druhov dokumentu,
- orientovať sa v používaných písmach, rozoznávať druhy a rezy písma, fonty a pod.
- vedieť vhodne navrhnúť a následne aj vytvoriť dokument, či už v kancelárskych balíkoch alebo v systéme LATEX,
- vedieť vytvoriť aj náročnejšie dokumenty (multimediálne, interaktívne),
- vedieť pripraviť prezentáciu svojich výsledkov.

Celkovým prínosom predmetu pre študenta by malo byť získanie schopnosti spracovať pri použití niektorého z uvedených nástrojov svoju záverečnú prácu na profesionálnej alebo takmer profesionálnej úrovni.

1.2. Základy typografie

Ak študenti majú osvojené určité prvky práce s kancelárskymi nástrojmi, tak oblasť typografie je pre nich zväčša „terra incognita“. Preto sme ako prvú do predmetu zaradili časť venujúcu sa typografickej úprave, ktorú je možné charakterizovať ako usporiadanie vzhľadu dokumentu, ktoré je dosiahnuteľné pomocou nástrojov, ktoré má typograf alebo grafik k dispozícii. Sem zaraďujeme napríklad druh a veľkosť písma,[1] vzdialenosti medzi riadkami, usporiadanie textu na strane, používanie obrázkov a tabuliek a pod.

Študenti sa tak oboznámia s tým, že tvorba elektronických dokumentov predstavuje súhrn viacerých činností ako:

- prepis textových predlôh do digitálnej formy alebo ich úprava,
- prevedenie obrazových predlôh do digitálnej podoby a ich konverzia do vhodného formátu,
- úprava digitalizovaných obrazových predlôh pomocou grafických programov (retuš, výrez, farebná úprava a pod.),
- tvorba grafických prvkov pomocou grafických programov,
- kontrolná a záverečná tlač, pri priemyselnej tlači aj vytvorenie záverečného súboru.

V rámci typografického minima sa študenti naučia to, čo by mal každý potenciálny autor vedieť pred prípravou svojho dokumentu:

[1] Toto si samozrejme vyžaduje aj oboznámenie s typografickými jednotkami merania, ktoré sú pre mnohých taktiež veľkou neznámou.

- ako upraviť text do odstavcov, čo je odseková odrážka a ako sa používa, čo je východový riadok, medziriadkový preklad ap.,
- pravidlá delenia slov, aké slová a slovné spojenia sa nedelia,
- aké majú byť medzislovné medzery, aká je ideálna medzislovná medzera, aká by mala byť minimálna a maximálna medzislovná medzera,
- ako sa používajú rozdeľovacie a interpunkčné znamienka, percento, paragraf, skratky, označenia jednotiek merania a aké medzery sa medzi nimi sádzu,
- aké rezy písma kombinovať pri zmiešanej sadzbe,
- ako používať poradovú sadzbu, ako má byť odsadená, ako vytvoriť obsah a register dokumentu,
- ako sádzať tabuľky, ako sa sádzu ich popisy a ako sa umiestňujú v texte,
- ako sádzať grafy a obrázky, ako sa popisujú a umiestňujú,
- ako sa tvoria bibliografické odkazy a zoznam literatúry.

1.3. Kancelárske programy

V rámci predmetu „Elektronické spracovanie a prezentácia dokumentov“ sa venujeme dvom kancelárskym balíkom. Jednak komerčnému MS Office a potom tiež slobodnej alternatíve OpenOffice.org resp. LibreOffice.org. Základy ich používania ovládajú všetci študenti, nakoľko sa s nimi oboznamujú už na nižších stupňoch vzdelávania. Avšak problematiku nastavení vzhľadu dokumentu riešia iba intuitívne.

Pri práci s týmito kancelárskymi balíkmi sa zameriavame predovšetkým na otázky definovania vlastného štýlu dokumentu, vytvoreniu obsahu, ktorý by sa generoval sám, bez potreby jeho prerábania pri každej zmene či doplnení názvov kapitol. Osobitne dôležitým prvkom je sadzba matematických rovníc a vzorcov, vytváranie kvalitných tabuliek a grafov. Z hľadiska úpravy dokumentu venujeme pozornosť výberu vhodného typu písma, jeho základnej veľkosti a rezu, vhodnej voľby zvýrazňovania textu a pod. Vzhľadom na to, že na fakulte prevládajú informatické odbory, je potrebné venovať sa aj pravidlám sadzby výstupov zdrojového kódu programu.

1.4. Systém LaTeX

Za ťažisko predmetu môžeme označiť zvládnutie práce s typografickým systémom LaTeX. Tento je voľne šíriteľným nástrojom a jeho používanie je bezplatné. Navyše je pomerne rozšírený v akademických a vedeckých kruhoch, kde sa stal populárnym najmä vďaka výborne zvládnutej práci s matematickými formulami. Medzi študentmi, ale aj mnohými učiteľmi je rozšírené používanie kancelárskeho MS Word-u. Príčiny tohto stavu sú hneď dve:

- Systém LaTeX nie je editorom typu WYSIWYG. Práca s ním je podobná programovaniu, kedy sa v ľubovoľnom textovom editore napíše zdrojový

kód, ktorý je následne kompilovaný do podoby výsledného `pdf` súboru. Pre začínajúceho užívateľa, poznačeného predchádzajúcim používaním kancelárskych nástrojov, je určitou bariérou skutočnosť, že výsledok svojho snaženia okamžite nevidí.
- Pre používanie systému LaTeX je potrebné mať určité minimálne vedomosti. je treba vedieť, akú tzv. triedu dokumentu chceme vytvoriť, aké rozširujúce balíčky pri tom budeme potrebovať a pod. Nie je teda možný intuitívny prístup a postup formou „k výsledku sa nejako doklikáme“.

Prvý problém sa už dnes čiastočne odstraňuje pomocou takých nástrojov, ako sú `texworks`, `texmaker` alebo `texstudio`. Pri použití týchto nástrojov sa zobrazenie výsledku stáva otázkou jedného kliknutia. Takisto je pre vizualizáciu výsledku možné použiť online nástroje ako napríklad `writelatex.com`.

Problém určitých minimálnych poznatkov už odstrániť možné nie je. Trvá však iba do chvíle, kým užívateľ nie je za svoju námahu odmenený vysokou kvalitou získaného dokumentu. Určité úsilie, ktoré je potrebné pre zvládnutie práce vynaložiť sa vysoko zúročí pri spracovaní zložitejších dokumentov a rozsiahlych projektov ako sú bakalárske práce, diplomové a dizertačné práce, knižné publikácie a pod.

Pri porovnaní práce s kancelárskymi balíkmi a systémom LaTeX tak študenti dospievajú k poznaniu, že je potrebné riešiť optimálny výber použitého nástroja. Kým pre krátke a jednoduché dokumenty ako napr. list sa hodia skôr kancelárske balíky (čomu zodpovedá aj ich prívlastok office), ktoré ho umožňujú pomerne rýchlo a efektívne takmer bez námahy vytvoriť. Naopak, pre spracovanie zložitejších dokumentov jednoznačne víťazí systém LaTeX.

1.5. Prezentácie výsledkov

Poslednou oblasťou, ktorej sa v rámci predmetu „Elektronické spracovanie a prezentácia dokumentov“ venujeme je otázka prezentácie výsledkov. Opäť máme k dispozícii viaceré nástroje.

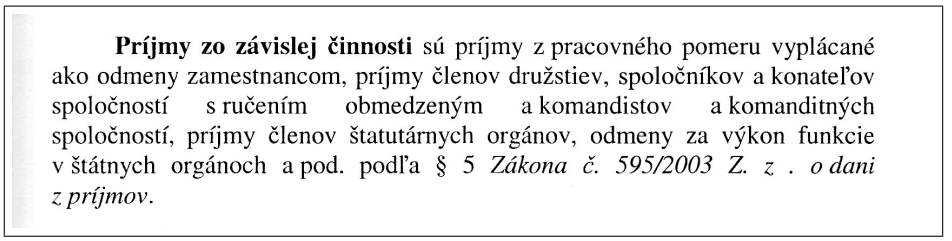
S pomedzi kancelárskych aplikácií je to MS PowerPoint alebo jeho otvorená analógia Impress. V náväznosti na preberané základy práce so systémom LaTeX predstavujeme pre tieto účely triedu dokumentov `Beamer`.

Po absolvovaní predmetu by mal byť študent schopný vytvoriť kvalitnú prezentáciu, mal by vedieť, že je vhodné si zvoliť bezpätkové písmo. Takisto by nemal vo svojich prezentáciách za každú cenu používať rôzne prechodové efekty, ktoré pôsobia často rušivo. Mal by si byť vedomý, že rôzne textúry na pozadí môžu „zneviditeľniť“ obsah prezentácie a že farebnosť písma by mala byť dostatočne kontrastná vo vzťahu ku farbe pozadia.

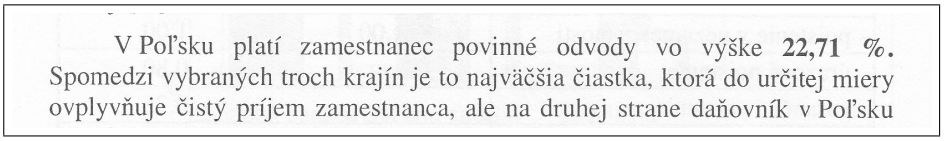
2. Niektoré nedostatky pri sadzbe dokumentov

Asi najbežnejšou chybou, ktorá sa často objavuje nielen v prácach študentov, ale aj učiteľov sú tzv. medzislovné rieky, ktoré vznikajú výskytom príliš veľkých

medzier pri zarovnávaní odstavcov. Text je potom obtiažne čitateľný, pôsobí rušivo a nevyvoláva požadovaný efekt jednoliatej šedi. Na odstránenie tohto nedostatku pritom niekedy stačí len povoliť delenie slov na konci riadku, či vhodne pracovať s pevnými medzerami.


Príjmy zo závislej činnosti sú príjmy z pracovného pomeru vyplácané ako odmeny zamestnancom, príjmy členov družstiev, spoločníkov a konateľov spoločností s ručením obmedzeným a komandistov a komanditných spoločností, príjmy členov štatutárnych orgánov, odmeny za výkon funkcie v štátnych orgánoch a pod. podľa § 5 *Zákona č. 595/2003 Z. z . o dani z príjmov*.

Obr. 1. Ukážka nedokonale zvládnutej sadzby hladkého textu z vydanej učebnice. Chybná sadzba na treťom a poslednom riadku

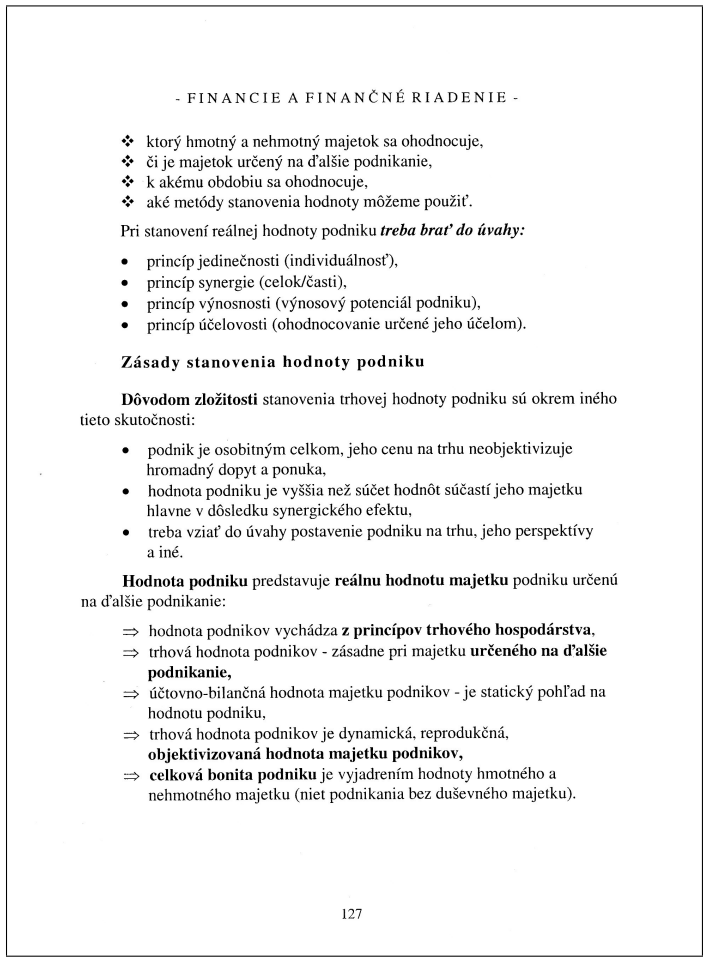

V Poľsku platí zamestnanec povinné odvody vo výške **22,71 %.** Spomedzi vybraných troch krajín je to najväčšia čiastka, ktorá do určitej miery ovplyvňuje čistý príjem zamestnanca, ale na druhej strane daňovník v Poľsku

Obr. 2. Ukážka nedokonale zvládnutej sadzby hladkého textu z vydanej učebnice. Chybná sadzba medzery oddeľujúcej percentá

Ako ukážky si všimnime obrázky 1 a 2. Na obrázku 1 si všimnime najmä chybnú prácu s medzerami, kde použitie pevných medzier pri jednopísmenných predložkách a spojkách spôsobilo následné neprimerané roztiahnutie ostatných medzislovných medzier. To pôsobí pri čítaní rušivo. Podobne na poslednom riadku tejto ukážky je vidieť príliš veľkú medzeru oddeľujúcu symbol paragrafu od jeho číselného použitia (mala byť použitá polovičná alebo aspoň pevná neoddeliteľná medzera. To isté platí aj o označení čísla zákona a takisto oddelenie písmen medzerami v skratke pre zbierku zákonov je nevhodné. Podobná chyba sa opakuje aj na obrázku 2, kde na konci prvého riadku mala byť pre oddelenie percent použitá polovičná (alebo aspoň pevná) medzera.

Na inej strane tej istej publikácie sa môžeme stretnúť so sadzbou vzorca pre budúcu hodnotu v podobe, ako ju ilustruje obrázok 4. Tu vidíme jednak nevhodné použitie symbolu hviezdičky pre násobenie, nehovoriac už ani o jej vertikálnom umiestnení v riadku. Rovnako vo vysvetlivkách ku vzorcu by mali byť príslušné symboly označenia vysádzané rovnakým rezom písma, ako sú vo vzorci. Teda správne by príslušný text mal vyzerať takto:

$$BH_a = a \cdot \frac{(1+1)^n - 1}{i},$$



- FINANCIE A FINANČNÉ RIADENIE -

❖ ktorý hmotný a nehmotný majetok sa ohodnocuje,
❖ či je majetok určený na ďalšie podnikanie,
❖ k akému obdobiu sa ohodnocuje,
❖ aké metódy stanovenia hodnoty môžeme použiť.

Pri stanovení reálnej hodnoty podniku ***treba brať do úvahy:***

- princíp jedinečnosti (individuálnosť),
- princíp synergie (celok/časti),
- princíp výnosnosti (výnosový potenciál podniku),
- princíp účelovosti (ohodnocovanie určené jeho účelom).

Zásady stanovenia hodnoty podniku

Dôvodom zložitosti stanovenia trhovej hodnoty podniku sú okrem iného tieto skutočnosti:

- podnik je osobitným celkom, jeho cenu na trhu neobjektivizuje hromadný dopyt a ponuka,
- hodnota podniku je vyššia než súčet hodnôt súčastí jeho majetku hlavne v dôsledku synergického efektu,
- treba vziať do úvahy postavenie podniku na trhu, jeho perspektívy a iné.

Hodnota podniku predstavuje **reálnu hodnotu majetku** podniku určenú na ďalšie podnikanie:

⇒ hodnota podnikov vychádza **z princípov trhového hospodárstva**,
⇒ trhová hodnota podnikov - zásadne pri majetku **určeného na ďalšie podnikanie,**
⇒ účtovno-bilančná hodnota majetku podnikov - je statický pohľad na hodnotu podniku,
⇒ trhová hodnota podnikov je dynamická, reprodukčná, **objektivizovaná hodnota majetku podnikov,**
⇒ **celková bonita podniku** je vyjadrením hodnoty hmotného a nehmotného majetku (niet podnikania bez duševného majetku).

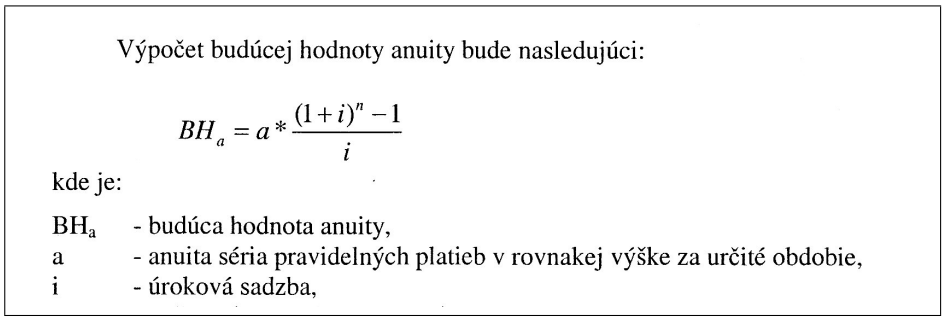
127

Obr. 3. Ukážka nedokonale zvládnutej poradovej sadzby. Ako by sadzač chcel ilustrovať čo najviac alternatív označenia odrážok

kde je:

BH_a budúca hodnota anuity,
a anuita séria pravidelných platieb v rovnakej výške za určité obdobie,
i úroková sadzba,

Na obrázku 3 nemožno prehliadnuť akúsi nerozhodnosť, aké označenie odrážok zvoliť. Len na tejto stránke sa stretáme až so štyrmi rôznymi symbolmi a na ďalších stranách publikácie by sme našli ešte ďalšie alternatívy. Mali by sme ctiť zásadu, že pre odrážky rovnakej úrovne v poradovej sadzbe je potrebné použiť rovnaké symboly a rovnaké zarážky od okraja sadzobného obrazca. Na tejto stránke si môžeme všimnúť aj dvojtvaré zvýrazňovanie textu – raz tučným rezom písma, druhýkrát je to pre zmenu tučnou kurzívou. Aj tu by bolo vhodné pridržať sa



Výpočet budúcej hodnoty anuity bude nasledujúci:

$$BH_a = a * \frac{(1+i)^n - 1}{i}$$

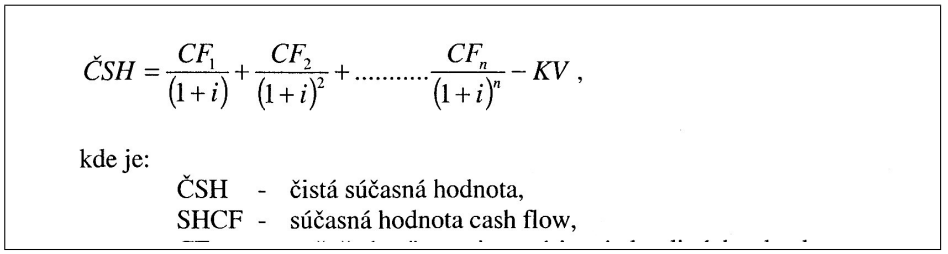
kde je:

BH_a - budúca hodnota anuity,
a - anuita séria pravidelných platieb v rovnakej výške za určité obdobie,
i - úroková sadzba,

Obr. 4. Ukážka nezvládnutej sadzby matematiky vo vydanej učebnici. Je použitý neadekvátny znak pre násobenie a vysvetlivky majú odlišný rez písma oproti vzorcu samotnému

jednotnosti. Navyše, nejednotnosť môže viesť ku zneprehľadneniu celého textu a komplikuje pochopenie samotného obsahu (čitateľ rozmýšľa, prečo je označenie odlišné, či ide o rovnaké premenné alebo nie).

Samostatným problémom je otázka sadzby matematiky. Na niekoľkých ukážkach si ilustrujeme niektoré najbežnejšie chyby.



$$ČSH = \frac{CF_1}{(1+i)} + \frac{CF_2}{(1+i)^2} + \ldots\ldots\ldots \frac{CF_n}{(1+i)^n} - KV\,,$$

kde je:

ČSH - čistá súčasná hodnota,
SHCF - súčasná hodnota cash flow,

Obr. 5. Ukážka nesprávnej sadzby matematického vzorca. nesprávne výpustky, vysvetlenie symbolov opäť rezom písma nezodpovedá vzorcu

Na obrázku 5 neprehliadneme už spomenutý problém s odlišným rezom písma pre označenie premenných vo vzorci a vo vysvetlivkách. Navyše, pribúda problém s výpustkami, ktoré sa majú sádzať ako tri bodky. Akékoľvek navyšovanie tohto počtu nie je v súlade s normou. Taktiež majú byť v danom prípade vysádzané v strede riadku na úrovni znamienok aritmetických operácií, ktoré je potrebné opakovať nielen pred výpustkami, ale aj po nich. Správna sadzba vzorca má teda vyzerať takto:

$$\frac{CF_1}{1+i} + \frac{CF_2}{(1+i)^2} + \cdots + \frac{CF_n}{(1+i)^n} - KV.$$

Chybe vo výpustkoch sa neubránili ani pri sadzbe českej verzie známej zbierky úloh z matematickej analýzy od B. P. Demidoviča, kde sa môžeme stretnúť so sadzbou nekonečného radu tak, ako ilustruje 6. Sadzač ako by sa nevedel rozhodnúť,

3. **První srovnávací kritérium.** Mějme kromě řady (1) ještě řadu

$$b_1 + b_2 + \ldots + b_n + \ldots = \sum_{n=1}^{\infty} b_n$$

Jestliže pro každé $n \geq n_0$ platí nerovnost

Obr. 6. Ukážka z českej verzie Demidovičovej zbierky úloh z matematickej analýzy. Výpustky sú v nesprávnej výške a sú nejednotné

či má výpustky sádzať vo forme troch bodiek, alebo štyroch. Pritom podľa normy majú byť výpustky vždy v podobe troch bodiek a to v danom prípade na úrovni stredu riadku, teda v rovine znamienok matematických operácií. Správna sadzba by teda mala vyzerať takto:

$$b_1 + b_2 + \cdots + b_n + \cdots = \sum_{n=1}^{\infty} b_n$$

Iným príkladom, z tej istej publikácie, môže byť ukážka sadzby integrálov, tak ako ju vidíme na obrázku 7.

2346. $\int_0^{+\infty} e^{-ax} \cos bx\, dx \quad (a > 0).$

2347. $\int_0^{+\infty} e^{-ax} \sin bx\, dx \quad (a > 0).$

Obr. 7. Ukážka z českej verzie Demidovičovej zbierky úloh z matematickej analýzy. Znak pre Eulerovo číslo a symbol diferenciálu majú byť sádzané vzpriameným rezom písma

Tu si môžeme všimnúť, že sadzač zvolil nevhodné použitie kurzívy, pričom ju použil aj pre označenie Eulerovho čísla e a takisto aj pre symbol diferenciálu na konci integrálov. Správne by teda mali byť tieto integrály vysádzané takto:

$$\int_0^{+\infty} \mathrm{e}^{-ax} \cos bx \,\mathrm{d}x \quad (a > 0), \qquad \text{resp.} \qquad \int_0^{+\infty} \mathrm{e}^{-ax} \sin bx \,\mathrm{d}x \quad (a > 0).$$

Záver

Ako sme videli z chýb na ukážkach, ktoré mimochodom nie sú v publikáciách nijak ojedinelé, správna úprava sadzby textu v žiadnom prípade nie je vrodenou

schopnosťou ani záležitosťou intuície. Naopak, ukazuje sa, že má zmysel a je potrebné vzdelávať sa aj v tejto oblasti. Ak teda hovoríme o počítačovej gramotnosti ako treťom prvku schopností nových generácií a o nástupe digitálneho veku, mali by sme sa zamýšľať aj nad tým, že zvládnutie určitého typografického minima je nedeliteľnou súčasťou tejto počítačovej gramotnosti. Ak môžeme dnes pozorovať aj snahy a vyradenie klasického ručného písma a jeho nahradenie akýmsi klonom písma tlačeného, či zľahčovanie úlohy písomného prejavu tvrdeniami, že texty sa už aj tak spracúvajú na počítačoch, potom sa treba zamýšľať aj nad tým, ako viesť ku uspokojivým výsledkom. Je teda na zamyslenie, či by základy typografie nemali byť postupne začlenené aj do obsahu vzdelávania na nižších stupňoch vzdelávania. Veď informatika a počítačové spracovanie textu nie je len o tieňovanom bielom písme na zelenom mramorovom pozadí a podobných absurdných požiadavkách.

Poďakovanie. Tento príspevok vznikol s podporou grantov KEGA-011ŽU-4/2014 „Experimentálna matematika – zviditeľnenie neviditeľného“ a VEGA-1/0890/14 „Stochastické modelovanie rozhodovacích procesov v motivovaní ľudského potenciálu.“

Literatúra

[1] PANÁK, J., ČERPAN, M., DVONKA, V., KARPINSKÝ, Ľ., KORDOŠ, P., MIKULA, M., JAKUCEWICZ, S.: *Polygrafické minimum*, Zväz polygrafie na Slovensku, typoset Bratislava 2000, ISBN 80-967811-2-X.

[2] RYBIČKA, J.: *LaTeX pro začátečníky*, Brno, KONVOJ 2003, ISBN 80-7302-049-1.

[3] RYBIČKA, J., ČAČKOVÁ, P., PŘICHYSTAL J.: *Průvodce tvorbou dokumentů*, Bučovice, Nakladatelství Martin Stříž 2011, ISBN 978-80-87106-43-3.

[4] ŠÍP, R.: *Typografické minimum*, Zväz polygrafie na Slovensku, SOU polygrafické Bratislava 2000, ISBN 80-967598-5-X.

Kontaktné adresy

RNDr. Aleš Kozubík, PhD., Katedra matematických metód, Fakulta riadenia a informatiky, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovenská republika, *Aktuálna adresa*: SOIT, 010 01 Žilina, Slovenská republika,
E-mailová adresa: `alesko@frcatel.fri.uniza.sk`

RNDr. Rudolf Blaško, PhD., Katedra matematických metód, Fakulta riadenia a informatiky, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovenská republika, *Aktuálna adresa*: SOIT, 010 01 Žilina, Slovenská republika,
E-mailová adresa: `beerb@frcatel.fri.uniza.sk`, `http://frcatel.fri.uniza.sk/~beerb/`

6. ročník
Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf

OSSConf 2014: 121–130

TVORBA DOKUMENTOV PRE OBLASŤ VZDELÁVANIA POMOCOU OPEN SOURCE NÁSTROJOV

PAVOL LAJČIAK (SK)

Abstrakt. Článok sa zaoberá možnosťami tvorby prevažne textových dokumentov zo zameraním na oblasť vzdelávania na vysokých školách a univerzitách s využitím open source softvérových nástrojov a technológií, pomocou ktorých môžeme vytvárať tieto dokumenty elektronickou cestou. Zameriavame sa na vlastnosti a štruktúru textových dokumentov, no na okraj sú spomenuté aj dokumenty iných typov. Textové dokumenty kategorizujeme do rôznych kategórii podľa svojho cieľa a učenia. Tiež je tu uvedený prehľad softvérových nástrojov a formátov, ktoré podporujú tvorbu textových dokumentov. Zaoberáme sa aj samotným procesom tvorby elektronického dokumentu. Ďalej sa zaoberáme systémami na distribúciu a správu elektronických dokumentov a výhodami a nevýhodami jednotlivých foriem šírenia – formou tlačeného alebo elektronického dokumentu.

Kľúčové slová. dokument, elektronický dokument, štruktúra, model.

CREATING EDUCATION DOCUMENTS IN OPEN SOURCE TOOLS

Abstract. The paper explores the possibilities of creating mainly text documents focused to education at colleges and universities with the use of open source software tools. We describe various software tools and technologies by which we can create mainly text documents electronically. We focus on properties and the structure of text documents, but we also mention other types of documents. We categorize Text documents in various categories according to their objective and learning. There is an overview of software tools and formats that support creating text documents. We also deal with the process of creating electronic document. Furthermore, we are dealing with systems for the distribution and management of electronic documents and benefits and disadvantages of the various forms of spreading — in the form of a printed or electronic document.

Keywords. document, electronic document, structure, model.

Úvod

S pokusmi o záznam, reprodukciu a odovzdávanie informácií rôzneho druhu sa v histórii ľudstva stretávame od nepamäti. Svedčia o tom rôzne nájdené historické artefakty v rámci archeologických a historických výskumov. Ľudstvo sa snažilo informácie zaznamenať rôznym spôsobom a na rozličné prenosové média. V rámci histórie ľudstva a ľudskej spoločnosti môžeme za míľniky pokladať napríklad nájdené nástenné maľby, drevené, hlinené tabuľky, kožené a papyrusové

zvitky, cez vynájdenie kníhtlače až po dnešné softvérové a hardvérové technológie založené na digitálnych počítačoch.

Dokumenty môžeme vytvárať rozličným spôsobom a to ručne, pomocou písacích strojov alebo pomocou počítačov s príslušným softvérovýcm vybavením. V článku prehľadového charakteru sa zameriavame hlavne na tvorbu dokumentov pomocou open source technológií, ktoré podporujú jednotlivé fázy procesu tvorby dokumentov.

1. Dokument

Dokumenty rôzneho druhu sprevádzajú ľudstvo celou jeho históriou. Dôležité informácie rôzneho druhu sa zaznamenávali pomocou rôznych nástrojov na rôzne médiá. Z histórie poznáme drevené a kamenné tabuľky, papyrusové a kožené zvitky, ručne písané dokumenty, ručne písané knihy, ktoré hlavne v stredoveku písali a prepisovali mnísi v kláštoroch. Výroba takýchto dokumentov bola fyzicky a časovo náročná a ich výroba bola aj drahá. S vynájdením kníhtlače sa ich výroba značne zjednodušila. K ďalšiemu výraznému zjednodušeniu došlo používaním počítačov.

Dokumenty môžeme kategorizovať do rôznych kategórií podľa rôznych kritérií a oblastí ľudského poznania. Kritéria na rozdelenie do rozličných kategórií môže byť napríklad tieto: dĺžka dokumentu, oblasť ľudského poznania, vedný odbor. V rámci knižníc existuje medzinárodné desatinné triedenie. Dokumenty tiež môžeme deliť napríklad podľa literárneho žánru – román, rozprávka, poviedka, epos, bájka, básnická zbierka.

Podľa cieľového záznamového médiá môžeme dokumenty deliť na fyzické a elektronické. Dnes veľká väčšina dokumentov, ktoré sú nakoniec vytlačené do fyzickej podoby sú spracovávané elektronickou cestou alebo existuje ich fyzické aj elektronické vydanie. Každá výsledná forma dokumentu má svoje výhody aj nevýhody. Otázkou je ako s týmito dokumentmi zachádzať. Systém spravovania fyzických dokumentov sa vyvíjal celé stáročia. V posledných desaťročiach nastal veľký rozmach elektronických dokumentov, tiež dochádza k digitalizácii dokumentov, ktoré sú zaznamenané na fyzických médiách. Zároveň sa vyvíjajú systémy na správu elektronických a fyzických dokumentov. Je dôležité si uvedomiť, že fyzické ako aj elektronické dokumenty tvoria jeden svet informačnej spoločnosti.

Preto existuje oprávnená otázka, čo to vlastne dokument je. Existuje viacero definícií dokumentu a nie je možné ich všetky uviesť. Napríklad slovník Free Dictionary definuje dokument ako písomný záznam, ktorý poskytuje informáciu (zvlášť informáciu oficiálnej povahy) [1]. Ďalej slovník Oxford Dictionary definuje dokument ako úryvok písanej, tlačenej alebo elektronickej podstaty, ktorý poskytuje informáciu alebo fakt, ktorý slúži ako oficiálny záznam [2].

2. Štruktúra dokumentu

Na štruktúru dokumentu sa môžeme pozerať z rôznych uhlov pohľadov. Každý dokument sa skladá z rôznych prvkov, ktoré tento dokument vytvárajú. Každý z týchto prvkov môže byť priradený k určitej vrstve. Vrstvy v dokumente majú súvis s ľudským myslením a vnímaním, ale sú aj vrstvy technického a informatického charakteru. Jednotlivé prvky môžu byť napr. na fyzickej vrstve, grafickej vrstve, jazykovednej vrstve, technickej vrstve. Podobne ako technický informačný systém má svoje vrstvy, tak svoje vrstvy má aj dokument, či už ide o fyzický dokument, alebo elektronický dokument. Elektronický dokument obsahuje viacej vrstiev ako fyzický dokument a môže obsahovať prvky, ktoré sú pomocou fyzického dokumentu ťažko realizovateľné, alebo nerealizovateľné. Niektoré pohľady sa pozerajú na dokument ako na monolitickú entitu. V článku opomenieme ručnú tvorbu dokumentov, ktorá má svoje miesto pri tvorbe dokumentov, má svoju krásu a používa sa hlavne v umeleckej tvorbe. Zameriame sa na tvorbu dokumentov a štruktúru dokumentov pre oblasť vzdelávania.

2.1. Fyzická úroveň

Fyzický dokument sa na fyzickej úrovni skladá z bodov vytlačených na nosiči – médiu, ktorým je obyčajne papier. V prípade elektronického dokumentu ide o množinu dát zaznamenaných na pamäťovom médiu, ale aj množinu dát zobrazujúcu sa na zobrazovacej jednotke zariadenia používaného na využitie obsahu dokumentu.

2.2. Typografická úroveň

Každý dokument sa skladá zo znakov. Do množiny znakov patrí abeceda, diakritické znamienka, ktorým tiež hovoríme akcenty, číslice, zátvorky, menové znaky, matematické znaky, špeciálne znaky a ďalšie znaky. Písmená delíme na minuskuly (malé písmená) a majuskuly (veľké písmená). Dôležitým prvkom na úrovni typografie je tiež písmová osnova. Ide o sústavu súbežných vodorovných čiar – doťažníc. Ide o základnú, dolnú, strednú, hornú a akcentovú doťažnicu, okrem toho existuje horná a dolná indexová doťažnica. Ďalšími dôležitým prvkom na tejto úrovni je písmo a s ním spojené pojmy ako meranie písma, merná sústava, stupeň písma, veľkosť písma, druh písma, typ písma, rez písma, písmová rodina, duktus.

2.3. Úroveň členenia

Zo znakov môžeme zostavovať slová, vety, súvetia, odseky, nadpisy, časti, kapitoly, podkapitoly, podpodkapitoly. Na úrovni členenia môže dokument môže obsahovať ďalšie objekty a to obrázky, tabuľky, animácie, zvuky, videa, formuláre, skripty, grafy, matematické vzorce, chemické vzorce, notový zapis a ďalšie.

2.4. Lingvistická úroveň

Na úrovni lingvistiky pracujeme pri odborných dokumentoch pracujeme so spisovným jazykom. Zo znakov môžeme napríklad zostaviť aj bezvýznamový text, ktorý sa používa hlavne pri návrhu tlačovín. Príkladom bezvýznamového textu môže byť „Lorem ipsum dolor sit amet.“, ktorý si môžeme vygenerovať na stránke `lipsum.com`. Na lingvistickej úrovni môžeme hovoriť o sémantickej, morfologickej, syntaktickej a inej úrovni dokumentu.

2.5. Prezentačná úroveň

Je úroveň, ktorá hovorí o tom ako bude výsledný dokument prezentovaný. Toto závisí od použitého formátu dokumentu. V prípade, že ide o elektronický dokument záleží na použitom formáte, v ktorom je dokument uložený, ale aj na použitom zariadení, na ktorom je dokument využívaný.

2.6. Aplikačná úroveň

Hlavne v prípade elektronických dokumentov môžeme hovoriť o aplikačnej úrovni. Obsah elektronického dokumentu je využívaný v určitej aplikácii a okrem textového obsahu môžeme využívať v ňom mediálne a programové prvky – napríklad formuláre a skripty.

Okrem prvkov zložených zo znakov môžu sa v dokumentoch nachádzať aj netextové prvky, a to obrázky, tabuľky, animácie, zvuky, videa, formuláre, skripty, grafy, vzorce, notový zápis a podobne. Ich podpora závisí od konkrétneho formátu a programu.

3. Model dokumentu

Ako sme už spomenuli, každý dokument môže byť zložených z rôznych prvkov – objektov, má nejakú štruktúru. Túto štruktúru môžeme zachytiť pomocou modelu dokumentu. Tento môžeme zobraziť pomocou orientovaného grafu, prípadne popísať pomocou nejakého modelovacieho jazyka. Dokument môže obsahovať tieto prvky: nadpisy rôznych úrovní, odsek, obsah, register, citácia, krížový odkaz, hypertextový odkaz. Okrem prvkov zložených zo znakov, sa v dokumentoch môžu nachádzať aj netextové prvky ako číslované a nečíslované zoznamy, obrázky, animácie, zvuky, tabuľky, videa, formuláre, skripty, grafy, vzorce, notový zápis a podobne. Ktoré konkrétne prvky dokument obsahuje, záleží od triedy dokumentu. Ich podpora závisí od konkrétneho formátu a programu. Príklad je DOM (Document object model), kde objekty v dokumente XML alebo HTML sú dostupné pomocou neho. Takéto logické rozčlenenie dokumentu nám umožňuje oddelenie obsahu od prezentácie. Štruktúru dokumentu môžeme modelovať aj pomocou jazykov XML a UML.

4. Kategorizácia dokumentov

Ako sme spomenuli na začiatku, dokumenty môžeme kategorizovať podľa rôznych kritérií. Každý dokument patriaci do rovnakej kategórie má isté spoločné znaky a podobnú štruktúru. V tejto časti chceme vymenovať niektoré dokumenty používajúce sa v oblasti vzdelávania. Ide o kategorizáciu podľa typu dokumentu. Vo vzdelávaní sa používajú rôzne typy dokumentov a sú to seminárna práca, ročníková práca, bakalárska práca, diplomová práca, ŠVOČ, vedecká monografia, odborná monografia, vysokoškolská učebnica, skriptá, učebný text, návody na cvičenia, rigorózna práca, dizertačná práca, habilitačná práca, iné záverečné a kvalifikačné práce a ďalšie. Podrobnejší prehľad rôznych dokumentov z oblasti vzdelávania môžete nájsť napríklad v Akademickej príručke. Pri kategorizácii dokumentov môžeme ako jeden z parametrov použiť dĺžku dokumentu alebo množstvo informácii v ňom obsiahnutých. Pre tento účel boli zavedené pojmy ako normostrana, ktorá obsahuje 30 riadkov po 60 znakov, čo je spolu 1800 znakov. Väčšou jednotkou je autorský hárok, ktorý zahŕňa 20 normostrán, čo je 20 strán po 1800 znakov, čo spolu tvorí 36 000 znakov.

Dokumenty môžeme ďalej rozdeliť na textové, netextové, interaktívne, neinteraktívne, s pevnou sadzbou, s flexibilnou sadzbou, s responzívnym dizajnom, neresponzívnym dizajnom, multimediálne, nemultimediálne.

5. Proces tvorby dokumentu

S tvorbou dokumentu je spojený proces vytvárania dokumentu a ako každý proces, aj tento proces má svoje fázy a stavy, od jeho štartu a až do chvíle, kým je k dispozícii výsledný dokument. Proces tvorby dokumnentu môžeme znázorniť rôznym spôsobom — napríklad pomocou vývojového diagramu alebo orientovaného grafu. Nadradeným k procesu tvorby dokumentu je publikačný proces, ktorý obsahuje širšiu množinu krokov.

Do procesu tvorby a publikovania dokumentov môže byť zapojené rôzne množstvo ľudí vrátane autora dokumentu. Dôležitým prvkom pri tvorbe dokumentu je výsledné určenie a typ dokumentu. Dokumenty môžeme vytvárať rôznym spôsobom, článok sa zameriava na tvorbu pomocou digitálnych technológií. Výsledný dokument je potrebné vytvárať s ohľadom na výsledné médium, na ktorom sa daný dokument bude používať.

Pre proces tvorby a publikovania dokumentov je potrebné mať vhodnú hardvérovú, softvérovú a personálnu základňu. Na tvorbe dokumentu sa môže okrem autora obýčajne podieľa pomerne široký tím ľudí. Pri procese tvorby dokumentu je potrebné vykonať množstvo operácií. Ide o intelektuálne operácie, technické operácie, personálne operácie.

Tento proces môžeme rozdeliť napríklad na tieto fázy – plánovanie, písanie, revidovanie a publikovanie [3]. Na inom mieste sa uvádzajú tieto fázy publikačného

procesu: písanie, editovanie, produkcia, publikovanie, distribúcia, predaj, propagovanie.

Pri tvorbe a publikovaní dokumentu sa môžeme stretnúť z niektorými z týchto krokov. Ich vymenovanie nie je úplné a ani konečné. Ide napríklad o tieto kroky: stanovenie kritérií pre tvorbu dokumentu — určenie typu dokumentu — určenie rozsahu dokumentu — určenie štýlu dokumentu — určenie cieľovej skupiny — určenie typu výstupného média — stanovenie výstupných parametrov, rozlíšenia, DPI — príprava dokumentu — príprava štruktúry — tvorba osnovy — zber materiálov — zhromažďovanie materiálov — zoraďovanie materiálov — usporadúvanie informácii — brainstorming — myslenie — tvorba vlastných materiálov — citovanie — formálna úprava dokumentu — editovanie — korektúra — typografická, jazyková — revízie — spolupráca — publikovanie — predtlačová príprava — tlač — viazanie — vydanie — šírenie — distribúcia — diseminácia — propagovanie — uchovávanie — ochrana dokumentov — zamknutie dokumentu.

6. Nástroje a formáty na tvorbu dokumentov

6.1. Nástroje na tvorbu dokumentov

Pre tvorbu dokumentov potrebujeme mať potrebné zázemie / infraštruktúru. Potrebujeme mať vhodný hardvér, softvér ako aj potrebnú počítačovú gramotnosť. Vedieť ovládať potrebnú aplikáciu alebo softvérový systém. Existuje veľa aplikácií systémov na tvorbu dokumentov a každý z nich podporuje isté vstupné a výstupné formáty. Ak daný softvér nepodporuje určitý formát, vždy existuje možnosť konverzie pomocou ďalšieho softvéru.

Existujú rôzne prístupy k tvorbe dokumentov. Bez použitia počítačov môžeme dokumenty vytvárať ručne, manuálne. Čo sa týka tvorby dokumentov pomocou počítačov, existujú rôzne prístupy k tvorbe dokumentov.

Prvým prístupom k tvorbe dokumentov je WYSIWYG (What You See Is What You Get). Do tejto kategórie patria textové pocesory, ktoré sú obyčajne súčasťou kancelárskeho balíka. Okrem textových procesorov do tejto kategórie patria aj DTP (DeskTop Publishing) aplikácie. Medzi textové procesory patria otvorené aplikácie LibreOffice Writer, Apache OpenOffice.org Writer, Abiword. Proprietárne aplikácie Microsoft Word. Pre operačný systém Mac OS je to Apple Pages. Pre operačný systém Android sú to Kingsoft Office, Polaris Office, Think-Free Office. Okrem toho existujú online kancelárske balíky ako Google Dokumenty a ZOHO. Medzi otvorené DTP programy patrí Scribus.

Druhým prístupom k tvorbe dokumentov je prístup WYSIWYM (What You See Is What You Mean). Pri tomto prístupe sa používa sada príkazov, pomocou ktorých formátujeme text. Do tejto kategórie patria značkovacie a formátovacie jazyky. Sú to napríklad jazyky a systémy ako TEX, LATEX, docbook, asciidoc,

HTML, XHTML, XML, SGML a ďalšie. Tretím prístupom je hybrid medzi WYSIWYG a WYSIWYM. Ide vlastne o WYSIWYM, kde ale dokument tvoríme v prostredí WYSIWYG. Klasickým prestaviteľom je aplikácia LyX.

Okrem tvorby dokumentov v niektorom z vyššie uvedených programov môžeme tvoriť elektronické dokumenty vo forme elektronických kníh v niektorom z formátov pre elektronické knihy. Tvoriť elektronické knihy môžeme rôznym spôsobom. Z otvoreného softvéru sa vyvíja program Calligra Author, ktorý je na túto činnosť priamo určený. Elektronické knihy môžu byť v rôznom výstupnom formáte. Medzi najpoužívanejšie formáty patria PDF, epub, mobi. Elektronické knihy môžeme čítať pomocou softvérovej čítačky alebo môžeme použiť hardvérovú čítačku. Každá čítačka podporuje iné formáty. Taktiež za zmienku stojí program Calibre, ktorý slúži na manažovanie elektronických kníh.

Pri spracovaní textu pomocou textových procesorov je jedným z najväčších problémov vzájomná nekompatibilita medzi textovými procesormi. Ďalším problémom je tiež tiež nekompatibilita medzi verziami toho istého textového procesora.

6.2. Ďalšie podporné nástroje nástroje

Ako sme spomínali, do textových dokumentov môžeme vkladať ďalšie objekty. Veľmi častými objektami v dokumentoch sú obrázky. Tieto môžu byť rastrové alebo vektorové. Rastrové obrázky môžeme tvoriť alebo upravovať pomocou programu Gimp, vektorové zasa pomocou programu Inkscape, LibreOffice Draw, Apache OpenOffice Draw, Xara Xtreme.

6.3. Formáty elektronických dokumentov

Pri tvorbe elektronických dokumentov sa stretávame so zdrojovým formátom a výstupným formátom dokumentu. Ak ide o tvorbu dokumentov pomocou WYSIWYG, zdrojový a výstupný formát je totožný. V prípade, že tvoríme dokumet metódou WYSIWYM, zdrojový formát a výstupný formát sú rôzne. Zdrojový formátom je čistý textový súbor so syntaxou v niektorom z jazykov TEX, asciidoc, docbook, XML, SGML, šablóny, eLML, HTML. Výstupným formátom môže byť napríklad postscript, PDF alebo iné. Pre elektronické knihy môže byť výstupným formátom PDF, MOBI, EPUB, DOC, FB2, HTML, DOCX, PRC, TXT, RTF. Medzi dôležité výstupné formáty patria napríklad Postcript, PDF, ODF. Dôležitým štandardom je PDF. Adobe PDF 1.7. je definovaný štandardom ISO 32000-1. Open Document Format je zas definovaný štandardom ISO/IEC 26300:2006.

6.4. Nástroje na konverziu dokumentov

Na konvertovanie dokumentov môžeme použiť rozličné nástroje. Medzi najuniverzálnejšie patrí asi konvertor pandoc. Ďalšími užitočnými nástrojmi sú konvertory latex2html, latex2rtf a convert.

7. Výstupné médium

Výstupným médiom môže byť papier alebo zobrazovacia jednotka. V oblasti tlačených dokumentov existuje medzinárodná norma ISO 216, ktorá definuje formáty papierov, rozdelených do sérií A, B a C. Pri tlači je definovaná hustota tlače. Medzi bežne používané hustoty tlače patria hodnoty 150, 300, 600 dpi, či dokonca menej používaných 1200 dpi.

V prípade zobrazovacích jednotiek je situácia komplikovanejšia. Existujú zariadenie s rôznou uhlopriečkou displeja a rôznym rozlíšením. V tejto súvislosti bol zavedení parametri PPI – Pixel Per Inch – počet obrazových bodov na jeden palec. Zobrazovacou jednotkou dnes môže byť inteligentný telefón, tablet, phtablet, tabletphone, konvertibilný laptop, roztrúsený laptop, AIO (All-In-One), čítačka elektronických kníh, inteligentné hodinky a podobne. Dôležitú úlohu pre kvalitu čítania (prezerania) tiež hrá pozorovacia vzdialenosť, ktorá úzko súvisí s rozlišovacou schopnosťou ľudského zraku. Hustotu obrazových bodov na palec môžeme vypočítať podla vzťahu.

Parameter PPI môžeme definovať podľa nasledujúceho vzorca

$$PPI = \sqrt{\frac{R_x^2 + R_y^2}{D}},$$

kde R_x je rozlíšenie v smere X, R_y je rozlíšenie v smere Y a D je uhlopriečka obrazovky u čítačky elektroních kníh. Podľa vzorca bol vypočítaný parameter PPI pre zvolené jednotlivé čítačky. Podobne by mohlo byť PPI vypočítane pre zobrazovacie jednotky mobilných telefónov, tabletov a podobne. Z vypočítaných parametrov vidíme, že PPI displejov čítačiek zaostáva za DPI tlačených dokumentov.

7.1. Čítačky elektronických kníh

Čítačky sú špeciálne zariadenia určené na čítanie elektronických kníh. Mnohé z nich obsahujú zobrazovaciu jednotku založenú na elektronickom papieri, ktorého zobrazovací princíp sa líši od klasického LCD displeja. V tabuľke 1 je urobený výber niektorých v súčasnosti na našom trhu dostupných čítačiek. V tabuľke 2 je súhrn podporovaných formátov jednotlivými čítačkami.

8. Ďalší softvér na prácu s dokumentami

Medzi systémy pre prácu s elektronickými dokumentami patria systémy pre elektronické vzdelávanie LMS. Dôležité pre výmenu elektronických kurzov sú štandardy. Medzi tieto štandardy patria IMS, AICC, SCORM a Experience API, ktoré je známe aj pod názvami xAPI, Tin Can API, Next Generation SCORM alebo SCORM2.

Tabuľka 1. Čítačky elektronických kníh

Názov	Uhlopriečka [inch]	Rozlíšenie	PPI
Amazon Kindle Paperwhite 2	6	1024 × 768	213,13
Nook Simple Touch GlowLight	6	800 × 600	166,67
Sony PRS-T2	6	800 × 600	166,67
PocketBook 626 Touch Lux 2	6	1024 × 758	212,34
PocketBook Color Lux	8	800 × 600, 4096 farieb	125
Sony Mobius	13,3	1200 × 1600, 16 stupňov sivej	150

Tabuľka 2. Čítačky elektronických kníh – podporované formáty

Názov	Formáty
Amazon Kindle Paperwhite 2	PDF, MOBI, AZW, PRC, TXT
Nook Simple Touch GlowLight	PDF, EPUB
Sony PRS-T2	EPUB, PDF, TXT
PocketBook 626 Touch Lux 2	PDF, MOBI, EPUB, DOC, FB2, HTML, DOCX, PRC, TXT, RTF
PocketBook Color Lux	PDF, EPUB, DOC, FB2, HTML, PRC, TXT, RTF
Sony Mobius	PDF

Dalšie sytémy sú systémy určené pre spoluprácu na dokumentoch umožnujúce udržiavanie verziovanie a revízie dokumentov. Pre čisté textové súbory sú to systémy git, cvs, svn. Spoluprácu na dokumentoch umožňuje aj textový procesor Abiword a online textový procesor Etherpad.

Dalšími systémami sú systémy na správu dokumentov a to napríklad Alfresco a OpenKM.

Záver

Cieľom tohto článku bolo priblížiť problematiku tvorby dokumentov hlavne pomocou open source a iných bezplatne dostupných nástrojov Obsahuje prehľad nástrojov a pomáha sa zorientovať v týchto nástrojoch a poukazuje na ne ako na alternatívy k súčasnému komerčnému softvéru. Tieto nástroje sú mnohokrát dostatočujúce a nie je potrebné používať drahý proprietárny softvér, ktorý síce obsahuje veľké množstvo funkcií. Funkcie zahrnuté v komerčnom softvéri sú väčšinou nepoužívané, alebo ich potrebujeme len zriedkavo a sme ich schopní suplovať pomocou iného nástroja.

Použitie otvoreného softvéru nám, okrem značnej finančnej úspory, prináša aj ďalšie benefity a možnosť využitia staršieho hardvéru. Ďalšou výhodou je otvorenosť a tým možnosť úpravy a jeho rozširovania.

Stačí sa naučiť ovládať pár jednoduchých príkazov a technológií a s pomerne malým úsilím dokážeme vytvoriť profesionálne vyzerajúce dokumenty s vysokou typografickou kvalitou. Pomocou rozširujúcich balíkov pre systém LaTeX dokážeme dokumenty rozšíriť napríklad o interaktívne prvky.

Otvorený softvér podporuje množstvo formátov, ďalšie formáty dokážeme získať pomocou použitia konverzných nástrojov.

Literatúra

[1] The Free Dictionary, `http://www.thefreedictionary.com/document`.
[2] Oxford Dictionaries, `http://www.oxforddictionaries.com/definition/english/document`.
[3] Calligra Author, `http://userbase.kde.org/Calligra_Author`.
[4] MEŠKO, D. et al.: *Akademická príručka*, Martin, Osveta 2005, ISBN 8080632006, str. 496.
[5] BERAN, V.: *Typografický manuál*, Náchod, MANUÁL 1994, ISBN 80-901824-0-2.
[6] RYBIČKA, J., ČAČKOVÁ, P., PŘICHYSTAL, J.: *Průvodce tvorbou dokumentů*, Bučovice, Nakladatelství Martin Stříž 2011, ISBN 978-80-87106-43-3.
[7] Sony to bring its 13.3", Mobius Touch E Ink Digital Paper to the US, `http://www.e-ink-info.com/sony-bring-its-133-mobius-touch-e-ink-digital-paper-us`.

Kontaktná adresa

Ing. Pavol Lajčiak, Katedra informatiky, Pedagogická fakulta, Katolícka univerzita v Ružomberku, Hrabovská cesta 1, 034 01 Ružomberok, Slovenská republika,
E-mailová adresa: `pavol.lajciak@ku.sk`, `http://skola.pavolmaria.org/`

6. ročník
Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf 2014

OSSConf 2014: 131–137

CODE GENERATION IN UML .FRI

JOZEF PAĽA (SK) AND MIROSLAV LOUMA (SK)

Abstract. The article's purpose is to explain principles of code generation inside the UML .FRI software developed as an open source CASE tool licensed under GNU/GPL license. Article emphasise benefits of code generation, possibilities of its use in various fields, techniques to achieve flexible code generation and possible code generation's benefits, which could positively impact educational system. Last part is dedicated to the future plans of further implementation of reverse code engineering inside UML .FRI.

Key words and phrases. Code generation, CASE tool, DSM, open source, XML.

Generovanie kódu v UML .FRI

Abstrakt. Cieľom príspevku je vysvetliť princípy generovania kódu v rámci softvéru UML .FRI vyvíjaného ako otvorený softvér a CASE nástroj licencovaný pod GNU/GPL licenciou. Príspevok zdôrazňuje výhody generovania kódu, možnosti jeho použitia v rôznych oblastiach, techniky na dosiahnutie flexibilného generovania kódu a možné výhody generovania kódu, ktoré by mohli pozitívne vplývať na systém vzdelávania. Posledná časť je venovaná budúcim plánom pre ďalšiu implementáciu týkajúcu sa reverzného inžinierstva v rámci UML .FRI.

Kľúčové slová. Generovanie kódu, CASE nástroj, DSM, otvorený softvér, XML.

Introduction

UML .FRI is a project ran by Faculty of Management Science and Informatics at the University of Žilina with direct participation of its students. This project is a part of the larger system of education through project work. Students are offered to take a part in one of the various projects during their master's degree studies, where they are led by a more experienced tutor to research and develop solutions in long-term and large-scale project. During regular lessons students usually work just on a small one-man term project strongly connected to the attended course and mostly without, or just with small connections to other courses they are taking during studies. However, during project work, they cooperate with other students on larger tasks. Students get necessary experience and skills for their future professional career such as working in a team, management and cooperation during working on larger tasks, how the development life cycle works, how working on a small system's component can affect system as a whole and much more. [1]

UML .FRI is a CASE tool that uses techniques of domain-specific modeling (DSM). CASE tools aim to make the software's development easier. Users can use

it to plan development of their projects and to model their application's design and behaviour through various types of currently supported modeling languages such as UML, DFD, ERD or BPMN. Furthermore, they can create their own diagram types using XML language to define meta-model required for that purpose. [1]

However, supporting only modeling is not what the modern CASE tools are about. They offer various other utilities to make the development easier and smoother. One example is code generation. After user finishes modeling of system, he has to implement it and it seems like starting from a scratch, even though the system was already modeled and described using previously created diagrams.

Developer has to do a lot of repetitive work, such as creating folder and file structure. All this mundane, repetitive work could be automated. And that is where code engineering steps in. Developers can simply generate source code and its structure based on the created model and therefore, they can focus on development itself rather than repetitive preparation of it. It also helps to maintain code consistency and its quality. [2]

1. Current state

Current stable version of UML .FRI does not support code generation at all. However, it was a part of UML .FRI in one of the very first versions. At that time, it was implemented directly into application's core. Core of the UML .FRI went through a massive rebuild since then and since students are part of the project for three terms only, the team member, who developed code generation algorithms, is not a part of the project anymore. The consequence is that the code generation is not usable with the newest version of the application's core and reimplementing of the algorithms was not priority until now.

1.1. Plug-in system

As explained above, the main problem with previous version was tight coupling between code generation and the application's core. Any change in the core affects the code generation algorithms and requires refactoring and reimplementation of said algorithms to reflect core's changes. A solution to this problem is usage of UML .FRI's API and plug-in system, which would make code generation independent of changes inside the application's core.

Plug-in system in UML .FRI is based on client server architecture. Plug-in acts as a client and communicates with application's core using messages exchanged through interface [1]. Therefore, the way how messages are being processed inside the application is not important and only response sent back to the plug-in matters. This response should stay the same regardless the core's version.

1.2. Implementation

UML .FRI is developed as DSM tool. One of the advantages of this approach is separation of metamodel from source code. This leads to greater flexibility and possibility to use the tool in various domains [3]. To follow those principles, it is necessary to make code generation flexible as well. Furthermore, it is necessary to offer easy-to-use way to define and use various templates to generate code (or different types of files, for example documentation).

It should be possible to generate source code from class diagram, as well as SQL commands from entity-relationship diagram using the same tool just with different generation templates. Therefore, it is possible for users to not only define their own metamodels and diagrams, but also the way how those diagrams will be transformed into output files.

In order to achieve this goal, similar principle currently implemented for metamodels is used. Code generation templates are defined as XML files with structure representing how the concrete elements, defined inside specific metamodel, should be transformed into the output file.

To make it easier to understand, following example could be used. Worker in a car factory receives order to make two cars, one red with full equipment and one yellow with standard equipment. Worker has manual on how to construct a car. It says how to build it in general (what parts are needed and how to put them together), however, it gives possibility to specify some parameters (for example colour or equipment).

To express this in UML .FRI internal structure, there is a model (worker's plan) with two elements called "car". Each element got their attributes (colour, equipment). Code generation receives this model and compares elements with an XML file. Algorithm searches XML file and looks for element with an ID "car". Found element defines process of car's transformation from the model into the output. Furthermore, it defines the way how to utilize the output based on attributes inside the model (car will have correct colour and equipment).

```
<Template diagram="Class Diagram" name="C++" type="Code">
    <Element id="Package">
            <Directory name="{name}" value="[_0-9a-zA-Z]+">
                    <Recursive>
                        <AllowElement id="All" />
                    </Recursive>
            </Directory>
    </Element>
</Template>
```

Example above shows concrete XML template for class diagram inside UML .FRI. Root element defines template and its attributes, such as diagram where the template can be used and name of the template (in this case it is "C++" which

means code generation algorithm will generate source code in C++ programming language from the given class diagram).

Inside the root element there are definitions of all the elements that can be processed by this template. In this case there is element "Package" only. Body of this element defines the way how it has to be processed. Element "Directory" explains that new folder has to be created with name defined as an attribute inside class diagram. Package inside class diagram might include another class diagram, therefore, there is element "Recursive" which ensures that if there are any elements inside the processed package defined in the class diagram, they will be processed.

The real template also includes definition for class element. However, this definition is more complex and it is too lengthy to be listed in its entirety. Example below shows picture of diagram with one class and generated code for that class:

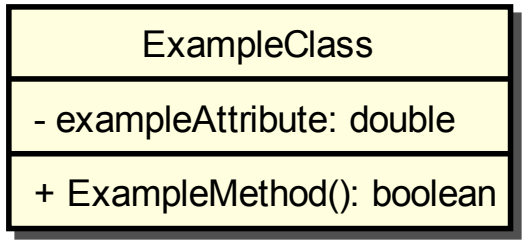


Figure 1. Example diagram

subsubsection*ExampleClass.h

```
class ExampleClass{
private:
    double exampleAttribute;
public:
    boolean ExampleMethod();
};
```

ExampleClass.cpp.

```
#include "ExampleClass.h"

boolean ExampleClass::ExampleMethod(){

}
```

To return to the example with car production, it is possible to use code generation and to define template for it. It could look like this:

```
<Template diagram="Car Production Plan Diagram" name="Car builder"
          type="Car construction">
    <Element id="Car">
```

```
            <Coachbuilder color="{color}" />
        <Interior equipment="{equipment}" />
    </Element>
</Template>
```

It is possible for the users to build their own custom templates for their own custom diagrams this way. However, one more thing is necessary to be done before this template for car production can be used. Elements “Coachbuilder” and “Interior” are similar to element “Directory” in previous example. They define what has to be done to generate parent element (element with the ID “Car” in this case). The definition is expressed using a virtual method "generate", which is overridden in element classes. Those elements use generalization and they all overrides parents’ method “generate”. This method is called when generation algorithm processes XML file. Therefore, it is necessary to define this “generate” method for elements “Coachbuilder” and “Interior”. Once this is done, those algorithms can be reused for example for sport cars, trucks or buses.

2. Usage in education

Students of Faculty of Management Science and Informatics at the University of Žilina are being taught about advantages of software modeling since their first courses. However, while working on small projects, significance of the modeling might not be evident enough. Code generation might help student to see advantages of this process. As explained at the beginning of the article, code generation helps to automate repetitive work and it is very useful technique for maintaining quality code [2]. Students might understand benefits of modeling better if they are able to experience it themselves. They are able to simply focus on learning proper and effective programing techniques instead of spending their time with writing basic class structure.

Furthermore, this might help during other courses which are less connected to the coding. As explained before, generation was implemented with principles of universality. Therefore, teachers can use UML .FRI to define their own metamodels. Students can use them to create diagrams, which will help them to analyze and understand topic they are currently studying. After that they are able to get defined output and use it for further studies with help of generation algorithms.

Learning of how the SQL databases work is a good example. Teacher is able to prepare metamodel for entity-relationship diagram. Then students use it to define table structure for their projects and they are able to use code generation to generate SQL commands to build up and initialize table structure inside an SQL server. There is little benefit for students to do this manually. Instead, they are able to focus on working with data itself. Similar examples could be applied for other courses.

3. Comparison with the other open source tools

There are various UML tools available on the Internet. This comparison is focused on open source products, which is the field where UML .FRI belongs as well. Code generation is a part of all compared products. This fact just underlines importance of decision to implement this functionality into UML .FRI, so it can compete with other similar products.

First example of the other UML tool is Umbrello, which is available for various Linux distributions as well as Windows as a part of "KDE for Windows" installation. This tool offers generation of code for various programming languages. The environment is easy to use. However, generated code includes unnecessary lines. For example, if there is class defined with one public method, generated C++ files still includes placeholders for private, protected and public attributes and methods even if there is none of them defined in diagram.

Next open source UML tool compared is ArgoUML. This tool is available for all major platforms (Windows, Linux and Mac OS). Great advantage is possibility to run the tool within Java Web Start. This allows using of this tool on all platforms that Java supports. When it comes to code generation, the tool offers less languages compared to Umbrello. On the other hand, it is possible to generate one class in multiple languages at the same time. Disadvantage of this tool is the way how it processes packages. If package includes another class diagram with more elements, those are ignored. Therefore, user has to generate packages manually one by one. Tool offers option to generate code for a whole project, unfortunately it did not seem to work during testing.

Last tested software is Modelio, which is also available for all major platforms similarly to ArgoUML. Graphical user interface of this tool gives the most modern impression of the the compared tools. This tool, similarly to UML .FRI, offers extending of functionality through downloadable modules. However, some of them are not open source and those are usually paid as well. Unfortunately, this includes also code generation modules. Company developing this tool is offering various paid modules, each for one different language. This is main disadvantage compared to UML .FRI and other tools, which offer this functionality for free and as a part of their open source code policy.

To summarize this comparison, code generation is clearly important part of the UML tools. UML .FRI has currently XML templates for three languages only (C++, Delphi and Python), which is lower number compared to other tools. On the other hand, it is possible to define those templates easily and even users alone should be able to do it. This allows creation of generation templates not only for different programming languages, but also for different types of diagrams. All the tools mentioned above were limited to generate code only from limited number of languages and only from UML diagrams. Therefore, universality can be considered as the main advantage of the UML .FRI.

3.1. Where to download the tools

All the tools mentioned are available online through the following websites:

- UML .FRI – `http://umlfri.org/`.
- ArgoUML – `http://argouml.tigris.org/`.
- Umbrello – `http://umbrello.kde.org/`.
- Modelio – `http://www.modelio.org/`.

4. Future plans

Modern case tools offer not only code generation, but also reverse engineering, process of analysing source code to generate appropriate diagrams. Users are able to generate diagrams out of the source code using this feature. There are various reasons for using it, for example, developers are able to see if the result follows the structure defined in previously created diagram. They are able to find out what has changed during development. Therefore, reverse code engineering helps to keep the diagrams up to date.

UML .FRI should offer this possibility as well while using the same principles as code generation does. The same XML template could be used to generate diagrams out of the source code, however, there are several issues, which need to be solved, before this feature is fully functional.

References

[1] PAĽA, J., LOUMA, M., ŠUŠUK, R., BELÁS, M., ŠANDOR, E., LOKAJ, M.: *UML .FRI – Latest Development Changes and Future Plans*, Journal of Information, Management and Control Systems, Vol. 12, No. 1, 2014, `http://kifri.fri.uniza.sk/ojs/index.php/JICMS/article/download/1560/717`.

[2] FERTALJ, K., BRCIC, M.: *A Source Code Generator Based on UML Specifications*, International Journal of Computers and Communications, Issue 1, Volume 2, 2008, `http://shonen.naun.org/multimedia/UPress/cc/cc-23.pdf`.

[3] JANECH, J.: *UML/DSM nástroj UML .FRI*, Zborník príspevkov medzinárodnej konferencie OSSConf, 2009, ISBN 978-80-89276-16-5, `http://ossconf.soit.sk/images/zborniky/zbornik2009.pdf`.

Contact addresses

Bc. BEng. Jozef Paľa, Žilinská univerzita, Slovenská republika,
E-mail address: `bax007@gmail.com`

Bc. Miroslav Louma, Žilinská univerzita, Slovenská republika,
E-mail address: `louma@st.fri.uniza.sk`

6. ročník Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf
OSSConf 2014: 139–150

LaTeX & TABULKY & TIPY & TRIKY \\

JIŘÍ RYBIČKA (CZ)

Abstrakt. Sazba tabulek představuje zejména v odborných textech velmi frekventovanou součást. Základní přístup ke tvorbě tabulek se významně liší, použijeme-li majoritně používaný systém pro zpracování textů (např. Open Office Writer) oproti systémům pracujícím na principu TeX. Většina uživatelů pohlíží na tabulku jako na grafický objekt, proto jim plně vyhovuje interaktivní přístup, jehož základní nevýhodou je nízká nebo žádná automatizovatelnost a problematická typografická kvalita. Dávkový přístup implementovaný v systému LaTeX naproti tomu komplikuje vytvoření požadovaného tvaru tabulky. Článek se zabývá některými možnostmi usnadnění tvorby tabulek při zachování nebo rozšíření automatizovatelnosti a současného dosažení požadované typografické kvality výsledku.

Klíčová slova. Tabulka v LaTeXu, `tabular`, `tabbing`, `parbox`, `catcode`.

LaTeX & TABLES & TIPS & TRICKS \\

Abstract. Typesetting of tables is frequently used component of texts especifically scientific articles and books. The basic approach to creating tables differ significantly if we use majority systems for text processing (eg. Open Office Writer) and systems operating on the principle of TeX. Most users considers the table as a graphical object, so they fully complies with interactive approach whose main drawback is low or no automatizing and problematic typographic quality. Batch approach implemented in a system LaTeX on the other hand is complicated to create the desired shape of the table. Article deals with some tips to facilitate the creation of tables while achieving the automatical processing and the desired typographical quality of the result.

Keywords. LaTeX table, `tabular`, `tabbing`, `parbox`, `catcode`.

1. Požadavky na sazbu tabulek

Sazba tabulek představuje zejména v odborných textech zcela nezbytnou součást konstrukce dokumentu. Stejně jako u jiných prvků jsou i na tabulky kladeny určité požadavky.

Tyto požadavky můžeme rozdělit na typografické (to by mělo být prvotní hledisko) a technické. Dostatečně podrobně tuto problematiku rozebírá ve své diplomové práci Talandová (2006). Na stranách 17–22 uvádí přehled typografických zásad a pravidel, požadavky na technické zpracování shrnuje na stranách 23–32. Základním výsledkem těchto dvou částí je konstatování, že typografické systémy tabulky samozřejmě podporují, ale nejsou obvykle vedeny typografickými zásadami; rovněž přístupy k řešení tabulek jsou velmi rozdílné a dosažení téhož efektu si může vyžádat diametrálně odlišnou pracnost.

Správně typograficky upravená tabulka je především přehledná, vyvážená, má správně zarovnané údaje a čtenáři přináší maximální užitek. Dosažení tohoto cíle vyžaduje obvykle kombinaci mnoha prvků a určitou volnost při jejich využívání.

Od doby kovové sazby se výrazně rozšířily možnosti sazby tabulek. S rozvojem počítačové grafiky se doména tabulkových nástrojů postupně přesunula do této oblasti, čímž se zcela uvolnily možnosti využívání dříve zcela nedostupných prvků – linek nejrůznějších typů, tlouštěk a barev, používání podbarvených polí, seskupování tabulkových polí do libovolných skupin, manipulace s interním prostorem a materiálem v polích atd.

Samozřejmě že ne všechny tyto nástroje jsou využívány k dosažení typograficky optimálního výsledku. To však není obvykle není vnímáno jako nedostatek, zatímco určitá komplikovanost nástrojů v některých systémech bývá značnou překážkou pro uživatele.

1.1. Malovat, nebo programovat?

Základním přístupem, který je uživateli silně preferován, je interaktivní (grafický) návrh tabulky – malování, zahrnující téměř neomezené možnosti volby způsobu ohraničení a podbarvení, manipulace se šířkami a výškami polí, vizuální značkování obsahu polí a snadné přecházení mezi jednotlivými poli. Je pochopitelné, že jednoduchost tohoto přístupu, neustálá vizuální kontrola výsledku a přítomnost řady nástrojů příliš nedávají šanci systémům postaveným na principu TEX – tedy programování, jejichž přístup vyžaduje neinteraktivní práci, vkládání příkazů a relativně komplikované řešení situací, které v interaktivní verzi nepřinášejí prakticky žádné problémy.

Na obhajobu neinteraktivních systémů lze však říct, že zdaleka ne vždycky je jejich použití nevýhodné. Naopak, existuje řada případů, kdy vhodným využitím bohatého aparátu tvorby maker dosáhneme vytčeného cíle rychleji, s minimálním úsilím při případných úpravách vzhledu a se zajímavými možnostmi integrace tabulek do okolního materiálu dokumentu. Nezanedbatelná není ani typografická kvalita, v některých situacích má dokonce neinteraktivní systém nástroje, které nemají v interaktivních protějšcích žádnou obdobu.

Základní manipulace s předdefinovanými příkazy lze zjistit z mnoha zdrojů a učebnic, jednou z možností je učebnice Rybičky (2003). V těchto textech jsou však uvedeny do značné míry jen základní rekvizity, jejichž použití v mnoha běžných dokumentech s sebou nese největší nevýhodu neinteraktivních systémů – těžkopádný a nepřehledný zápis.

Podívejme se tedy na několik drobných tipů, které nám mohou pomoci v neinteraktivním přístupu tvorby tabulek v systému LATEX a jeho následnících. Tipy vycházejí z dlouholetých a osvědčených zkušeností s dokumenty různých druhů.

2. Některé možnosti řešení

Nerad používám cizí makra.

Petr Olšák

Nerad znovu vynalézám kolo.

Petr Sojka

Uvedené citáty možná nejsou zcela autentické, ale vyjadřují dvojí diametrálně odlišný přístup k řešení problémů – buď si všechno vyrobím sám, umím to velmi dobře a mám nad celým řešením plnou kontrolu, nebo využiji již existujících balíčků, které ovšem mohou mít netušené vnitřní vazby a skryté zrady a často je budu muset přizpůsobit konkrétní situaci.

Praktické případy však pravděpodobně leží někde mezi těmito krajními mezemi. Je vhodné vědět, že podobný problém už někdo řešil (a možná dobře vyřešil), na druhé straně místo dlouhého hledání vhodného balíčku je daleko efektivnější napsat dvě nebo tři vhodná makra a danou konkrétní situaci tím vypořádat. Nebude to třeba řešení obecné, ale o to mnohdy vůbec ani nejde.

V každém případě přístup neinteraktivní, tedy „programátorský“, má svou největší výhodu v automatizovatelnosti. Vhodnou cestou je prakticky vždy vytvoření nějakých vlastních příkazů, které zefektivní zápis zdrojového textu a umožní také často velmi výrazně usnadnit jakékoliv další změny a zásahy. Nebraňme se tedy vlastním příkazům – v opačném případě je vhodnější zůstat u malování (a také asi u jiného systému).

2.1. Obejdeme se bez `tabular`?

Když se řekne „tabulka v LaTeXu“, prakticky každému se ihned vybaví nejfrekventovanější nástroj – prostředí `tabular`. Je však vždy nutné tento do značné míry komplikovaný přístup použít?

Pro sazbu tzv. otevřených tabulek (v nich stačí obvykle pouze dvě vodorovné linky, jinak nevyžadují žádné orámování) můžeme použít prostředí `tabbing`, v jednoduchých případech vystačíme pouze s boxy, u nichž stanovíme vhodnou šířku. Příklad – pár údajů o teplotách naměřených dnes v 7 hodin ráno. Zdrojový text může mít tento tvar:

```
\def\radek#1;#2.{\noindent\makebox[50mm][l]{#1}%  1. sloupec
                 \makebox[10mm][r]{#2}\par}    %  2. sloupec

{\bfseries \radek Město; Teplota [$^\circ$C].}
\smallskip\hrule width 60mm\smallskip
  \radek Ostrava; 12,8.
  \radek Olomouc; 14,2.
  \radek Bruntál; 9,8.
```

```
  \radek Brno; 13,3.
  \radek Trenčín; 15,5.
  \radek Žilina; 14,3.
  \radek Banská Bystrica; 11,6.
\smallskip\hrule width 60mm
```

Po vysázení dostaneme:

Město	**Teplota [°C]**
Ostrava	12,8
Olomouc	14,2
Bruntál	9,8
Brno	13,3
Trenčín	15,5
Žilina	14,3
Banská Bystrica	11,6

2.2. Zalamování řádků v buňkách

Zcela přirozeným požadavkem je, aby se materiál v buňce tabulky mohl lámat na řádky. To ovšem ani v prostředí `tabbing`, ani v prostředí `tabular` běžně možné není. Nepátrejme nad příčinami tohoto na první pohled málo pochopitelného stavu a pojďme se podívat, jak tuto nevýhodu obejít. Budeme se zabývat prostředím `tabular`.

Obsahem každého tabulkového pole je tzv. LR-box, tedy box v omezeném horizontálním režimu. Jeho součástí je nejčastěji jednořádkový text. Potřebujeme-li více řádků, musíme si pomoci nějakým trikem.

Nejjednodušší možností je použít uvnitř pole další prostředí `tabular`, uvnitř něho pak můžeme vkládat více řádků. Toto vnořené pole však musíme poněkud ošetřit, aby výsledek splňoval alespoň základní vzhledové požadavky.

Vložená tabulka je jednosloupcová, ale potřebujeme u ní odstranit mezisloupcovou mezeru na levé i pravé straně, aby se tato mezera nesčítala s mezisloupcovou mezerou pole, do něhož tabulku vkládáme. Druhá úprava spočívá v úpravě řádkování vnořené tabulky – dost často potřebujeme, aby řádky byly o něco hustší než řádky okolní tabulky, protože implicitní světlo v tabulkových polích je příliš malé a potřebujeme je roztáhnout. To ovšem neplatí pro vnořené tabulky.

```
% víceřádkové tabulkové pole:
\def\pole#1#2{{\def\arraystretch{1}%
    \begin{tabular}{@{}#1@{}}#2\end{tabular}}}

% tabulkové pole se zarovnáním na horní okraj:
\def\polet#1#2{{\def\arraystretch{1}%
    \begin{tabular}[t]{@{}#1@{}}#2\end{tabular}}}
```

```
\def\arraystretch{1.3}% běžné světlo v tabulkách

% příklad tabulky:
\begin{tabular}{|l|c|} \hline
\bfseries Město & \bfseries\pole c{Teplota\\{}[$^\circ$C]}
                                  \\\hline\hline
Ostrava                     & 10,8 \\\hline
Olomouc                     &  9,2 \\\hline
Žilina                      & 11,3 \\\hline
\pole l{Banská\\ Bystrica} & 11,6 \\\hline
\end{tabular}
```

Po vysázení dostaneme (v první tabulce je makro `\pole`, ve druhé je použito makro `\polet`):

Město	**Teplota [°C]**
Ostrava	10,8
Olomouc	9,2
Žilina	11,3
Banská Bystrica	11,6

Město	**Teplota [°C]**
Ostrava	10,8
Olomouc	9,2
Žilina	11,3
Banská Bystrica	11,6

Všimněte si, jak je v těchto tabulkách rozdílný prostor nad vnořeným polem. V prvním případě je mezera menší (neuplatní se zde nastavení `\arraystretch`), zatímco ve druhém případě se vlivem zarovnání na horní řádek víceřádkového pole uplatní nastavená řádková mezera.

Nepřípustné zarovnání čísel ve druhém sloupci vyřešíme v následující části.

2.3. Jak na číselná data

Požadavek zarovnání čísel tak, aby stejné řády byly pod sebou, je zcela nesporný. Máme-li prezentovat data v tabulce, děláme to právě proto, aby čtenář **na první pohled** viděl, jaké jsou mezi číselnými hodnotami rozdíly.

Podíváme-li se kolem sebe na nejrůznější dokumenty pocházející z nejrůznějších systémů, můžeme konstatovat, že se v tomto směru objevuje masivní množství chyb. Jedna z nejčastějších je demonstrována na předchozím příkladu: sloupec je zarovnán na střed, ale čísla nemají stejné šířky, takže různé řády plavou různě

pod sebou. Tím se přehled o velikostech jednotlivých hodnot stírá, čímž je vlastně popírán hlavní důvod, proč se data do tabulek umisťují.

Řešením jistě je sloupec upravit na pravý okraj. Vznikne tím tvar, který jsme uvedli v prvním příkladu jednoduché tabulky: Popisek sloupce je zpravidla podstatně delší než číselné údaje, vzniká tím tedy esteticky nepřípustný zející volný prostor s daty namačkanými na pravé čáře. Správně by data měla splňovat dvě podmínky současně: měla by být na střed, ale stejnými řády pod sebou.

Toho lze docílit mimo jiné tím, že čísla budou mít stejný počet číslic a všech ostatních prvků (desetinných čárek, znamének apod.). Prvky, které absentují, tedy doplníme vhodnými mezerami. Postupovat však nebudeme tak, jak to dělají zcela laičtí uživatelé v interaktivních systémech, že budeme mlátit do mezerníku tak dlouho, až budeme na obrazovce zhruba spokojeni a pak se raději ani nebudeme dívat na tištěný výstup. Mezery dodáme ve zcela přesných velikostech.

V některých systémech se za tímto účelem vyskytuje tzv. figurální mezera – tedy mezera o šířce číslice. Číslice, které lze použít v tabulkách, mají totiž identické šířky – jsou to tzv. neproporcionální číslice. Kde v LaTeXu máme figurální mezeru? Lze jí docílit například příkazem `\hphantom{0}`.

Ano, nic složitého, ale psát do předchozí tabulky u „kratšího“ čísla (v případě složitějších tabulek do každého řádku) text `\hphantom{0}9,2` je přinejmenším těžkopádné. Už i tak málo přehledný zápis tabulky se tímto „instrumentem“ pěkně zamíchá. Vhodným řešením je využít k tomuto účelu jednoznakový příkaz (aktivní znak), který umožní i snadnou optickou kontrolu, pokud si zápis tabulky vhodně zformátujeme v editoru. Definujme tedy figurální mezeru jako například znak vykřičník, který předtím učiníme aktivním znakem:

```
\catcode`\!=\active
\def!{\hphantom{0}}
```

Tabulku upravíme tak, aby ve zdrojovém textu bylo snadno vizuálně kontrolovatelné, že všechna čísla jsou stejně široká:

```
\begin{tabular}{|l|c|c|} \hline
\bfseries Příčina ukončení testu &
  \bfseries \pole c{Absolutní \\ četnost} &
  \bfseries \pole c{Relativní\\četnost [\%]}      \\\hline
Opisování z taháku               & 180    & !25,0 \\\hline
Použití mobilního telefonu       & !45    & !!6,3 \\\hline
Vyžádání pomoci od spolužáka     & 405    & !56,2 \\\hline
Vložení řešení do kalkulačky     & !90    & !12,5 \\\hline
\bfseries Celkem $N$             & 720    & 100,0 \\\hline
\end{tabular}
```

Po vysázení dostaneme:

Příčina ukončení testu	**Absolutní četnost**	**Relativní četnost [%]**
Opisování z taháku	180	25,0
Použití mobilního telefonu	45	6,3
Vyžádání pomoci od spolužáka	405	56,2
Vložení řešení do kalkulačky	90	12,5
Celkem N	720	100,0

Mírně odlišný problém je použití znaménka minus, jehož výskyt nesmí narušit zarovnání na stejné řády. K tomu účelu si můžeme vytvořit jednoduché makro pro box s nulovou šířkou:

```
\def\mm{\makebox[0pt][r]{$-$}}
```

a použít je v následujícím příkladu:

```
\begin{tabular}{|l|c|} \hline
\bfseries Hodnocené příčiny   &
  \bfseries \pole c{Relativní\\četnost [\%]} \\\hline
Vliv speciálního předmětu     & 25,0          \\\hline
Negativní zkušenosti ze SŠ    & !\mm6,3       \\\hline
Penalizace u hodnocení prací  & 56,2          \\\hline
Tendence usnadnit si práci    & \mm12,5       \\\hline
\bfseries Celkový výsledek    & 62,4          \\\hline
\end{tabular}
```

Po vysázení dostáváme:

Hodnocené příčiny	**Relativní četnost [%]**
Vliv speciálního předmětu	25,0
Negativní zkušenosti ze SŠ	−6,3
Penalizace u hodnocení prací	56,2
Tendence usnadnit si práci	−12,5
Celkový výsledek	62,4

2.4. Poznámky v tabulce

Navážeme na předchozí rekvizity podobným řešením dalších přídavných symbolů, které se vyskytují u prezentovaných dat. Jedná se nejčastěji o různé poznámkové

symboly – hvězdičky, mečíky, písmena. Samotná poznámka by pak měla být sázena jako součást tabulky a měla by mít označení, které se nemíchá s běžnými poznámkami pod čarou. Často se odkazy na tyto lokální poznámky řeší nečíselně, aby nemohlo dojít k záměně s údaji v tabulce.

Základní prvek je podobně jako v případě znaménka minus box s nulovou šířkou, tentokrát se zarovnáním doleva:

```
\def\hve{\makebox[0pt][l]{*}}
```

Analogicky definujeme podobná makra pro případné další značky.

Lokální tabulkovou poznámku lze řešit jako zvláštní řádek tabulky s odpovídajícím zarovnáním přes všechny její sloupce. K tomu si vyrobíme makro:

```
\def\tabpozn#1#2#3{\multicolumn{#1}{@{}l@{}}{%
      \parbox[t]{#2}{\footnotesize\raggedright #3}}}
```

a použijeme je v následujícím příkladu:

```
\begin{tabular}{|l|c|} \hline
\bfseries Hodnocené hypotézy           &
  \bfseries Hodnota kritéria                        \\\hline
H$_0$ Podvádění je naučené             & 143,8\hve  \\\hline
H$_0$ Plagiátorství lze snížit výukou  & !74,4\hhve \\\hline
H$_0$ Penalizace je dostatečná         & !!7,9\ha   \\\hline
H$_0$ Pomůže softwarová detekce        & !13,4\hb   \\\hline
\tabpozn{2}{90mm}{*\,pro $p<0{,}05$; **\,pro $p<0{,}01$;\\
  $^a$\,hypotéza nebyla zamítnuta;\\ $^b$\,hypotéza byla zamítnuta.}
\end{tabular}
```

Po vysázení dostaneme tvar:

Hodnocené hypotézy	**Hodnota kritéria**
H_0 Podvádění je naučené	143,8*
H_0 Plagiátorství lze snížit výukou	74,4**
H_0 Penalizace je dostatečná	$7{,}9^a$
H_0 Pomůže softwarová detekce	$13{,}4^b$

* pro $p < 0{,}05$; ** pro $p < 0{,}01$;
[a] hypotéza nebyla zamítnuta;
[b] hypotéza byla zamítnuta.

2.5. Odstavcové buňky

Dalším frekventovaným požadavkem je umístění souvislého textu do tabulkového pole. Tento stav je obvykle v interaktivních systémech implicitní. Sazba odstavcového textu do tabulkových polí má však své zvláštnosti. Obvykle se jedná o sazbu do užšího rozměru, je tedy nezbytné, aby byla zarovnána na prapor, někdy na střed. Prostředí `tabular` však nabízí jako jedinou možnost definovat sloupec se

zarovnáním do bloku. Vypomůžeme si však jednoduchým makrem, které nám v řadě případů velmi usnadní práci – tabulkovým odstavcem řešeným pomocí parboxu:

```
\def\strdn{\vrule depth .5em width 0pt}
\def\txpar#1#2{\parbox[t]{#1}{\raggedright #2\strdn}}
```

Vertikální čára nulové šířky vkládaná na konec odstavcového materiálu zde zastupuje činnost řádkování tabulky definované pomocí `\arraystretch`, protože to zde nefunguje. Lokálními redefinicemi makra `\strdn` můžeme snadno ovlivňovat situaci v jednotlivých tabulkách dokumentu. Podobně lze řešit i prostor v horní části pole. Samotné makro `\txpar` je poněkud těžkopádné. Na jeho základě si však můžeme v každé tabulce definovat vhodné následníky, jak je ilustrováno na následujícím příkladu. Makro `\tx` představuje materiál, který je ve všech čtyřech sloupcích tabulky stejně široký.

```
\def\tx#1{\txpar{0.2\textwidth}{#1}}

\begin{tabular}{|l|l|l|l|}\hline
\tx{\bfseries Humánní etologie}      &
   \tx{\bfseries Výzkumy komunikace} &
   \tx{\bfseries Psychologie}        &
   \tx{\bfseries Pedagogika}         \\\hline
\tx{{\bfseries Trevarthen:}  \\
 Teorie intersubjektivity    \\
 Reciprocita                 \\
 Protokonverzace             \\
 Vrozené motivace ke komunikaci}     &
\tx{{\bfseries Jakobson:}    \\
 Lingvistika                 \\
 Rituály kontaktu            \\[\medskipamount]
 {\bfseries Watzlawick:}     \\
 Pragmatika komunikace       \\[\medskipamount]
 Videofeedback}                      &
\tx{{\bfseries Bowlby:}      \\
 Teorie citové vazby         \\[\medskipamount]
 {\bfseries Vygotsky, Stern:}\\
 Vývojové teorie             \\
 Atribuční teorie}                   &
\tx{{\bfseries Tausch / Tausch, Rogers:}\\
 Humanistická pedagogika     \\[\medskipamount]
 {\bfseries Bruner:}         \\
 Kulturní teorie učení}              \\\hline
\end{tabular}
```

Po vysázení dostaneme tento výsledek:

Humánní etologie	**Výzkumy komunikace**	**Psychologie**	**Pedagogika**
Trevarthen: Teorie intersubjektivity Reciprocita Protokonverzace Vrozené motivace ke komunikaci	**Jakobson:** Lingvistika Rituály kontaktu **Watzlawick:** Pragmatika komunikace Videofeedback	**Bowlby:** Teorie citové vazby **Vygotsky, Stern:** Vývojové teorie Atribuční teorie	**Tausch / Tausch, Rogers:** Humanistická pedagogika **Bruner:** Kulturní teorie učení

Zde je „nastřelena“ šířka sloupce na hodnotu `0.2\textwidth`, ale můžeme postupovat i daleko promyšleněji. Chceme (a to je také dost častý požadavek), aby tabulka byla široká přesně jako šíře sazby. Necháme si tedy odpovídající šířku spočítat například takto:

```
\newlength{\lntxpar}
\lntxpar=\textwidth              % vyjdeme ze šířky sazby
\advance\lntxpar by -8\tabcolsep % odečteme mezisloupcové mezery
\divide\lntxpar by 4             % vydělíme počtem sloupců
\def\tx#1{\txpar{\lntxpar}{#1}}
```

Stejnou tabulku pak dostaneme v následujícím provedení:

Humánní etologie	**Výzkumy komunikace**	**Psychologie**	**Pedagogika**
Trevarthen: Teorie intersubjektivity Reciprocita Protokonverzace Vrozené motivace ke komunikaci	**Jakobson:** Lingvistika Rituály kontaktu **Watzlawick:** Pragmatika komunikace Videofeedback	**Bowlby:** Teorie citové vazby **Vygotsky, Stern:** Vývojové teorie Atribuční teorie	**Tausch / Tausch, Rogers:** Humanistická pedagogika **Bruner:** Kulturní teorie učení

2.6. Umístění tabulky do dokumentu

Tabulka jako celek není pevnou součástí textu. Protože by měla být přehledná, je nevýhodné, pokud systém dovolí její běžný zlom na koncích stránek. Každá tabulka také musí mít popisek systematicky umisťovaný buď nad tabulku, nebo pod tabulku (v celém textu však jednotně). Tyto popisky se často přenášejí i do seznamů tabulek.

Pro umístění tabulky do dokumentu lze s výhodou využívat plovoucího prostředí `table`. V základním tvaru tohoto prostředí je k dispozici i příkaz `\caption` definující popisek tabulky. Implicitní tvar obojího nám však nemusí vyhovovat a navíc je vhodné mít možnost všechny tabulky dokumentu centrálně ovládat (nastavit vhodné písmo, mezery apod.).

Všechny tyto důvody vedou k vytvoření vlastního makra pro vkládání tabulek. Jeho zjednodušený tvar může být například následující:

```
% fonty:
\def\tabpopisfnt{\small\sffamily\itshape}
\def\tabulfnt{\small\sffamily}
% popisek:
\def\mujpopisek#1{\refstepcounter{table}
    \parbox{\textwidth}{\raggedright \tabpopisfnt #1}}
% samotná tabulka:
\long\def\tabulka#1#2\endtab{\begin{table}[htb] %plovoucí
        \centering                    % materiál na střed
        \tabulfnt                     % písmo pro tabulky
        #2                            % umístění materiálu tabulky
        \par\medskip\mujpopisek{#1}   % popisek pod tabulkou
        \end{table}}
```

Použití makra demonstruje následující příklad, v němž využijeme řadu uvedených rekvizit:

```
\tabulka{Plánování podnikání a~hodnocení přípravy na vysoké škole}
\newsavebox{\pombox}
\lntxpar=\textwidth
\sbox{\pombox}{Celkem}
\advance\lntxpar by -\wd\pombox
\advance\lntxpar by -8\tabcolsep
\divide\lntxpar by 3
\def\tx#1{\txpar{\lntxpar}{#1}}
\begin{tabular}{|l|c|c|c|} \hline
  & \tx{\bfseries Uvažujete o~tom, že byste ... podnikali?}
  & \tx{\bfseries Myslíte si, že Vás... jiné i~na to, ... }
  & \tx{\bfseries Máte na své fakultě ... předmětů?}        \\\hline
 Ano    & !3 !!(7,3\,\%) & !7 !(17,1\,\%) & !38 !(92,7\,\%) \\\hline
 Ne     & 27 !(65,9\,\%) & !4 !!(9,8\,\%) & !!0 !!(0,0\,\%) \\\hline
 Nevím  & 11 !(26,8\,\%) & 30 !(73,2\,\%) & !!3 !!(7,3\,\%) \\\hline
 Celkem & 41 (100,0\,\%) & 41 (100,0\,\%) & !41 (100,0\,\%) \\\hline
\end{tabular}
\endtab
```

Po vysázení dostáváme tabulku č. 1, ve které je změněno písmo a popisek tabulky je umístěn pod ní.

	Uvažujete o tom, že byste v budoucnu (horizont 4 let) podnikali?	**Myslíte si, že Vás fakulta dokáže připravit mimo jiné i na to, abyste v budoucnu sami začali podnikat?**	**Máte na své fakultě možnost studia ekonomických předmětů?**
Ano	3 (7,3 %)	7 (17,1 %)	38 (92,7 %)
Ne	27 (65,9 %)	4 (9,8 %)	0 (0,0 %)
Nevím	11 (26,8 %)	30 (73,2 %)	3 (7,3 %)
Celkem	41 (100,0 %)	41 (100,0 %)	41 (100,0 %)

Tab. 1: Plánování podnikání a hodnocení přípravy na vysoké škole

3. Závěr

Praxe přináší stále nové a nové náměty, které je potřeba řešit. Podobným způsobem bychom tedy mohli pokračovat ještě dále – samostatnou kapitolou jsou barvy v tabulce (ohraničující čáry i tabulková pole) , řadu problémů je potřeba řešit, když se materiál tabulky nevejde do stanoveného prostoru atd.

Hlavním cílem článku však není předkládat množství hotových řešení – důležitý je zejména postup: pokud možno jednoduchými prostředky dosáhnout požadovaného efektu, byť se jedná o zcela lokální řešení, jehož obecná podoba v nějakém balíčku by byla nesrovnatelně náročnější a koneckonců i pro uživatele těžkopádnější. Ukazuje se, že cesta, kterou hodně prosazuje P. Olšák například svým projektem OPMac (Olšák, 2014), je schůdná i pro méně zdatné uživatele TEXu a jemu příbuzných systémů. Investice do naučení tvorby maker se podle praktických zkušeností určitě bohatě vyplatí.

Reference

[1] OLŠÁK, P.: *OPmac – rozšiřující makra plainTEXu*, 2014, `https://math.feld.cvut.cz/ftp/olsak/opmac/opmac-u.pdf`.
[2] RYBIČKA, J.: *LATEX pro začátečníky*, Brno, KONVOJ 2003, ISBN 80-7302-049-1.
[3] TALANDOVÁ, P.: *Přístupy ve zpracování tabulek v systémech DTP*, diplomová práce, MZLU v Brně, 2006.

Kontaktní adresa

doc. Ing. Jiří Rybička, Dr., Ústav informatiky, Provozně ekonomická fakulta, Mendelova univerzita v Brně, Zemědělská 1, 613 00 Brno, Česká republika,
E-mailová adresa: `rybicka@mendelu.cz`, `http://akela.mendelu.cz/~rybicka`

6. ročník Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf

OSSConf 2014: 151–158

THE HONEYPOTS AS NETWORK FORENSICS TOOLS

PAVOL SOKOL (SK)

Abstract. Honeypots are not only common tools for network security, but also for network forensics. This paper offers a discussion about honeypots as network forensics tools. Using of generic model of network forensics process we focus on the usage of the honeypots within this process. In paper we outline the issues related to usage of honeypots in individual phase of network forensics, especially in phase of detection, collection, investigation and presentation.

Key words and phrases. honeypot, network forensics, forensic tool, digital evidence.

HONEYPOTY AKO NÁSTROJE SIEŤOVEJ FORENZNEJ ANALÝZY

Abstrakt. Honeypoty sú nielen všeobecnými nástrojmi pre sieťovú bezpečnosť, ale aj nástrojmi vhodnými pre sieťovú forenznú analýzu. Tento článok ponúka diskusiu o honeypotoch ako forenzných nástrojoch. Použitím všeobecného modelu pre proces sieťovej forenznej analýzy, sa zameriavame na použitie honeypotov v rámci tohto procesu. V článku načrtávame problémy súvisiace s použitím honeypotov v jednotlivých fázach sieťovej forenznej analýzy, najmä vo fáze detekcie, zberu, vyšetrovania a prezentácie.

Kľúčové slová. honeypot, sieťová forenzná analýza, forenzný nástroj, digitálna evidencia.

Introduction

Due to rapid growth of information and transfer messages, network security has become an increasingly important part of modern society. Traditionally, information security is primarily defensive and it uses tools such as firewalls and intrusion detection systems to protect the information. In case the protection of the information system is broken and the attacker controls the system, the security strategy, tools and methods are changed. In this case, it is necessary to use the tools and methods of the network forensics.

Palmer defines **network forensics** as "the use of scientifically proven techniques to collect, fuse, identify, examine, correlate, analyse, and document digital evidence from multiple, actively processing and transmitting digital sources for the purpose of uncovering facts related to the planned intent, or measured success of unauthorized activities meant to disrupt, corrupt, and compromise system components as well as providing information to assist in response to recovery from these activities" [1]. The objective of network forensics is to identify malicious

activities from the traffic logs, discover their details, and to assess the damage [2]. The result of network forensics investigation is **digital evidence**.

For the traditional computer forensic discipline, there are a lot of proven investigative techniques and methods. Since 2001, various digital forensic models were proposed to handle the networked environments. In this paper we focus on **a generic process model for network forensic analysis**, based on various existing digital forensics models. They formalize a methodology specifically for network - based investigation [3]. The network forensics specialists go through all these processes in order to find the network attacks and the source of the attack. The general process of network forensics is formalized into main eight steps [3] representing the subsections of first section.

There are lots of proposed models for network forensic that consist of many different phases. Researchers developed many frameworks that implement these phases. One of the network forensic frameworks is honeypot framework that is explained in detail bellow. **Honeypots** play an important role for forensics and investigation of networks. Earlier, the data for forensic analysis was collected from security products like firewalls and intrusion detection systems only. With their development, honeypots have become key contributors in capturing the attack data, which is analysed and investigated, hence facilitating the process of network forensics [4].

As mentioned above, honeypot presents a model for network forensic. It consists in an environment, where vulnerabilities have been deliberately introduced, in order to observe attacks and intrusions [5]. Concurrently, the honeypot is "a computing resource, whose value is in being attacked" [6]. A **honeynet** extends the concept of a single honeypot to a highly controlled network of honeypots [7]. The term honeynet was coined by **The Honeynet Project** [8].

In this paper we focus on relationship between honeypots and network forensics. We discuss the **usage of honeypots** as **network forensics tools**. We choose generic process model for network forensic analysis to show how we can use the honeypots in process of network forensics.

1. Related works

There are some related papers, related to research presented in this paper. In [9] authors illustrate the usefulness of deploying multiple simple honeypot sensors to discover new tools or new methods used against a computers or servers. Network forensics investigation has greatly benefited from having access to data sets of honeypots. A. Almulhem in [10] focuses on various aspects of network forensics, related technologies and their limitations. According to Alhulhem, the honeypot is an ideal tool for studying attackers closely and capturing their tools, methods, goals from an investigative perspective. F. Raynal in [11] presented the definition of honeypot forensics to outline the network forensics investigation aided

by honeypot functionality. In [3] authors focus on the studying network forensics methodologies and tools in addition to developing a well understanding of honeypots terminologies and their value in network forensics.

2. Honeypots as network forensics tools

In this section we discuss the usage of honeypots as **network forensic analysis tools (NFATs)**. This tools allow administrators to monitor the networks, gather all information about anomalous traffic, assist in network crime investigation and help in generating a suitable incident response [12]. NFATs also help in analysing the insider theft and misuse of resources, predict attack targets in the near future, perform risk assessment, evaluate network performance, and protect intellectual propriety [3].

NFATs capture the entire network traffic, allow the users to analyse the network traffic according to their needs, and discover significant features about the traffic. NFATs synergize with IDSs and Firewalls and make possible the long term preservation of network traffic records for quick analysis [13]. To show how we can use the honeypots as NFATs we use the generic process model for network forensic analysis according to Pilli. We discuss the individual honeypots according to the individual phases of this generic process model in the following subsections.

2.1. Preparation

The first phase of generic framework is **preparation**. In this phase, investigators prepare tools, techniques, methods, monitoring authorizations and management support to ensure that maximum and quality evidence. Network forensics is only applicable to environments, where network security tools (sensors) like intrusion detection systems, packet analysers, firewalls, traffic flow measurement software are deployed at various strategic points on the network [3].

The fundamental principle of honeypot deployment lies in no-detection of honeypot. According to Defibaugh-Chavez et al. [14], an ideal honeypot is „mirror a real system exactly and is thus difficult to detect but unfortunately existing honeypot technology is far from ideal“. In general, there are several specific features for honeypots, but real production systems do not have this features. Before the deploying the honeypots, it is necessary to deal with the following issues:

- There is no or minimal network activity on the honeypot.
- All interactions with the honeypot are logged extensively.
- Bandwidth from honeypots is often restricted to prevent a compromised honeypot from damaging other networks.
- Low interaction honeypots do not implement a full feature set.
- Emulated environments have often significant software overhead compared to real systems.

2.2. Detection and collection

The alerts, generated by various security tools, indicating a security breach or policy violation, are observed. Unauthorized events and anomalies noticed will be analysed. The presence and nature of the attack are determined from various parameters. A quick validation is done to assess and confirm the suspected attack [3]. From the perspective of detection phase, the all honeypots are used in this phase since the main aim of each honeypot is to attract the attackers and detect them. Within **collection phase**, the data are acquired from the sensors, used to collect the traffic data. A well-defined procedure, using reliable hardware and software tools, must be in place to gather maximum evidence, causing minimum impact to the victim. This phase is very significant as the traffic data change at a rapid pace and it is not possible to generate the same trace later. The amount of data logged will be enormous, requiring huge memory space and the system must be able to handle different log data formats appropriately [3].

In detection phase and collection phase, there are a number of challenges in network forensics, especially useful data, data sources and data granularity [10].

The useful network events and records (**useful data**) determine the minimum representative attributes for each event, so that the least amount of information with highest probable evidence is stored. At this point, it is necessary to answer to question, when the honeypot sensor indicates the attack (unauthorized events or anomalies). Some authors [6], [7], [4] considered that all connections can be considered as attacks against the honeypot. This statement is based on the fact that legitimate connections are not normally directed to honeypots. Usage of new honeypots implementation changes a view to this issue. Decision about nature of the attack is important for identifying useful data for network forensics. This decision is a challenge for future research.

Data sources means that since a typical network has several possible sources of data, it is desirable to collect data from all the possible sources. On the other hand, data granularity is an issue related to selecting data sources. This issue lies in the decision how much details data should be kept. The interaction of honeypot is related to above mentioned challenges in network forensics. The **level of interaction** can be defined as the maximum range of attack possibilities that a honeypot allows an intruder to have. Usually, it is distinguished between low-interaction and high-interaction honeypots. The difference between these classes is the extent, to which the attacker is allowed to interact with the system.

The **low-level interaction honeypots** that detect attackers using software emulation of characteristics of a particular operating system and network services on the host operating system. First example of low-interaction honeypot is **Dionaea** [15]. It captures attack payloads and malware. The another example is **Kippo** [16]. It only emulates an SSH shell. Each incoming connection to Kippo sensor is an attempt to login via SSH (username and password) and the creation

of SSH session. **HoneyD** [17] emulates a lot of various operating systems and optionally provides basic service emulation.

As we showed in above mentioned examples **data sources in low-interaction honeypots** are individual services running on specific ports. **Data granularity** in this type of honeypot is limited to several information, especially IP address, port, service, combination of login and password, spatial information etc.

On the other hand, in order to get more realistic information about attackers, their methods and attacks, we use complete operating system with all services. This type of honeypot is called **high-level interaction honeypot**. The goal of this type of honeypot is to give the attacker access to a real operating system, where nothing is emulated or restricted [18]. Examples of this type of honeypot are **Argos** [19], **Sebek** [20] etc.

Compared with the low-level interaction honeypots, the high-level interaction honeypots have more data sources. This is due to the fact that is like real systems. Besides the above mentioned services, the examples of data source are shell sensors, CPU sensors, memory and disk sensors etc. The data granularity is linked to this since administrator is able to collect all available data (e.g. memory utilization, sequence of commands).

2.3. Preservation

In perspective of **data integrity** the outcome of the forensics process can be adversely affected if the collected data are altered either deliberately or accidentally. In most cases, the honeypots' data are sent to central location with the hash of data. In distributed forensics tools or frameworks [21], the authors propose an efficient data storage method, based on distributed storage, pre-selection, and specific data compression. Also, they outline a general framework, using distributed techniques' supports, easy integration of known attribution methods, and effective cooperation within the system.

In the **legal issues** of honeypots the data as **legal evidences** are important. **Convention on Cybercrime** [22] outlines the fundamental framework for digital evidence investigation. Under Article 16 of the Convention on Cybercrime competent national authorities (e.g. law enforcement) are authorized to obtain the expeditious preservation of specified computer data, including traffic data, which has been stored by means of a honeynet, in particular where there are grounds to believe that the computer data is particularly vulnerable to loss or modification.

2.4. Examination and analysis

Examinations involve forensically processing large amounts of collected data using a combination of automated and manual methods to assess and extract data of particular interest, while preserving the integrity of the data [23]. The information

obtained from the digital evidence is used to identify who, what, where, when, how and why of the incident. Redundant information and unrelated data is removed and minimum representative attributes are identified, so that the least information with the highest probable evidence needs analysis [3]. Within the next phase of generic framework – **analysis phase**, there is a need to analyse the results of the examination, using legally justifiable methods and techniques, to derive useful information, which addresses the questions that were the impetus for performing the collection and examination [23].

In these phases, investigators can use a number of honeypots as **Honeysnap** [24] or **Hflow2** [25]. **Honeysnap** is a tool for processing tcpdump data. It is designed to be a command-line tool for parsing single or multiple pcap data files and producing a 'first-cut' analysis report that identifies significant events within the processed data. **Hflow2** is a data coalesing tool for honeypot and network analysis. It allows coalescing data from snort, p0f, sebekd into a unified cross related data structure stored in a relational database.

2.5. Investigation

The main aim of the **investigation phase** is to determine the path from a victim network or system through any intermediate systems and communication routes, back to the point of attack source. The packet captures and statistics obtained are used for attribution of the attack. This phase may require some additional features from the analysis phase and hence these two phases are iteratively performed to arrive in the conclusion [3].

There are several honeypots' solution in investigation phase. Example is the paper [26] proposes a novel **IP traceback scheme** based on honeypots. In this proposed system, a **digital watermark** is put into a honeypot and the probe-scan-entraps the attacks through the honeypot, which sequentially induces the attacks to visit the digital watermark. The main objective is to reconstruct the attack route and location of the hacker's address. Another example is the **Internet Threat Monitoring** [27] that tracks threats such as denial of service (DoS) and distributed Denial of Service (DDoS).

2.6. Presentation

Last phase of generic framework is **preservation** that summarizes and provides explanation of conclusions. The observations are presented in an understandable language to the organizations management and legal personnel, while providing explanation of the various standard procedures, used to arrive at the conclusion. The systematic documentation is also included to meet the requirements [3].

We can find several implementations of honeypots, which presents data collected from honeypots. First example is **Hpfriends** [28] that is a sharing model and web frontend for the Hpfeeds data-sharing platform. Another example is

HoneyMap [29] that shows the attacks collected by the honeypots in real time. It shows these attacks and searches the geographical location of its corresponding IP address. Other examples of usage of Hpfeeds is **HpfeedsHoneygraph** [30] that indexes data collected from honeypots in addition to extracting data from them. It shows the attacks as a tree graph. It also searches the country where the IP address is and the DNS information, if it is available.

3. Conclusions and future works

The network attacks are becoming an everyday reality. Protection against these attacks plays an important role in network security and also in the network forensics. In paper we focus on network forensics and its special framework – honeypots. Paper outlines the honeypots and its relationship to generic framework for network forensics. Honeypots are strong tools in perspective of network forensics. Their usage has some advantages and several limitations.

In future work we focus on the analysis of usage of honeynets and virtual honeynets in network forensics. Very interesting challenge for future works are legal issues related to honeynets and virtual honeynets as network forensics' tools.

Acknowledgment. We thank colleagues from Czech chapter of The Honeynet Project for their comments and valuable input. This paper is funded by the VVGS grant under contract No. VVGS-PF-2013-109.

References

[1] PALMER, G.: *A Road Map for Digital Forensic Research*, Digital Forensic Research Workshop, Utica, 2001. pp. 15–30.

[2] CHNADRAN, R.: *Network Forensics*, Know Your Enemy: Learning about Security Threats, Ed. L. Spitzner, Second Edition, Addison Wesley Professional, 2004, pp. 281–325.

[3] PILLI, E. S., JOSHI, R. C., NIYOGI, R.: *Network forensic frameworks: Survey and research challenges*, Digital Investigation, vol. 7, pp. 1–12, April 2010.

[4] JOSHI, R. C., SSARDANA A.: *Honeypots: A New Paradigm to Information Security*, Science Publishers, 2011.

[5] POWELL, D., STROUD, R.: *Conceptual Model and Architecture of Maftia*, MAFTIA Project (IST-1999-11583), Deliverable D21, January 2003.

[6] THE HONEYNET PROJECT: *Know Your Enemy: Learning about Security Threats*, (2nd Edition), Honeynet Alliance, 2004, p. 768.

[7] ABBASI, F. H., HARRIS, R. J.: *Experiences with a Generation III virtual Honeynet*, Telecommunication Networks and Applications Conference (ATNAC), 2009 Australasian, IEEE, 2009.

[8] HONEYNET PROJECT: `http://project.honeynet.org/`.

[9] CHEN, P., et al.: *Comparative survey of local honeypot sensors to assist network forensics*, Systematic Approaches to Digital Forensic Engineering, 2005, First International Workshop on. IEEE, 2005.

[10] ALMULHEM, A.: *Network forensics: Notions and challenges*, Signal Processing and Information Technology (ISSPIT), 2009 IEEE International Symposium on. IEEE, 2009.
[11] RAYNAL, F., et. al.: *Analyzing the network*, IEEE Security & Privacy, vol. 2, no. 4, pp. 72–78, July 2004.
[12] SIRA, R.: *Network Forensics Analysis Tools: An Overview of an Emerging Technology*, GSEC (1.4), 2003.
[13] COREY, V., et. al.: *Network forensics analysis*, IEEE Internet Computing, 6 (6), pp. 60–66, 2002.
[14] DEFIBAUGH-CHAVEZ, P., et al.: *Network based detection of virtual environments and low interaction honeypots*, Information Assurance Workshop, IEEE, 2006.
[15] DIONAEA PROJECT: `http://dionaea.carnivore.it/`.
[16] KIPPO PROJECT: `https://code.google.com/p/kippo/`.
[17] HONEYD PROJECT: `http://www.honeyd.org/`.
[18] SPITZNER, L.: *The Honeynet Project: Trapping the Hackers*, IEEE Security & Privacy, pp. 15–23, March/April 2004.
[19] ARGOS PROJECT: `http://www.few.vu.nl/argos/`.
[20] SEBEK PROJECT: `http://projects.honeynet.org/sebek`.
[21] TANG, Y., DANIELS, T.E.: *A simple framework for distributed forensics*, Distributed Computing Systems Workshops, 25th IEEE International Conference on. IEEE, pp. 163–169, 2005.
[22] COUNCIL OF EUROPE: *ETS No. 185 – Convention on Cybercrime*, `http://conventions.coe.int/Treaty/en/Treaties/Html/185.htm`.
[23] KENT, K., et. al.: *Guide to Integrating Forensics into Incident Response*, Special Publication 800-86, Computer Security Division Information Technology Laboratory, National Institute of Standards and Technology, Gaithersburg, MD August 2006.
[24] HONEYSNAP PROJECT: `https://projects.honeynet.org/honeysnap/`.
[25] HFLOW2 PROJECT: `https://projects.honeynet.org/hflow`.
[26] Yi, Z., et al.: *IP traceback using digital watermark and honeypot*, Ubiquitous Intelligence and Computing, Springer Berlin Heidelberg, 426–438, 2008.
[27] MUNIVARA P., et. al.: *IP Traceback for Flooding attacks on Internet Threat Monitors (ITM) Using Honeypots*, 2012.
[28] HPFRIENDS PROJECT: `http://hpfriends.honeycloud.net`.
[29] HONEYMAP PROJECT: `http://www.honeynet.org/node/960`.
[30] HONEYGRAPH PROJECT: `http://sourceforge.net/projects/honeygraph/`.

Contact address

RNDr. JUDr. Pavol Sokol, Department of Computer Science, Faculty of Science, Pavol Jozef Šafárik University in Košice, Jesenná 5, 040 01 Košice, Slovak Republic,
E-mail address: `pavol.sokol@upjs.sk`, `http://pavolsokol.science.upjs.sk`

6. ročník Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf

OSSConf 2014: 159–166

VISUALIZATION OF THE ACTIVE NETWORK CONNECTIONS IN VIRTUAL HONEYNETS

PAVOL SOKOL (SK) AND TERÉZIA MEZEŠOVÁ (SK)

Abstract. Paper focuses on visualization of the active network connections in virtual honeynets based on operating system-level virtualization. It focuses on usage of visualization for purpose of monitoring and control of virtual honeynets. In paper we outline the design of proposed system and its place within distributed virtual honeynet. We also discuss the implementation of parser and visualization part and we focus on information, which are visualized in proposed system.

Key words and phrases. visualization, network connection, honeynet, virtualization.

VIZUALIZÁCIA AKTÍVNYCH SIEŤOVÝCH SPOJENÍ VO VIRTUÁLNYCH HONEYNETOCH

Abstrakt. Článok sa venuje vizualizácii aktívnych sieťových spojení vo virtuálnych honeynetoch založených na virtualizácii na úrovni operačného systému. Zameriava sa na použitie vizualizácie na účely monitorovania a kontroly virtuálnych honeynetov. V článku načrtávame návrh systému a jeho miesto v rámci distribuovaného virtuálneho honeynetu. Tiež diskutujeme o implementácii parsera a vizualizačnej časti a venujeme sa informáciám, ktoré sa vizualizované v navrhovanom systéme.

Kľúčové slová. vizualizácia, sieťové spojenie, honeynet, virtualizácia.

Introduction

In the current information society there is security gap between ability to secure information systems, computer networks and the actual level of security. For this reason, it is necessary to search for new ways of protecting critical infrastructure of the organizations. One of the relatively new approaches of information and information system infrastructure security is concept of honeypots and honeynets.

A **honeypot** can be defined as "a computing resource, whose value is in being attacked". A honeypot computer system is "a system that has been deployed on a network for the purpose of logging and studying attacks on the honeypot". These systems are purposely made insecure in order to lure attackers to study their tools, methods and motivations [1]. Honeypots can be classified according to their level of interaction, purpose, role and deployment.

A **honeynet** extends the concept of a single honeypot to a highly controlled network of honeypots [2]. It can be defined as a special kind of high-interaction honeypot. The honeynet is composed of four core parts [3]:

- **Data control** - purpose of this part is to control and contain the attacker's activity. This is the most important function and it always has to be given high priority when implementing a honeynet [4]. Honeynet data control can reduce the risk of being misused by intruders. Once the intruders take control over one of the honeynets, the intruder can use the compromised honeypot to attack other hosts, so it is necessary to control the invaders [1].
- **Data capture** monitors and captures attackers' activities within the honeynet.
- **Data collection** is necessary when organization manages more than one honeynet.
- **Data analysis** is used for analysing and tracking the captured attacks or some other malicious activities [1].

A **virtualization** can be defined as "a technology that allows multiple virtual machines to run on a single physical machine" [5]. According to Amit Singh, the virtualization is "a framework or methodology of dividing the resources of a computer into multiple execution environments by applying one or more concepts or technologies, such as hardware and software partitioning, time-sharing, partial or complete machine simulation, emulation, quality of service, and many others" [6]. According to definitions of the honeynet and virtualization, a **virtual honeynet** can be defined as all honeypots running on a single computer [5].

When talking about **visualization in network security** we have to remember that we are working with abstract and often multidimensional data. As such classic bar graphs and scatter plots used for visualization in other science fields are often used only to provide secondary statistic information. Visualization theory suggests dividing data into several hierarchical levels to minimize visual noise and allow the user to control how extensive details he wants to see.

This paper provides a discussion of specific approach to data control in virtual honeynet. This approach is based on the **decision** of virtual honeynet's **administrator** in process of outgoing network traffic control. In our approach we use **data visualization in data control** of virtual honeynet. Data visualization allows administrators to monitor active network connections. Based on this monitoring the administrator can decide what he will do in virtual honeynet (e.g. shutdown the honeypot, block specific connections etc.)

The main contribution of this paper lies in design and implementation of system, which helps honeynet's administrator to control the active network connections in virtual honeynets.

1. The design of of proposed system

In our research [7] we focus on **virtual honeynets based on operating system-level virtualization** and a distributed virtual honeynet. This **distributed virtual honeynet** is composed of virtual honeynets based on operating system-level

virtualization and **master control center** (shortly MCC). In the virtual honeynets there are the sensors in host operating system and the honeypots (virtual machines) do not have access to the sensors.**MCC** is a separate computing system that controls virtual honeynets. It includes a database where data is stored (**main database**). In this paper we focus on one part of MCC. Main aim of which is to monitor virtual honeynets. This monitoring is based on visualization of active network connections related do virtual honeynets. The architecture of the proposed system and its place within distributed virtual honeynet is illustrated in the Fig. 1.

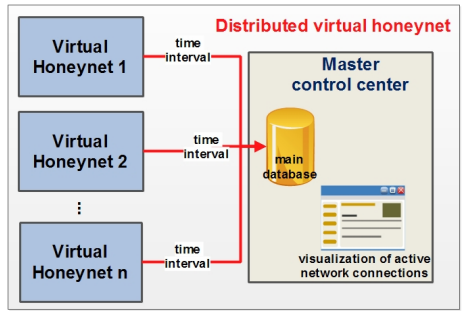


Figure 1. The place of the proposed system within the distributed virtual honeynet

Proposed system is implemented as **client-server web application** and the visualization of the data is carried out on the client's side. The proposed system includes following visualization:

- visualization of current network activity,
- visualization of hardware utilization,
- visualization of other available information.

For purpose of **visualization of current network activity**, we use **graph structure**, where the **vertices** of graph are the endpoints (source and destination) of network communications. The connections between those points produce the **edges of graph**. For the purpose of quick and efficient distinguishing the vertices that represent honeypots from vertices representing attackers, we display the vertices representing a honeypot significantly bigger. It is necessary that the information about the connections is visible and recognizable at a glance. For purposes of the proposed system, the following information is relevant:

- **source IP address**, which represents the attacker,
- **destination port**, according to which we can determine the service for which it was attacked,
- **protocol** used for communication and,
- **state of network connection**, in which the connection is currently located, if it is a TCP connection.

State of the network connection in proposed system is encoded into the width and colour of the line as illustrated on figure 2.

Visualization of hardware utilization includes the memory, CPU and disk usage of virtual honeypots and host operating system. Examples of **visualization of other available information** are total number of network connections, currently logged in users etc. This information are shown either as an info window with the information or as discoloration. We focus on these information in the following sections in more detail.

2. The implementation of proposed system

In proposed system we used **D3.js** [8] JavScript library that supports a declarative approach to handling the elements in **Document Object Model** (DOM). D3.js allows this model to connect any data to DOM and to continue working with them. It is a quick and flexible library that supports large data sets and dynamic behaviours for interaction and animation.

2.1. The implementation of parser

All information is obtained from the main database in an unprocessed text form (raw format). On receiving a request from the client's part of the system the server part of system reads the information from the main database. All data are preprocessed and they are sent to client as an array of objects of **JavaScript Object Notation** (JSON) [9].

We use the scripting language **PHP** in version **5** to communicate with the main database. PHP is also used to implement the parser. Since the output of the netstat command, which obtains information about active connections, differs from the output of commands that obtain information on honeypots, parser has two usage modes. Individual information about the state of honeypots are numbers, separated by space. The parser reads the data in the order, in which they are stored. Subsequently it adds the appropriate attribute to JSON object, in which they are written. In case that parser obtains the object of active connections, it obtains a raw text, which firstly has to be divided by line. Subsequently, parser reads each line separated by space.

2.2. The implementation of visualization part

The visualization of active connections is directly made in a web browser via JavaScript. As soon as the application is started, information about number of virtual honeynets and honeypots is obtained. On this basis, the grid layout of honeypots and honeynets is determined. In case, there is one virtual honeynet, the entire virtual honeynet is shown. If there are more virtual honeynets, they are displayed in a grid. This view is shown in Fig. 3.

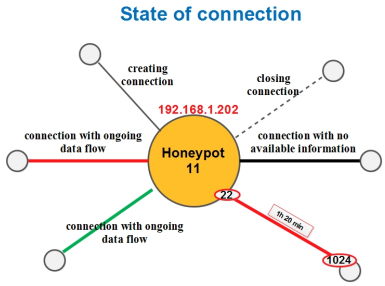


Figure 2. State of network connections in proposed system

Figure 3. Visualization grid of virtual honeynets (VH is abbreviated term from virtual honeynet)

To show the network connection, the client requests appropriate information from the database based on time, since it is a real-time visualization. The data obtained in JSON format are then read and each connection is drawn according to the relevant rules. For clarity, text information is not immediately shown. We decided to show them at the user's request. The users get a basic overview about the status of the virtual honeynets at the beginning. The user gets all specific information after the place mouse cursor on the various objects (IP addresses and ports). In Fig. 4 and Fig. 5 we can see the visualization of information about connection (Fig. 4), respectively honeypot's status (Fig. 5).

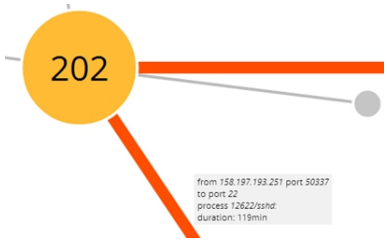


Figure 4. Visualization of information about connection

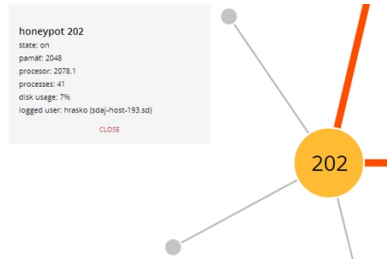


Figure 5. Visualization of information about honepot's status after the click on honeypot's vertex

Vertices, which represent honeypots, are plotted in colour. The shade of the colour is graded according to the memory and CPU utilization of honeypots. Particular value of the utilization is regularly obtained together with information about active connections. Same data are also available for the host operating system as for virtual honeypots. The shades of colour of honeypot's vertex are shown in Fig. 6.

In addition to colour distinction of vertices representing honeypots the proposed system distinguishes the colour of vertices representing attackers. The proposed system allows administrator to specify the criteria for colour coding the vertices representing attackers. Examples of these criteria are a specific service, a specific connection, or IP address from specific subnet.

In previous section we outlined the design of visualization of **state of network connections** in proposed system and the colour differentiation of types of connection. This distinction by colour is basic and the administrator can use the **advanced colour distinction of types of connection**. It is based on the **type of connection** (TCP, UDP) and the **state of TCP connection** (e. g. listen, established, closed etc.). Advanced colour distinction of types of connection is shown in Fig. 7.

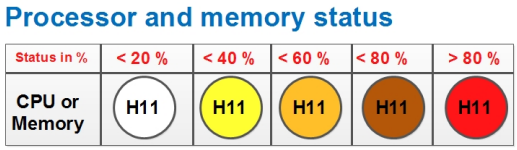


Figure 6. The shades of the colour of honeypot's vertex

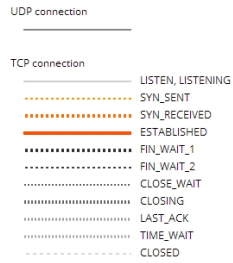


Figure 7. Advanced colour distinction of types of connection

Implementation of proposed system also includes the **control of honeypots**. In this case, the administrator decides to shut down the specific honeypot, he can do it using the right-click of the mouse on vertex of this honeypot.

3. The main database and showing data

Data sent to main database by virtual honeynets in regular time intervals are stored in the format, in which they are outputted by operating system commands used to retrieve the information. Following information is stored in the main database in the form: virtual honeynet identifier, honeypot identifier, data type, data and timestamp.

We distinguish between these data types and their attributes:

- **memory information**: size of total, free, used memory, cache memory, shared memory and size of total swap and shared swap memory,
- **disk usage**: total, free and used disc space,
- **active logged users** and where they are signed in from,
- **number of active connections**: number of TCP and UDP connections,
- **CPU usage**: CPU usage, number of proccesses,

- **active network connections**: protocol, source and destination IP address and port, state of connection and the associated process ID,
- amount of transferred data.

All this information is collected from the virtual file system – `/proc` file system using the above mentioned tools. All tools we have chosen are distributed under an open source license.

Since it is a web application, it is necessary to ensure the **integrity of showing data**. One way to implement cryptographic data integrity is usage of **HMAC hash function**, where the hash function is computed using by a secret code. The problem is that this secret code would be either embedded directly in the application or served to application. One of the possible solutions to ensure data integrity would be usage of **HTTPS protocol**.

4. Related works

The Honeynet Project [10] focuses on the issues of visualization in honeypots and honeynets. This organization created some very interesting implementations. Examples of these are following: [11] and [12]. **HoneyMap** [11] shows the attacks collected by the honeypots in real time. It shows this attacks and searches the geographical location of its corresponding IP address. On the other hand **HpfeedsHoneygraph** [12] indexes data collected from honeypots in addition to extracting data from them. It shows the attacks as a tree graph. It also searches the country where the IP address is and the DNS information, if it is available. These examples are visualization of data from honeypots and their main purpose is to present data collected from honeypots. Proposed solution uses the visualization data from the whole virtual honeynets to monitor and control them. Similar system to our proposed system represents **distributed Honeypot log management** [13]. Authors design and implement a log management server, which automatically and periodically collects log files from honeypot. Information collected by each honeypot is sent to the log management server in secure manner. Subsequently, it parses the information into the database server, where users can search for information via the web interface. Proposed system visualizes the active network connections from virtual honeynets, not directly from honeypots. Proposed system partially implemented the data control of honeynet.

5. Conclusions and future works

This paper focuses on data control in virtual honeynets based on visualization in network security. In paper we proposed system, which shows active network connections and allows administrator to monitor and control virtual honeynets. We outline the design of proposed system and its implementation.

In the future, we will focus primarily on connection between vertices representing attackers. Also we will focus on the amendment of logic in visualization and completion of spatial information of vertices representing attackers.

Acknowledgment. We thank colleagues from Czech chapter of the Honeynet Project for their comments and valuable input. This paper is funded by the VVGS grant under contract No. VVGS-PF-2013-109.

References

[1] SPITZNER, L.: *the Honeynet Project: Trapping the Hackers*, IEEE Security & Privacy, pp. 15–23, March/April 2004.
[2] YASINSAC A, MANZANO Y.: *Policies to enhance computer and network forensics*, In: Proceedings of the IEEE workshop on information assurance and security, New York, pp. 289–95, 2001.
[3] CHNADRAN, R.: *Network Forensics*, In Know Your Enemy: Learning about Security Threats, Ed. L. Spitzner, Second Edition, Addison Wesley Professional, pp. 281–325, 2004.
[4] THE HONEYNET PROJECT: *Know Your Enemy: Learning about Security Threats*, (2nd Edition), Honeynet Alliance, p. 768, 2004.
[5] ABBASI, F. H., HARRIS, R. J.: *Experiences with a Generation III virtual Honeynet*, Telecommunication Networks and Applications Conference (ATNAC), Australasian, IEEE, 2009.
[6] SINGH, A.: *An introduction to virtualization*, Retrieved May 1 (2004), 2009.
[7] SOKOL, P., PISARČÍK, P.: *Kernel modul of senzors for OS-virtualization honeynet*, Proceedings ICTIC (Proceedings in Conference of Informatics and Management Sciences), ISBN 978-80-554-0648-0, ISSN 1339-231X, vol. 2, issue 1, pp. 427–432, 2013.
[8] D3JS PROJECT: `http://d3js.org/`.
[9] RFC 4627: *The application/json Media Type for JavaScript Object Notation (JSON)*, `http://www.ietf.org/rfc/rfc4627.txt`.
[10] THE HONEYNET PROJECT: `http://honeynet.org/`.
[11] SCHÖSSER, M.: *HoneyMap – Visualizing Worldwide Attacks in Real-Time*, `http://www.honeynet.org/node/960`.
[12] YUCHIN CH. J.: *HpfeedsHoneyGraph – Automated Attack Graph Construction for Hpfeeds Logs*, `http://www.honeynet.org/node/957`.
[13] VISOOTTIVISETH, V, et al.: *Distributed honeypot log management and visualization of attacker geographical distribution*, Computer Science and Software Engineering (JCSSE), 2011 Eighth International Joint Conference on. IEEE, 2011.

Contact addresses

RNDr. JUDr. Pavol Sokol, Department of Computer Science, Faculty of Science, Pavol Jozef Šafárik University in Košice, Jesenná 5, 040 01 Košice, Slovak Republic,
E-mail address: `pavol.sokol@upjs.sk`, `http://pavolsokol.science.upjs.sk`

Terézia Mezešová, Department of Computer Science, Faculty of Science, Pavol Jozef Šafárik University in Košice, Jesenná 5, 040 01 Košice, Slovak Republic,
E-mail address: `tmezesova@gmail.com`

6. ročník Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf
OSSConf 2014: 167–178

VYBRANÉ KRÍZOVÉ SITUÁCIE F/OSS SIEŤOVÝCH SLUŽIEB NA MSÚ V BANSKEJ BYSTRICI

PETER TUHÁRSKY (SK)

Abstrakt. V tomto článku si predstavíme niekoľko vybraných krízových situácií F/OSS sieťových služieb, s akými sa stretlo Oddelenie informatiky na Mestskom úrade v Banskej Bystrici. Budeme diskutovať o okolnostiach a možných preventívnych opatreniach. Uvedené príklady môžu administrátorom poslúžiť ako inšpirácia alebo varovný príklad pri riešení podobných problémov.

Kľúčové slová. Open Source, samospráva, bezpečnosť, spam, Postfix, proxy.

SELECTED CRISIS SITUATIONS OF F/OSS-OPERATED NETWORK SERVICES ON MUNICIPALITY OF BANSKÁ BYSTRICA

Abstract. This paper presents selected crisis situations that has Department of IT on Municipality of the city of Banská Bystrica dealt with on F/OSS-operated network services. We will discuss circumstances and possible preventive measures. The examples might serve as inspiration or warning message.

Keywords. Open Source, Municipality, Security, spam, Postfix, proxy.

Úvod

Samospráva Mesta Banská Bystrica vo veľkej miere využíva free/open-source softvér (F/OSS) pri zabezpečovaní svojich potrieb IT. Samospráva pritom zahŕňa nielen samotný Mestský úrad, ale aj ďalších niekoľko desiatok organizácií – mestskú políciu, materské školy, školské jedálne, zariadenia sociálnych služieb a dlhodobej starostlivosti atď. Tieto jednotky boli počas ostatných rokov postupne pripojené k sieti MsÚ pomocou MPLS, čím sa podstatne zvýšila zložitosť sieťovej infraštruktúry, ktorá sa v súčasnosti rozprestiera už po celom území mesta. Alikvótne stúpli aj nároky na sieťové služby. Napríklad počet aktívnych schránok elektronickej pošty stúpol z 320 v roku 2010 na 440 v roku 2014.

Samotné začlenenie značne heterogénneho pôvodného softvéru a hardvéru týchto zariadení do sieťovej infraštruktúry, starostlivosť o ne, a samozrejme sprístupnenie sieťových služieb takým používateľom, ktorí sú s nimi len slabo, ak vôbec, oboznámení, predstavuje samostatnú kategóriu problémov ktorú netreba zvlášť objasňovať nikomu, kto sa podobnými aktivitami už zaoberal.

Na pozadí projektu elektronizácie (OPIS), pre oddelenie informatiky (OIT) sú takéto integračné aktivity organizačne aj logisticky náročné, a svojou naliehavosťou majú tendenciu odsunúť do úzadia starostlivosť o bežiace systémy. Ako sa však ukazuje, takejto zmene priorít treba v záujme udržania kvality a dostupnosti služieb odolávať.

1. Prípad – Mailserver

Príznaky: Používatelia spozorovali výpadky funkcionality poštového servera – správy neprichádzali, alebo len s veľkým oneskorením.

Diagnostika: Záťaž servera bola úplne mimo hraníc chodivosti (load > 50). Hľadal som príznaky neobvyklých aktivít analyzátorom logov (pflogsumm) a našiel som 250000 správ z jednej platnej lokálnej adresy. Keďže použitie skutočnej, existujúcej adresy odosielateľa z našej poštovej domény je povolené len pre autentifikovaných používateľov, a keďže logy obsahovali adekvátne záznamy o prihláseniach tohto používateľa, bolo zrejmé, že došlo k zneužitiu účtu.

Riešenie: Po zablokovaní inkriminovaného účtu nastala okamžite úľava systému a spriechodnenie poštových služieb. V spoole SMTP servera Postfix bolo ešte 150000 nedoručených správ, tieto boli vymazané skriptom približne v tomto duchu:

```
riadkov=‘mailq | wc -l‘ ; riadkov2=‘expr $riadkov - 2‘
   mailq | head -n $riadkov2 | grep -v ’^ *(’ | awk ’BEGIN { RS
= "" } { if ($7 == "pouzivatel@banskabystrica.sk" ) print $1 }’ |
tr -d ’*!’ | postsuper -d -
   #Iné použiteľné rozlišovacie parametre: $1=ID, $2=message
size, $7=sender, $8=recipient1, $9=recipient2
```

Analýza a následné opatrenia: Zamestnanci majú prístup k pracovnej e-mailovej schránke cez Internet. Táto funkcionalita je oceňovaná a žiadaná, no prináša so sebou aj riziko, napr. vo forme odchytenie hesla keyloggerom pri prihlasovaní k pracovnej e-mailovej schránke, napr. na infikovanom počítači mimo kontroly OIT. Inou možnosťou je phishing, ako aj slovníkové uhádnutie hesla robotom; touto variantou sa budeme zaoberať ďalej. Odchytenie hesla z prebiehajúcej komunikácie nebolo pravdepodobné – jednak pre relatívne vyššiu obtiažnosť oproti iným metódam zistenia hesla, jednak pre používanie TLS (vtedy nebola známa zraniteľnosť).

Každopádne, bolo jasné, že útočníci sa dostali k menu a heslu používateľa a použili ho na rozposielanie správ, čím preťažili spamové filtre a spôsobili zahltenie. Samotný fakt, že k tomuto útoku došlo, ukazuje rastúcu sofistikovanosť útočníkov – nielenže získali meno a heslo, ale museli ho správne priradiť ku konkrétnemu serveru a konkrétnej službe, a potom úspešne zneužiť, pričom všetko pravdepodobne prebehlo plne automaticky.

Pri správe poštového servera sa bezpečnostné opatrenia zvyčajne zameriavajú proti vonkajšiemu zneužitiu (napr. open-relay) a proti prijímaniu nevyžiadanej pošty (spam). Týmto cieľom je podriadené množstvo nastavení servera a vyhradený špecializovaný filtrovací softvér. Oprávnení, autentifikovaní používatelia predstavujú „vnútorný okruh“ a ochrániť server voči zneužitiu z ich strany je paradoxne ťažšia úloha.

Udalosť potvrdila význam organizačných opatrení a školení ohľadom používania hesiel; kompromitovaný účet patril používateľke z nedávno pričleneného zariadenia ktorá ešte nebola prešla bezpečnostným školením a zrejme si nezmenila svoje dočasné úvodné (jednoduchšie) heslo.

Udalosť naznačuje správnosť testovania nielen prichádzajúcej, ale aj odchádzajúcej pošty pomocou antivírusových a antispamových filtrov; tento konkrétny útok síce neobsahoval vírus (preto nebol zachytený antivírusom) a z hľadiska spamu bol dostatočne "nízkoprofilový" aby prekonal spamový filter, ale toto nemusí byť pravidlom pre všetky útoky; práve filter odchádzajúcej pošty môže byť tou poslednou bariérou ktorá zabráni masívnemu odosielaniu neželanej pošty. Ostatne, práve zahltenie spamových filtrov vyvolalo viditeľné príznaky preťaženia poštového systému a spomalilo distribúciu spamu; nebyť filtrov, mohlo dôjsť k nepozorovanému odoslaniu rádovo väčších objemov správ kým by došlo k odhaleniu prieniku. Pri veľkosti inkriminovaných správ 6kb a uplink konektivite 10 Mbit/s, každú hodinu mohlo odísť 10000/8/6*3600=750000 správ.

Paradoxne, javí sa byť výhodou, ak je poštový systém dimenzovaný takým spôsobom, aby s prehľadom zvládal bežnú prevádzku, no aby nebol úplne odolný voči zahlteniu. Ide najmä o pridelený počet CPU, operačnú pamäť, nastavené obmedzenia SMTP servera, ale aj prenosového pásma v smere do Internetu (riešime pomocou QoS na strane firewallu). Inými slovami, aj v prípade, že to hardvérové možnosti dovoľujú, nemusí byť vždy pozitívom, ak poštovému serveru pridelíme všetky dostupné prostriedky. Na základe dlhodobého pozorovania prevádzky som poštový server nadimenzoval tak, že zaručuje komfortné používanie pošty pri bežných cca 13000 doručených správach denne a záťaži v rozpätí 0,1 – 0,5.

V takýchto situáciách sa môže prejaviť značný rozdiel v správaní systému aj v závislosti od zvolenej „filozofie“ kontroly správ. V našom prípade filtre pracujú v režime pred zaradením do fronty. Predstavuje to nárok na kapacitu servera dostupnú v reálnom čase, ale je to v súlade so štandardmi, pretože server v rámci relácie preberá zodpovednosť za definitívne prijatie alebo odmietnutie správy. Keby sme mali filtre aplikované až po zaradení do fronty, útok by sa neprejavil rýchlym zahltením kapacity servra, ale len postupným spomaľovaním doručovania správ kvôli narastajúcej fronte. Útok by tak bol zrejme odhalený o niečo neskôr.

Udalosť naznačuje trvalý problém ochrany hesiel. Keby útočník nebol taký nedočkavý a posielal správy pomalšie, kompromitácia hesla mohla ostať dlhý čas neodhalená. Z hľadiska používateľského komfortu však nie je reálne, zrušiť dostupnosť e-mailových účtov cez Internet, k čomu sú zvyčajne používané súkromné

počítače zamestnancov, mimo kontroly OIT. Ostáva teda na administrátorovi, aby aspoň v rámci dostupných možností urobil opatrenia na strane servera.

2. Prípad – Mailserver

Príznaky: Ako v prípade 1.

Diagnostika: Analýza logov neukázala konkrétneho odosielateľa, ale správy mali spoločnú doménu odosielateľa PAUsa.com. Cez náš server boli odoslané z IP adries z rôznych kútov sveta, čo ma viedlo k podozreniu na open relay. Ukázalo sa však, že správy boli odoslané cez autentifikovaný účet. Odhalenie tohto účtu bolo veľkým prekvapením – bol ním guest!

Riešenie: Akútne riešenie spočívalo v zmene hesla a vyčistení spoolu od inkriminovaných správ. Od prvej odoslanej útočnej správy po odstránenie poslednej zo spoolu uplynuli 2 hodiny.

Analýza: Keďže útok prichádzal z mnohých IP adries, zjavne nešlo o kompromitáciu konkrétneho počítača niektorého nášho používateľa, ale o koordinovaný útok prostredníctvom botnetu. Znepokojivým zistením bolo zneužitie účtu guest, z čoho vyplynulo viacero dôsledkov:

1. Zneužitý bol doménový účet určený pôvodne pre celkom iné účely.
2. Heslo muselo byť odhalené cieľavedomým skúšaním metódou pokus-omyl.

Používateľ guest nebol lokálnym používateľom operačného systému na poštovom servri. Prístup k odosielaniu správ majú len autentifikovaní doménoví používatelia s aktívnym príznakom prístupu k pošte; zhodou okolností, guest týmto príznakom disponoval, čo bolo následne prehodnotené.

Doména je silným nástrojom a integrujúcim prvkom sieťovej infraštruktúry. Sústreďuje množstvo účtov a poskytuje k nim informácie využívané v rámci rôznych sieťových služieb. V našej doméne LDAP máme implementovaných viacero schém a dôsledky jednotlivých atribútov nemusia byť pri vytváraní účtu hneď zrejmé. Drvivá väčšina účtov patrí používateľom a majú pomerne jednoliatu štruktúru atribútov a súvisiacich práv. Avšak zdá sa, že treba venovať zvýšenú pozornosť špeciálnym účtom vytvoreným za určitým účelom, a ich separácii od služieb ktoré nemusia využívať. Guest mal pôvodne slúžiť len pre technické účely Windows domény – ako vidno, aj účet vytvorený s istým konkrétnym úzkym cieľom môže priniesť celkom nečakané dôsledky do takých služieb, pre ktoré pôvodne nebol zamýšľaný, a môže predstavovať bezpečnostné riziko.

Udalosť ma viedla k vytvoreniu skriptu na vyhodnocovanie doménových účtov. Situáciu komplikuje z hľadiska OIT nie celkom transparentný proces zmien alebo ukončenia práce niektorých používateľov, najmä pokiaľ ide o pracovníkov externých organizácií alebo zamestnancov na dohodu apod. ktorí často potrebujú prístup k sieťovým službám.

Heslo používateľa guest nemohli útočníci zistiť pomocou keyloggera na napadnutom PC, pretože tento účet sa nepoužíva na prihlásenie k žiadnej verejne prístupnej sieťovej službe. Znamená to, že ho museli uhádnuť pomocou opakovaného skúšania. Tento fakt viedol k ďalším opatreniam.

Služby s najväčším potenciálom zneužitia, napr. ssh alebo login, sa vo všeobecnosti tešia pozornosti administrátorov a majú aj v defaultnom nastavení implementované obranné mechanizmy voči opakovanému chybnému prihláseniu, napríklad vo forme predlžujúceho sa intervalu pre zadanie hesla, alebo dočasné zablokovanie po piatich neúspešných pokusoch apod.

Avšak potenciál zneužitia môžu mať aj zdanlivo menej zaujímavé služby, ako napr. SMTP server, POP alebo IMAP server. Tieto služby navyše nie je možné umiestniť na nejaký obskurný port kde by unikli pozornosti útočiacich robotov. Všetkým službám, ktoré sú prístupné cez sieť a zahŕňajú prihlasovanie pomocou doménových účtov, treba venovať adekvátnu pozornosť, aby neboli zneužité k uhádnutiu hesla!

V našom prípade, hlavným kandidátom na zneužitie bol SMTP server. Postfix má v defaultnom nastavení pomerne benevolentné správanie k opakovanému prihláseniu, posielaniu hromadných správ apod. Síce to zaručuje nerušenú prevádzku poštového systému, no zneužitie je o to ľahšie a intenzívnejšie. Príslušné parametre ani nie sú zahrnuté v štandardnom distribučnom konfiguračnom súbore, takže nie sú „na očiach“ pri nastavovaní servera; sú skryté medzi stovkami iných defaultných parametrov s ktorými Postfix potichu pracuje. A ako je známe, do defaultných parametrov zložitých systémov sa zasahuje len v prípade skutočnej potreby zmeniť konkrétne správanie servera; ináč platí „never touch running system“. Mnohí správcovia akiste nadobudli skúsenosť toho typu, keď dobre myslená zmena niektorého parametra „dostihne“ správcu po dlhšom čase keď si už na ňu ani nespomenie a zrazu musí kvôli nej riešiť zdanlivo náhodný problém.

Nástroje pre bezpečnostné obmedzenia klientov Postfixu sú pomerne hrubé. Z hľadiska SMTP sa za klienta považuje stroj s konkrétnou IP, čiže môže ísť o PC z ktorého používateľ odosiela poštu, ale môže ísť aj o poštový server. Kým z hľadiska bežného používateľa by sme mohli pomerne pohodlne zaviesť bezpečnostný limit napr. 10 odoslaných správ za hodinu, takéto obmedzenie by mohlo vyvolať prakticky zablokovanie pošty prichádzajúcej od servra obsluhujúceho niektorú veľkú poštovú doménu, napr. gmail.com. Obmedzenia treba preto nasadzovať uvážene a na základe dlhodobého sledovania bežnej poštovej prevádzky.

V rámci konfiguračného súboru main.cf sa dajú klienti obmedziť napr. nasledovnými parametrami:

```
#Koľko spojení naraz môže nadviazať jeden klient (default 50):
  smtpd_client_connection_count_limit = 8 #Definovanie jednotky času (default 60 = 60 sekúnd): anvil_rate_time_unit = 60
```

```
#Koľko spojení môže klient nadviazať za jednotku času (default
0 = neobmedzene):
smtpd_client_connection_rate_limit = 30
#Koľko požiadaviek na doručenie môže klient urobiť za jed-
notku času bez ohľadu na to, či sú prijaté (default 0 = neobme-
dzene):
smtpd_client_message_rate_limit = 30
#Koľkým adresátom môže klient poslať správu za jednotku času
bez ohľadu na to, či sú prijaté (default 0 = neobmedzene):
smtpd_client_recipient_rate_limit = 120
#Klienti, ktorí sú vyňatí z obmedzení (default $mynetworks):
#smtpd_client_event_limit_exceptions
#Koľko relácií TLS môže urobiť vzdialený SMTP klient za jed-
notku času (default 0 = neobmedzene):
#smtpd_client_new_tls_session_rate_limit
```

Posledným preventívnym opatrením po tejto udalosti bolo nasadenie služby fail2ban, čo je log analyzér ktorý sleduje vybrané služby a pri prekročení povoleného počtu pokusov zablokuje inkriminovanú IP adresu na stanovený čas.

3. Prípad – Mailserver

Mailserver bol premigrovaný do virtuálneho prostredia aby dostal k dispozícii výkonnejší hardvér a zároveň sa oň mohol deliť s ďalšími infraštruktúrnymi servrami. Odvtedy 2-3x denne zatuhol do úplnej nepoužiteľnosti, vyťažené 4 CPU AMD Opteron 2,6GHz na 100%.

Diagnostika: Záťaž išla na vrub procesov filtra Amavis, takže prvotné pátranie sa zameriavalo na možný ďalší útok. Analýza logov však neodhalila žiadnu podozrivú aktivitu. Antivírus využívaný filtrom Amavis tiež nejavil žiadne nezvyčajné ťažkosti. Záťaž na strane Amavisu teda mohla byť dôsledkom a nie príčinou tuhnutia. Okrem toho, záťaž jednotlivých poštových subsystémov nikdy predtým neviedla k takémuto úplnému zastaveniu servera.

Ako príčina sa ponúkala niektorá zo zmien, ktoré nastali v čase pred problémami:

1. Bol vymenený server (hw) za typovo totožný no dlhšie nepoužívaný.
2. Boli vymenené niektoré moduly pamäte.
3. Nasadená nová verzia hypervízora.
4. Nastavenia CPU afinity v rámci hypervízora kvôli výkonovej izolácii virtuálnych serverov.
5. Nastavenie parametra hypervízora **sched.cpu.vmspconsolidate** pre využitie jadier CPU na jednom puzdre a nie viacerých.

Riešenie: Memtest neodhalil žiadny problém operačnej pamäte. Zrušenie afinity nepomohlo. Problém vymizol po tom, ako som vrátil server na pôvodný hw (okrem diskov).

Analýza: Z hľadiska obmedzení rozpočtu, ako aj zmyslu pre nákladovú efektívnosť, infraštruktúru zabezpečujú aj staršie, výkonovo stále dostatočné servre. Používanie starého hardvéru však nie je bez rizika. Porucha základnej dosky alebo procesora sa môže prejavovať zvláštnymi spôsobmi. V tomto prípade bol chybou ovplyvnený iba poštový server, ostatné virtuálne servre ani hypervízor nejavili žiadne príznaky, čo odpútavalo pozornosť od problému hw. Na druhú stranu, ani nové zariadenia nie sú nutne spoľahlivé. Relatívne najviac porúch na blade serveroch sme zaznamenali počas prvého roku používania (2008). Po záručnej výmene inkriminovaných súčastí potom servre fungovali ďalšie roky už takmer bez problémov. Za najspoľahlivejší preto považujem taký server, ktorý už prešiel aspoň dvoma rokmi používania. K opotrebovaniu však nevyhnutne skôr či neskôr dochádza a treba s ním počítať. Je vhodné mať v zálohe výpočtový výkon, ktorý umožní nahradiť prípadný výpadok jedného zo serverov a získať tým čas k nákupu nového hardvéru.

4. Prípad – Mailserver

Klasické príznaky zahltenia poštového servera. Analýza logov neodhalila cudzí útok ani vnútorné problémy servera. Problém spôsobila chybová notifikácia nového CMS servera, nasadeného v rámci projektu elektronizácie služieb mesta. Nedostupnosť databázy počas plánovanej odstávky viedla k vygenerovaniu niekoľkých tisíc správ v priebehu minút. Bol dosiahnutý bezpečnostný limit SMTP servera ktorý následne odmietol ďalšie spojenia. Po uplynutí timeout-u dorazili ďalšie správy a cyklus sa opakoval, až kým som nezablokoval príslušný systémový účet a nevyčistil frontu. Dodávateľ potom implementoval potrebné obmedzenia na strane notifikačnej služby.

Ponaučenie: Nasadenie nových systémov do infraštruktúry môže v nečakanej chvíli zasiahnuť zabehnuté systémy nečakaným spôsobom.

5. Prípad – Proxy

Príznaky: V denných špičkách pomalé načítavanie stránok, miestami hraničiace s úplnou nefunkčnosťou.

Diagnostika: Vysoká záťaž servera, zrejme v rovine pamäťového subsystému, mierne zvýšené swapovanie.

Riešenie: Rozdelenie služieb servra na dva fyzicky oddelené servre.

Služby pripojenia na Internet zabezpečoval 8 rokov starý server. V jeho náplni bol nielen proxy/cache server Squid, ale aj filtrovanie stránok programom Dansguardian (DG) a antivírusom.

Tieto úlohy po niekoľko rokov s prehľadom plnil, hoci postupne stúpali nároky uvedených programov a aj samotný internetový obsah sa stal objemnejším a komplikovanejším. Pripojením ďalších organizácií ešte viac vzrástli nároky, avšak pri internetovom pripojení s prenosovým pásmom 3 Mbit/s server týmto nárokom stačil. Po zmene poskytovateľa pripojenia s nárastom pásma na garantovaných 20 a neskôr 30 Mbit/s, sa však server už stal úzkym hrdlom pre tok dát.

Rozšírenie pásma bolo vzhľadom k rastu siete potrebným a vítaným krokom, umelé spomaľovanie nebolo reálnou možnosťou. Nebolo ňou ani vyradenie filtra obsahu – síce by okamžite vyriešilo výkonnostný problém, no znamenalo by otvorenie veľkého bezpečnostného rizika v podobe prieniku škodlivého softvéru do počítačov vo vnútornej sieti. Spoliehať sa iba na lokálny antivírus na koncových PC by znamenalo degradáciu bezpečnostnej politiky. Generická vrstva ochrany, ktorá bráni prieniku potenciálne nebezpečných súborov bez ohľadu na aktuálnosť alebo neaktuálnosť vírusovej databázy, je dôležitým prvkom bezpečnosti.

Ponechanie filtra vyžadovalo riešenie niekoľkých rovín problému. Jednou bol zjavne nedostatočný výkon hw, ktorý už nezachránilo ani rozšírenie RAM; hoci už nedochádzalo k swapovaniu, procesy DG nestíhali požiadavky dostatočne rýchlo spracúvať a tak neostávali voľné procesy pre obsluhu ďalších požiadaviek. Ďalšie ladenie výkonnostných parametrov DG už neprinášalo badateľné výsledky a z hľadiska používateľov bolo pripojenie stále neuspokojivé.

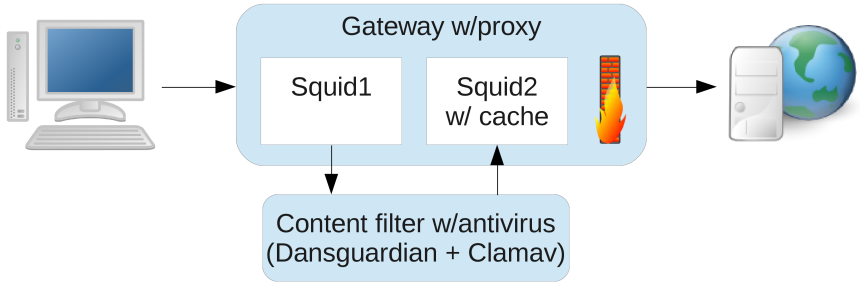
Keďže nebol k dispozícii výkonnejší dedikovaný server, bolo potrebné služby a záťaž rozdeliť. Prvý server ostal na pôvodnom hw a rieši iba proxy/cache, druhý zabezpečuje filtrovanie obsahu a kvôli výkonu a škálovateľnosti je umiestnený na výkonnom hw ako virtuálny server. Uskutočniť takéto rozdelenie v časovom strese však bolo podstatne komplikovanejšie, než by sa zdalo.

Prvý problém nastal, keď sa filtrovací server nedostal do siete. Všetko vyzeralo v poriadku, nastavenia boli OK, ostatné guesty fungovali. Chyba bola spôsobená jedným zo 4 fyzických sieťových rozhraní, ktoré nebolo zapojené do siete a hypervízor ho pridelil práve tomuto servru. Po identifikovaní rozhrania a jeho vyradení z network poolu, komunikácia začala fungovať.

Druhým problémom bolo stanovenie štruktúry služieb. DG nie je plnohodnotným proxy serverom, hoci sa voči klientom tvári ako transparentný proxy. Diagnostika takejto komunikácie sa ukazuje ako náročná a z hľadiska routovania bolo najvýhodnejšie, aby vstupná aj výstupná komunikácia prebiehala cez gateway server. Nakoniec sa ako schodné ukázalo nasledujúce usporiadanie zobrazené na obr. 1.

Rozbehnutie dvoch inštancií Squidu na jednom stroji však nebolo triviálne. Každý Squid musí mať:

- svoj vlastný init.d skript,
- svoj vlastný PID súbor (nastavený v init.d skripte),
- svoj vlastný konfiguračný súbor (tiež nastavený v init.d skripte),
- svoj vlastný komunikačný port,



Obr. 1. Usporiadanie v 5. prípade – Proxy

- svoje vlastné log súbory,
- samostatné adresáre pre log súbory.

Inštaláciu som si zjednodušil nainštalovaním distribučného Squid 2 aj 3, avšak init.d skript pre verziu 2 som nahradil skriptom verzie 3 a potom už len upravoval. Dochádzalo však k zahlcovaniu diskového priestoru partície pre /var/log spôsobenému nefunkčnosťou rotácie logov. Logrotate.d mal problémy s logsúborom pre Squid1 – síce súbor korektne premenoval, ale neukončil Squid daemon a ten do súboru naďalej zapisoval. Aby logrotate.d korektne riešil dve inštancie Squidu, bolo potrebné prinútiť ho aby hľadal správny PID procesu, obracal sa na správny spúšťač a použil správny konfiguračný súbor (-f ...)

Ďalší problém bol takmer mysteriózny – vždy asi do hodiny po štarte sa objavil jeden proces DG so záťažou CPU 11% ktorý nemizol a vytrvalo brzdil premávku; v jeho fronte totiž trvale čakali nevybavené ďalšie požiadavky. Keďže som sa dočítal, že novšia verzia DG mala byť lepšie uspôsobená pre funkciu transparentnej proxy, snažil som sa ju nasadiť.

Táto verzia už nebola k dispozícii v distribučnom balíku a s kompiláciou boli problémy. Niektoré parametre som musel dať priamo do kompilačných parametrov v Configure (napr. predvolené cesty), ináč nebol ochotný ani akceptovať konfiguračný súbor ponúknutý init.d skriptom, podobne musel byť zakompilovaný používateľ a skupina (bez ohľadu na prítomnosť nastavení v konfiguračnom súbore programu) apod. Ani po týchto krokoch sa mi ho nepodarilo spojazdniť – síce naštartoval bez zjavných problémov, ale na žiadnu komunikáciu jednoducho neodpovedal. Pre akútnosť problémov s pripojením som sa touto cestou nemohol ďalej zaoberať a musel som sa vrátiť k starej distribučnej verzii 2.10.1.1.

Mal som podozrenie, že uvedený proces DG tuhne z toho dôvodu, že DG je stavaný ako transparentný proxy pre koncových klientov, a možno sa nepočítalo s možnosťou, že sa naň bude obracať už Squid. Teoreticky sa dalo uvažovať o NATovaní apod. Najprv som sa však sústredil na vyladenie zvyšných logovaných problémov DG, napr:

ACL::checklistMatches WARNING: 'fast_mime' ACL is used but there is no HTTP reply – not matching

rep_mime_type som musel zmeniť na req_mime_type

DG temporary disabling (Bad Gateway) digest from

Musel som správne nastaviť parametre cache_peer na oboch Squid-och:

proxy-only no-digest no-query no-netdb-exchange no-tproxy

Zmyslom bolo, aby Squid1 nekešoval, neťahal sám, na nikoho iného sa neobracal, len na DG. Potrebné k tomu boli aj tieto parametre:

```
never_direct allow all
  cache_mem 1 kb
```

Druhému Squidu boli uvedené rovnaké direktívy cache_peer pretože takisto bolo potrebné zakázať pokusy o koordináciu s „partnerským proxy“ v podobe DG, ktorý takejto koordinácii beztak nerozumie. Oba Squidy boli prepnuté do režimu intercept. Ďalšie problémy nasledovali:

Failed to select source for 'http://...'

Zriedkavá hláška v logu, že Squid nemohol určiť spôsob stiahnutia. Príčinou je, že preťažený DG odmietal ďalšie konexie a Squid1 mal zakázané ísť vlastnou cestou (never_direct).

Squid2 - WARNING: Forwarding loop detected for:

Problém s nastaveniami x_forwarded_for. Indikáciou tejto príčiny bolo aj to, že Squid2 videl ako zdroj požiadavky DG, nie pôvodného klienta. Okrem správneho nastavenia x_forwarded_for bolo potrebné správne nastaviť aj ACL k tomuto parametru:

```
acl my_other_proxy src 10.2.2.5
  follow_x_forwarded_for allow localhost
  follow_x_forwarded_for allow my_other_proxy
  follow_x_forwarded_for deny all
  forwarded_for on
```

Nechcel som však, aby údaj prostredníctvom koncovej proxy opustil našu sieť, pretože by to vzdialeným servrom zbytočne prezrádzalo, čo máme vnútri siete. Pre Squid2 preto platilo:

forwarded_for off

DG zase nemal pridávať svoj parameter x_forwarded_for do hlavičky dopytu, lebo Squid2 by potom videl požiadavky z jedinej adresy. Má však rešpektovať existujúcu hlavičku, preto:

```
forwarded_for = off
  usexforwardedfor = on
```

Naďalej sa u Squid1 objavovali chyby typu „failed to select source“ a „TCP connection to proxy2.misbb.sk/3128 failed:“ kvôli vyčerpaniu voľných procesov DG. Nastavil som počet dostupných pripojení v Squid1 na totožný s počtom DG pomocou direktívy max_conn v parametri cache_peer, ale nepomohlo (Squid nemohol vybaviť požiadavky klienta).

Keď som hľadal možnosti zvýšenia počtu dostupných procesov DG, obával som sa, že budem musieť vytvoriť dva paralelné servre, na ktoré by sa Squid1 postupne obracal ako na cache_peer. Našťastie to nebolo potrebné. Zistil som, že v rámci distribúcie tak kernel ako aj DG majú dostatočnú rezervu, takže limit sa dá zvýšiť na podstatne vyššiu hodnotu, než uvádza základná dokumentácia ako strop (250 „pre veľmi veľké siete“). Alikvótne som zvýšil limit v Squid1.

Výsledok:

- Prestali sa objavovať chybové správy v logu Squid1
- Prestali sa objavovať zamrznuté procesy
- DG Filter prestal byť úzkym hrdlom internetového pripojenia

DG v distribučnej verzii má stále niektoré problémy. Časť problémov bola odstránená v poslednej alfa verzii, no tá sa do distribúcie nedostane a nepodarilo sa mi ju, ako som uviedol, spojazdniť zo zdrojových kódov. Nová stabilná verzia nie je na obzore, DG je v podstate mŕtvy projekt. Hľadal som možnosti náhrady. Komerčné možnosti existujú, napr. Smoothwall, no v súčasnej rozpočtovej situácii hľadám najprv F/OSS riešenia. V tejto sfére som však žiadny iný program porovnateľný s DG nenašiel. Určitú nádej ponúka jeho fork e2guardian, ktorý v súčasnosti testujem.

Záver

Predstavili sme si niekoľko krízových situácií z bežnej prevádzky a ich riešenia. Sieťová infraštruktúra založená na F/OSS má dobré predpoklady k tomu, aby fungovala spoľahlivo, výkonne a bezpečne. Dokáže pracovať pomerne dlhé obdobia prakticky bez významného zásahu. Ako však ukazuje skúsenosť, tento nedostatok starostlivosti sa časom začne prejavovať ako "investičný deficit", ktorý sťažuje prevenciu a riešenie krízových stavov.

V ideálnom prípade by sa viacerým problémom dalo predísť, napríklad rozsiahlym predbežným testovaním alebo podrobným ladením parametrov. Žiaľ, nárast rozsahu siete a jeho koordinácia, nárast počtu a zložitosti telekomunikačných prostriedkov, softvéru, integrácie existujúcich a nových systémov, za poklesu rozpočtu na IT, generuje množstvo akútnych úloh a nedáva dostatočný priestor pre plánované implementácie a profylaxiu.

Napriek tomu, že F/OSS systémy dokážu bežať do značnej miery navonok bezúdržbovo, starostlivosť o ne netreba podceňovať. Okrem samozrejmých bezpečnostných aktualizácií, kontroly záloh, stavu hardvéru a diskových úložísk, koncepčného

plánovania škálovateľnosti, rastúcu pozornosť si vyžadujú problémy súvisiace s nárastom sieťovej konektivity a sofistikovanosti útočníkov so záujmom o zneužitie. Účinné opatrenia na ochranu systémov, ako je napríklad voľba silného hesla a rozumné zaobchádzanie s ním, sú však z veľkej časti závislé na počítačovej gramotnosti používateľov, a táto oblasť trpí vo všeobecnosti dlhodobým deficitom. OIT sa snaží tento deficit zmierniť systematickými školeniami zamestnancov.

Administrácia systémov môže pre laikov vyzerať ako neuchopiteľná činnosť, ťažko kvantifikovateľná a zdanlivo dokonca zbytočná – „veď to funguje samo“. Používatelia vnímajú požiadavky podľa svojho vlastného hodnotového systému, ktorý môže byť diametrálne odlišný od potrieb a rizík vnímaných administrátorom. Používateľ môže urgovať riešenie „vážneho problému“, ktorým je načítavanie stránky v trvaní 10 sekúnd namiesto 3, kým administrátor zápasí, aby spojenie vôbec udržal v chode bez ohrozenia bezpečnosti siete. Hoci je táto práca zameraná na systémy, jej náročnosť napokon aj tak spočíva v riešení priorít v komunikácii s používateľmi a menežmentom.

Kontaktná adresa

Mgr. Peter Tuhársky, Oddelenie informatiky, Mesto Banská Bystrica, Slovenská Republika,
E-mailová adresa: peter.tuharsky@banskabystrica.sk

6. ročník Žilina 2.- 4. júla 2014
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSSConf
OSSConf 2014: 179–187

OPTIMALIZACE SÍTĚ LINEK MHD VE VYBRANÝCH SOFTWARECH

RICHARD TUREK (CZ)

Abstrakt. Příspěvek se zabývá matematickým modelováním návrhu linkové sítě a přidělováním vozidel MHD metodou PRIVOL ve vybraných softwarech. V úvodu je uvedena problematika tvorby sítě linek včetně vstupní charakteristiky metody PRIVOL. V následující části je popsán způsob zápisu modelu při řešení konkrétní úlohy v tabulkovém kalkulátoru Calc a optimalizačním software Gusek. V závěru příspěvku je provedeno zhodnocení výsledků.

Klíčová slova. Lineární programování, Calc, Gusek, síť linek MHD.

OPTIMIZING TRANSPORT NETWORK IN SELECTED SOFTWARES

Abstract. The paper deals with mathematical modeling of the draft line network and the allocation vehicles of public transport by PRIVOL in a selected softwares. In the introduction is given problem of making public transport route network, including baseline characteristics methods PRIVOL. The following the example of notation the model in solving concrete tasks in spreadsheet Calc and in optimization software Gusek. In conclusion is made assessment.

Keywords. Linear programming, Calc, Gusek, Transport Network.

Úvod

Jedním ze základních problémů při řešení hromadné osobní dopravy je tvorba linkové sítě. K řešení problému tvorby sítě linek MHD je možno přistoupit různými způsoby. Nejčastěji je využíván tzv. zkušenostní přístup, kdy pověření zaměstnanci dopravců zajišťují možnosti přímého spojení mezi vybranými zastávkami, zpravidla na základě svých logických úvah vyplývajících z historicky vzniklých vazeb. Nedílnou součástí pro jejich rozhodování jsou také výstupy z dopravních průzkumů.

Pokročilejší přístup v dopravní praxi v tuzemsku i v zahraničí představují modely na tvorbu systému linek MHD, které využívají pokročilé metody operačního výzkumu. Problémy výběru linek z dané množiny linek, resp. její tvorby na základě poptávky cestujících a dopravní nabídky určené počtem a kapacitou autobusů resp. odjezdy spojů jsou formulované a řešené jako:

- úlohy celočíselného lineárního programování,
- úlohy dynamického programování,

- optimalizační úlohy v grafech.

Při řešení optimalizačních úloh je nutné využít specializovaný software. V současnosti existuje řada specializovaných softwarů na optimalizaci dopravních procesů. Jejich rozšíření v dopravní praxi je však vázáno na jejich pořizovací cenu. Mezi tyto softwary patří např. Xpress-IVE, LINDO, CPLEX nebo Matlab. Úlohy lineárního programování (LP) je možné řešit také v nekomerčních softwarech. Jednou z možností využití těchto software je získání řešení matematického modelu při optimalizaci linek MHD. Na použití tohoto přístupu je zaměřen předložený článek.

1. Metoda PRIVOL

Na řešení úlohy, kdy je třeba vybrat linky MHD tak, aby byla pokryta poptávka cestujících a určit jaký počet vozidel je třeba přiradit linkám je využívána metoda PRIVOL (PŘIřazení VOzidel Linkám), která patří mezi úlohy celočíselného lineárního programování, viz např. literatura [1], [2]. Uvedený model byl v minulosti sestaven řešitelským kolektivem Výskumného ústavu dopravného v Žilině, a. s., pod vedením prof. RNDr. Jana Černého DrSc., Dr. h. c.

Každá optimalizační metoda vyžaduje před vlastním řešením určitá vstupní data.

Vstupními daty v případě metody PRIVOL jsou:

- širší množina linek,
- intenzity cestujících na jednotlivých úsecích za zvolené časové období,
- informace o vozidlovém parku (typy vozidel a jejich kapacita),
- oběžné doby na jednotlivých linkách (počty oběhů).

Matematický model metody PRIVOL může mít více tvarů, konkrétní tvar se volí v závislosti na variabilitě vstupních podkladů a zvoleném optimalizačním kritériu. V dalším textu bude uvedena varianta modelu PRIVOL použitelná v podmínkách systémů MHD provozovaných jedním druhem dopravního prostředku. Optimalizačním kritériem je počet autobusů, který bude minimalizován.

Význam jednotlivých symbolů použitých v matematickém modelu je následující:

x_{lj} – počet vozidel j-tého typu, které budou nasazeny na linku l,
k_j – kapacita j-tého typu vozidla,
N_l – počet oběhů vozidla na lince l za hodinu,
$q(h)$ – přepravní proud na úseku h,
L_o – širší množina linek,
L_h – množina linek obsluhujících danou hranu,
J – množina typů vozidel,
H – množina hran.

Matematický model minimalizující počet vozidel

$$\min f(x) = \sum_{l} \sum_{j} x_{lj}, \tag{1}$$

za podmínek

$$\sum_{l \in L_h} \sum_{j \in J} k_j x_{lj} N_l \geq q(h), \quad h \in H, \tag{2}$$

$$x_{lj} \in Z^+ \cup \{0\}, \quad j \in J, \quad l \in L_0. \tag{3}$$

Výraz (1) reprezentuje účelovou funkci. Podmínka (2) zajišťuje, že na každé hraně bude nabídnuto minimálně tolik míst, kolik je průměrně požadováno (za stanovený čas). Podmínka (3) zajišťuje, že proměnná bude nabývat celočíselných nezáporných hodnot. Symbol Z^+ znamená množinu kladných celých čísel.

2. Aplikace modelu PRIVOL v podmínkách MHD Prostějov

V této části bude pozornost věnována charakteristice MHD Prostějov a použití modelu PRIVOL při racionalizaci provozu MHD Prostějov v podmínkách ranní přepravní špičky.

2.1. Definování problému, charakteristika MHD Prostějov

Současný stav MHD v Prostějově je charakteristický dvěma základními nedostatky, formulovanými v [4, 5]. Prvním nedostatkem je nepřiměřeně velký počet linek vzhledem k velikosti města, což přispívá k nepřehlednosti, zejména pro cestujícího, který MHD v Prostějově nevyužívá pravidelně. Druhým nedostatkem jsou nepravidelné intervaly mezi spoji jednotlivých linek, které zhoršují orientaci cestujících o odjezdech (příjezdech) spojů.

Městskou hromadnou dopravu ve městě Prostějov zajišťuje dopravce FTL – First Transport Lines a.s. Území města je obsluhováno prostřednictvím 15 linek a jedné linky komerční, na které neplatí tarif MHD. Systém tras linek MHD je radiálně -– okružní, což je dáno situováním vnitřního okruhu ve vnější hranici historického jádra města. Radiálně z něj pak vycházejí další trasy.

2.2. Software Calc

Tabulkový kalkulátor Calc nabízí známé uživatelské prostředí, kdy se do jednotlivých buněk zadávají potřebná data. V případě modelu LP se jedná o zápis účelové funkce, omezujících podmínek a označení buněk, ve kterých mají být vyčísleny proměnné. Pro samotné řešení je určena doplňková aplikace Řešitel.

Postup při řešení konkrétní optimalizační úlohy.

V první řadě je třeba přehledně zapsat strukturu modelu a vstupní data. Jejich rozvržení je libovolné, musí však být dodržena pravidla, která vyžaduje funkce

Řešitel. Většina koeficientů jsou přímo zadané hodnoty, v uvedené úloze se jedná o kapacitu vozidel a intenzity cestujících.

Účelová funkce zahrnuje proměnnou (C4:Q4), která modeluje počty přiřazených vozidel. S ohledem na skutečnost, že nástroj Řešitel předpokládá v účelové funkci součin, byla do této doplněna konstanta o hodnotě 1 (C5:Q5). Jejich skalární součin (SUMPRODUCT(C4:Q4;C5:Q5)) vyjádříme do vybrané buňky (S4), ve které později získáme její hodnotu.

Následně dosadíme hodnoty ze vstupních dat do omezujících podmínek, viz blok (C15:Q46). Řádky představují jednotlivé omezující podmínky a sloupce obsluhu hran jednotlivými linkami. V případě levých stran zapíšeme součiny kapacity k a počtu oběhů N (C11:Q11) tam, kde je po vybrané hraně vedena linka, která ji obsluhuje. Na pravé strany nerovností dosadíme intenzity přepravního proudu na hranách (sloupec T).

K tomu, aby bylo možno řešit takto zapsaný model v aplikaci Řešitel, je ještě třeba vyjádřit skalární součiny levých stran a proměnných v účelové funkci do samostatného sloupce R za účelem zajištění provázanosti. Např. pro první podmínku vyjádříme tento součin jako (SUMPRODUCT(C15:Q15,C4:Q4)) do buňky (R15). Zápis modelu je uveden na obr. 1.

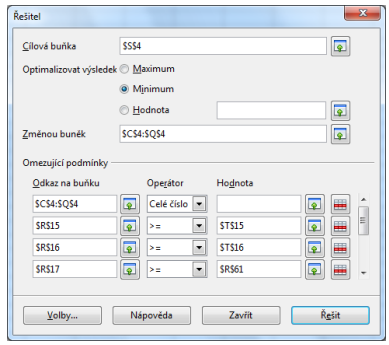
S4 =

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1																				
2		ÚF																		
3			x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12	x13	x14	x15		Celkem	
4		Přiřazení																		
5		Konstanta	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1			
6																				
7		Vstupní údaje																		
8			L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	L11	L12	L13	L14	L15			
9		N	1,66	1,3	1,2	1	1,2	1,07	1,03	1,3	1	1,11	1,3	1,2	1,25	1,3	1,03			
10		k	90	90	90	90	90	90	90	90	90	90	90	90	90	90	90			
11		k*N	149,4	117	108	90	108	96,3	92,7	117	90	99,9	117	108	112,5	117	92,7			
12																				
13		Omezující podm.																		
14			x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12	x13	x14	x15	součin LS		q(h)
15		H1	149,4		108	90	108	96,3	92,7	117		99,9	117						≥	1135
16		H2						96,3								117	92,7		≥	88
17		H3	149,4		108	90	108		92,7	117			117		112,5	117	92,7		≥	612
18		H4										99,9			112,5				≥	30
19		H5	149,4		108	90	108	96,3	92,7	117	90				112,5				≥	1166
20		H6									90	99,9	117		112,5				≥	164
21		H7	149,4	117	108					117				108					≥	601
22		H8		117		90	108	96,3	92,7		90			108	112,5				≥	897
23		H9														117	92,7		≥	42
24		H10			108									108		117	92,7		≥	478
25		H11	149,4	117						117									≥	146
26		H12						96,3											≥	83
27		H13				90													≥	173
28		H14		117					92,7					108			92,7		≥	130
29		H15									90	99,9							≥	177
30		H16									90				112,5				≥	103
31		H17										99,9	117						≥	66
32		H18		117		90	108		92,7		90	99,9	117	108	112,5				≥	613
33		H19		117		90	108		92,7		90	99,9	117	108	112,5				≥	559
34		H20					108	96,3			90	99,9	117						≥	206
35		H21					108				90	99,9							≥	74
36		H22			108											117	92,7		≥	423
37		H23												108					≥	35
38		H24			108		108				90	99,9				117	92,7		≥	283

Obrázek 1. Zápis matematického modelu na pracovní ploše

Po ukončení zápisu modelu otevřeme nástroj Řešitel, který se nachází pod záložkou Nástroje. V dialogovém okně Řešitel (obr. 2) je potom nutné zadat potřebné informace. Nejdříve zadáme adresu buňky, ve které má být vyčíslena hodnota účelové funkce (S4) a označíme oblast proměnných buněk (C4:Q4). Poté

doplníme odkazy na levé strany (sloupec R) a pravé strany podmínek (sloupec T) a vybereme typ omezení. Nezápornost proměnných definujeme prostřednictvím tlačítka Volby.



Obrázek 2. Dialogové okno Řešitel

V okamžiku, kdy máme nadefinovanou účelovou funkci, omezující a obligatorní podmínky spustíme optimalizační výpočet prostřednictvím tlačítka Řešit. Po ukončení optimalizačního výpočtu se zobrazí dialogové okno Výsledek Řešitele s informacemi, zda bylo nalezeno optimální řešení. Potvrzením tlačítka OK získáme na příslušeném listě (obr. 3) hodnotu účelové funkce (S4), hodnoty proměnných (C4:Q4) a výsledky součinů levých stran (sloupec R).

S4 =SUMPRODUCT(C5:Q5;C4:Q4)

	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1																			
2	ÚF																		
3		x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12	x13	x14	x15		Celkem	
4	Přiřazení	1	1	3	2	1	1	1	1	1	1	0	1	1	1	0		16	
5	Konstanta	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1			
6																			
7	Vstupní údaje																		
8		L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	L11	L12	L13	L14	L15			
9	N	1,66	1,3	1,2	1	1,2	1,07	1,03	1,3	1	1,11	1,3	1,2	1,25	1,3	1,03			
10	k	90	90	90	90	90	90	90	90	90	90	90	90	90	90	90			
11	k*N	149,4	117	108	90	108	96,3	92,7	117	90	99,9	117	108	112,5	117	92,7			
12																			
13	Omezující podm.																		
14		x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12	x13	x14	x15	součin LS		q(h)
15	H1	149,4		108	90	108	96,3	92,7	117		99,9	117					1167,3	≥	1135
16	H2						96,3								117	92,7	213,3	≥	88
17	H3	149,4		108	90	108		92,7	117			117		112,5	117	92,7	1200,6	≥	612
18	H4										99,9			112,5			212,4	≥	30
19	H5	149,4		108	90	108	96,3	92,7	117	90				112,5			1269,9	≥	1166
20	H6									90	99,9	117		112,5			302,4	≥	164
21	H7	149,4	117	108					117				108				815,4	≥	601
22	H8		117		90	108	96,3	92,7		90			108	112,5			904,5	≥	897
23	H9														117	92,7	117	≥	42
24	H10			108									108		117	92,7	549	≥	478
25	H11	149,4	117						117								383,4	≥	146
26	H12						96,3										96,3	≥	83
27	H13				90												180	≥	173
28	H14		117					92,7					108			92,7	317,7	≥	130
29	H15									90	99,9						189,9	≥	177
30	H16									90				112,5			202,5	≥	103
31	H17										99,9	117					99,9	≥	66
32	H18		117		90	108		92,7		90	99,9	117	108	112,5			908,1	≥	613
33	H19		117		90	108		92,7		90	99,9	117	108	112,5			908,1	≥	559
34	H20					108	96,3			90	99,9	117					394,2	≥	206
35	H21					108				90	99,9						297,9	≥	74
36	H22			108											117	92,7	441	≥	423
37	H23												108				108	≥	35
38	H24			108		108				90	99,9				117	92,7	738,9	≥	283

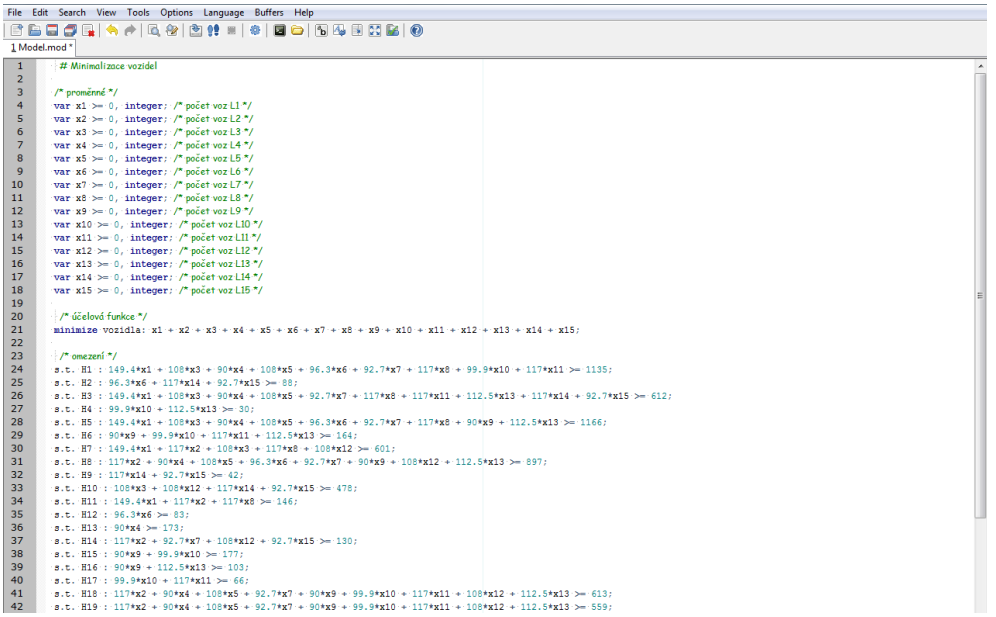
Obrázek 3. Matematický model se získanými výsledky

2.3. Software Gusek

S ohledem na omezený počet proměnných při řešení v procesoru Calc je k řešení rozsáhlejších úloh LP nutné využít specializovaný software. Jedním z dostupných OSS softwarů je software Gusek. Primárním modelovacím jazykem Guseku je GNU Mathematical Programming Language (GMPL), který je srozumitelný i pro uživatele, kteří běžně modelovací jazyky nevyužívají. Mezi tyto uživatele patří např. technologové hromadné osobní dopravy.

Při spuštění programu Gusek se zobrazí hlavní pracovní okno (Obr. 4) obsahující menu, ikony a pracovní plochu. Menu a ikony slouží k obsluze vytvářených, resp. již vytvořených souborů v programu Gusek, např. otevírání a ukládání souborů prostřednictvím záložky File. Pracovní plocha slouží k řešení konkrétních příkladů prostřednictvím příkazů, které se zadávají do dialogových řádků.

Na začátku zápisu modelu je třeba příkazem var nadefinovat proměnné a obligatorní podmínky. Podmínky nezápornosti zapíšeme obvyklým způsobem, celočíselnost proměnných definujeme příkazem integer. V dalším kroku zapíšeme prostřednictvím příkazu minimize popř. maximize účelovou funkci. Následně zapíšeme omezující podmínky, do kterých dosadíme hodnoty ze vstupních dat. Pro zápis omezujících podmínek nevyžaduje syntaxe jazyka GMPL žádné specifické příkazy ale pro lepší čitelnost je vhodné uvést klíčovou zkratku s.t. (subject to). Zápis modelu ukončíme klíčovým příkazem end. Příklad zápisu modelu je znázorněn na obr. 4.



Obrázek 4. Zápis matematického modelu na pracovní ploše

Před provedením výpočtu je potřeba vytvořený soubor uložit s příponou mod. Optimalizační výpočet následně spustíme prostřednictvím záložky *Tools Go* nebo příslušné ikony. V případě, že budeme požadovat vypsání výsledků v samostatném souboru, označíme v záložce *Tools* položku *Generate Output File on Go*. Software pak vygeneruje výstupní soubor s příponou out.

2.4. Software LP Solve

Za účelem srovnání softwaru Gusek s obdobným OSS nástrojem pro řešení úloh lineárního programovaní byl vybrán program LP Solve. Tento software podobně jako Gusek umožňuje využít programovací jazyk GMPL, pro který platí výše popsané výhody. LP Solve má také velmi podobné uživatelské prostředí.

Při vlastním řešení modelu v LP Solve však program zobrazí chybovou hlášku týkající se obligatorní podmínky zajišťující celočíselnost řešení. Pro ověření správnosti zápisu podmínky byl tento zápis použit u úlohy menšího rozsahu, kdy tento problém nenastal. Tímto se potvrdila skutečnost, že konkrétní úlohu nelze v tomto softwaru řešit. Určitou nevýhodou tohoto softwaru je tedy omezená možnost celočíselného programování.

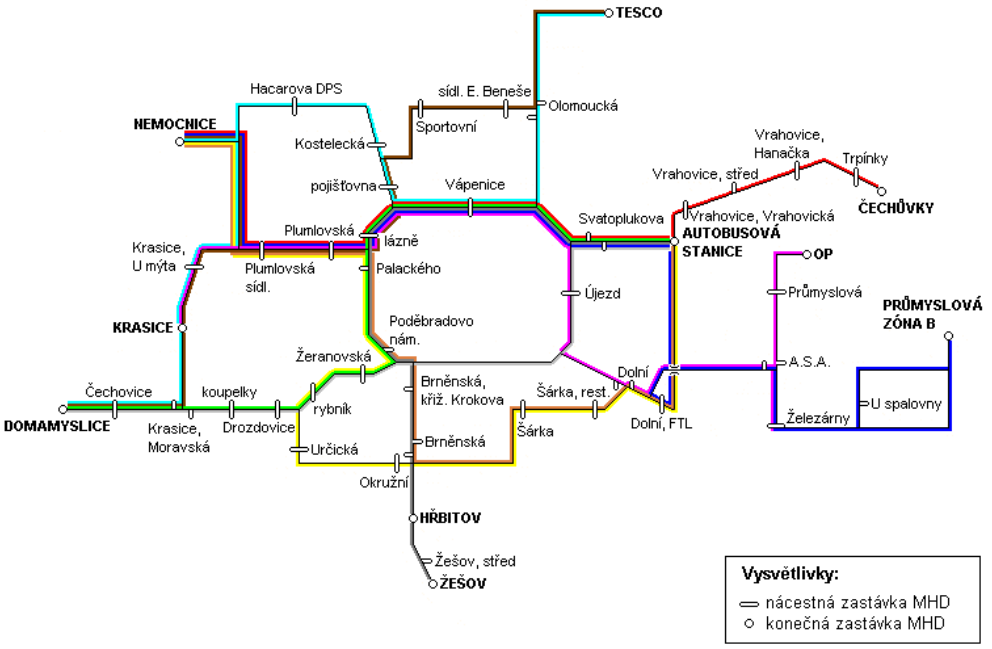
3. Zhodnocení

Po provedení optimalizačních výpočtů v obou softwarech bylo provedeno srovnání výsledků. Na základě tohoto srovnání byla zjištěna shoda v počtu přiřazených vozidel. Výsledky, které byly získány po provedení výpočtu, jsou uvedeny v tabulce 1 a znázorněny na obr. 5.

Tabulka 1. Vyhodnocení počtu nasazených vozidel *(*v současném stavu nejsou vozidla přidělena konkrétní lince)*

Linka	1	2	4	5	6	7	9	11	15	16	19	21	32	41	74	$\sum$
Stávající stav*	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	19
Calc	1	1	3	2	1	1	1	1	1	1	0	1	1	1	0	16
Gusek	1	1	3	2	1	1	1	1	1	1	0	1	1	1	0	16

V důsledku racionalizace došlo ke snížení počtu autobusů a také linek. Výsledek metody PRIVOL činí 16 autobusů obsluhujících 13 linek. Navrhované řešení představuje efektivní využití autobusů. Linky č. 19 a 74 byly zřejmě zrušeny vzhledem k malým intenzitám přepravního proudu v daných relacích a částečnému kopírování trasy jiné linky. Snížení počtu linek přispívá ke zpřehlednění sítě MHD Prostějov a eliminaci souběhů linek. Každý autobus je nově přidělen jedné lince, což umožňuje zpravidelnění provozu.



Obrázek 5. Schéma linek MHD Prostějov po optimalizaci

V relacích původně obsluhovanými linkami č. 19 a 74 budou cestující nuceni nově přestupovat. S ohledem na obdobné vedení tras linek zajišťujících alternativní spojení se však v obou případech bude jednat pouze o 1 přestup. Analýza alternativních spojení a možností řešení přestupů je uvedena v [5].

Závěr

Zpracování optimalizačních úloh v tabulkovém procesoru Calc vyžaduje běžné uživatelské znalosti. K přepisu modelu dochází formou tabulky, která obsahuje jednotlivé omezující podmínky a slouží jako podklad pro optimalizační výpočet realizovaný prostřednictvím aplikace Řešitel, v níž se definují jednotlivé zákonitosti. Tento způsob představuje dostupný nástroj pro řešení úloh středního rozsahu. Řešení úlohy LP v optimalizačním software Gusek vyžaduje znalost programovacího jazyka GMPL protože při zpracování dochází k transformaci modelu do tohoto jazyka, uvedený jazyk však neklade na uživatele žádné větší nároky. Výhodou je, že syntaxe jazyka GMPL srozumitelně koresponduje s algebraickou formou matematického modelu. Uvedený přístup tak mohou využít i uživatelé bez znalosti programovacích jazyků. Největší výhodou software Gusek je možnost řešit úlohy velkého rozsahu.

Reference

[1] ČERNÝ, J., KLUVÁNEK, P.: *Základy matematickej teórie dopravy*, Bratislava, VEDA 1991, s. 279, ISBN 80-224-0099-8.
[2] JANÁČEK, J.: *Matematické programování*, Žilina, ŽU v Žilině 1999, 1. vydání, s. 225, ISBN 80-7100-573-8.
[3] SUROVEC, P.: *Technológia hromadnej osobnej dopravy: cestná a mestská doprava*, Žilina, ŽU v Žilině, 1998, 1. vydání, s. 117, ISBN 80-7100-494-4.
[4] *Studie městské hromadné dopravy města Prostějova včetně komplexní dopravní obslužnosti průmyslové zóny*, UDIMO spol s. r. o., 2007.
[5] TUREK, R.: *Matematické modelování vybraných problémů MHD Prostějov*, Diplomová práce. Ostrava. Vysoká škola báňská – Technická univerzita Ostrava, 2009.
[6] *FTL – First Transport Lines a.s.*, `http://www.ftl.cz/stranky/28/mhd-v-prostejove/`.
[7] *Calc*, `http://www.openoffice.cz/calc`.
[8] *Gusek*, `http://gusek.sourceforge.net/gusek.html`.
[9] *LPSolve*, `http://gusek.sourceforge.net/gusek.html`.

Kontaktní adresa

Ing. Richard Turek, Ph.D., Divize bezpečnosti a dopravního inženýrství, Centrum dopravního výzkumu, v.v.i., Líšeňská 33a, 636 00 Brno, Česká republika,
E-mailová adresa: `richard.turek@cdv.cz`

OTVORENÝ SOFTVÉR VO VZDELÁVANÍ, VÝSKUME A V IT RIEŠENIACH
Zborník príspevkov medzinárodnej konferencie OSSConf 2014,
konanej 2.–4. júla 2014 v Žiline

Prvé vydanie 2014
Elektronická sadzba programom pdfLaTeX Rudolf Blaško
Vydal SOIT, Bratislava, Žilinská univerzita v Žiline
Tlač C–PRESS, s. r. o., Košice
Rozsah 188 strán
Náklad 100 ks

ISBN 978-80-970457-4-6

ISBN 978-80-970457-4-6