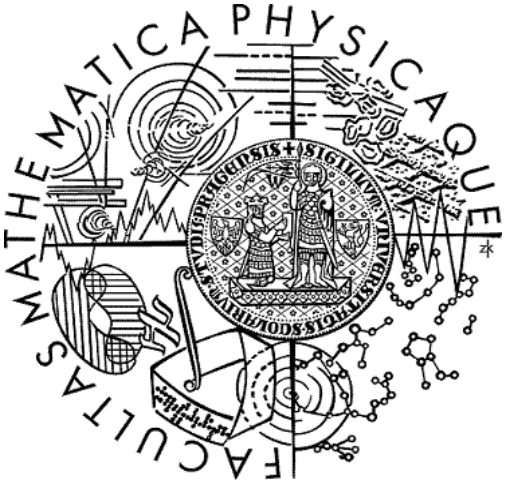
Univerzita Karlova v Praze

Matematicko–fyzikální fakulta

DIPLOMOVÁ PRÁCE



Lada Oberreiterová

Akviziční systém odborné literatury s podporou informovaného rozhodování

Katedra softwarového inženýrství

Vedoucí diplomové práce: RNDr. Michal Kopecký, Ph.D.

Studijní program: Informatika

Na tomto místě bych ráda poděkovala všem, kteří mi byli nápomocni při psaní této práce. Zejména děkuji RNDr. Michalovi Kopeckému, Ph.D. za cenné rady a připomínky. Děkuji také PhDr. Anně Součkové, vedoucí Ústřední knihovny VŠCHT Praha, za konzultace z oblasti knihovnictví. Dále sem patří i poděkování všem blízkým za jejich trpělivost a podporu v době vzniku této práce.

Prohlašuji, že jsem svou diplomovou práci napsala samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce.

V Praze dne 11. prosince 2007 Lada Oberreiterová

Název práce: Akviziční systém odborné literatury s podporou informovaného rozhodování
Autor: Lada Oberreiterová
Katedra : Katedra softwarového inženýrství
Vedoucí práce: RNDr. Michal Kopecký, Ph.D.
e–mail vedoucího: Michal.Kopecky@mff.cuni.cz
Abstrakt:

Knihovny jsou důležitým zdrojem informací nezbytných pro vědeckou a výzkumnou činnost, pro práci v akademickém prostředí či rozvoj naší společnosti. Knihovny investují ročně velké částky do nákupu informačních zdrojů pro rozšíření knihovního fondu, a to především proto, aby co nejlépe uspokojily informační potřeby svých čtenářů. Pro optimalizaci akviziční politiky využívají knihovny různé prostředky. Mezi významné kontrolní a rozhodovací mechanismy patří i statistické rozbory, které se stávají nezbytným podkladem pro obhajobu vynaložených finančních prostředků, pro plánování jejich dalšího využití a pro vyhodnocení efektivity využívání dostupných zdrojů.

Cílem diplomové práce je analyzovat, navrhnout a implementovat systém, který poskytne podporu při plánování investic a zkvalitňování knihovního fondu. Nejprve je pozornost věnována procesům a postupům, které v knihovnách probíhají při akvizici, dále získávání znalostí důležitých pro specifikaci požadavků na aplikaci a její analýzu. Po úvodní analytické části tato práce popisuje volbu technologií a vlastní návrh aplikace. Implementace a testování výsledné aplikace celou práci uzavírá. Výsledkem implementace je databázová aplikace pro zpracování dat o informačních zdrojích a o jejich využívání, archivace těchto dat v databázi pro budoucí použití, nabídka analýz těchto dat a statistické výstupy v různých formátech.

Klíčová slova: akvizice, knihovna, statistické rozbory, reporty, OLAP

Title: Customized literature acquisition system with support for qualified decision making
Author: Lada Oberreiterová
Department: Department of Software Engineering
Supervisor: RNDr. Michal Kopecký, Ph.D.
Supervisor's e–mail address: Michal.Kopecky@mff.cuni.cz
Abstract:

Libraries are important and essentials resources for scientific research, work in academic environment and development of our society. Libraries invest vast sums of money to purchase information resources in order to extend their materials and by doing so, satisfying the needs of their readers. Several means to optimize the acquisition policy are utilized. Statistical analysis is one of the significant decisive and control mechanisms which is essential foundation for the vindication of financial expenses, planning for their further use and objective examination of available resources utility.The aim of my diploma thesis is to analyse and to make a proposal and to implement the system which allow facilitation at the scheduling of the investments and to improve the quality of the library materials.

At first the focus is dedicated to a processes and techniques which pass over during the acquisition, next obtaining the knowledges significant for the unique demands of applications and analysis. After the preliminary part of analysis this tesis describe the choice of the technologies and my personal concept of application. My tesis is concluded with the implementation and the testing procedure of the resulting application. The result of the implementation is a database application which process the data about the informations resources and about their utilization, next the archiving of this data in database for the further use and at last the proposal of analyses of the data and the statistic outgoing in several formats.

KeyWords: library, statistics, reports, analysis, OLAP

OBSAH

1 Úvod

Během studia či práce často vyhledáváme dokumenty a články obsahující důležité informace, které nám pomáhají řešit nejrůznější problémy. K vyhledávání můžeme využít celou řadu nástrojů počínaje bibliografickými databázemi přes nejrůznější katalogy až po vyhledávací nástroje jako Google, Scholar a podobné. Některé zdroje jsou volně dostupné na Internetu, ale velká část informačních zdrojů je placená a tím pro mnoho uživatelů nedostupná. Jedna z možností, jak uspokojit naše informační potřeby, jsou knihovny.

Knihovny investují ročně velké částky do nákupu informačních zdrojů za účelem rozšíření knihovního fondu a to především proto, aby co nejlépe uspokojily informační potřeby svých čtenářů. O investicích rozhoduje knihovník, který se musí zajímat o potřeby čtenářů a dostupné finance investovat co nejefektivněji. Vzhledem k narůstajícímu množství dokumentů je důležité, aby knihovník kladl důraz na kvalitu a preferoval ji před kvantitou.

Většina z nás chodí do knihovny, půjčuje si dokumenty, využívá elektronické informační zdroje dostupné na Internetu a nevnímá, jaké úsilí stálo knihovníka, aby fond knihovny byl kvalitní, aby byly k dispozici relevantní dokumenty, které co nejvíce pokrývají naše informační potřeby.

V období volby tématu diplomové práce jsem začínala pracovat v knihovně VŠCHT Praha. Zde jsem se stále častěji setkávala s pojmy jako např. akvizice, způsob doplňování knihovních fondů a postupně zjišťovala, jak zoufalý je nedostatek nástrojů, které by knihovníkovi usnadnily rozhodování jak investovat finanční zdroje. Navštívila jsem proto i další knihovny, abych se s touto problematikou seznámila podrobněji.

Po mnoha konzultacích jsem se rozhodla analyzovat, navrhnout a implementovat software, který knihovnám usnadní zpracování a vyhodnocování statistických dat, pomůže jim sledovat informační potřeby čtenářů a na základě výsledků zkvalitňovat a rozšiřovat knihovní fond. Aplikaci jsem nazvala LibrarEX (Library Exchange).

1.1 Struktura práce

Motivaci pro výběr tématu diplomové práce a hlavním cílům práce je věnována kapitola 2. Práce se zabývá analýzou, návrhem a implementací aplikace, která poskytne knihovníkovi podporou při plánování investic a zkvalitňování knihovního fondu. Existují různé metodiky, které doporučují jak správně vyvíjet a řídit softwarové systémy. Volbou vhodné metodiky pro vývoj aplikace se zabývá kapitola 3.

Kapitola 4 shrnuje výsledky analýzy a z ní vycházející specifikaci požadavků. Vzhledem k tomu, jak důležité je porozumět jednotlivým požadavkům, je první část této kapitoly věnována seznámení s řešenou problematikou. Další část se zabývá samotnou specifikací požadavků a vizuálním modelováním systému. Součástí analýzy je také celkové seznámení se strukturou vstupních a výstupních dat aplikace.

Kapitola 5 popisuje architekturu navrhovaného systému, volbu technologie, obsahuje návrh vlastní databáze, podrobněji se věnuje designu jednotlivých částí systému a vybraným implementačním detailům. Způsobem testování jak během vývoje aplikace tak i výsledného systému se zabývá kapitola 6.

2 Motivace

„Veřejná knihovna je místní branou do světa vědomostí a základním předpokladem celoživotního vzdělávání, nezávislého rozhodování a kulturního rozvoje jednotlivců i společenských skupin.“
(Manifest IFLA/UNESCO o veřejných knihovnách, 1994)

Knihovny jsou důležitým zdrojem informací nezbytných pro vědeckou a výzkumnou činnost, pro práci v akademickém prostředí či rozvoj naší společnosti.

Dynamický rozvoj informačních technologií v posledních desetiletích s sebou přináší markantní nárůst objemu dat, se kterými pracují databáze, informační systémy či samotní uživatelé. Tento trend se nevyhnul ani knihovnám.

Zejména během několika posledních let dochází k rozsáhlejší digitalizaci knihoven a rozšiřování knihovního fondu o velké množství elektronických informačních zdrojů, které jsou zpřístupňovány přes webové rozhraní. Přístup k jednotlivým dokumentům je řešen na základě licenčních smluv uzavíraných mezi knihovnami a poskytovateli elektronických informačních zdrojů nebo na základě tzv. konsorciálních licencí, které zastřešují větší počet institucí sdružených za účelem společného nákupu a provozování elektronických informačních zdrojů.

Knihovny, které nakládají s vlastními rozpočty a s finančními prostředky grantových programů, zodpovídají za jejich účelné využití. Aby kvalitně uspokojily informačních potřeby, je výběr jednotlivých informačních zdrojů stěžejní. Pro optimalizaci akviziční politiky využívají knihovny různé prostředky a to jak v rámci jedné instituce, tak i v rámci spolupráce jednotlivých knihoven sdružených v konsorciích.

Mezi významné kontrolní a rozhodovací mechanismy patří i statistické rozbory. Výsledky statistických šetření týkajících se využívání informačních zdrojů se stávají nezbytným podkladem pro obhajobu investic finančních prostředků, plánování jejich dalšího využití a pro objektivní posouzení vytíženosti dostupných zdrojů. Statistická data poskytují cenné informace při rozhodování o prodloužení licencí či nákupu nového informačního zdroje.

V posledních letech se počet poskytovatelů nabízejících přehledy využívání informačních zdrojů významně zvýšil a při uzavírání licenčních smluv je často kladen požadavek na dostupnost mj. právě i již zmíněných statistických dat. Nevýhodou je, že statistické údaje jsou dostupné v různých formátech (csv, xls, txt, htm, atd.) s odlišnou strukturou dat a s podporou různých metodik jejich sledování (COUNTER, ICOLC, vlastní, atd.) v závislosti na konkrétním poskytovateli. Zpracování těchto různorodých výstupních formátů je časově velice náročné a při následném vyhodnocování získávaných údajů i – do jisté míry – omezující. A to již vůbec nebereme na zřetel možnou chybovost při opakovaném zpracování rozdílných formátů. Důležité je pracovat i s dalšími parametry, které umožní získat statistické rozbory s větší vypovídající schopností. Rozšiřující parametry mohou být

- *impakt faktor* – informuje o průměrné citovanosti časopisu,
- *počet článků* – vyjadřuje počet dostupných plných textů článků v jednotlivých časopisech,
- *počet kapitol* – uvádí počet plných textů kapitol knihy či jiné monografie,
- *cena dokumentu.*

Nevýhodou manuálního zpracování dat je také skutečnost, že existují i dokumenty přístupné od vícero poskytovatelů, tzn. z více zdrojů ve více kolekcích. Sjednocení všech potřebných údajů pro získání požadovaných výstupů je tedy velmi náročné. A pokud bychom chtěli např.

vyhodnocení investic za delší časové období, zabere zpracování příslušného objemu heterogenních dat velmi mnoho času.

2.1 Cíl práce

Cílem této práce je analyzovat, navrhnout a implementovat systém, který bude zpracovávat data o informačních zdrojích a jejich využívání, archivovat tyto data v databázi pro budoucí použití, nabízet analýzy nad těmito daty a poskytovat statistické výstupů v různých formátech. Výstupní data budou podporovat informované rozhodování při plánování investic tak, aby byla co nejlépe pokryta oblast zájmu o informační zdroje, přičemž by bylo maximálně efektivně naloženo s dostupným rozpočtem.

3 Vývoj softwarového systému

Vývoj softwarového produktu obecně zahrnuje celou řadu činností, postupů a aplikovaných metod. Je možné najít velké množství publikací, článků a skript, které se věnují různým metodikám formálního popisu vývojového procesu, které doporučují, jak správně vyvíjet a řídit softwarové projekty.

3.1 Výběr metodiky

Před zahájením práce na aplikaci, jejíž návrh a implementace je cílem diplomové práce, jsem se snažila najít metodiku, která by se pro její vývoj nejvíce hodila. Metodiku můžeme definovat jako formální popis celého vývojového procesu nebo jeho částí [10].

Při seznamování se s metodikami jsem narazila na několik názorů na výběr metodik. V knize [12] doporučuje autor pro volbu metodik pravidlo: „Vezměte to, co vám vyhovuje, a ostatní nechte ležet." V knize [10] autor zdůrazňuje, že prvotním cílem je vytvořit fungující a udržovatelný program a ne za každou cenu splnit všechny požadavky metodiky. Ne každý dokument, který metodika požaduje, má pro náš projekt význam Proto je vhodné přizpůsobit metodiku vlastním potřebám.

Rozhodla jsem se tedy použít postup, který bude pro vývoj navrhovaného systému nejvíce vyhovující. Zvolila jsem iterativní přístup, který definuje vývoj software opakováním sledu několika základních fází. V mém případě se bude jednat o čtyři fáze: 1. analýza a specifikace požadavků, 2. návrh softwarového systému, 3. implementace a 4. testování. V diplomové práci bude implementován první životní cyklus softwarového díla, který zahrnuje vývoj systému zaměřeného na zpracování a vyhodnocování statistik elektronických informačních zdrojů s připravenou podporou pro práci s tištěnými informačními zdroji. Dalo by se namítnout, že použití klasického iterativního postupu, kdy nové požadavky vznikají v průběhu vývoje, je u aplikace závisející na struktuře databáze nevhodné. Toto tvrzení bude pravdivé ve chvíli, kdy v další iteraci dojde ke změnám v datovém schématu, které vyvolají větší zásahy do předchozí verze systému. Rozšíření systému o novou funkčnost nemusí znamenat velké změny v existujícím databázovém schématu, ale spíše přidání nové databáze. V případě aplikace LibrarEX by se mohlo jednat o databázi uchovávající statistiky využívání bibliografických databází, výsledky anket, atd. Také se může jednat o rozšíření o nové reporty, atd.

V diplomové práci jsem se rozhodla použít rozšířený a často používaný univerzální jazyk pro vizuální modelování systémů, jazyk UML (Unified Modeling Language), který byl navržen, aby spojil softwarové inženýrství a existující postupy modelovacích technik. Není vázán na žádnou specifickou metodiku, což umožňuje jeho použití společně se všemi existujícími metodami. V roce 1997 sdružení OMG (Object Management Group) jazyk UML přijalo, a tím se stal prvním průmyslovým standardem objektově orientovaného jazyka pro vizuální modelování [11]. Díky standardizaci a z toho vyplývající rozšířenosti existuje i řada zajímavých nástrojů pro tvorbu diagramů. Další důvodem pro použití tohoto jazyka, je snadná dostupnost dostatečného množství odborné literatury.

4 Analýza a specifikace požadavků

Tato kapitola popisuje procesy a postupy, které v knihovnách při akvizici probíhají, a dále uživatelské požadavky, které je potřebné pro podporu těchto procesů implementovat. Požadavky byly průběžně konzultovány s uživateli, kteří budou program používat.

Sekce 4.2 a 4.3 popisují současnou podobu zpracování a vyhodnocování statistických dat a porovnávají tento způsob řešení s řešením navrhovaným.

V sekci 4.4 je uveden výsledný seznam obecných a funkčních požadavků na systém. Funkční požadavky jsou vizualizovány pomocí tzv. diagramu uživatelských případů (usecase), který popisuje základní funkčnost programu.

Sekce 4.5 popisuje vybrané části systému pomocí vizuálního modelování.

Důležitou součástí analýzy databázové aplikace je důkladné seznámení se se vstupními daty. Jejich struktuře se věnuje sekce 4.6 a struktuře výstupních dat sekce 4.7.

Sekce 4.8 seznamuje s výsledkem hledání existujícího software řešícím podobnou problematiku. Kladla za cíl zjistit, zda existuje program, který je pro řešení dané problematiky funkčně dostačující. Dále jsem chtěla najít inspiraci pro návrh funkcí vlastní aplikace a identifikovat nedostatky, kterým bych se chtěla vyhnout.

4.1 Seznámení s problematikou

Na začátku práce bylo důležité seznámit se s principem fungování knihovny a nalézt základní ukazatele mapující její činnost – jaké zdroje jsou využívány, jak jsou zpřístupňovány dokumenty, podle čeho se rozhoduje o investicích do informačních zdrojů, jaké existují typy licenčních smluv, jakým způsobem se sleduje využívání informačních zdrojů, jaká data o využívání informačních zdrojů mají a mohou mít knihovny k dispozici, co ovlivňuje rozhodování knihovníka při rozšiřování a zkvalitňování knihovního fondu, atd.

Během mnoha sezení se zaměstnanci knihoven jsem získala informace, z nichž vyplynuly požadavky na aplikaci a na její návrh aplikace. Následující sekce se budu věnovat právě těmto informacím. V sekci 4.1.1 je vysvětleno několik základních pojmů, které pomohou v porozumění textu práce. Další sekce 4.1.2 je zaměřena na akvizici a aspekty ovlivňující rozhodování knihovníka při doplňování knihovního fondu. V sekci 4.1.3 je rozebrána problematika konsorcií a licenčních smluv, které jsou neopomenutelnou součástí akvizice a současného knihovního světa. Sekce 4.1.4 se věnuje seznámení s informačními zdroji a jejich rozdělení podle různých kritérií, které vymezuje několik různých pohledů na tyto zdroje. Seznámení s dostupnými daty, které lze využít pro statistické rozbory, se bude věnovat sekce 4.1.5. Poslední sekce této části 4.1.6 je zaměřena na představení exitujících standardů a metodik pro sledování využívání elektronických informačních zdrojů.

4.1.1 Slovník pojmů

Knihovna – Kulturní, informační a vzdělávací instituce, která shromažďuje, zpracovává a uchovává organizovanou sbírku dokumentů a poskytuje knihovnické a informační služby uživatelům [1].

Vysokoškolská knihovna – Specializovaná knihovna vysoké školy, která získává, zpracovává a zpřístupňuje dokumenty s cílem informačního a dokumentačního zabezpečení výuky a vědeckovýzkumné činnosti. [1].

Informační zdroj – Informační objekt, který obsahuje dostupné informace odpovídající informačním potřebám uživatele. Informační zdroj může být tištěný, zvukový, obrazový nebo elektronický (včetně zdrojů dostupných online) [1].

Elektronický informační zdroj – Informační zdroj, který je uchováván v elektronické podobě a je dostupný v prostředí počítačových sítí nebo prostřednictvím jiných technologií distribuce digitálních dat (např. na discích CD–ROM). V bibliografickém popisu elektronických zdrojů se používá tohoto termínu pro obecné označení druhu dokumentu [7].

Elektronický archiv – Organizovaná sbírka digitálních dokumentů, shromážděná za účelem jejich dlouhodobého uchování. Může se jednat o digitalizované dokumenty, tj. tištěné druhy dokumentů převedených do digitální podoby, nebo o dokumenty vytvořené již jako digitální [7].

Impakt faktor – Významný koeficient citační analýzy, který je dnes používán k hodnocení důležitosti či kvality vědeckých časopisů. Tento koeficient je definován jako poměr počtu citací, které byly zaznamenány v hodnoceném roce na všechny články publikované v daném časopise za předchozí dva roky, k počtu všech článků publikovaných v těchto dvou letech. Například výpočet impakt faktoru časopisu pro rok 2006 bude vypočítán následovně:

$$\textit{Impakt faktor 2006} = \frac{\text{Počet citací v roce 2006 na články daného časopisu publikované v roce 2004 a 2005}}{\text{Celkový počet článků publikovaných v daném časopise v roce 2004 a 2005}}$$

Akvizice dokumentů – Doplňování knihovního fondu. Zahrnuje zjišťování, vybírání a získávání dokumentů do fondu knihovny s ohledem na funkce, zaměření a poslání knihovny, obsahovou strukturu jejího fondu (profil knihovního fondu) a potřeby jejích uživatelů. K získávání fyzických dokumentů do fondu knihovny dochází čtyřmi způsoby: povinný výtisk, nákup, dar, výměna [6].

Knihovní jednotka – Dokument získaný knihovnou na zvláštním fyzickém nosiči za účelem plnění primárních funkcí knihovny [8].

Knihovní fond – Odborně knihovnicky zpracovaná, uložená a zpřístupňovaná sbírka knihovních jednotek v určité knihovně [1].

Metoda Konspektu – Souborná charakteristika fondu určité knihovny bez podrobných bibliografických informací. Užívá standardizovaný formát a metodiku (kódy a indexy) charakterizující tematickou strukturu knihovního fondu [1]. Základní hierarchie: 24 předmětových kategorii, 500 skupin konspektu, 4000 předmětů.

Konsorcium – Konsorciem se rozumí skupina několika institucí (např. vysokoškolské knihovny), které se sdružily za účelem společného nákupu a provozování elektronických informačních zdrojů. Konsorcium nemá právní subjektivitu a může být zcela neformální nebo podepřené např. společnou smlouvou o spolupráci [4].

Licenční smlouva – Smlouva, kterou nositel práva – poskytovatel licence – poskytuje nabyvateli licence oprávnění k výkonu práva dílo užít (licenci). Jedná se tedy o právo autorské dílo užívat nebo s ním nakládat v rozsahu a podle podmínek stanovených smlouvou. Nabyvatel licence se zavazuje poskytnout nositeli práva odměnu. Licence je poskytována jako

výhradní nebo nevýhradní, k jednotlivým způsobům užití nebo ke všem způsobům užití, s právem poskytnout oprávnění zcela nebo zčásti třetí osobě (podlicence) nebo bez tohoto práva. Licenční smlouva musí obsahovat ujednání o výši odměny, případně ujednání o bezplatnosti licence [1].

Široké spektrum termínů knihovnictví a informační vědy nabízí volně dostupný rozsáhlý výkladový slovník Informační věda a knihovnictví[1]. Případně je možné využít i služby českých knihoven "Ptejte se knihovny"[2].

4.1.2 Akvizice – knihovník

Doplňování a zkvalitňování knihovního fondu je jedním z nejdůležitějších úkolů informační instituce. V současné době nepředstavuje rozšiřování knihovního fondu pouhý jednorázový nákup informačního zdroje, ale stále častěji se objevuje také forma investic v podobě předplaceného přístupu k informačním zdrojům pro konkrétní období. Tato varianta nabízí větší možnosti obměny a zkvalitňování knihovního fondu na základě aktuálních a průběžných informačních potřeb čtenářů.

Akviziční pracovník by měl znát potřeby čtenářů, provoz celé knihovny, složení fondu, zdroje dokumentů, v současné době by se měl také dobře orientovat na internetovém trhu informačních zdrojů a v neposlední řadě by měl být i dobrým ekonomem.

Práce s elektronickými informačními zdroji se již stala běžnou součástí odborných knihovnických činností. Byla realizována velká řada projektů, jejichž cílem bylo usnadnit a sjednotit přístup k elektronickým zdrojům, byly vytvořeny rozsáhlé oborově zaměřené fulltextové databáze, atd. Mezi nejvýznamnějšími projekty patří VPK[3], STM[4], JIB[5], PEC[6] a dále Infozdroje.cz[7] či „Informace pro knihovny“[8].

S elektronickými informačními zdroji přicházejí i nové možnosti akvizice. Mezi jedny z nejdůležitějších patří vznik konsorcií. Smysl vzniku konsorcií spočívá ve vzájemné spolupráci knihoven při nákupu a získávání výrazně širšího spektra informačních zdrojů za finanční částky, které jsou pro jednotlivé knihovny přijatelné. Díky tomuto trendu dochází v knihovnách k ukončení nákupu některých dokumentů v tištěné podobě, které byly pro knihovnu finančně zatěžující a jejichž elektronická podoba je dostačující. Formy konsorcií mohou být různé, stejně jako způsoby zpřístupnění dokumentů a jejich financování.

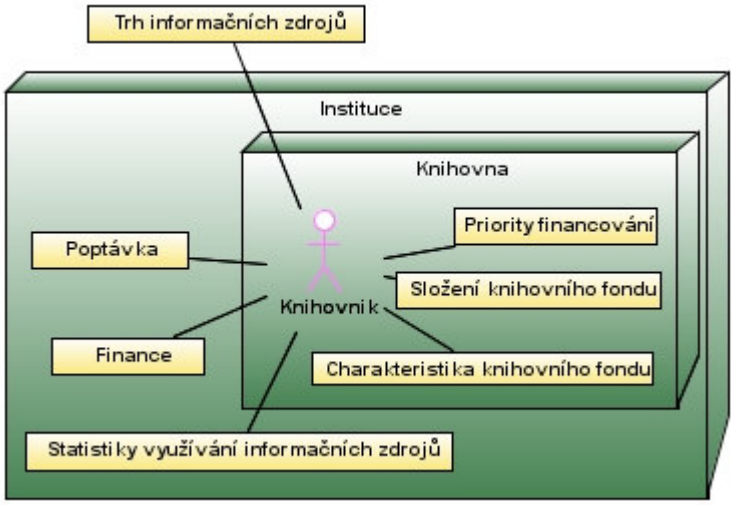
Další novou možností akvizice je předplácení/nákup tzv. balíčků. Jedná se o soubor dokumentů, které jsou poskytovatelem nabízeny jako jeden celek za zvýhodněnou cenu ve srovnání s cenami jednotlivých dokumentů. Nejvíce je využíván nákup elektronického archivu.

Dostupnost informačních zdrojů se za poslední roky velice zlepšila, ale přináší s sebou i negativní aspekt, který se týká kvality poskytovaných informací. V prostředí Internetu se vedle kvalitních a užitečných informačních zdrojů nalézají též zdroje poskytující neúplné, nedostatečné a někdy dokonce i zavádějící informace. Jedním z důležitých úkolů knihovníka je proto zajistit uživatelům přístup ke spolehlivým a relevantním informacím.

[1] http://vydavatelstvi.vscht.cz/informace/uid_isbn-80-7080-599-4_cze.html
[2] http://www.ptejteseknihovny.cz
[3] http://www.vpk.cz
[4] http://www.portalstm.cz
[5] http://www.jib.cz/V?RN=271773895
[6] http://pec.cuni.cz
[7] http://www.infozdroje.cz
[8] http://knihovnam.nkp.cz

Jak je vidět, pohled akvizičního pracovníka se během několika posledních let musel rozšířit o sledování a pochopení nových forem akvizice a jejich výhod/nevýhod. Naštěstí s příchodem a rozvojem nových technologií, získávají knihovníci i nové nástroje, které jim práci usnadňují.

Na obr. 1 je zobrazeno, co vše musí akviziční pracovník brát v úvahu.



obr. 1: Vlivy na nákupní chování knihovníka

Úkolem profilování knihovního fondu je zajištění a stanovení optimální struktury a obsahu fondu. Pracovník akvizice vychází z následujících faktorů:

- ***Stanovení profilu knihovny*** (tematická a druhová struktura fondu, rozsah fondů, důležitou roli hraje i provenience dokumentů, institucionální, autorské a časové hledisko).
- ***Metody hodnocení fondů*** (klasifikační, dokumentační a citační analýza, sledování a vyhodnocování statistik zájmů čtenářů o jednotlivé tituly).
- ***Sledování poptávky po konkrétních publikacích*** (anketa, hodnocení využití mezi–knihovní výpůjční služby, atd.).

Pro detailnější seznámení s uvedenými faktory a konkrétními metodami doporučuji elektronický informační zdroj [2]. V následujícím textu se věnuji vybraným faktorům, jejichž znalost je důležitá pro hodnocení investic do informačního zdroje.

- ***Priority financování informačních zdrojů*** – Každá knihovna, včetně své uživatelské obce a dalších složek, které ji tvoří, je jedinečná. Proto i ustanovení týkající se strategie budování fondů jsou jedinečná a měla by odrážet úlohu konkrétní knihovny i charakter uživatelské obce, které slouží. Jednotlivé knihovny si vypracovávají vlastní metody hodnocení fondů, z kterých pak stanoví priority financování dokumentů. Dokumenty s vyšší prioritou budou zařazovány do fondu knihovny přednostně. Nastavení těchto priorit v ústřední knihovně VŠCHT Praha uvádím jako příklad:
    - Priorita 1: Zabezpečení informační potřeby celoškolských předmětů a ostatních akreditovaných předmětů
    - Priorita 2: Zajištění informační potřeby strategických vědeckých a výzkumných programů školy – sledování moderních trendů vývoje oboru

- Priorita 3: Kompletnost sbírek – systematické budování souborů významných referátových, encyklopedických a seriálových děl
- Priorita 4: Ostatní informační zdroje, které nespadají do předchozích bodů

- ***Finanční prostředky*** – Určujícím faktorem objemu akvizice zůstává rozpočet knihovny a stanovení priorit využití financí. Např. vysokoškolské knihovny disponují finančními prostředky vymezenými z rozpočtu dané vysoké školy, kromě toho mohou získat významné finanční prostředky v rámci různých projektů, programů a grantů.

- ***Dostupnost informačního zdroje*** – Míra vynaloženého úsilí pro získání dokumentu může být pro čtenáře jedním z rozhodujících faktorů. Většina uživatelů v dnešní době využije spíše dokumenty přístupné z místa pracoviště a teprve ve chvíli, kdy dostupné dokumenty neposkytují požadované informace, jsou ochotni podniknout další kroky jako např. navštívit knihovnu osobně. Čím bude získání dokumentu obtížnější, tím bude počet čtenářů ochotných podniknout potřebné kroky k jeho získání menší. Tento aspekt by měl brát v úvahu knihovník při rozšiřování a zkvalitňování knihovního fondu a položit si při rozhodování otázky jako např.: Bude dostupnost časopisu po zrušení předplatného rozumná? Jak nákladné bude pro čtenáře získání článku tohoto časopisu? atd.Vzhledem k jedinečnosti knihoven je definování „rozumné dostupnosti“ různé. Jako příklad uvádím priority dostupnosti dokumentu pro ústřední knihovnu VŠCHT Praha.
  - Priorita 1: Dostupnost v síti VŠCHT, neomezený počet současně pracujících
  - Priorita 2: Dostupnost v síti VŠCHT, omezený počet současně pracujících
  - Priorita 3: Dostupnost v prostorách ústřední knihovny VŠCHT
  - Priorita 4: Dostupnost v jiné knihovně vzdálené do 10min od ÚK VŠCHT
  - Priorita 5: Dostupnost online po zaregistrování se v jiné knihovně
  - Priorita 6: Dostupnost prostřednictvím mezi–knihovní výpůjční služby z ČR
  - Priorita 7: Dostupnost prostřednictvím mezi–knihovní výpůjční služby ze zahraničí
  - Priorita 8: Finančně náročné získání dokumentu

4.1.3 Konsorcia

Knihovny využívají možnost uzavírat konsorciální licenční smlouvy pro přístup k elektronickým informačním zdrojům. Konsorciální přístup přináší knihovnám několik výhod, ale i nevýhod. Mezi výhody konsorcia patří lepší licenční podmínky, nižší ceny, možnost získat grantové podpory, přístup k širšímu spektru informačních zdrojů. Mezi nevýhody patří např. fakt, že dokumenty zpřístupněné v rámci konsorcia nebudou po jeho skončení již déle k dispozici. Další nevýhodou může být zdánlivě dobrá investice do velkého množství dokumentů, ale průměrná využitelnost všech jednotlivých předplacených dokumentů může být výrazně nižší.

Typy konsorciálních licenčních smluv se liší ve způsobu a množství zpřístupněných informačních zdrojů. Mezi nevýznamnější typy konsorciálních licenčních smluv patří:

- ***Multilicence pro přístup členů konsorcia do všech plných textů vydavatelství***

- ***Multilicence pro přístup do plných textů všech titulů předplácených členy konsorcií*** **–** tento přístup je nazýván jako „*křížový přístup*„, kdy každý člen odebírá jím vybrané tituly a navíc získává přístup k titulům předpláceným ostatními členy konsorcia.

- ***Multilicence pro přístup do vybrané kolekce*** – jedná se většinou o přístup do vydavatelem definované, obvykle subjektové kolekce nebo o přístup do kolekce titulů, kterou si vytvořil sám zákazník. Jako příklad může sloužit smlouva, kterou uzavřely knihovny s vydavatelstvím Elsevier – byly vytvořeny jednotlivé kolekce podle oborových zaměření (UTL1 až UTL5). Pro jednotlivé kolekce byl vypočítán váhový klasifikátor, který se používá dále pro výpočet ceny za danou kolekci. Každá knihovna, která chce přistupovat do UTL kolekce musí předplácet časopisy ve stanovené výši nebo zaplatit odpovídající částku vypočítanou na základě několika faktorů (velikost instituce, atd.). Tato částka se během přístupného období navyšuje každoročně o stanovená procenta. Knihovna má přístup k časopisům UTL kolekce, která je vytvořena uměle, a dále má přístup k časopisům, které předplácí, aby splnila požadavky na přístup do UTL kolekce. Po skončení konsorcia bude mít knihovna nadále přístupné časopisy, které předplácela, ale nebude mít přístupné časopisy, které byly v dané UTL kolekci. To se pak řeší nákupem archivů.

- ***Národní licence*** – neomezený přístup do plných textů všech titulů vydávaných jedním vydavatelstvím pro všechny definované subjekty na určitém geografickém území.

Licence bývají odlišné i vzhledem k dojednaným podmínkám mezi konkrétním konsorciem a vydavatelem. Licenční smlouvy se uzavírají na různá období od jednoho roku až po několik let. Pokud se jedná o smlouvu uzavřenou na více let, je již v této smlouvě stanoven meziroční nárůst cen, což může být jak výhodou, tak i nevýhodou. V rámci konsorcia je pak pro jednotlivé jeho členy stanovena výše předplatného. Tato výše je většinou proměnlivá nejčastěji v závislosti na velikosti konkrétního člena konsorcia.

Jedním z důležitých aspektů při dojednávání licenčních smluv je dostupnost statistických dat o využívání elektronických informačních zdrojů. Tyto statistiky mohou být dále využity např. při nákupu nových titulů či při obměně stávajících na základě jejich skutečné potřeby, nikoliv na subjektivním rozhodování podle přání jednotlivců.

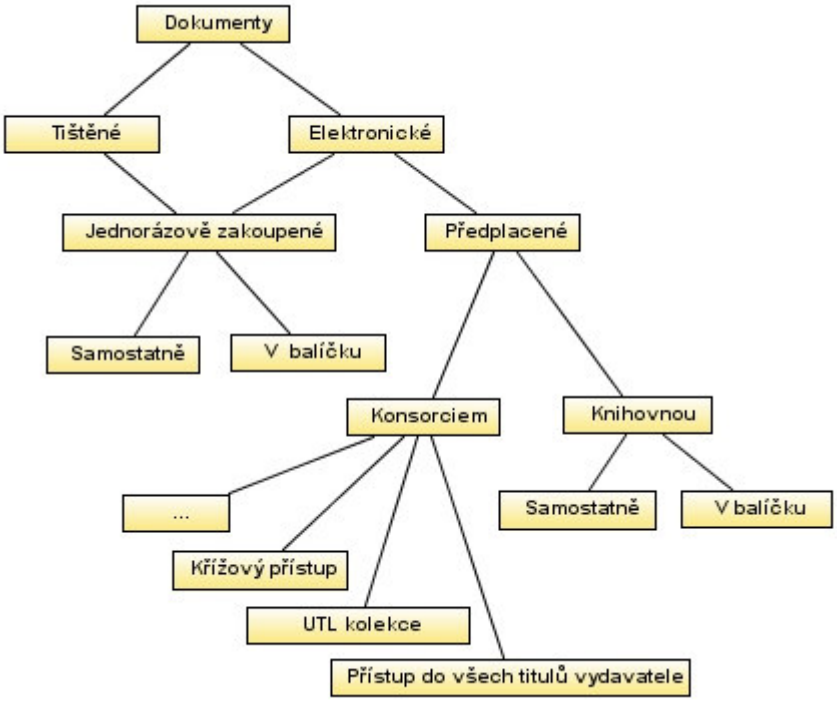
Dalším z důležitých aspektů je také počet současně pracujících uživatelů. Čím je tento počet vyšší, tím zaplatí instituce větší poplatky za zpřístupnění zdrojů. Je tedy důležité správně stanovit, kolik bude potřeba současných přístupů, a regulovat tak celkové finanční náklady. O nastavení počtu současných přístupů k danému zdroji je třeba se rozhodovat podle skutečného a statisticky prokázaného zájmu.

Knihovny se stávají členy řady konsorcií, což jim umožňuje přístup k velkému spektru elektronických, oborově zaměřených periodik. V České republice je významným správcem konsorciálních přístupů do fulltextových databází z celého světa společnost Suweco. V rámci této společnosti byla vytvořena významná konsorcia jako např. Elsevier Science[9] s 39 členy, John Wiley and Sons[10] s 40 členy, Springer Verlag[11] se 71 členy a Blackwell[12] s 26 členy. Tato konsorcia mají uzavřenou licenční smlouvu do roku 2008 nebo 2009. Po skončení každé licenční smlouvy se knihovny musí rozhodovat, zda nakoupí jednotlivé tituly, nebo se sdruží opět do konsorcia. Při rozhodování budou velice důležitým faktorem mj. právě i již zmiňované statistické rozbory, které ukáží, zda členství v konsorciu bylo pro knihovnu výhodné, nebo zda by bylo výhodnější předplácet jednotlivé tituly.

[9] http://www.suweco.cz/cz/online-konsorcium-elsevier.asp
[10] http://www.suweco.cz/cz/online-konsorcium-wiley.asp
[11] http://www.suweco.cz/cz/online-konsorcium-springer.asp
[12] http://www.suweco.cz/cz/online-konsorcium-blackwell.asp

4.1.4 Informační zdroje

Informační zdroje lze rozdělit podle základních kategorií, typů, tématicky atd.. Pro návrh aplikace je nutné zohlednit mj. i dále uvedené a popsané faktory, které hrají při vyhodnocování a dalším zpracování významnou roli. Jejich váha je, z pohledu knihovny jakožto investora, dostatečným důvodem, proč bychom se na ně měli zaměřit. Rozdělení informačních zdrojů je zobrazeno na obr. 2.



obr. 2: Základní rozdělení informačních zdrojů

Dokumenty v knihovně můžeme rozdělit podle několika kritérií. Jejich výběr je činěn s přihlédnutím k specifikaci řešeného problému.

Rozdělení podle typu dokumentu

- ***tištěné dokumenty*** – jedná se o dokumenty, které jsou v knihovně uloženy a připraveny k vypůjčení v papírové podobě.
- ***elektronické dokumenty přístupné přes Internet*** – dokumenty přístupné online, na základě licenční smlouvy. Přístup k dokumentu je řešen podle IP adresy instituce nebo přihlášením do zabezpečených stránek.
- ***elektronické dokumenty přístupné z hmotných nosičů*** – jedná se o dokumenty zakoupené např. na CD, DVD a přístupné na počítačích v prostorách knihovny.

Rozdělení podle typu vlastnictví dokumentu

- ***koupené dokumenty*** – dokumenty, které má knihovna v trvalém vlastnictví.
- ***předplacené dokumenty*** – dokumenty, které má knihovna přístupné jen na určité období na základě licenční smlouvy.

Rozdělení podle způsobu nákupu dokumentu

- ***tituly koupené či předplácené jednotlivě***
- ***tituly koupené či předplácené v rámci balíčku*** – příkladem dokumentů koupených v rámci balíčku může být nákup elektronického archivu (backfile). Tento typ balíčku je sestaven přímo poskytovatelem. Dalším příkladem investování do balíčku dokumentů může být konsorciální přístup k dokumentu. U dokumentů zakoupených v rámci balíčku není důležitá cena jednotlivých dokumentů, ale cena celého balíčku Při zvažování investic pak musíme vyhodnotit, co bude cenově výhodnější.

Rozdělení podle typu licenční smlouvy

- ***nekonsorciální licenční smlouva*** – smlouva uzavřená mezi jednotlivou knihovnou a poskytovatelem.
- ***konsorciální licenční smlouva*** – konsorciální licence je uzavřena mezi poskytovatelem a konsorciem.

Rozdělení podle typu konsorciálního přístupu

- ***přístup do všech titulů vydavatele***
- ***křížový přístup*** – každý člen konsorcia nakoupí jednotlivé tituly, které jsou pak přístupné ostatním členům konsorcia
- ***UTL kolekce*** (vysvětleno v kapitole 4.1.3)

4.1.5 Statistická data

Sledování statistických údajů je důležité pro všechny strany účastnící se procesu tvorby, zpřístupňování a nákupu informačních zdrojů. Statistická data jsou důležitá nejen pro knihovny a instituce, které poskytují finanční zajištění nákupu a provozu informačních zdrojů, ale i pro vydavatele a poskytovatele informačních zdrojů.

Statistická data poskytují zpětnou vazbu o využívání vkládaných finančních prostředků a jsou podkladem pro další finanční plánování. Jsou užitečným nástrojem pro zjišťování informačních potřeb a poskytují cenné informace v rozhodovacích procesech týkajících se prodloužení licencí či nákupu nového informačního zdroje. Samozřejmě platí i opačný postoj, kdy tato data mohou být podkladem pro vyřazení titulu z nabídky konsorcia (knihovny), a jsou taktéž velmi užitečná pro další vyjednávání s poskytovateli nabídek. Hodnoty statistických ukazatelů mohou knihovny informovat i o zkvalitnění informační výchovy uživatelů či o potřebě lepší propagace některých informačních zdrojů.

Důležitost statistických dat si uvědomuje stále více poskytovatelů, vydavatelů i knihoven. Při uzavírání licenčních smluv mezi knihovnou, konsorciem a poskytovatelem je požadavek na poskytování statistických dat o míře využití elektronických informačních zdrojů stále častější. Dle mezinárodních doporučení jsou již dnes tato data jedním z velmi důležitých bodů rozhodovacího procesu. Čím dál častěji mohou při rozhodování o zakoupení zdroje (balíčku, jednotlivé publikace, dokumentu) být oním pověstným zrnkem na misce vah.

Zajímala jsem se o dostupnost statistických dat v několika knihovnách.

Pro statistické rozbory jsem se rozhodla využít tato statistická data:

- využívání elektronických informačních zdrojů
- využívání tištěných informačních zdrojů
- cena jednotlivých dokumentů, cena balíčků
- impakt faktor časopisů
- počet přístupných článků časopisu, počet přístupných kapitol knih

Využívání elektronických informačních zdrojů

Důležitým zdrojem statistických dat jsou přehledy využívání elektronických informačních zdrojů získané od jednotlivých poskytovatelů.

Mezi sledované statistické veličiny jsem zařadila:

- počet přístupů – udává potenciální zájem o informační zdroj.
- počet zobrazení/stažení plných textů – ilustruje skutečný přínos pro uživatele. Tento údaj dává důležitou zpětnou vazbu o využívání informačního zdroje.
- počet hledání – odpovídá počtu zobrazení plných textů v režimu vyhledávání. Tento údaj odráží skutečný zájem o informační zdroj.
- počet zobrazení/stažení abstraktů – odráží zájem uživatelů o danou oblast. Např. pokud uživatelé navštíví velké množství abstraktů daného dokumentu a přitom počet stažení plného textu tohoto dokumentu je malý, může to svědčit např. o nevhodné formě abstraktu, jeho nedostatečné vypovídací schopnosti nebo jiných faktorech, díky nimž přestane být článek pro uživatele zajímavý nebo užitečný.
- počet zobrazení/stažení obsahů dokumentů.
- počet zobrazení/stažení dle typu dokumentu (PDF, HTML, atd.) – je to jen orientační údaj, který je důležitý hlavně pro vydavatele.

Knihovník tak dostane informaci, že v rámci daného poskytovatele bylo např. během ledna 2007 x_1 přístupů k plným textům článků konkrétního časopisu. Z toho bylo zobrazeno x_2 článků ve formátu PDF, x_3 článku ve formátu HTML a x_4 článků v jiných formátech (tzn. $x_1 = x_2 + x_3 + x_4$). Dále knihovník zjistí, že x_5 článků daného časopisu bylo zobrazeno vyhledáváním a x_6 „procházením“ stránek (tzn. $x_1 = x_5 + x_6$). Mezi statistikami může být i údaj x_7 o počtu zobrazených abstraktů článků sledovaného časopisu a x_8 o počtu zobrazených obsahů daného časopisu.

Sledované statistické veličiny se mohou u různých poskytovatelů lišit, stejně jako formát, ve kterém jsou tato data poskytnuta uživatelům – knihovnám. Existují i časopisy, jejichž články nabízí více poskytovatelů, protože jsou dostupné v rámci několika kolekcí. Pro hodnocení těchto časopisů je důležitá centralizace získaných statistických údajů, která sjednocuje informace o využívání časopisů. Statistická data jsou vyhodnocována většinou poskytovatelů jednou měsíčně.

Problematika využívání elektronických informačních zdrojů se řeší již několik let. V ČR byla již v roce 2002 v rámci programu L1[13] provedena analýza dostupných statistik informačních

[13] http://www.stk.cz/li01018

zdrojů. Při průzkumu statistik se objevily problémy způsobené tím, že poskytovatelé často dodávali data neporovnatelná, v nevhodných formátech, v nedostatečné míře nebo jinak nevyhovující. Jako řešení bylo navrženo využití standardu Code of Practice[14] (dále jen CoP) , který byl v té době vyvíjen a jehož první verze byla vydána mezinárodní iniciativou Project Counter v lednu 2003.

V současné době se situace ohledně poskytnutých statistických dat o využívání elektronických informačních zdrojů zlepšuje. Větší poskytovatelé již podporují, nebo začínají podporovat, standard CoP, čím se kvalita a porovnatelnost statistických dat výrazně zlepšuje.

Statistická data jsem získala od Ústřední knihovnou VŠCHT Praha. Spolu s vedoucí Phdr. Součkovou jsme požádali velkou část dodavatelů o zpřístupnění statistických dat a většina z nich nám tyto informace poskytla. Získali jsme data o využívání elektronických časopisů od 14 hlavních poskytovatelů (pokrývají 96% zpřístupněných elektronických časopisů), data o využívání elektronických knih od 6 poskytovatelů (pokrývají 100% zpřístupněných elektronických knih). Z 80% je podporován standard CoP.

Využívání tištěných informačních zdrojů

U tištěných informačních zdrojů jsou statistické přehledy nejčastěji získávány z knihovního systému. Důležitými statistickými veličinami jsou

- počet výpůjček
- počet rezervací
- počet prodloužení
- počet požadavků na kopii
- počet požadavků na výpůjčku

Impakt faktor

Spektrum časopisů na našem trhu se neustále rozšiřuje. Impakt faktor se stal určitým měřítkem kvality vědeckého časopisu. Slouží k hodnocení časopisů na základě citovanosti.

Seznam impakt faktorů jednotlivých časopisů je pravidelně vydáván. V ČR je k dispozici např. v základní knihovně Akademie věd ČR [9]. Seznam impakt faktorů časopisů je dostupný v databázi Web of Sience – Journal Citation Reports.

Cena

Dalším důležitým ukazatelem pro statistické rozbory je cena. Pokud se zajímáme o cenu, musíme rozlišovat cenu koupeného informačního zdroje od ceny předpláceného dokumentu. U předplatného se stanovuje cena vždy pro celý běžný rok, tzn. pokud dojde ke změně ceny, projeví se tato změna až na přelomu předplatného období.

Pokud budeme znát cenu dokumentů/balíčků, můžeme sledovat cenu stažení plného textu a zjistit, který z dokumentů/balíčků je pro knihovnu finančně méně či více výhodný. Do

[14] http://www.projectcounter.org/code_practice.html

vyhodnocování je ale nutné zapojit ještě další faktory, zejména pokud se knihovna rozhoduje o předplácení nebo naopak o jeho zrušení.

U publikací vycházejících po delší časové období (např. časopisy) je možné ceny v aktuálním roce a další podrobné informace získat v databázi Ulrich's Periodicals Directory[15].

Počet článků, počet kapitol, počet stránek

Také rozsah informačních zdrojů je důležitý pro statistické rozbory. Počet stažení plných textů informačního zdroje odpovídá počtu stažení jednotlivých článků, kapitol. Pokud informační zdroj bude menšího rozsahu, celkový počet stažení článků může být sice menší než u rozsáhlejšího časopisu, ale průměrný počet stažení na článek může být naopak výrazně vyšší. Parametry informující o rozsahu dokumentu tedy umožní porovnávat a vyhodnocovat nové údaje, jako je např. průměrná cena stažení článku, nebo jen průměrná cena článku.

4.1.6 Standardy, projekty

K nejvýznamnějším skupinám, které se zajímají o problematiku a to, jak jsou elektronické informační zdroje využívány, lze zařadit COUNTER[16], ICOLC[17], ARL E–metrics[18] a projekt NISO Standard Z39.7[19].

COUNTER

Counting Online Usage of Networked Electronic Resources (COUNTER) je mezinárodní projekt pracující a udržující standard pro měření zájmu o online–informační zdroje, zaměřený na tvorbu a poskytování důvěryhodných, konzistentních a kompatibilních statistik [5]. Projekt je postaven na již existujících projektech (ICOLC, ARL e–metrics) a výsledkem je vznik mezinárodního standardu Code of Practice (CoP), který jednoznačně definuje požadavky, jak vytvářet uživatelské statistiky a usnadňuje záznam, výměny a interpretace dat o využívání online informačních zdrojů [3].

ICOLC

International Coalition of Library Consortia (ICOLC) je mezinárodní sdružení, které vzniklo za účelem snadnější komunikace institucí na téma problematiky konsorcií [3]. V rámci sdružení je též řešena problematika využívání on–line přístupných informačních zdrojů. Ve směrnicích ICOLC byly definované minimální požadavky na dodávání uživatelských statistik. Tyto směrnice jsou však pouze doporučeními, nejedná se o mezinárodní standard.

4.2 Současné řešení

Zaměřím se na současnou situaci ve zpracování statistických dat o elektronických informačních zdrojích. Abychom získali užitečné statistické rozbory s dobrými vypovídacími vlastnostmi je nutné sledovat nejen data poskytovaná jednotlivými vydavateli, ale i další údaje vypovídající o dokumentu (cena dokumentu, impakt faktor, počet článků, počet kapitol, atd.).

[15] http://www.ulrichsweb.com/ulrichsweb
[16] http://www.projectcounter.org
[17] http://www.library.yale.edu/consortia/
[18] http://www.arl.org/stats/initiatives/
[19] http://www.niso.org/emetrics/

V současné době se pracuje jen se statistickými daty získanými individuálně od jednotlivých poskytovatelů. Protože jednotlivé přehledy využití elektronických informačních zdrojů má knihovna od jednotlivých poskytovatelů zvlášť, v různých formátech, s odlišnými sledovanými položkami, je pro knihovníka velmi komplikované získat rozumné propojení a následné vyhodnocení těchto údajů v rámci více poskytovatelů se stává prací vskutku sysifovskou. Z toho důvodu se v současné době vyhodnocuje pouze zájem o dokumenty konkrétního poskytovatele, kdy se bere v úvahu pouze 10 až 20 časopisů s největším počtem stažených plných textů a několik dokumentů s nejmenším počtem stažených plných textů. Ale i tato vyhodnocení jsou velmi pracná, zejména v případech, kdy se nejedná o formát (např. xls) nabízející třídění dat podle zvoleného kritéria.

Pokud by měl knihovník při současném způsobu zpracování dat zapojit do vyhodnocování i zmíněné další statistické údaje a další aspekty jako např. nákup v rámci balíčku, konsorcia atd., tak se před nim objevuje velice obtížný a časově náročný úkol.

Následující výčet shrnuje nevýhody současného řešení:

- velká obtížnost zpracování údajů dostupných v různých, vzájemně neporovnatelných formátech
- velká obtížnost zpracování dat podporujících různé metodiky pro sledování statistik
- vysoká časová a manuální náročnost zapojení dalších důležitých položek pro statistické rozbory
- velká obtížnost vyhodnocení všech elektronických dokumentů nezávisle na poskytovateli
- nepřehlednost velkého množství vzájemně nekompatibilních dat
- nesleduje se využitelnost nakoupených balíčků, konsorciálních přístupů
- nemožnost vyhodnotit efektivitu nákupu

4.3 Navrhované řešení

Cílem navrhovaného řešení je odstranit, nebo co nejvíce zmírnit nevýhody současného způsobu zpracování a vyhodnocování statistických dat o využívání informačních zdrojů.

Hlavní nevýhodou současného řešení je obtížnost zpracování dat dostupných v různých formátech, někdy i s odlišnými statistickými údaji. Aplikace LibrarEX umožní uživateli zpracování těchto dat omezit až na „jedno kliknutí tlačítka“, které spustí proces převodu datových souborů do jednotné podoby a jejich následný import do databáze. V tomto případě se jedná o spuštění hromadného úkolu, který může začít stažením dat z Internetu, jejich uložením na lokální disk, pokračovat jejich zpracováním a skončit jejich importem do databáze. Uživatel aplikace může využít i zpracování jednotlivých souborů se vstupními daty, kdy po jejich převodu do jednotné podoby lze v dialozích aplikace provést kontrolu načtených dat a vlastní import těchto dat může provést později, až sám uzná za vhodné.

Aplikaci lze snadno rozšířit o podporu nových formátů prostřednictvím skriptovacího jazyka. Skripty zpracovávají soubory s daty a ukládají je do jednotné podoby, do XML souboru. Je nutné definovat, jaký skript bude použit pro vygenerování konkrétního XML souboru, a jaké parametry budou skriptu předány. Tyto údaje jsou uloženy ve filtrech.

Aplikace umožňuje kromě statistických dat získaných od poskytovatelů publikací vkládat i data obsahující informace o ceně, impakt faktoru a základní data o jednotlivých dokumentech, poskytovatelích, vydavatelích, typu přístupu, atd. Tato data mohou být zpracována nezávisle na sobě.

Aplikace umožňuje následně hodnotit investice do dokumentů podle zvolených kritérií. Výsledné hodnocení dokumentů může být dále použito pro výpočet efektivity nákupu a plánování nových investic.

Uložení dat v databázi navíc řeší problém s nepřehlednosti velkého množství dat v mnoha souborech různých formátů.

Ve chvíli, kdy jsou kompatibilní a vzájemně porovnatelná data uložena v databázi, může uživatel vytvářet potřebné reporty. Pomocí vhodné volby parametrů reportu precizněji specifikuje oblast zájmu. Dále je může uživatel procházet podle několika pohledů. např. data seskupená podle jednotlivých poskytovatelů, podle typu dokumentů, atd.

4.4 Požadavky

Požadavky zachycují, co by měl systém dělat.

Rozlišujeme dva typy požadavků:

- Obecné požadavky – specifikují vlastnosti nebo omezující podmínky daného systému.
- Funkční požadavky – určují, jaké chování bude systém nabízet.

V mnoha dokumentech je kladen důraz na kvalitní zpracování požadavků jako na velmi důležitou část projektu, jejíž zanedbání je jednou z hlavních příčin neúspěchu celého projektu. Věnovala jsem této části velkou pozornost, aby požadavky na systém vyhovovaly většině uživatelů.

Stručný seznam požadavků na aplikaci LibrarEX je uveden v následujících dvou kapitolách.

4.4.1 Obecné požadavky

1. Licence GNU/GPL
2. GUI klient – zpracování dat, správa databází, reporty
3. Webový klient – reporty, analýza dat
4. Uživatelský systém – přístup do aplikace zabezpečen heslem (GUI klient)
5. Podpora více jazyků – čeština, angličtina, připraveno na rozšíření o další jazyky
6. Instalátor

4.4.2 Funkční požadavky

Aplikace rozlišuje dva typy uživatelů - administrátor a knihovník.

Administrátor má na starosti správu vstupních dat a zajištění jejich úplnosti v databázi. Pro práci si nainstaluje klienta GUI aplikace, který mu umožní provádět všechny potřebné

operace. V rámci správy uživatelského nastavení je možné nastavit jazyk aplikace a parametry pro připojení k databázi.

Do správy vstupních dat je zahrnuto stažení dat z internetu a jejich konverze do jednotného formátu. Je nutné zaručit rozšiřitelnost o zpracování nových formátů vstupních dat. Dále je požadováno přehledné zobrazení seznamu definovaných konverzí (dále je pro definici konverze používáno jméno *filtr*). Filtry je možné přidávat, editovat a odstranit.

Kromě zpracování jednotlivých vstupních souborů je požadováno definování hromadného úkolu zpracování dat, který vykonává více operací na jedno spuštění. Hromadné úkoly je možné přidávat, editovat a odstranit.

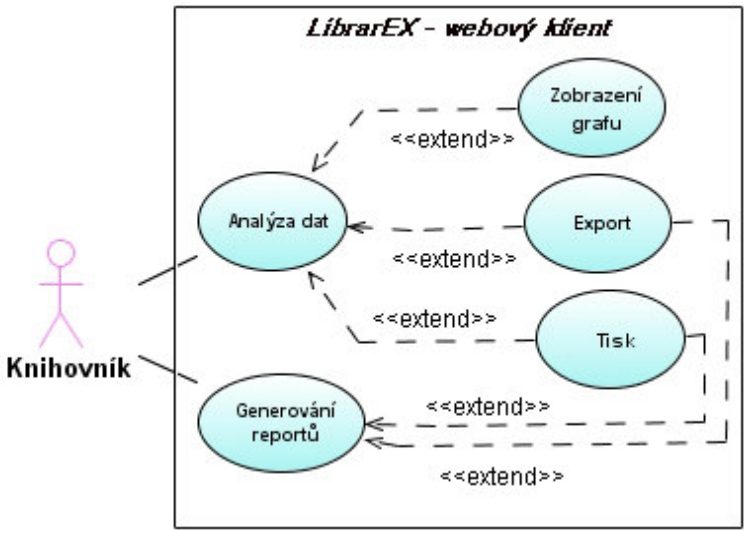
Data převedená do jednotného formátu je možné kontrolovat, editovat a následně importovat do databáze. Kontrola dat umožňuje opravu/doplnění povinných položek jednotlivých záznamů. Do databáze se importují pouze záznamy obsahující všechny povinné položky. O neúplných záznamech se ukládá informace do log souboru. Kromě úplnosti dat je nutné zaručit odstranění duplicitních záznamů. Po skončení importu mohou být zobrazeny informace o jeho výsledku.

Administrátor má k dispozici potřebné funkce pro správu databáze, generování a správu reportů. Data vytvářených reportů se načítají podle parametrů zadaných uživatelem.

Knihovník může prostřednictvím webového prohlížeče generovat reporty podle zadaných parametrů. Výsledné reporty jsou dostupné ve formátech PDF, XLS a CSV. K analýze dat je využito jejich zobrazení v kontingenčních tabulkách nebo graficky názorných grafech. Zobrazená data a grafy je možné exportovat do XLS formátu a vytisknout.

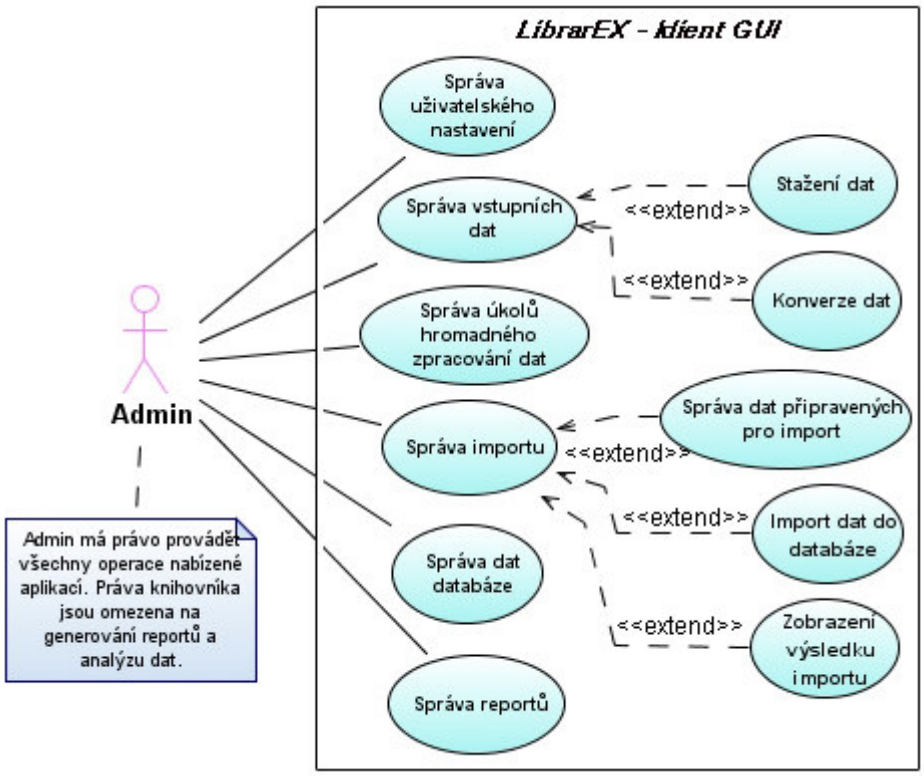
4.4.3 Případy užití

Sekce seznamuje s požadavky na systém pomocí diagramů případů užití (Use Case Diagram), které specifikují vzory chování systému a definují vztahy mezi případy užití systému a aktéry stojícími vně systému.



obr. 3: Funkční specifikace systému – webový klient

Diagramy případů užití zobrazující nejvyšší úroveň funkčnosti systému jsou zobrazeny na obr. 3 a obr. 4. V diagramu jsou zaznamenané funkční požadavky popsané v části 4.4.2.



obr. 4: Funkční specifikace systému – klient GUI

4.5 Vizuální modelování systému

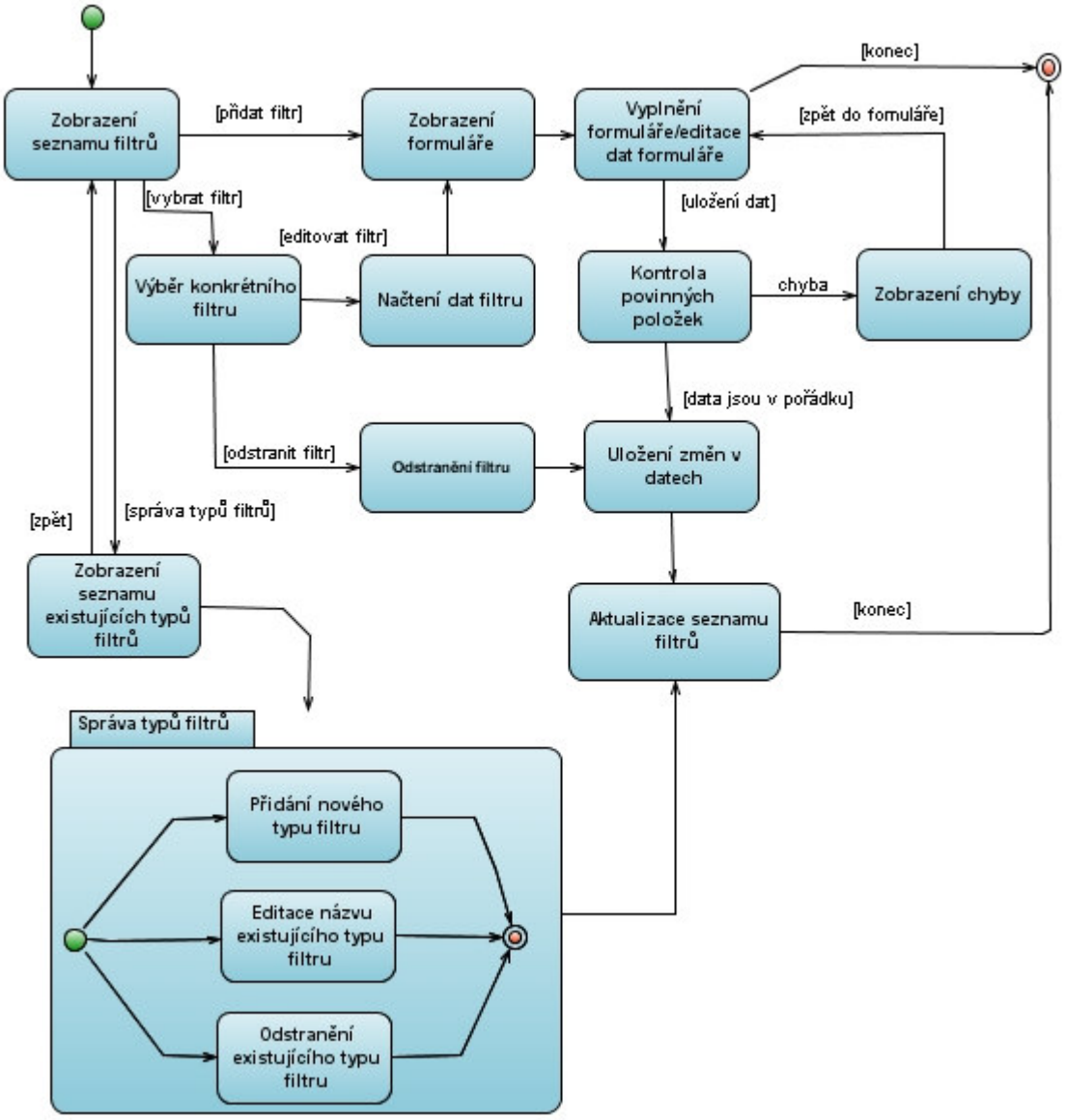
Tato kapitola seznamuje s chováním vybraných částí navrhovaného systému pomocí vizuálního modelování.

4.5.1 Správa filtrů

Jedním z požadavků na systém je zpracování vstupních dat, která jsou k dispozici v souborech různých formátů a odlišného obsahu. Další důležitý požadavek je rozšiřitelnost o zpracování nových, obsahově či formátem odlišných souborů.

Splnění těchto požadavků systém řeší pomocí skriptovacího jazyka Jython. Pro různé vstupní soubory jsou naprogramovány skripty, které data z těchto souborů převádí do jednotné podoby, do XML souboru. Aby bylo vytvoření nového skriptu co nejvíce usnadněno, aplikace nabízí předprogramované skripty, do kterých musí být dopsány jen potřebné speciální funkce.

Pro převod vstupních dat se spouští vybraný skript, kterému jsou předány parametry potřebné pro jeho vykonání a uložení odpovídajících dat. Jaké parametry a jaké hodnoty těchto parametrů se mají předat konkrétnímu skriptu má na starosti *filtr*, což je objekt, který obsahuje důležité parametry a hodnoty pro zpracování a převod vstupních dat do XML souboru.

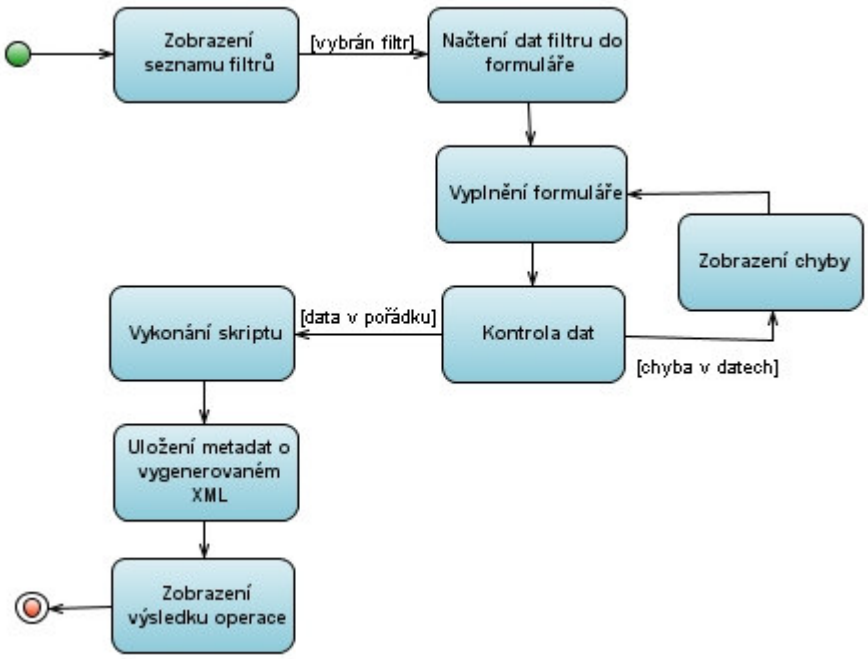


obr. 5: Stavový diagram – Správa filtrů

Správu filtrů je znázorněna na obr. 5. Uživatel může filtry přidávat, editovat a mazat. Pro přehlednost a lepší orientaci jsou filtry rozděleny podle nadefinovaných typů (např. filtr pro zpracování statistických dat, filtr pro zpracování základních dat o dokumentu). Uživatel má opět možnost tyto typy filtrů přidávat, editovat a mazat.

4.5.2 Generování XML

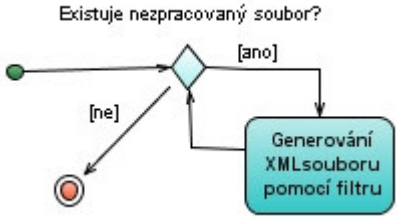
Proces vygenerování XML souboru s využitím filtru je znázorněn na obr. 6. Uživatel vybere filtr, který chce použít, a načte jeho data do formuláře. Do formuláře se musí vyplnit cesta k souboru se vstupními daty nebo zvolit použití předvolené cesty ve filtru. Dále je potřeba vyplnit období, pro které se data vztahují. Před spuštěním samotného skriptu probíhá kontrola zadaných dat a ve chvíli, kdy skončí úspěšně, je skript spuštěn. Tento skript vytváří požadovaný XML soubor. Metadata nového XML (např. název, datum vytvoření, typ, atd.) jsou uložena a uživatel je informován o výsledku operace.



obr. 6: Stavový diagram – Generování XML

Dávka – hromadné zpracování vstupních dat

Filtr umožňuje v jedné chvíli zpracovat jeden vstupní soubor s daty a vygenerovat jeden XML soubor. Při zpracování měsíčních statistik často nastane situace, že máme k dispozici více souborů a hodilo by se jejich hromadné zpracování. Tuto operaci umožňuje *dávka*. Na obr. 7 je zobrazen proces zpracování dávky.



obr. 7: Aktivity diagram – Dávka

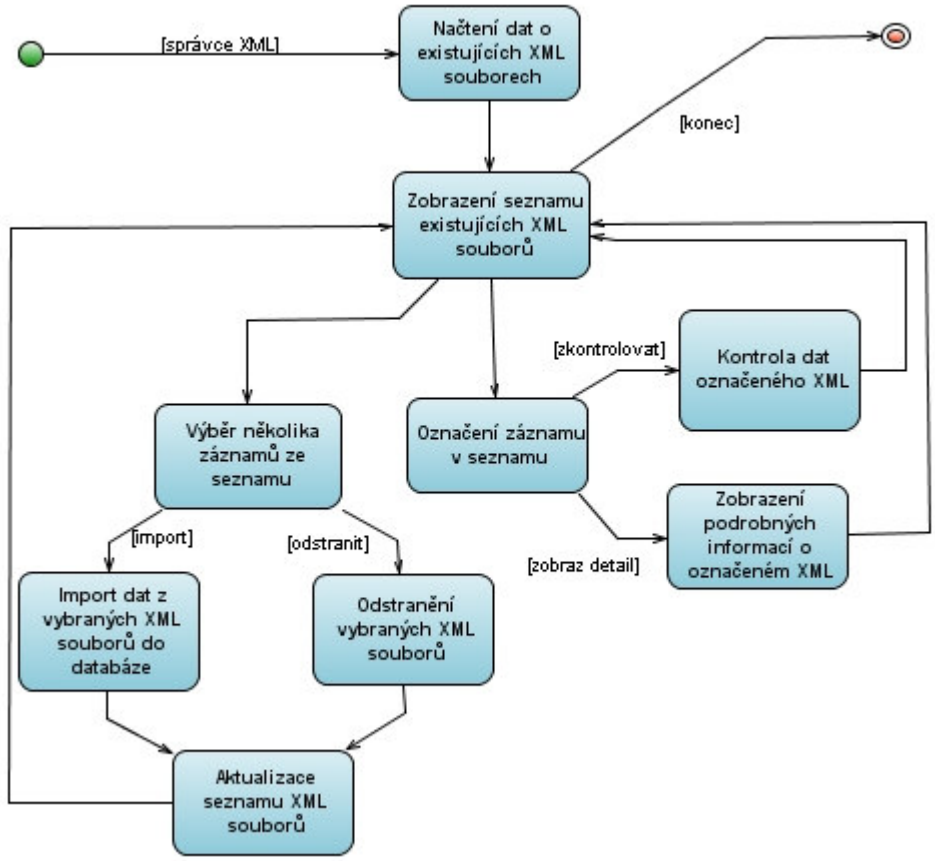
Úkol – hromadné zpracování dat

Hromadné zpracování vstupních dat je jedním z požadavků na systém. Kromě výše uvedené dávky aplikace nabízí ještě komplexnější operaci pro hromadné zpracování dat – úkol. V rámci úkolu je možné provést stažení dat z Internetu, vygenerování XML souborů a případně i import dat do databáze. *Úkol* je užitečný pro definování opakujících se operací – např. pro zpracování měsíčních statistických dat.

4.5.3 Správa XML souborů

Správa XML souborů zahrnuje kontrolu dat v XML souborech, zobrazení podrobných informací o XML souboru (např. datum vytvoření XML souboru, typ XML, použitý filtr,

název souboru se vstupními daty), import dat z vybraných XML souborů do databáze a vymazání vybraných XML souborů. Správa XML souborů je znázorněna na obr. 8.



obr. 8: Stavový diagram – Správa XML souborů

Kontrola dat XML souboru zahrnuje jejich načtení a zobrazení seznamu, který obsahuje identifikační hodnoty jednotlivých záznamů. Tento seznam má řádky zbarvené červeně a černě podle toho, zda záznam na daném řádku má uvedené všechny povinné položky (černá barva) nebo některé povinné položky chybí (červená barva). Uživatel může jednotlivé záznamy otevřít v editačním dialogu a upravit potřebné hodnoty. Kontrola XML souboru není nutnou podmínkou pro jeho import do databáze, ale ve chvíli, kdy některý záznam nebude mít vyplněné všechny povinné údaje, nebude do databáze uložen. Výsledek importu obsahuje počet úspěšně naimportovaných záznamů a počet nenaimporotvaných záznamů, o kterých je dále uložena podrobnější ínformace v logu importu. Po naimportování dat XML souboru je tento soubor uložen v adresáři záloh a následně smazán ze seznamu aktuálních XML souborů.

4.6 Struktura vstupních data

V části 4.1.5 jsem se věnovala statistickým datům, která jsou použita pro vyhodnocování využitelnosti dokumentů, pro reporty a analytické procházení dat. Základní obrázek o datech pro aplikaci jsme si tedy již vytvořili. V této části pohled na data trochu rozšíříme o další informace.

Jak již bylo uvedeno, statistická data obsahují údaje o využívání elektronických dokumentů, využívání tištěných dokumentů, ceně dokumentů, impakt faktoru článků, rozsahu dokumentů. Aplikace musí kromě statistických dat pracovat i s daty, která nesou základní informace o

dokumentech, institucích, konsorciích, poskytovatelích, vydavatelích a kategoriích dokumentů.

Důkladné seznámení se vstupními daty je důležité pro návrh databáze, proto jsem se snažila získat co nejvíce vzorků dat, které by mi pomohly porozumět datům, se kterými bude aplikace pracovat. Ve chvíli, kdy jsem narazila na nějaké nejasnosti, konzultovala jsem je s pracovníky knihoven, případně hledala odpověď v odborných publikacích nebo na Internetu. K dispozici jsem měla vzorky dat týkající se využívání elektronických informačních zdrojů, seznamy časopisů s informacemi o impakt faktoru, seznamy zpřístupněných dokumentů v rámci jednotlivých konsorcií, seznamy dokumentů s informací o ceně, atd.

Pro názorné seznámení se strukturou vstupních dat přidávám vybrané ukázky viz. obrázky obr. 9, obr. 10, obr. 11 a obr. 12.

Následuje seznam hlavních předpokladů, které byly při návrhu databáze brány v potaz. Po konzultaci s pracovníky knihovny byly tyto předpoklady schváleny jako vyhovující.

- Pro konkrétní rok bude existovat vždy jen jedna cena dokumentu.
- Statistická data o využívání informačních zdrojů budou obsahovat měsíční přístupy k dokumentům.

ISI Web of Knowledge℠ Journal Citation Reports GO HOME LOG OUT

Mark	Rank	Abbreviated Journal Title (linked to journal information)	ISSN	Total Cites	Impact Factor	Immediacy Index	Articles	Cited Half-life
☐	1	CA-CANCER J CLIN	0007-9235	5266	63.342	16.526	19	2.9
☐	2	NEW ENGL J MED	0028-4793	177505	51.296	12.743	303	6.9
☐	3	ANNU REV IMMUNOL	0732-0582	15482	47.237	9.500	24	6.3
☐	4	ANNU REV BIOCHEM	0066-4154	16761	36.525	4.433	30	8.5
☐	5	REV MOD PHYS	0034-6861	20672	33.508	6.656	32	>10.0
☐	6	NAT REV CANCER	1474-175X	13189	31.583	4.675	80	3.4
☐	7	PHYSIOL REV	0031-9333	16209	31.441	4.906	32	7.2
☐	8	NAT REV MOL CELL BIO	1471-0072	14132	31.354	3.894	85	3.6
☐	9	SCIENCE	0036-8075	361389	30.028	5.555	885	7.7
☐	10	CELL	0092-8674	132528	29.194	6.403	352	8.7
☐	11	NAT REV IMMUNOL	1474-1733	11098	28.697	4.628	78	3.3
☐	12	NAT MED	1078-8956	43664	28.588	5.261	153	5.3

obr. 9: Impakt faktor

Basic Description | Other Editions/ Formats | Abstracting/ Indexing & Article Access | Publisher & Ordering Information | Advertising, Rights, Demographics | Reviews

JCR Web | WorldCat | ScienceDirect | ULRICH'S RESOURCE LINKER

Click highlighted text for a new search on that item.

Table of Contents: TOC

ISSN:	0304-3770
Title:	Aquatic Botany
Publishing Body:	Elsevier BV
Country:	Netherlands
Status:	Active
Start Year:	1975
Frequency:	8 times a year
Volume Ends:	# 4,
Document Type:	Journal; Academic/Scholarly
Refereed:	Yes
Abstracted/Indexed:	Yes
Media:	Print
Language:	Text in English
Price:	EUR 1,218 subscription per year in Europe to institutions JPY 161,800 subscription per year in Japan to institutions USD 1,362 subscription per year to institutions Except Europe And Japan (effective 2006)
Subject:	BIOLOGY - BOTANY
Dewey #:	579.177
CODEN:	AQBODS
Special Features:	Includes Advertising, Bibliographies, Illustrations, Abstracts, Book Reviews
Article Index:	Index Available
Editor(s):	Dr. G Bowes (Editor-in-Chief), Jan Vermaat (Editor-in-Chief)
URL:	http://www.elsevier.com/locate/aquabot
Description:	Concerned with fundamental studies on structure, function, dynamics and classification of plant-dominated aquatic ecosystems.

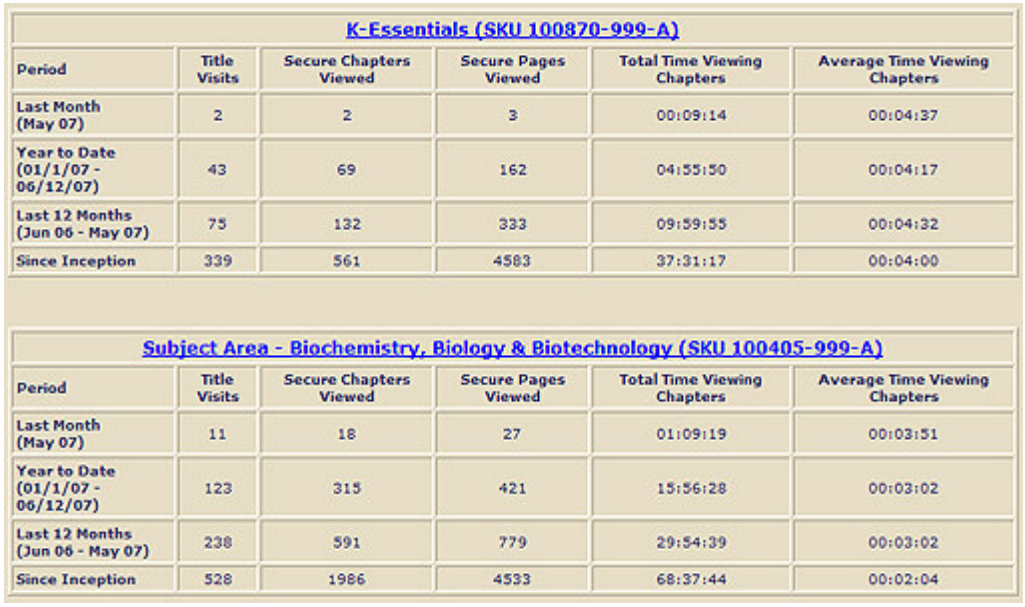
obr. 10: Základní informace o dokumentu + Cena

Journal Report 1: Number of Successful Full-Text Article Requests by Month and Journal for Czech -Institute of Chemical Technology June 2007

Journal Report 1 (R2): Number of Successful Full-Text Article Requests by Month and Journal

	Publisher	Platform	Print ISSN	Online ISSN	Jan-2007	Feb-2007	Mar-2007	Apr-2007	May-2007	Jun-2007	YTD Total	YTD HTML	YTD PDF
Total for all journals	Wiley	Wiley Interscience			1242	1069	1258	1188	1172	890	6819	293	6526
About Campus	Wiley	Wiley Interscience	1086-4822	1536-0687	0	0	0	0	0	0	0	0	0
Acta Biotechnologica	Wiley	Wiley Interscience	0138-4988	1521-3846	0	1	0	0	1	0	2	0	2
Acta Biotechnologica 1980-2000	Wiley	Wiley Interscience	0138-4988	1521-3846	0	0	0	0	7	44	51	0	51
Acta hydrochimica et hydrobiologica	Wiley	Wiley Interscience	0323-4320	1521-401X	1	9	1	2	2	1	16	0	16
Acta Hydrochimica et Hydrobiologica 1973-1997	Wiley	Wiley Interscience	0323-4320	1521-401X	0	0	0	0	0	0	0	0	0
Acta Polymerica	Wiley	Wiley Interscience	0323-7648	1521-4044	0	0	0	1	0	0	1	0	1
Acta Polymerica 1979 - 1997	Wiley	Wiley Interscience	0323-7648	1521-4044	0	0	0	0	4	0	4	0	4
Advanced Engineering Materials	Wiley	Wiley Interscience	1438-1656	1527-2648	3	0	1	2	2	0	8	0	8
Advanced Functional Materials	Wiley	Wiley Interscience	1616-301X	1616-3028	5	6	5	11	0	1	28	0	28
Advanced Materials	Wiley	Wiley Interscience	0935-9648	1521-4095	26	9	9	5	11	3	63	0	63

obr. 11: COUNTER – využívání elektronických informačních zdrojů



K-Essentials (SKU 100870-999-A)

Period	Title Visits	Secure Chapters Viewed	Secure Pages Viewed	Total Time Viewing Chapters	Average Time Viewing Chapters
Last Month (May 07)	2	2	3	00:09:14	00:04:37
Year to Date (01/1/07 - 06/12/07)	43	69	162	04:55:50	00:04:17
Last 12 Months (Jun 06 - May 07)	75	132	333	09:59:55	00:04:32
Since Inception	339	561	4583	37:31:17	00:04:00

Subject Area - Biochemistry, Biology & Biotechnology (SKU 100405-999-A)

Period	Title Visits	Secure Chapters Viewed	Secure Pages Viewed	Total Time Viewing Chapters	Average Time Viewing Chapters
Last Month (May 07)	11	18	27	01:09:19	00:03:51
Year to Date (01/1/07 - 06/12/07)	123	315	421	15:56:28	00:03:02
Last 12 Months (Jun 06 - May 07)	238	591	779	29:54:39	00:03:02
Since Inception	528	1986	4533	68:37:44	00:02:04

obr. 12: Vlastní formát – využívání elektronických informačních zdrojů

4.7 Struktura výstupních dat

Při návrhu struktury databáze hrají významnou roli i požadované výstupy, tj. jaké informace a v jaké podobě bude uživatel chtít získávat. Aplikace umožní vytvářet různé reporty, procházet data pomocí kontingenčních tabulek a zobrazovat data v grafech.

4.7.1 Reporty

Jak již bylo zmiňováno, mají získané statistické rozbory pomoci knihovníkovi ve finančním plánování, zkvalitňování knihovního fondu a získávání podkladů pro obhájení finančního hospodaření. V kapitole 4.1.4 jsem se věnovala vlastnímu rozdělení dokumentů podle různých kritérií, které lze nyní využít při návrhu reportů. Stejně jako existují různé typy přístupu k dokumentům, tak budou mít odlišnou informační povahu i generované reporty. Podle požadavků a po konzultacích s knihovníky jsem navrhla následující základní reporty:

- Měsíční využívání elektronických informačních zdrojů
- Roční statistiky elektronických informačních zdrojů
- Roční statistiky balíčků elektronických informačních zdrojů
- Měsíční využívání tištěných informačních zdrojů
- Roční statistiky tištěných informačních zdrojů

Každý základní typ reportu nabízí uživateli několik variant provedení, což záleží na samotném návrhu těchto reportů. Reporty mohou například vyhodnocovat využívání jednotlivých dokumentů, využívání všech dokumentů daného poskytovatele a řadu dalších ukazatelů. Pro každý typ reportu je možné pomocí volitelných parametrů přesněji specifikovat oblast zájmu uživatele – zda ho zajímají časopisy či knihy, konkrétní poskytovatel, atd.. Zajímavým a důležitým parametrem pro reporty je hodnocení investice do dokumentu. Jedná se o položku, která je dopočítávána programem na základě získaných statistických dat, dat o ceně či přiřazené prioritě (důležitosti) financování dokumentu. Tato položka rozděluje

dokumenty do skupin podle efektivity investice a nabízí jednodušší vyhodnocení investic finančních prostředků. Výsledné reporty je možné ukládat ve formátech PDF, XML, HTML, CSV, XLS a RTF.

4.7.2 Analýza dat

Provádět analýzu dat a získávat z nich relevantní informace je (kromě reportů) dále možné i procházením kontingenčních tabulek. Kontingenční tabulky jsou interaktivní a umožňují prohození řádků a sloupců a tedy zobrazení různých souhrnů zdrojových dat. Díky hierarchické struktuře sloupců a řádků umožňují kontingenční tabulky zobrazovat rovněž různou granularitu statistických dat.

4.8 Existující software

Z analýzy vyplývá, že v současnosti zřejmě není implementován software, který by řešil výše nastíněný problém.

Vyslovení tohoto tvrzení předcházelo studium informací na Internetu, čtení článků, konzultace v knihovnách či písemná korespondence, kdy jsem kontaktovala autority, věnující se dané problematice delší dobu. Seznámila s několika projekty z oblasti knihovnictví a statistik např. LibQUAL+, ies National Center for Educational Statistics. Žádný z uvedených projektů však nesleduje a nevyhodnocuje statistiky jednotlivých informačních zdrojů přístupných konkrétní knihovně.

Ani zmíněné projekty COUNTER, ICOLC a ARL E-metrics, které umožňují sledování přístupů k informačním zdrojům, neřeší výše nastíněný problém. Pokud bych měla zmínit odlišnosti těchto projektů a aplikace, která je cílem této diplomové práce, nebude jich málo – např.:

- Navrhovaná aplikace LibrarEX je určena knihovnám a ne poskytovatelům, její cíl je tudíž úplně odlišný od cíle výše uvedených projektů, které sledují pouze statistiky dokumentů.
- Uvedené projekty podporují vždy jeden konkrétní standard či předpis pro sledování statistických údajů. Knihovny však mají k dispozici statistická data od jednotlivých poskytovatelů – jedním z cílů aplikace je sjednotit všechny dostupné statistické údaje do jednotného formátu.
- Aplikace LibrarEX pracuje s mnohem více statistickými údaji (cena, počet článků, atd.), které poskytují kromě vyhodnocování přístupů k dokumentům i další zajímavé pohledy na data.
- Aplikace LibrarEX nabízí reporty generované pro všechny dokumenty přístupné knihovně a také pohledy na data z různých úhlů – jako např. podle poskytovatele, konsorcia, kategorie, jednotlivých dokumentů.

5 Návrh a implementace

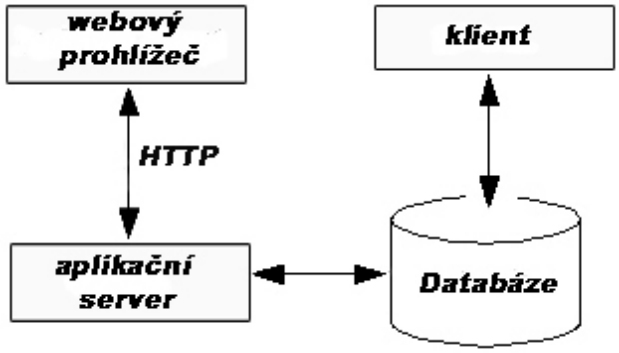
Tato kapitola se zabývá architekturou systému, popisuje komponenty, ze kterých se aplikace skládá, co je úkolem jednotlivých částí a jak jsou vzájemně provázány.

V sekci 5.1 je popsáno základní rozdělení aplikace do tří logických celků, kterými se podrobně zabývají kapitoly 5.3, 5.4 a 5.5. Podrobný návrh jednotlivých částí systému může být ovlivněn použitou technologií, proto se sekce 5.2 zabývá právě volbou technologie pro implementaci aplikace.

5.1 Architektura a struktura aplikace

Úkolem aplikace je spravovat statistická data a získávat analytické informace. K analytickým informacím se přistupuje buď lokálně nebo přes Internet. Aplikaci tedy můžeme rozdělit do následujících logických celků:

- ***databáze*** – je centrálním uložištěm dat aplikace. K datům se přistupuje prostřednictvím klienta a aplikačního serveru viz. obr. 13.
- ***klient*** (tlustý klient kombinovaný s middleware) – zpracovává statistická data, řídí jejich import do databáze, transformaci dat mezi databázemi a generování reportů. Uživatel má k dispozici grafické uživatelské rozhraní pro správu dat a databází. Klientská část aplikace obsahuje databázovou vrstvu, která maximálně odstiňuje programátora od vlastních databázových modelů a zároveň nabízí metody pro práci s daty v databázi a získávání požadovaných dat z databáze.
- ***webový klient*** – pomocí běžně dostupných webových prohlížečů lze procházet a zobrazovat reporty a kontingenční tabulky obsahující data, která jsou zprostředkována aplikačním serverem. Přístup k datům přes webové rozhraní nabízí několik výhod. Jednou ze stěžejních je fakt, že si klienti nemusí nic instalovat na vlastním počítači, vše je nainstalováno centrálně a spravováno pověřenou osobou.



obr. 13: Architektura systému

5.2 Technologie

Volba technologie může návrh aplikace znatelně ovlivnit, proto je již při návrhu důležité seznámit se s technologiemi, které budou použity pro implementaci. Tato sekce se bude zabývat zvolenými technologiemi a důvodem jejich volby.

5.2.1 Programovací jazyky

Jako hlavní programovací jazyk jsme zvolila objektově orientovaný programovací jazyk Java. Důvodem volby tohoto jazyka byla jeho přenositelnost na různé systémy a dostupnost celé řady knihoven a nástrojů, které jsou poskytovány zdarma. Dalším důvodem byla i moje předchozí dobrá zkušenost s tímto jazykem.

Aplikace dále používá skriptovací jazyk Python (Jython). Pomocí něj se získávají a zpracovávají vstupních dat aplikace do XML formátu. Zajišťuje snadnou rozšiřitelnost o zpracování nových formátů vstupních dat nebo o funkce, které získávají datové soubory z Internetu.

Jython je interpretem jazyka Python. Je napsaný čistě v jazyce Java a převádí zdrojový kód jazyka Python přímo do bajtového kódu jazyka Java. Jython vyžaduje jen Java Virtual Machine (JVM) a knihovny, což zaručuje přenositelnost mezi platformami. Tyto výhody jsou dalším argumentem pro jeho použití.

Další jazyky používané v aplikaci: značkovací jazyk XML, jazyk XPath, DTD jazyk pro popis XML struktury, dotazovací jazyk MDX, dotazovací jazyk SQL, JSP, CSS, HTML..

5.2.2 Hibernate

Hibernate[20] je rozšířený Open Source (licence LGPL) systém, který poskytuje objektově relační mapování (ORM) pro javovské prostředí. Mezi výhody můžeme zařadit použití objektově orientovaného přístupu a práci se záznamy v databázi jako s objekty. Další výhodou je přenositelnost mezi velkým počtem databázových systémů. Kromě toho Hibernate nabízí různé způsoby zadávání dotazů např. použití vlastního plně objektově orientovaného jazyka Hibernate Query Language (HQL) nebo přímo jazyka SQL nebo omezujících podmínek v Javě (criteria API).

Jak je vidět, Hibernate nabízí spoustu výhod pro snadný vývoj databázového systému. Důležité je však dobré seznámení s tímto systémem, abychom předešli budoucím problémům a naopak uměli využít jeho výhod.

5.2.3 Databázový systém

Vhledem k použití technologie Hibernate ORM je možné aplikaci provozovat nad různými databázovými stroji, které tato technologie podporuje. Jako základní databázový systém jsem zvolila PostgreSQL. Důvodem této volby byla jeho dostupnost jako OpenSource (pod BSD licencí), vynikající pověst pro svou spolehlivost a bezpečnost, osobní dobré zkušenosti, spustitelnost na všech rozšířených operačních systémech, jednoduchá administrace a dostupnost kvalitní dokumentace.

5.2.4 XML

XML (*eXtensible Markup Language*) je značkovací jazyk, který byl vyvinut a standardizován konsorciem W3C [17]. Jazyk XML je v dnešní době standardním jazykem používaným pro výměnu dat mezi aplikacemi a často je využíván jako formát pro jejich ukládání.

[20] http://www.hibernate.org/

Jak již bylo zmíněno, aplikace LibrarEX využívá tento jazyk pro uložení vstupních dat do jednotného formátu. Není to však jediné využití tohoto jazyka, dále je využíván pro ukládání informací o vytvořených XML souborech, o filtrech definujících způsob zpracování vstupních dat a také o nastavení aplikace.

Pro práci s XML dokumenty jsem použila rozhraní DOM i rozhraní SAX. Jako parser XML jsem zvolila Dom4j.

DOM (Document Object Model)

Objektový model dokumentu reprezentuje XML dokument jako stromovou strukturu, kde každému elementu odpovídá jeden uzel stromu. Výhodou této reprezentace dat je možnost procházení dokumentu podle našich potřeb, snadná modifikace jednotlivých uzlů, jejich mazání a vkládání. Jeho nevýhodou je paměťová náročnost, která omezuje jeho použití na zpracování méně rozsáhlých dat.

Rozhraní DOM bude vzhledem k rozsahu zpracovávaných dat použito pro uložení konfiguračních dat aplikace, pro správu metadat vytvořených XML souborů s daty pro import do databáze a také pro správu filtrů. Ve všech jmenovaných případech se předpokládá, že XML soubory nedosáhnou velikosti, která by při jejich zpracování vedla k přetečení paměti.

SAX (Simple API for XML)

SAX je rozhraní pro zpracování XML dokumentu řízené událostmi. Dokument je zpracováván sekvenčně a každý element vyvolává událost, která může v aplikaci spustit konkrétní činnost. Výhoda událostmi řízeného přístupu je v jeho rychlosti a malé pamětové náročnosti.

Rozhraní SAX je tak vhodné pro zpracování velkých souborů. V aplikaci se musí použít SAX pro XML soubory určené pro import dat do databáze. Tyto soubory jsou vygenerovány a naplněny daty ze vstupních datových souborů. zejména u statistických dat se předpokládá velký rozsah. Aplikace dále umožňuje editaci jednotlivých elementů, což není při použití metody SAX příliš typické. Použitý parser Dom4j však změnu XML dokumentu i při použití rozhraní SAX umožňuje.

5.2.5 Model–View–Controller

Model–View–Controller (MVC) se uplatňuje při návrhu a implementaci informačních systémů. Jedná se architekturu, která rozděluje aplikaci do tří nezávislých komponent: model (datový model), view (uživatelské rozhraní) a controller (řídící logika). Výhodou MVC je větší flexibilita kódu a možnost modifikace některé z komponent bez velkého dopadu na komponenty ostatní.

5.2.6 Datové sklady, OLAP

Vzhledem k požadavkům na analytické zpracování dat je následující text věnován stručnému seznámení se základními pojmy z oblasti datových skladů. Pro podrobnější prostudování doporučuji další literaturu [13].

Datový sklad *je podnikový strukturovaný depozitář předmětově orientovaných, vzájemně provázaných, časově neměnných historických dat, používaný k získávání informací a podporu rozhodování. V datovém skladu jsou uložena detailní a sumární data.*

Bill Inmon

Vysvětlení vlastností uvedených v definici:

- **Předmětová orientace** – z pohledu uživatelů je obsah datového skladu organizovaný podle odborového zaměření. Oproti běžné relační databázi, kde je snaha o co nejmenší redundanci uložených dat, které je dosahováno jejich normalizací do 3NF a vnitřním provázáním jednotlivých logických funkčních celků, dochází v datovém skladu k denormalizaci, redundanci.

- **Integrovanost** – v datovém skladu je třeba shromáždit informace z mnoha různých zdrojů a seskupit je dle logického významu.

- **Nízká proměnlivost** – data jsou do datového skladu obvykle nahrávána ve větších dávkách (např. týdenních, měsíčních intervalech).

- **Historická data** – data jsou v datovém skladu obvykle udržována v historické podobě, nikoliv pouze v aktuálním stavu. To je dáno nutností provádět analýzy zaměřené na vývoj v čase.

Příprava a zavedení údajů do datového skladu je důležitou součástí každého datového skladu. Údaje pocházejí z různých nehomogenních zdrojů a musí být před zavedením do datového skladu vyčištěny a upraveny. V této souvislosti mluvíme o nástrojích a postupech ETL[21]:

- Extraction – získání dat prostřednictvím různých metod

- Transformation – ověření, čištění, úprava dat

- Loading – zavedení dat do datového skladu

Údaje jsou do datového skladu ukládány s ohledem na co nejlepší a nejrychlejší provádění složitých dotazů. Výsledný datový sklad poskytuje sjednocená a agregovaná data pro náročnější dotazování, pomocí nichž lze získat koncentrované informace obsažené v původních datech. Tyto informace podporují rozhodování, ať již s využitím specifických technik dolování dat nebo v navazujících systémech typu OLAP [14].

Ukázalo se, že relační databáze nejsou vhodné pro analýzu velkého množství dat. Místo nich se data ukládají výhradně do multidimenzionálních struktur. V souvislosti s datovými sklady a analýzou dat se setkáváme s pojmem *Multidimenzionální databáze*. Tyto databáze se od klasických databází liší zejména denormalizací, redundancí, optimalizací na složité analýzy a v důrazu na srozumitelnost uživateli. Multidimenzionální databáze obsahují dva druhy tabulek, jedná se o tabulku faktů (faktová tabulka) a tabulky dimenzí (dimenzionální tabulky).

Fakta jsou numerické hodnoty potřebné k analytickým výpočtům. V aplikaci LibrarEX se bude jednat např. o statistická data o využívání publikací, cenu publikací, impakt faktor, atd. Tabulka faktů je nejobjemnější a nejčastěji aktualizovanou tabulkou databáze.

Dimenze obsahují logicky nebo organizačně hierarchicky uspořádané údaje. Zachycují úhel pohledu na sledované ukazatele ve faktových tabulkách. V aplikaci LibrarEX se bude jednat např. o dimenzi poskytovatelů, dokumentů, času, atd. Tabulky dimenzí jsou poměrně malé a obecně obsahují stabilní data.

Data v datovém skladu jsou z logického pohledu členěna do schémat. Schéma odpovídá jedné analyzované funkční oblasti nazývané datové tržiště (datamart). Jádrem datového tržiště je

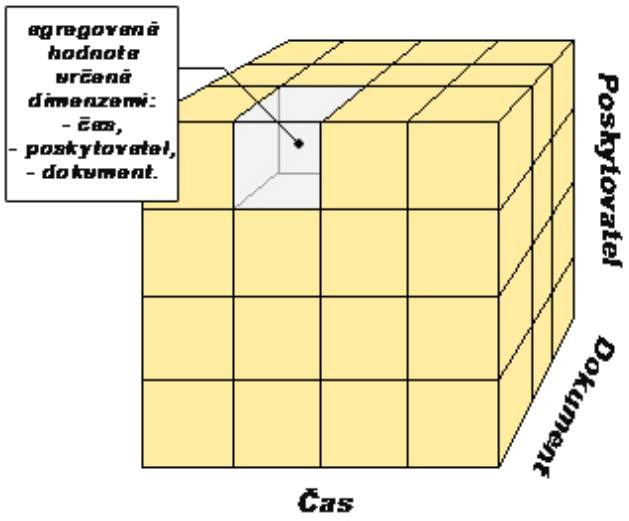
[21] http://en.wikipedia.org/wiki/Extract,_transform,_load

jedna nebo více **faktových tabulek**, které jsou pomocí cizích klíčů spojeny s tabulkami **dimenzí**.

Existují dvě základní schémata uspořádání faktových tabulek:

- **Hvězda (souhvězdí)** – obsahuje tabulku faktů, na kterou jsou přímo napojeny všechny tabulky dimenzí. Předpokládá se, že mezi jednotlivými dimenzemi nejsou žádné závislosti. *Souhvězdí* (galaxie) představuje schéma s více hvězdami, které sdílejí některé dimenzionální tabulky [14].
- **Sněhová vločka** – obsahuje tabulku faktů, na kterou jsou navázány jen dimenze na nejnižším stupni hierarchie, ostatní dimenze se napojí na některou z nižších dimenzí v hierarchické struktuře (na hierarchickou dimenzi byla použita normalizace 3NF).

OLAP (On Line Analytical Processing)[22] je volně definovaný řád principů, které poskytují dimenzionální rámec pro podporu rozhodování. Jak již bylo řečeno, systémy OLAP jsou typicky využívány pro analýzu velkého množství údajů, jejichž výsledkem jsou souhrny a reporty, které jsou důležité pro rozhodování. Existuje několik druhů OLAP. Mezi nejznámější patří MOLAP (Multidimensional OLAP), kde jsou data i agregace uloženy v multidimenzionální databázi (datová kostka). Dalším známým druhem je ROLAP (Relation OLAP), který používá pro uložení dat a agregací relační databázi. HOLAP (Hybrid OLAP) je kombinací MOLAP a ROLAP.



obr. 14: Krychle OLAP

V souvislosti s OLAP se setkáváme často s pojmem **krychle OLAP**. Jedná se o multidimenzionální datovou strukturu, ve které jsou uloženy výsledky agregace dat a to v průsečíku jednotlivých dimenzí. Dimenzi si můžeme představit jako jednu osu krychle. Příklad třídimenzionální krychle je na obr. 14.

Data z OLAP krychle se získávají pomocí specializovaného jazyka MDX (Multidimensional Expresions)[23] pro multidimenzionální databáze.

V souvislosti s přístupem k databázím OLAP z klientských aplikací se můžeme setkat s pojmem **kontingenční tabulka**[24] (Pivot Table). Kontingenční tabulka je interaktivní

[22] http://en.wikipedia.org/wiki/On_line_analytical_processing
[23] http://en.wikipedia.org/wiki/Multidimensional_Expressions

tabulka, která kombinuje a porovnává velké množství dat. Na rozdíl od klasické tabulky má několik speciálních vlastností – např. umožňuje výměnu řádků a sloupců, kombinaci a hierarchickou strukturu řádků a sloupců.

5.2.7 MDX

Jazyk MDX (Multidimensional Expressions)[25] je dotazovací jazyk pro vícedimenzionální struktury, který umožňuje definovat dotaz, na základě kterého jsou z krychle OLAP vybrány požadované údaje. MDX je jistým ekvivalentem dotazovacího jazyka SQL nad relačními databázemi.

MDX dotaz má následující strukturu:

```
SELECT
      [<specifikace_osy>
      [, <specifikace_osy>...]]
FROM [<specifikace_krychle>]
[WHERE [<specifikace_řezu>]]
```

Klíčové slovo SELECT je analogie příkazu SELECT pro výběr údajů v jazyce SQL. Příkaz SELECT specifikuje jednu či více osových dimenzí, které určují, jak bude výsledná část kostky vypadat. Jazyk MDX umožňuje práci až se 128 dimenzemi, které můžeme definovat buď slovně např. COLUMNS, ROWS, PAGES nebo číselně AXIS<index>.

Pomocí klauzule FROM je stanovena krychle, z které budou data vybrána.

Klauzule WHERE definuje výřez (omezení množiny údajů). Výsledná data pak budou obsahovat jen ty osy a dimenze, které splňují podmínku.

Následuje praktický příklad, pro který je nutné definovat pracovní krychli WORK_CUBE. První tabulka obsahuje přehled dimenzí a jejich úrovní. Druhá tabulka informuje o měrných jednotkách krychle.

Název dimenze	*Úrovně*	*Popis*
Document	Publisher, Document	Vydavatel a název dokumentu
Category	Cat1, Cat2, Cat3	Tématické třídění dokumentů
Provider	Provider	Jméno poskytovatele dokumentu
Date	Year, Month	Datum sledování statistických dat

tabulka 1: Pracovní krychle – dimenze

Název měrné jednotky	*Popis*
FullText	Počet stažených plných textů
PriceDown	Cena stažení jednoho plného textu
PriceDoc	Cena dokumentu

tabulka 2: Pracovní krychle – fakta (measures)

```
SELECT
```

[24] http://office.microsoft.com/cs-cz/excel/HP052743561029.aspx
[25] http://msdn2.microsoft.com/en-us/library/Aa216767(SQL.80).aspx

```
      {[Measures].[FullText],[Measures].[PriceDown]} ON COLUMNS,
      {[Dokument]} ON ROWS
FROM [WORK_CUBE]
WHERE {[Date].[2007]}
```

Výsledkem uvedeného příkladu by byla kontingenční tabulka 3. V tabulce jsou vidět dva sloupce pro měrné jednotky FullText a PriceDown informující o počtu stažení plných textů a ceně stažení pro danou skupinu dokumentů. Kliknutím na symbol plus (+) se zobrazí nové řádky, obsahující seznam vydavatelů dokumentů a přepočítané sledované hodnoty pro nové skupiny dokumentu. Vydavatelé jsou v dimenzi Document v hierarchii nejvýše, proto se po rozkliknutí zobrazí první. Pro každého vydavatele bude možné rozbalit seznam jeho dokumentů. V podmínce WHERE dotazu je definováno, že budou zobrazena jen data pro rok 2007.

	Measures	
Document	***FullText***	***PriceDown***
+Documents	4 080	385

tabulka 3: Výsledek příkladu

5.2.8 Mondrian

Zajímala jsem se o nástroje, pomocí kterých by bylo možné vytvářet různé reporty a procházet data z více pohledů. Vzhledem k obecnému požadavku 1, podle kterého má být výsledná aplikace dále šířena pod licencí GPL, může být v aplikaci použit opět pouze program, který dodržuje tento požadavek. Zaměřila jsem se na nekomerční produkty pro business inteligence (dále jen BI) a OLAP servery. Byla jsem příjemně překvapená množstvím projektů z nekomerční oblasti. K největším open source BI projektům patří Pentaho BI Platform[26] a JasperSoft[27]. Mezi další zajímavé projekty patří např. OSBI[28], Bizgres[29], BRIT[30] či Bee[31]. Z OLAP serverů moji pozornost nejvíce upoutal ROLAP server Mondrian[32] a MOLAP server JPalo[33].

Pentaho BI Platform je open source určený pro business intelligence, který slučuje projekty jako např. JFreeReport, Mondrian, Pivot, Weku, KETTLE, které jsou určené pro reporting, analýzu dat, data mining a ETL. Nabízí tak jednotné prostředí pro spouštění, zobrazování výsledků a práci s daty. Samotná Pentaho BI Platform i její součásti jsou napsány v Javě [16].

Mondrian je open source ROLAP (Relation OLAP) server napsaný v Javě a může běžet jako webová aplikace na aplikačním serveru. Pro OLAP dotazy používá speciální jazyk MDX (viz. kapitola 5.2.7), který převádí do SQL dotazů nad klasickou relační databází. Ve spolupráci s prohlížečem JPivot[34] zobrazuje výsledná data v multidimenzionálním formátu v kontingenční tabulce. Abychom mohli prostřednictví serveru Mondrian získávat data z relační databáze, musí být vytvořeno schéma definující multidimenzionální databázi. Schéma obsahuje logický

[26] http://mondrian.pentaho.org
[27] http://www.jaspersoft.com
[28] http://www.squidoo.com/osbi/
[29] http://www.bizgres.org/?page=6
[30] http://www.eclipse.org/birt/phoenix/
[31] http://www.beeproject.org/
[32] http://mondrian.pentaho.org
[33] http://www.jedox.com/en/enterprise-spreadsheet-server/excel-olap-server/palo-server.html
[34] http://jpivot.sourceforge.net/

model tvořený z kostek, hierarchií, členů a mapování tohoto modelu na databázi. Tento logický model je definován pomocí jazyka XML.

Při výběru jsem se ve výsledné fázi rozhodovala mezi použitím Pentaho BI Platform a serverem Mondrian. Rozhodla jsem se použít server Mondrian, protože nabízí požadovanou funkčnost. Použití celého komplexního nástroje, kterým je Pentaho, se mi jeví zbytečné.

5.2.9 Aplikační server

Vzhledem k volbě severu Mondrian jsem výběr aplikačního serveru omezila na aplikační servery s podporou JavaServer Pages. Rozhodla jsem se pro použití serveru Apache Tomcat, který je jedním z nejpoužívanějších volně dostupných JSP kontejnerů.

5.2.10 JasperReport

Hledala jsem vhodný nástroj, který bych použila pro generování reportů a našla jsem knihovnu JasperReport. Tento reportovací nástroj dovede vytvářet dokumenty ve formátech PDF, XML, HTML, CSV, XLS, RTF a TXT. Pro definici vzhledu dokumentu a práci s daty se využívá XML. Psaní samotného XML by bylo hodně složité, naštěstí existují nástroje, které tvorbu reportů ulehčují. Já jsem pro svou práci zvolila iReport, který nabízí grafický editor pro vytváření vzhledu reportů. Pokud máme hotový design reportu, musíme zařídit jeho naplnění daty. K tomu bylo potřeba implementovat interface JDataSource.

5.3 Databáze

Vytvoření konečného návrhu databáze byla jedna z časově náročnějších činností. Databáze prošla několika návrhy než získala svou konečnou podobu. Vývoj návrhů byl ovlivněn dodatečným upřesňováním některých požadavků na aplikaci a získáním podrobnějších informacích o vstupních datech aplikace.

Mezi důležité požadavky, které výrazně ovlivnily návrh databáze, patří požadavek na tvorbu reportů podle zvolených parametrů a požadavek na provádění analýz s daty. Oba tyto požadavky vyžadují časté agregační operace a složité dotazy do databáze v závislosti na parametrech zadaných uživatelem.

Při návrhu databáze jsem se rozhodovala mezi dvěma způsoby řešení. První řešení zahrnuje návrh transakční databáze, druhé řešení zahrnuje návrh analytické databáze. Transakční a analytické databáze plní odlišné úkoly, což vede i k odlišnosti návrhu jejich struktury.

1. řešení: Transakční databáze

Transakční databáze je typ databáze, která ukládá dynamická data a používá se v situacích, kdy je třeba shromažďovat, modifikovat a udržovat data, která se často mění. Transakční databáze jsou primárně používány v online zpracování transakcí (OLTP).

Základem tohoto řešení je návrh relační databáze s normalizovanými tabulkami. Data v této databázi budou aplikací aktualizována, přidávána a mazána.

Výhody řešení:

- Návrh jedné relační databáze

Nevýhody řešení:

- Dlouhotrvající import dat do databáze
- Dlouhotrvající vyhodnocení dotazů, které jsou složité a obsahují agregace

2. řešení: Analytická databáze

Analytická databáze je typ databáze, která uchovává statistická data a používá se v případě, kdy je potřeba sledovat trendy, zobrazovat dlouhodobá statistická data nebo provádět taktické a strategické obchodní předpovědi [15]. Analytické databáze jsou primárně používány v online analytickém zpracování (OLAP).

Základem tohoto řešení je návrh databáze optimalizované pro online analytické zpracování. Inspiraci jsem našla v oblasti Business inteligence, přesněji v problematice věnované datovým skladům. Seznámení s datovými sklady byla věnována kapitola 5.2.6.

Řešení je založeno na návrhu tří databází. První databáze (označme ji L0) odpovídá nulté vrstvě datového skladu. Aplikace do databáze L0 importuje data systémem 1:1. Při importu není nutné řešit žádná integritní omezení ani indexy. Databáze L0 tak obsahuje pouze dočasná data, která jsou dále zpracována do další vrstvy. Druhá databáze (označme ji L1) odpovídá první vrstvě datového skladu. Databáze L1 již splňuje třetí normální formu (3NF) a data uložená v této databázi musí být při převodu očištěná a konzistentní. Poslední databáze (označme ji L2) odpovídá druhé vrstvě datového skladu a jedná se o speciálně připravená data ve formátu odpovídajícím datovým tržištím. Nad touto databází lze snadno vyhodnocovat dotazy a získávat informace pro reporty a hloubkové analýzy.

Výhody řešení:

- Rychlý import dat z aplikace do databáze L0
- Rychlé vyhodnocení složitých dotazů nad databází L2

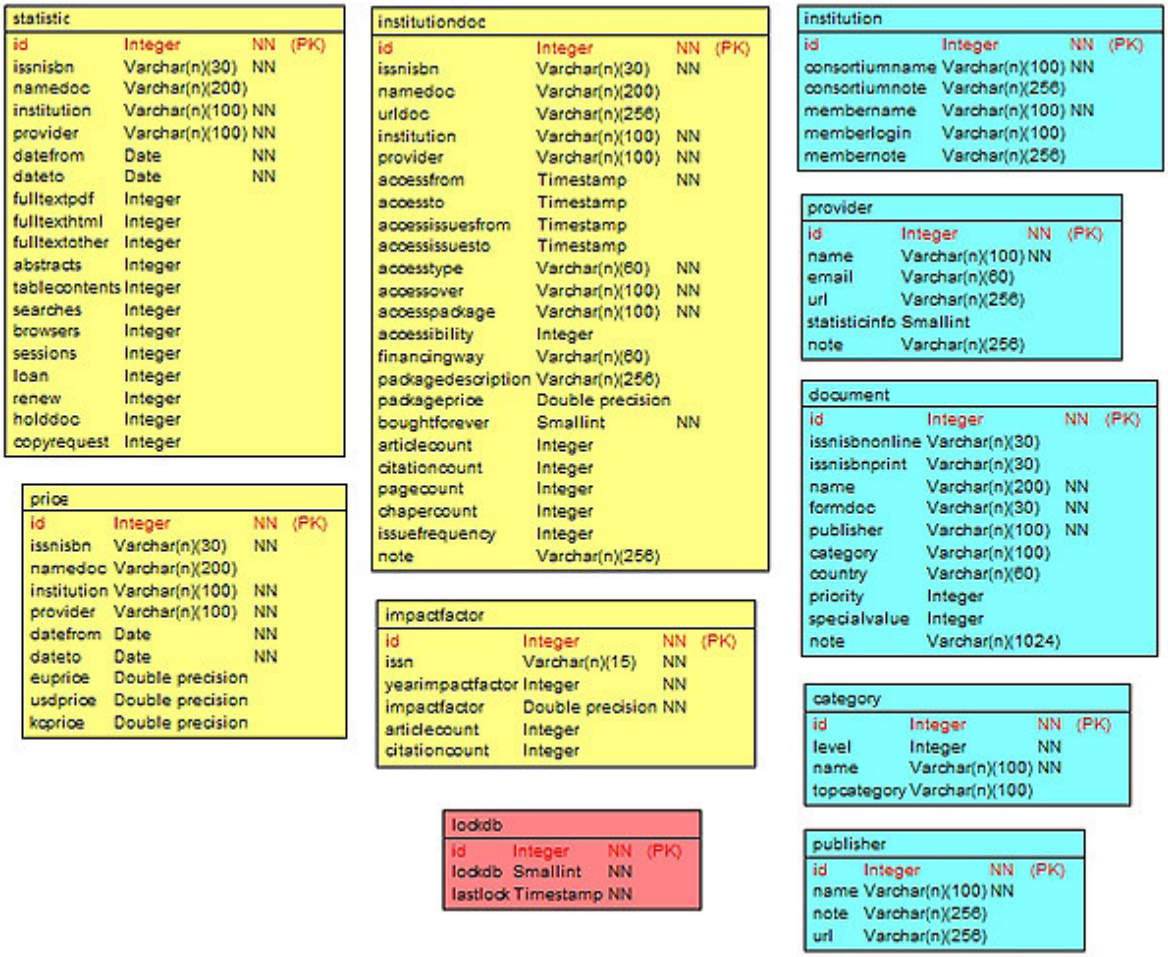
Nevýhody řešení:

- Návrh tří databází
- Nutnost konverzí mezi jednotlivými úrovněmi databáze pomocí datových pump

Vzhledem k požadavkům, které jsou uvedeny v kapitole 4.4, je pro softwarový systém výhodnější druhé řešení, které je přímo zaměřeno na práci se statistickými daty, jejich vyhodnocování a analýzu. Toto řešení nabízí rychlejší vyhodnocování složitějších dotazů, rychlejší import dat do databáze, což ocení zejména koncový uživatel. Dalším plusem této varianty je četnost přidávání a aktualizace dat, která je menší (např. statistická data o využívání informačních zdrojů získávaná měsíčně).

5.3.1 Databáze L0

Databáze L0 bude obsahovat dočasná data, která budou dále zpracována do databáze L1. Aplikace bude do databáze L0 importovat data 1:1, při importu se nebudou řešit žádná integritní omezení ani indexy. Jak je vidět z diagramu na obr. 15, databáze L0 neobsahuje žádné vazby mezi tabulkami. Při návrhu modelu databáze L0 bylo rozhodující rozdělení vstupních dat podle obsahové informace a skutečnost, že vstupní data jsou získávána z různých zdrojů, u nichž nedochází k obsahovému prolnutí.



obr. 15: Databázový model – databáze L0

Popis tabulek

Všechny tabulky obsahují jako primární klíč speciální atribut *id*, který je generován automaticky. S jednotlivými tabulkami databáze L0 seznamuje tabulka 4.

Název tabulky	***Obsah***
Statistic	Statistická data pro elektronické i tištěné dokumenty. Dále obsahuje povinné údaje o dokumentu, instituci a poskytovateli daného dokumentu. Tyto údaje jsou důležité pro transformaci do databáze L1. Důležitými a povinnými atributy jsou *datefrom* a *dateto*, které vymezují období zaznamenání statistických dat.
document	Základní data o dokumentu. Důležitými atributy jsou *issnisbnonline* a *issnisbnprint*. Tyto atributy jednoznačně identifikují daný dokument pomocí hodnoty ISSN v seriálových publikací nebo ISBN v případě monografie. Dalším důležitým povinným atributem je *formdoc*, který přináší informaci o typu dokumentu (např. časopis, kniha, …). Pro dokumenty je definován atribut *priority*, který informuje o důležitosti dokumentu pro danou instituci.
provider	Data o poskytovatelích informačních zdrojů. Poskytovatelé jsou jednoznačně určeni svým jménem – atribut *name*.

Název tabulky	*Obsah*
publisher	Data o vydavatelích informačních zdrojů. Vydavatelé jsou jednoznačně určeni svým jménem – atribut *name*.
price	Data o ceně dokumentu. Dále obsahuje povinné údaje o dokumentu, instituci a poskytovateli daného dokumentu. Tyto údaje jsou důležité pro transformaci do databáze L1. Důležitými a povinnými atributy jsou *datefrom* a *dateto*, které vymezují období, na které se vztahuje uvedená cena.
impactfactor	Data o impakt faktoru časopisu identifikovaného atributem *issn,* který obsahuje hodnotu ISSN daného časopisu. Impakt faktor se uvádí vždy pro určitý rok. Informaci o roce nese povinný atribut *yearimpactfactor*.
category	Data o tématickém rozdělení dokumentů. Kategorie je jednoznačně určena svým jménem a hloubkou umístění v hierarchii kategorií.
institution	Data o institucích a definuje vztah mezi knihovnou a konsorciem. Pokud budeme chtít zaznamenat knihovnu nezávislou na konsorciu, v té chvíli bude název konsorcia stejný jako název knihovny. Povinnými atributy jsou názvy institucí *consortiumname* a *membername*.
institutiondoc	Data o dokumentech, které jsou přístupné knihovně. Obsahuje detailnější informace o přístupu k dokumentu – období, po které má knihovna přístup k dokumentu, rozsah zpřístupněných dokumentů, typ přístupu.
lockdb	Zastává funkci semaforu, řídícího spouštění transakcí v databázi. Pro udržení konzistence dat a zamezení konfliktům v datech může v jedné chvíli probíhat pouze jedna transakce. Hodnota uložená v atributu *lockdb* informuje aplikaci o probíhající transakci, která vkládá, aktualizuje nebo maže data v databázích. Před spuštěním transakce se dotazem *„SELECT * from lockdb FOR UPDATE“* zjistí, zda jsou databáze odemčeny a nastaví se příznak zámku. Po skončení transakce budou databáze odemčeny změnou hodnoty atributu lockdb.

tabulka 4: Popis tabulek databáze L0

Popis atributů

Popis jednotlivých atributů je uveden v dokumentu „databaze.pdf“ na přiloženém CD.

Hodnoty atributů

Atributy, které nemusí mít vyplněnou hodnotu, používají jako výchozí hodnotu *null*.

Atribut *id*, který je primárním klíčem tabulek, je generován automaticky a jeho výchozí hodnota je 1.

Tabulka '*lockdb*' obsahuje atribut *lockdb*, který může nabývat hodnoty 1 nebo 0, a atribut *lastlock*, který informuje o čase posledního zamčení databáze. Hodnota 0 zamyká databázi po dobu běhu transakce. Hodnota 1 informuje, že neprobíhá žádná transakce, která by bránila spuštění požadovaných operací nad databázemi.

Indexy

Index je databázová struktura sloužící k optimalizaci zpracování dat (zrychluje vyhledávací a dotazovací procesy, definuje unikátní hodnoty sloupců tabulky). Indexy můžeme rozdělit podle typu, který blíže určuje, jakým způsobem má být přístup k primárním datům příslušné databázové tabulky optimalizován.

V databázi L0 jsou použity jen primární indexy, které jsou vytvořeny nad sloupcem obsahujícím primární klíč.

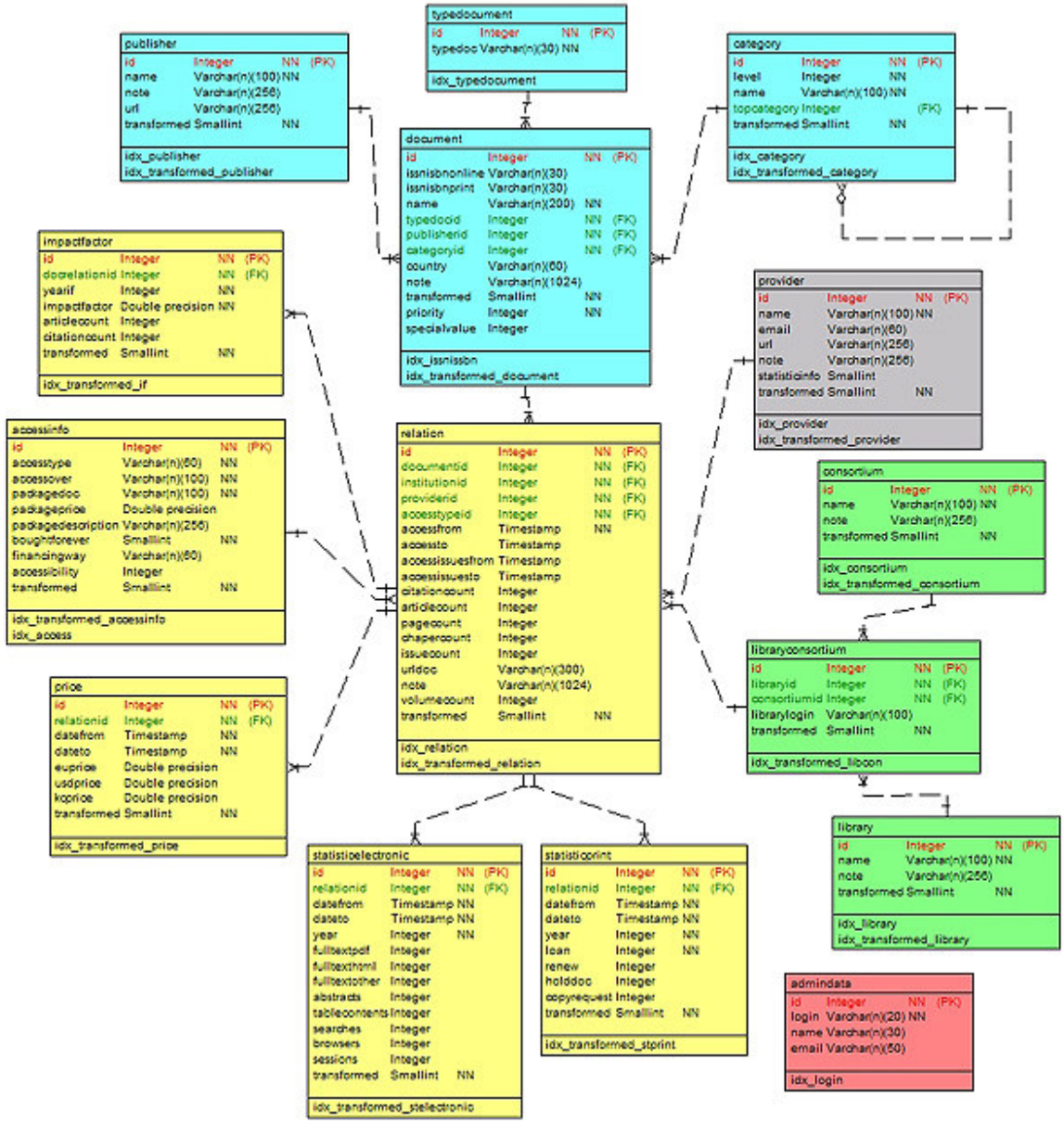
Uživatelské role a jejich práva

Přístup do databáze L0 bude umožněn jen uživateli, který bude mít nastavenou roli *librarex_admin.* Tato role dává uživateli všechna práva pro práci s databází a je určena pro administrátory aplikace.

5.3.2 Databáze L1

Databáze L1 splňuje 3NF a data v ní uložená musí být očištěná a konzistentní. Data z ní není možné mazat, editace je omezena na vybrané údaje. K datům se bude přistupovat jen ve chvíli, kdy bude prováděn převod dat mezi databázemi. Databáze L1 je plněna daty z databáze L0. Ve chvíli plnění databáze dochází k čištění dat, které zahrnuje např. odstranění duplicitních záznamů a vzájemné provázání importovaných záznamů (dohledání záznamů a nastavení cizích klíčů) .

Model databáze L1 je zobrazen v diagramu na obr. 16.



obr. 16: Databázový model – databáze L1

Popis tabulek

Všechny tabulky obsahují jako primární klíč speciální atribut *id*, který je generován automaticky. S jednotlivými tabulkami databáze L1 seznamuje tabulka 5.

Název tabulky	***Obsah***
document	Data o dokumentu. Důležitými atributy jsou *issnisbnonline* a *issnisbnprint*. Tyto atributy jednoznačně identifikují daný dokument pomocí hodnoty ISSN v případě časopisů nebo ISBN. Dalším důležitým povinným atributem je *typedocid*, který přináší informaci o typu dokumentu.
typedocument	Data o existujících typech dokumentů, s kterými databáze pracuje – např. časopis, kniha, encyklopedie, atd.
publisher	Data o vydavatelích dokumentů. Jednoznačným identifikátorem vydavatele je jeho jméno, kterému odpovídá atribut *name*.

Název tabulky	*Obsah*
category	Data o tématickém zařazení dokumentů, např. dokument, zabývající se analytickou chemií můžeme zařadit právě do kategorie analytická chemie, tato kategorie je podkategorií chemie, která dále spadá pod kategorii přírodní vědy. Kategorie je jednoznačně určena svým jménem a hloubkou umístění v hierarchii kategorií.
provider	Data o poskytovatelích informačních zdrojů. Poskytovatelé jsou jednoznačně určeni svým jménem – atribut *name.*
library	Data o knihovně. Knihovna je jednoznačně určena svým jménem – atribut *name*.
consortium	Data o konsorciu. Konsorcium je jednoznačně určeno svým jménem – atribut *name*.
libraryconsortium	Data o vztahu knihoven a konsorcií. Říká, že knihovna je členem těchto konsorcií, a naopak, že konsorcium má tyto členy.
relation	Data o přístupu knihovny k danému dokumentu. Umožňuje provázat s tímto přístupem další důležitá data (poskytovatele, typ přístupu, cenu, impakt faktor, statistická data). Jde o centrální tabulku této databáze.
accessinfo	Data o typu přístupu dané instituce k dokumentu. Informuje o tom, zda se jedná o přístup k tištěnému či elektronickému dokumentu, zda je přístup v rámci konsorcia, zda byl dokument pořízen v rámci balíčku nebo samostatně, zda byl dokument koupen nebo je předplácen.
price	Data o ceně jednotlivých dokumentů v konkrétním období. Během přístupu k dokumentu může dojít ke změně ceny, v té chvíli bude do této tabulky vložen nový záznam s provázáním na příslušný záznam tabulky relation.
impactfactor	Data o impakt faktoru časopisů. Impakt faktor je vždy vypočítáván pro konkrétní rok. Tabulka je provázána s tabulkou 'relation' jen ve chvíli, kdy se období navzájem prolínají.
statisticelectronic	Statistická data týkající se využívání elektronických informačních zdrojů. Statistická data elektronických a tištěných zdrojů byla rozdělena do separátních tabulek, protože ve většině případů budeme mít pro daný dokument a období k dispozici jen jeden typ těchto dat.
statisticprint	Statistická data týkající se využívání tištěných informačních zdrojů.
admindata	Data o uživateli aplikace. Tato tabulka je důležitá při přihlašování uživatele do aplikace. V té chvíli se ověřuje existence uživatele podle jeho uživatelského jména (loginu).

tabulka 5: Popis tabulek databáze L1

Popis atributů

Popis jednotlivých atributů je uveden v dokumentu „databaze.pdf“ na přiloženém CD.

Zde se věnuji pouze jednomu z atributů, který se vyskytuje ve většině tabulek. Jedná se o atribut *transformed*. Tento atribut nese informaci o tom, zda byl daný záznam po vložení či po poslední změně zpracován transformací, která propaguje změny do databáze L2.

Hodnoty dat

Atribut id je generován automaticky a jeho výchozí hodnota je 1.

Atribut *transformed* může nabývat hodnoty 0 nebo 1, které nahrazují boolean hodnoty. Hodnota 0 informuje o tom, že daný záznam nebyl po vložení nebo aktualizaci transformován do databáze L2. Hodnota 1 naopak informuje, že již proběhl převod záznamu do databáze L2.

Tabulka '*typedocument*' bude naplněna daty při instalaci aplikace. Tabulka bude obsahovat následující typy dokumentů: časopisy, knihy, encyklopedie, seriály. Výčet typů dokumentů není konečný a může být rozšířen o další položky.

Tabulka '*admindata*' bude také naplněna daty při instalaci aplikace. Do tabulky bude vložena informace o administrátorovi aplikace.

Atributy, které nemusí mít vyplněnou hodnotu, mají implicitní hodnotu *null*.

Relace

Vztahy mezi tabulkami jsou zobrazeny v diagramu na obr. 16. Význam jednotlivých symbolů použitých pro propojení tabulek vysvětluje tabulka 6.

Vysvětlení použitých symbolů pro vztahy mezi tabulkami	
⊣	povinná účast – vztahy typu 1:1, 1:N
⊣⋉	povinná účast – vztahy typu N:1, N:M
⊸⋉	volitelná účast – vztahy typu N:1, N:M

tabulka 6: Symboly pro vztahy mezi databázovými tabulkami

Nejčastěji je v databázi použit vztah typu 1:N. Dále je použit samoreferenční vztah typu 1:N v tabulce '*category*', kde se kategorie provazuje se svojí hierarchicky nadřazenou kategorií.

Indexy

V databázi L1 jsou kromě indexů nad primárními klíči a cizími klíči použity indexy nad jednotlivými sloupci, indexy nad více sloupci a unikátní indexy. Důvodem pro vytvoření dalších indexů jsou opakované dotazy se stejnou podmínkou nad konkrétním sloupcem tabulky. V diagramu na obr. 16 jsou tyto indexy zaznamenány vždy pod čarou ve spodní části konkrétní entity (název indexu začíná prefixem „idx_“).

Pro indexy, které jsou vytvořeny nad sloupcem *transformed (idx_transformed_)*, jsem zvolila bitmapový index, protože tento sloupec nabývá jen dvou hodnot. Pokud databázový stroj bitmapové indexy nepodporuje, bude použit B–strom index. Pro ostatní indexy je použit B–strom index.

Podrobnější popis vytvořených indexů je v dokumentu „databaze.pdf“ na přiloženém CD.

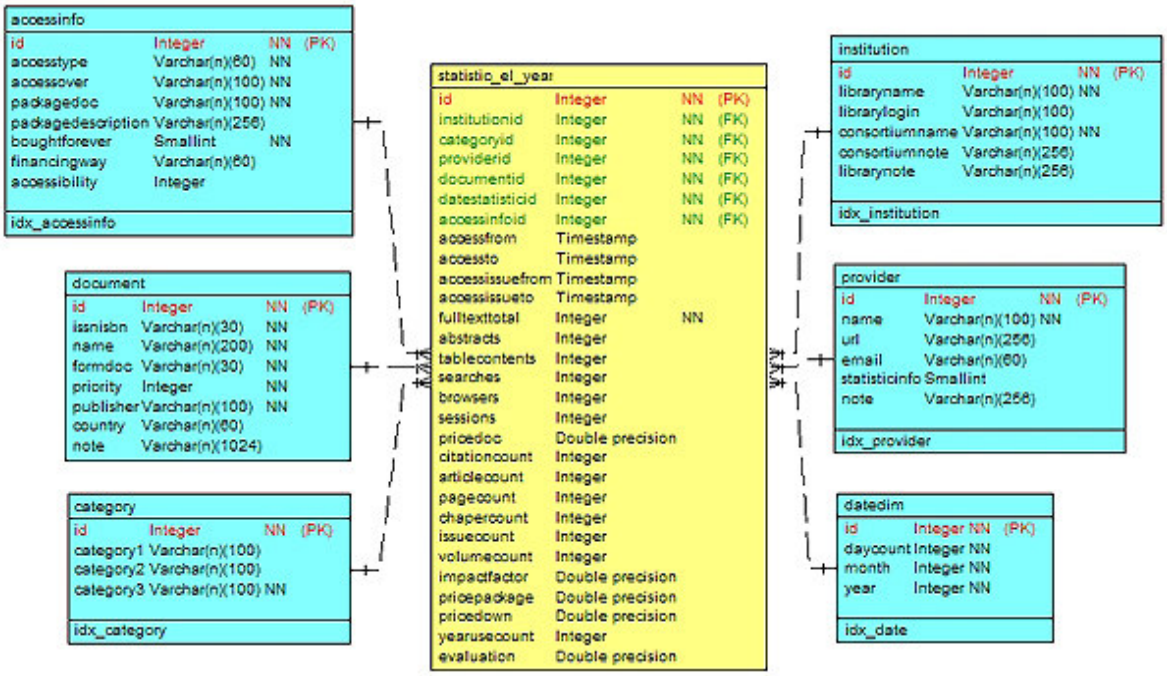
Uživatelské role a jejich práva

Přistupovat do databáze L1 bude moci jen uživatel, který bude mít nastavenou roli *librarex_admin.* Tato role dává uživateli všechna práva pro práci s databází a je určena pro administrátora aplikace.

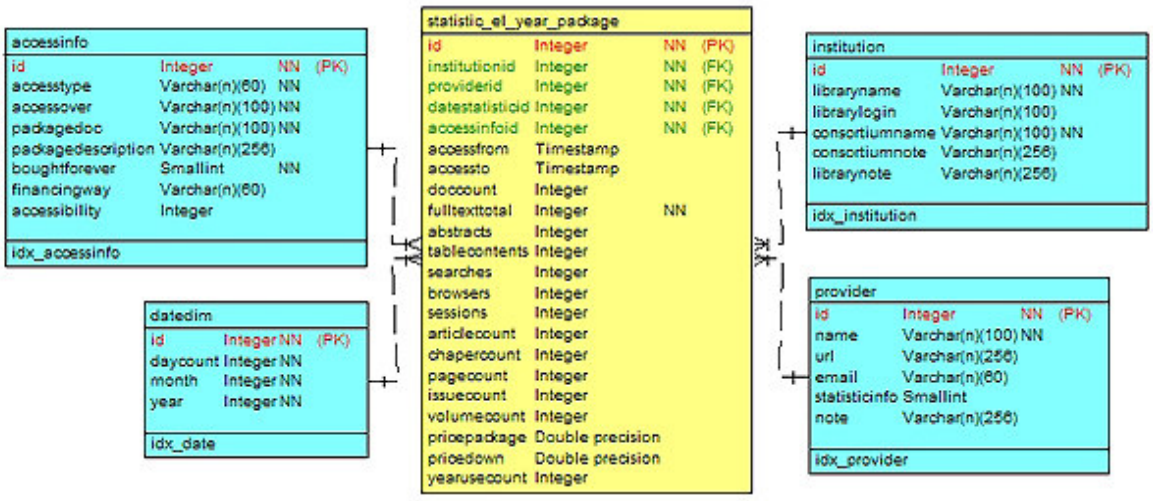
K vložení nových záznamů, nebo editování existujících záznamů v databázi L1 dochází jen během transformace dat z databáze L0.

5.3.3 Databáze L2

Databáze L2 je určena pro dotazování a rychlé získávání informací z dat pro reporty a analýzu. Z logického pohledu je databáze členěna do schémat, která odpovídají analyzované oblasti. V oblasti datových skladů jsou tato schémata nazývána DataMart (Datové tržiště). Název "*Datové tržiště*" používám v další části textu pro označení jednotlivých logických celků.



obr. 17:Schéma hvězdy datového tržiště SEY



obr. 18: Schéma hvězdy datového tržiště SEYP

Pro uspořádání faktových a dimenzionálních tabulek jsem zvolila schéma ***souhvězdí***, které vzniklo z více ***hvězd*** sdílejících některé dimenzionální tabulky.

Schémata dvou datových tržišť jsou zobrazeny na obr. 17 a obr. 18. Datová tržiště, i když sdílejí dimenzionální tabulky *accessinfo, datedim, institution* a *provider* (dimenzionální tabulky mají žluté pozadí), jsou z důvodu přehlednosti zobrazena separátně. Celkem pracujeme s pětí logickými celky, se kterými se seznámíme v následujícím textu. Schémata zbývajících datových tržišť jsou v dokumentu „datova_trziste.pdf“ na přiloženém CD.

Datové tržiště	***Dimenze***					
	Datum	***Dokument***	***Instituce***	***Poskytovatel***	***Přístup***	***Kategorie***
SEY	x	x	x	x	x	x
SEYP	x		x	x	x	
SEM	x	x	x	x	x	x
SPY	x	x	x	x	x	x
SPM	x	x	x	x	x	x

tabulka 7: Provázání datových tržišť s dimenzemi

S datovými tržišti a seznamem dimenzí seznamuje tabulka 7. Pro každé datové tržiště je křížkem zaznamenáno, s kterými dimenzemi pracuje.

Následuje popis jednotlivých datových tržišť a dimenzi.

Popis datových tržišť

Při návrhu datových tržišť jsem použila schéma hvězda. Toto schéma obsahuje jednu centrální faktovou tabulku, podle které jsem odvodila název datového tržiště. Popis jednotlivých datových tržišť obsahuje tabulka 8.

Návrh datových tržišť se opírá o analyzované požadavky na aplikaci a potřebné znalosti o problematice, které jsou uvedené v kapitole 4.1.

Název datového tržiště	*Popis*
SEY	Určeno pro získávání ročních analytických informací o elektronických dokumentech. Název faktové tabulky tohoto datového tržiště je 'statistic_el_year'.
SEYP	Určeno pro získávání ročních analytických informací o balíčcích elektronických dokumentů. Název faktové tabulky tohoto datového tržiště je 'statistic_el_year_package'.
SEM	Určeno pro získávání informací o využívání elektronických dokumentů během jednotlivých měsíců. Faktová tabulka neobsahuje informace o ceně, impakt faktoru, rozsahu dokumentu. Jedním z důvodů proč toto datové tržiště nepracuje s uvedenými údaji, je skutečnost, že data jsou poskytována za celý rok a tím jsou měsíční hodnoty neznámé. Název faktové tabulky tohoto datového tržiště je 'statistic_el_month'.
SPY	Určeno pro získávání ročních analytických informací o tištěných dokumentech. U tištěných dokumentů se předpokládán fyzické vlastnictví dokumentu navždy. Název faktové tabulky tohoto datového tržiště je 'statistic_print_year'.
SPM	Určeno pro získávání informací o využívání tištěných dokumentů. Faktová tabulka neobsahuje informace o ceně a rozsahu dokumentu. Název faktové tabulky tohoto datového tržiště je 'statistic_print_month'.

tabulka 8: Popis datový tržišť

Popis faktových tabulek

Stručná charakteristika dat ve faktové tabulce odpovídá popisu datového tržiště, ve kterém je centrálním prvkem. Popis jednotlivých atributů faktových tabulek je uveden v dokumentu „databaze.pdf“ na přiloženém CD.

Popis dimenzí

Dimenze jsou tabulky zachycují úhel pohledu na sledované ukazatele ve faktových tabulkách. S jednotlivými dimenzemi seznamuje tabulka 9.

Název dimenze	*Obsah*
Datum	Časové údaje, které upřesňují období sledovaných statistických dat. Tato dimenze obsahuje hierarchii – rok, měsíc. Nejmenší granularita je tedy měsíc. V dimenzionální tabulce se zaznamenává informace o počtu dní, které dané období zahrnuje. Tento atribut je využit ve chvíli, kdy období neodpovídá přesně měsíci nebo roku. Název tabulky této dimenze je 'datedim'.
Dokument	Data o dokumentech. Název tabulky této dimenze je 'document'.
Instituce	Data o institucích (konsorciích, knihovnách).

Název dimenze	*Obsah*
	Název tabulky této dimenze je 'instituce'.
Poskytovatel	Data o poskytovatelích dokumentů. Název tabulky této dimenze je 'provider'.
Přístup	Data, informující o formě přístupu k dokumentům. Název tabulky této dimenze je 'accessinfo‘.
Kategorie	Data o tématickém zařazení dokumentů. Pokud si představíme kategorie v datové struktuře strom, obsah záznamu v tabulce můžeme přirovnat k názvům kategorií při cestě od listu ke kořeni stromu. Předpokladem je hloubka stromu 3. Toto omezení bylo stanoveno po nastudování související problematiky (konspekt, knihovní fond) a následných konzultacích. Název tabulky této dimenze je 'category'.

tabulka 9: Popis dimenzí

Popis atributů

Popis atributů faktových tabulek a dimenzionálních tabulek je uveden v dokumentu „databaze.pdf“ na přiloženém CD.

Hodnoty dat

Atribut id je generován automaticky a jeho výchozí hodnota je 1.

Implicitní hodnota atributů, které nemusí být vyplněné je hodnota *null*.

Relace

Na obr. 17 a obr. 18 lze vidět vztahy mezi tabulkami pro datová tržiště SEY a SEYP. Použité znaky pro vztahy mezi tabulkami jsou stejné jako v databázi L1 a byly popsány již u této databáze.

Indexy

Základní přehled indexů je uvedeno v předchozích dvou kapitolách zabývajících se návrhem databáze L0 a databáze L1.

V databázi L2 jsou použity indexy nad primárními a cizími klíči a v tabulkách dimenzí jsou použity indexy nad jednotlivými sloupci, indexy nad více sloupci a unikátní indexy. V diagramech na obr. 17 a obr. 18 jsou tyto indexy zaznamenány vždy pod čarou ve spodní části dimenzionální tabulky.

Podrobnější popis vytvořených indexů je v dokumentu „databaze.pdf“ na přiloženém CD.

Uživatelské role a jejich práva

Přístup do databáze L2 mají uživatelé s rolí *librarex_admin* nebo *librarex_librarian.* Role *librarex_admin* dává uživateli všechna práva pro práci s databází a je určena pro administrátory aplikace. K editaci a mazání dat v databázi L2 dochází jen v průběhu převodu dat z databáze L1, který řídí aplikace. Role *librarex_librarian* je určená pro koncové uživatele, kteří se nestarají o plnění databází daty. Jejich práce je zaměřena na získávání

informací z dat, z čehož vyplývá i nastavení práv. Práva tohoto uživatele jsou omezena jen na dotazování se nad daty.

5.3.4 Struktura XML

Data pro import do databáze L0 jsou k dispozici v různých formátech a jsou různého obsahu. S novým poskytovatelem se navíc může objevit nový formát a obsah datového souboru se statistickými daty. Je tedy nutné řešit tuto situaci obecněji, abychom uměli zpracovat i nová data. Jako řešení jsem zvolila převod dat do jednotného formátu a až teprve tento formát bude využit pro import dat do databáze. Pro sjednocení formátu dat jsem vybrala jazyk XML, který je pro tento případ ideální.

Do databáze jsou importována data, která lze rozdělit na skupiny obsahující data o:

- dokumentech (ISSN/ISBN, název, vydavatel, atd.)
- využívání dokumentů
- ceně
- impakt faktoru
- vydavatelích
- poskytovatelích
- tématických kategoriích
- instituci a jejích dokumentech

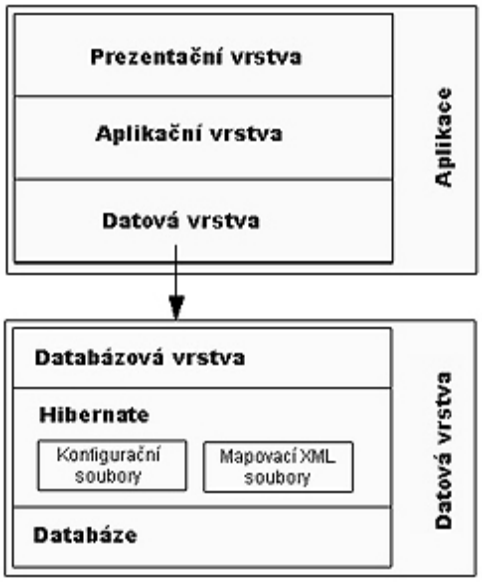
V úvahu přicházela dvě řešení jak navrhnout XML soubory pro přípravu dat.

1. Vytvořit jeden XML soubor, který bude obsahovat všechna data pro import do databáze. Výhodou tohoto řešení je existence jen jednoho typu XML souboru. Nevýhodou je, že vzniklá struktura XML se stává nepřehledná a vzhledem k provázání různých dat a obsahu více časových údajů i nešikovná. Data pro jednotlivé tématické skupiny jsou získávána samostatně. Dalším problémem, který by se v tomto řešení objevil, je, že každá skupina obsahuje jiné povinné údaje a jejich kontrola by při využití tohoto typu řešení byla složitá a časově náročná.
2. Vytvořit více XML souborů podle výše uvedeného obsahového rozdělení vstupních dat. Výhodou tohoto řešení je přehledné rozdělení do skupin podle obsahu získávaných vstupních dat. To umožní uživateli, aby se lépe orientoval ve vytvořených XML souborech. Dále se zjednoduší kontrola povinných údajů, protože pro každou skupinu známe její povinné údaje a víme o jakou skupinu se jedná.

Zvolila jsem druhé řešení, protože vzhledem k povaze vstupních dat je výhodnější. Jednotlivé typy XML souborů tedy odpovídají obsahovému rozdělení vstupních dat. Návrh databázového modelu pro databázi L0 je také rozdělen podle tématického obsahu vstupních dat. Tuto skutečnost jsem využila při návrhu struktury XML souborů a výsledné soubory mapují jednotlivé tabulky databáze L0. Pro bližší seznámení se strukturou jednotlivých XML souborů můžete využít dokument „strukturaXML.pdf“ na přiloženém CD.

5.4 Klient GUI

Klienta můžeme rozdělit do tří základních vrstev. Rozděleni je znázorněno na obr. 19.



obr. 19: Architektura klienta – vrstvy

Prezentační vrstva je nejvyšší vrstva, která zajišťuje komunikaci s uživatelem. Uživateli jsou prostřednictvím GUI zobrazovány dialogy, které nabízejí funkce aplikace. Implementováno pomocí Model View Controller.

Aplikační vrstva reprezentuje logiku aplikace. Součástí této vrstvy jsou algoritmy definující např. zpracování vstupních dat, import dat do databáze, transformaci dat mezi databázemi, způsob kontroly čistoty dat, atd..

Datová vrstva zajišťuje komunikaci mezi aplikační vrstvou a databází. Podrobnější členění této vrstvy je vidět na obr. 19.

- ***Databázová vrstva*** poskytuje vlastní API, které překrývá Hibernate API, pro jednodušší prácí s databází. Databázová vrstva maximálně odstiňuje vyšší vrstvy od vlastních databázových modelů, poskytuje metody pro získávání dat z databáze a správu dat v databázi (vložení, editace, transformace). Databázová vrstva také zajišťuje verifikaci uživatele při přihlášení do aplikace.

- ***Hibernate –*** součástí vrstvy implementované pomocí Hibernatu jsou mapovací a konfigurační soubory. Pomocí mapovacích XML souborů je definováno provázání objektů (třídy v Javě) s tabulkami daného databázového schématu. Konfigurační soubory obsahují základní vlastnosti, definují např. přístupy k mapovacím souborům, připojení k databázi, zobrazování ladících výpisů a jiné vlastnosti.

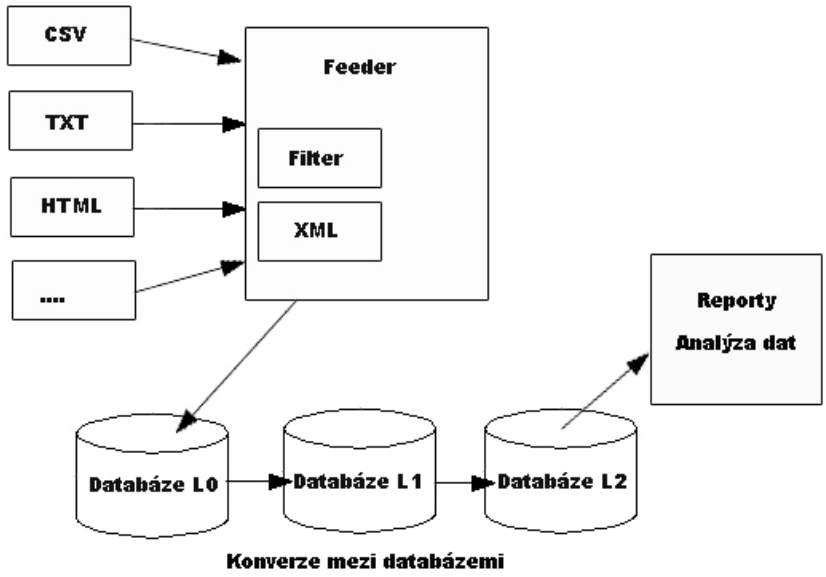
- ***Databáze –*** perzistentní úložiště dat.

Z pohledu uživatele lze funkčnost aplikace rozdělit do tří základních celků. Toto rozdělení je patrné z obr. 20.

- ***Sběr dat (Feeder)*** – nástroj pro práci s datovými soubory, jejich zpracování a uložení do databáze L0. Důležitým úkolem Feederu je převod vstupních dat do jednotného formátu. Pro uložení dat do jednotné podoby je použit jazyk XML. Bylo nezbytné

zajistit rozšiřitelnost o zpracování nových formátů a obsahově odlišných typů souborů. Od tohoto požadavku se odvíjel i návrh filtrů, které definují vlastnosti převodů. Feeder dále zajišťuje správu vytvořených XML souborů a následný import dat z XML do databáze L0. Jednou ze zajímavých funkcí Feederu je možnost vytvářet a spouštět úkoly hromadného zpracování dat, což ušetří uživateli mnoho času.

- ***Konverze mezi databázemi*** – transformace dat mezi databázemi spojená s kontrolou čistoty dat a duplicit.
- ***Reporty*** – vytváření reportů na základě zadaných parametrů.



obr. 20: Architektura systému – tok dat

Následující kapitoly se věnují návrhu a popisu jednotlivých částí systému.

5.4.1 Grafické uživatelské rozhraní

Klient je spustitelný s grafickým uživatelským rozhraním (dále jen GUI) a z příkazové řádky. Příkazová řádka omezuje funkčnost na:

- generování XML soubory prostřednictvím filtrů, dávek a úkolů
- import dat do databáze
- převod dat mezi databázemi

Výhodou práce s aplikací bez GUI je možnost naprogramovat skripty (např. shell skripty), které lze následně využít pro zautomatizování některých úkonů např. zpracování měsíčních statistik.

Aplikace spuštěná s GUI dává uživateli k dispozici veškerou funkčnost programu od správy filtrů po generování reportů. Výhodou práce v GUI je intuitivní ovládání aplikace prostřednictvím dialogů, zobrazení postgress baru při vykonávání dlouho trvající operace, atd.

Model–View–Controller

GUI aplikace LibrarEX bude postaveno na Model–View–Controller (dále jen MVC) architektuře. MVC rozděluje aplikaci do tří nezávislých komponent: model (datový model), view (uživatelské rozhraní) a controller (řídící logika). Použití jen jednoho MVC pro celou aplikaci by vzhledem k množství dialogů a jejich řídících prvků bylo nepřehledné. Rozhodla jsem se pro každý větší dialog aplikace použít vlastní MVC. Mezi výhody použití více MVC patří kromě lepší orientace ve zdrojovém kódu také možnost jednoduchých změn jednotlivých dialogů, dále i celého grafického návrhu aplikace. Komunikace mezi jednotlivými MVC bude řešena prostřednictvím modelů.

Dlouhotrvající operace

Ve chvíli, kdy bude spuštěna dlouhotrvající operace (import dat do databáze, převod dat mezi databázemi, atd.) je důležité informovat uživatele o průběhu této operace. Další problém, který je ve spojení s dlouhotrvajícími operacemi nutné řešit, je nepřekreslování „zamrzání" oken aplikace. Tyto problémy lze odstranit pomocí více vláken.

Pro implementaci GUI bude použita knihovna SWING, bude tedy důležitá správná práce s vlákny. Práce s GUI je obsluhována z jediného vlákna, tzv. Event Dispatch Thread (dále EDT). Toto vlákno obsahuje frontu událostí, které postupně zpracovává v pořadí, v jakém byly do fronty zařazeny. Abychom se vyhnuli problémům se zamrzáním aplikace a deadlockům je nutné dodržet dvě podmínky a to:

- všechny operace s komponentami Swingu budou prováděny ve vlákně EDT
- časově náročné operace by neměly být prováděny ve vlákně EDT

Pro usnadnění práce s vlákny bude použita třída SwingWorker.

Pro správu vláken jsou použity třídy Task, TaskScheduler, TaskComplete a ProgressBar. Task je abstraktní třída, která sjednocuje provádění dlouhotrvajících operací v novém vlákně. Pro práci s vlákny používá třídu SwingWorker a informuje o změnách stavu podle návrhového vzoru Observer. Hlavní metodou této třídy je abstraktní metoda task(), která implementuje konkrétní výpočet. Třída ProgressBar implementuje Observer a informuje uživatele o průběhu operace. Pokud při výpočtu dojde k chybě, je vyvolána výjimka, která je zpracována touto třídou. Průběh některých výpočtů může uživatel zastavit. Třída TaskScheduler zajišťuje, aby v jedné chvíli běžela spuštěná jen jedna dlouhotrvající operace. Metoda runTask() zajišťuje zmíněnou kontrolu, zaregistrování instance třídy Task pozorovatelské třídě ProgressBar a spuštění výpočtu. Třída TaskComplete obsahuje operace, které musí být vykonány po skončení výpočtu.

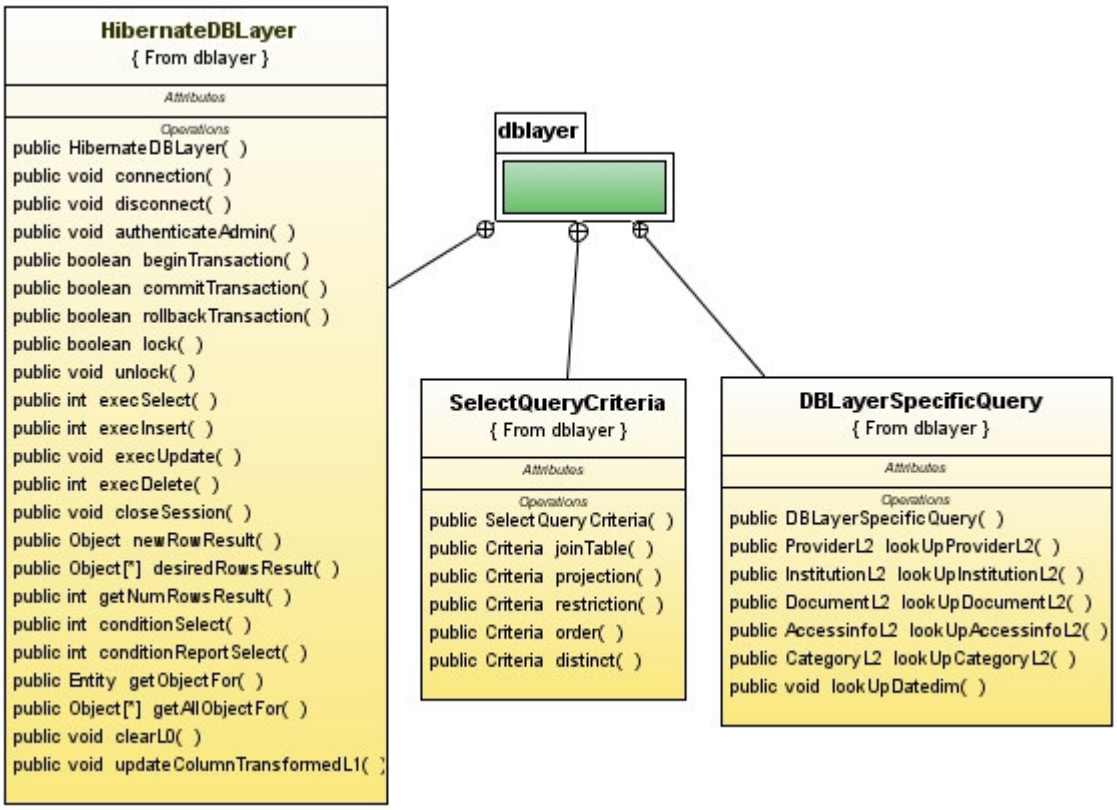
Výcejazyčné GUI

Jedním z požadavků je podpora více jazyků. Rozhodla jsem se lokalizaci řešit shromážděním veškerých hlášení a popisků GUI na jednom místě ve více verzích, kde jedna verze odpovídá jednomu jazyku. Přístup k libovolným textům bude řešen podle zadaného klíče a aktuálního nebo nastaveného locale. Tento způsob řešení umožní snadnou editaci a také rozšíření o další jazyky. V základní implementaci aplikace podporuje český a anglický jazyk.

5.4.2 Databázová vrstva

Databázová vrstva poskytuje vlastní API pro jednodušší práci s databází. Toto API je nástavbou nad Hibernate API a poskytuje řadu často používaných metod, čímž se zamezí opakovanému psaní kódu. Metody také usnadňují vykonání složitější operací nad databází.

Diagram na obr. 21 zobrazuje komponentu dblayer a jednotlivé třídy implementující databázovou vrstvu. V diagramu jsou uvedeny důležité metody jednotlivých tříd. Podrobnější informace o parametrech metod a samotných metodách jsou k dispozici v programátorské dokumentaci.



obr. 21: Databázová vrstva – návrh tříd

Třída HibernateDBLayer zajišťuje připojení a odpojení od databáze, správu transakcí, zamykání databáze, vykonání INSERT, UPDATE a DELETE příkazů, práci s výsledkem a smazání či aktualizaci všech dat databáze.

Pro práci s databází musí být vytvořena instance třídy HibernateDBLayer. Parametry pro připojení k databázi jsou získávány z konfiguračního souboru hibernate.cfg.xml zvolené databáze, načteného nastavení aplikace a z přihlašovacího dialogu, kde uživatel zadá login a heslo. Během připojení k databázi proběhne načtení mapovacích souborů a autentifikace uživatele. Práce s databází je ukončena funkcí disconnect(), která uzavře otevřené session a odpojí uživatele od databáze.

Správa transakcí je řešena pomocí metod beginTransaction(), commitTransaction() a rollbackTransaction(). Hibernate ORM nabízí metody pro práci s krátkými transakcemi. Je nutné metody implementovat tak, aby uměly pracovat i s dlouhotrvajícími transakcemi a zajistit běh jen jedné transakce v daném čase.

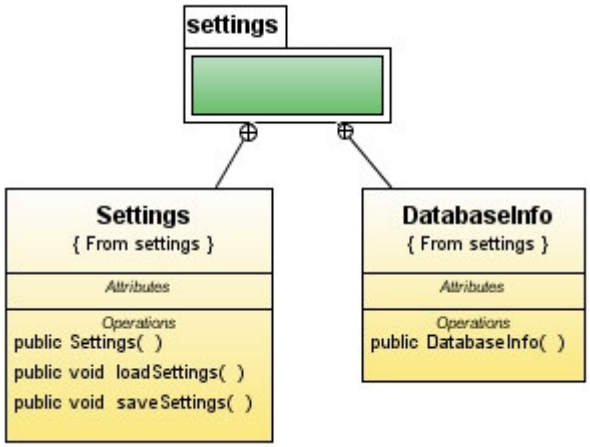
Během provádění operací INSERT, UPDATE a DELETE jsou databáze uzamčeny. Zamykání a odmykání databází řeší funkce lock() a unlock(). Vyhodnocení dotazu, vložení, aktualizace či smazání záznamu je prováděno metodami execSelect(), execInsert(), execUpdate() a execDelete(). Výběr dat z databáze je trochu komplikovanější, je nutné implementovat metody pro definování criteriaAPI dotazu. Dále jsou důležité metody pro práci s výsledkem newRowResult(), desiredRowsResult(). Pro získávání dat výsledků je použito lazy načítání objektů. Je tedy nutné si po dobu práce s výsledkem držet otevřenou session, kterou uzvíráme pomocí funkce closeSession().

Třída SelectQueryCriteria zajišťuje nastavení podmínek SELECT dotazu. Pro získávání dat z databáze jsem se rozhodla použít omezujících podmínek v Javě (criteria API) nabízené Hibernate. Metody této třídy zajišťují konstrukci kritérií dotazu a to spojení tabulek, projekci, restrikci, jednoznačnost a uspořádání výsledku.

Třída DBLayerSpecificQuery obsahuje často používané metody při vyhledávání konkrétních dat databáze.

5.4.3 Nastavení aplikace

Uživatel může před přihlášením do aplikace nastavit několik základních parametrů. Mezi tyto parametry patří volba jazyka aplikace, informace o databázích, ke kterým se aplikace připojuje, a kódování příkazové řádky, aby při spuštění aplikace z příkazové řádky byly správně zobrazovány české znaky. Zadané nastavení je uloženo do XML souboru a načítáno při každém spuštění aplikace. Návrh tříd pro správu nastavení je zobrazen na obr. 22.



obr. 22: Nastavení aplikace – návrh tříd

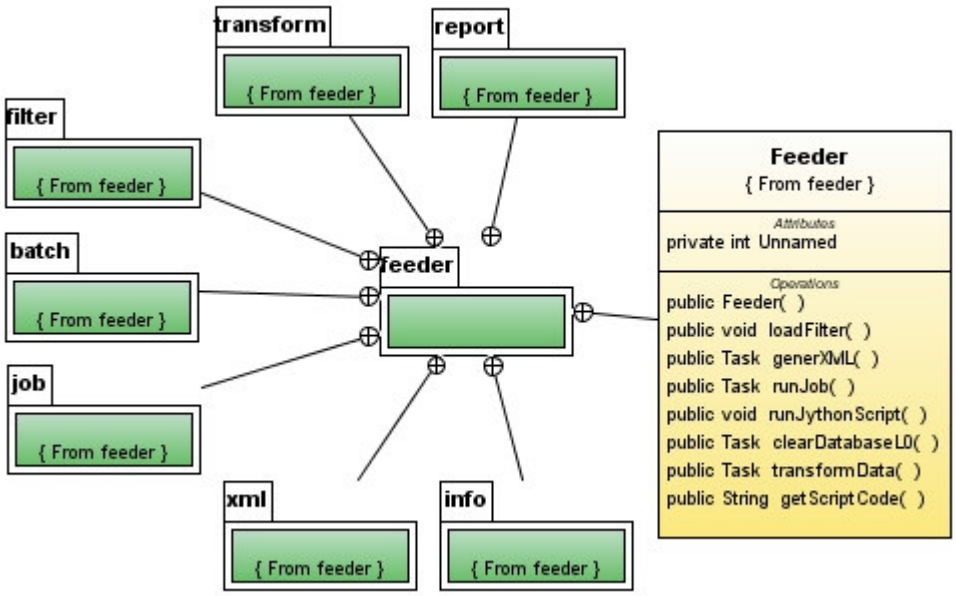
5.4.4 Přihlášení

Přístup do aplikace (klient GUI) je zabezpečen heslem. Uživatel musí zadat přihlašovací údaje, které jsou ověřeny úspěšným přihlášením do databáze L1 a k následným ověřením uživatelského jména v tabulce admindata. Přihlašovací údaje nejsou tedy uloženy nikde na disku počítače, čímž se zvýší bezpečnost aplikace. Přihlášením do aplikace se vytváří instance třídy login, ve které jsou uloženy přihlašovací parametry pro budoucí připojení k databázi L0, L1 i L2. Po autentifikaci uživatele a stažení dat pro naplnění comboboxů v dialozích aplikace, je uživatel od databáze odpojen. Opětovné připojení k databázi proběhne jen při spuštění importu dat do databáze (databáze L0), při spuštění transformace dat mezi databázemi (databáze L0, L1 a L2) nebo pro získání dat reportu (databáze L2).

5.4.5 Feeder

Feeder je název pro hlavní MVC, který pracuje jako centrální bod aplikace pro spouštění důležitých funkcí programu a uložení základních dat. Feeder MVC bude implementovat hlavní okno aplikace, které umožňuje uživateli ovládat aplikaci, přístup k dialogům, atd..

V diagramu na obr. 23 je zobrazen balíček feeder, který seskupuje balíčky filter (správa filtru), batch (správa dávek), job (správa úkolů), xml (správa XML), transform (transformace dat mezi databázemi), report (vytváření reportů) a info (dialogy pro zobrazení informací o generování XML a transformací dat mezi databázemi).

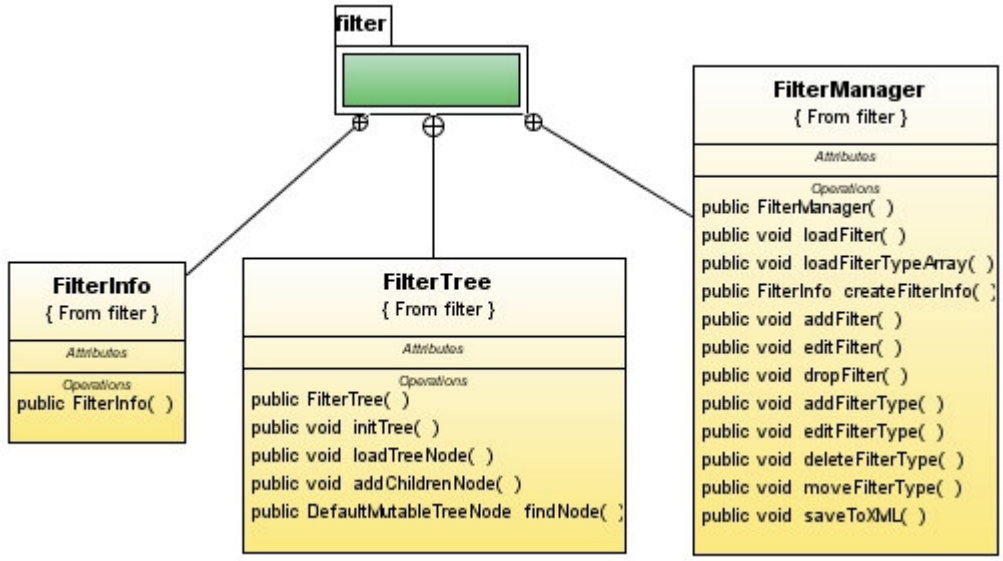


obr. 23: Komponenta feeder

5.4.6 Správa filtrů

Správou filtrů a to z pohledu chování systému se věnuje sekce 4.5.1. V této sekci se správě filtrů věnuji z pohledu návrhu jednotlivých tříd viz. diagram na obr. 24.

Třída FilterInfo zapouzdřuje data o filtru a příslušné get a set metody. Třída FilterManager obsahuje metody pro práci s XML souborem filter.xml, ve kterém jsou uloženy údaje o definovaných filtrech aplikace. Prostřednictvím metod této třídy je možné načíst data o jednotlivých filtrech a jejich typech a také provádět editaci, mazání či přidávání nových filtrů a jejich typů. Třída FilterTree je použita pro práci se stromem filtrů, který je zobrazen v hlavním okně aplikace a obsahuje všechny filtry rozdělené do větví podle jejich typů.



obr. 24: Správa filtrů – návrh tříd

5.4.7 Generování XML

Sekce 4.5.2 seznamuje s generováním XML souborů s daty pro import do databáze z pohledu chování systému. Jak již bylo uvedeno, pro vygenerování XML souboru je možné použít:

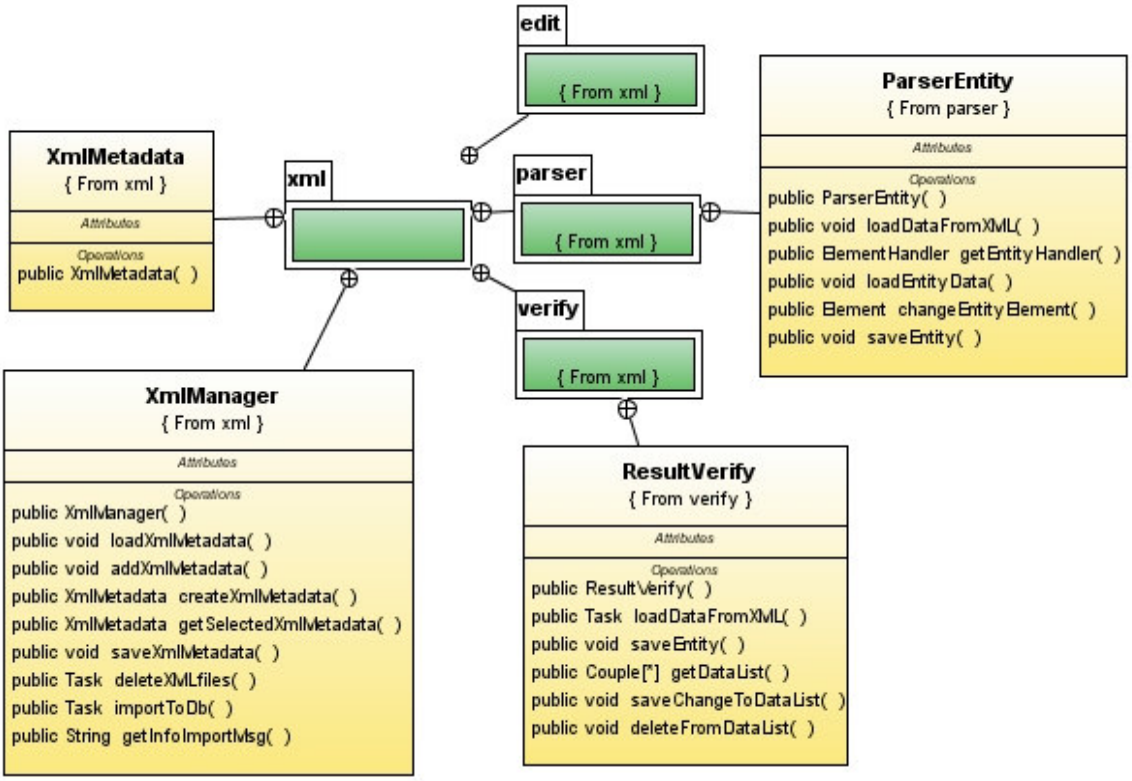
- filtr – vygenerování jednoho XML souboru podle zvoleného filtru
- dávku – vygenerování více XML souborů podle definovaných filtrů pro dávku
- úkol – stažení dat, spuštění dávek a import dat do databáze

V předchozí sekci jsem se věnovala správě filtrů. Aplikace implementuje i třídy pro správu dávek v balíčku batch a úkolů v balíčku job.

5.4.8 Správa XML

Správě XML souborů z pohledu chování systému se věnovala sekce 4.5.3. Správa XML souborů zahrnuje kontrolu a editaci dat v XML souborech, zobrazení podrobných informací o XML souboru, import dat z vybraných XML souborů do databáze a vymazání vybraných XML souborů. Na diagramu na obr. 25 je návrh tříd a balíčku, které implementují správu XML.

Třída XmlMetadata obsahuje data o XML souboru a příslušné get a set metody. Data o vytvořených XML souborech se ukládají do souboru metadata.xml. Třída XmlManager obsahuje metody pro práci se souborem metadata.xml, metody pro import dat do databáze či smazání požadovaných XML souborů bez jejich importu. Existuje více typů zpracovávaných XML souborů a to podle typu informace, kterou obsahují – např. data o dokumentech, statistická data, data o poskytovatelích. Třída ParserEntity hraje důležitou roli pro získání dat ze všech typů XML souborů a editaci těchto souborů. Vzhledem k možné velikosti souborů musí být použita metoda SAX, jinak by mohlo dojít k přetečení paměti. Pro kontrolu a editaci dat v XML souborech je implementována třída ResultVerify a třídy v balíčku edit, které umožňují zobrazení každého typu záznamu v dialogu a jeho následnou editaci.



obr. 25: Správa XML souborů – návrh tříd a balíčků

5.4.9 Konverze mezi databázemi

Cílem konverze je naplnit databázi L1 a L2 daty, které uživatel naimportoval do databáze L0. Ve chvíli, kdy jsou nová data zpracována do databáze L2, jsou přístupná i pro uživatele v generovaných reportech a při procházení kontingenčních tabulek.

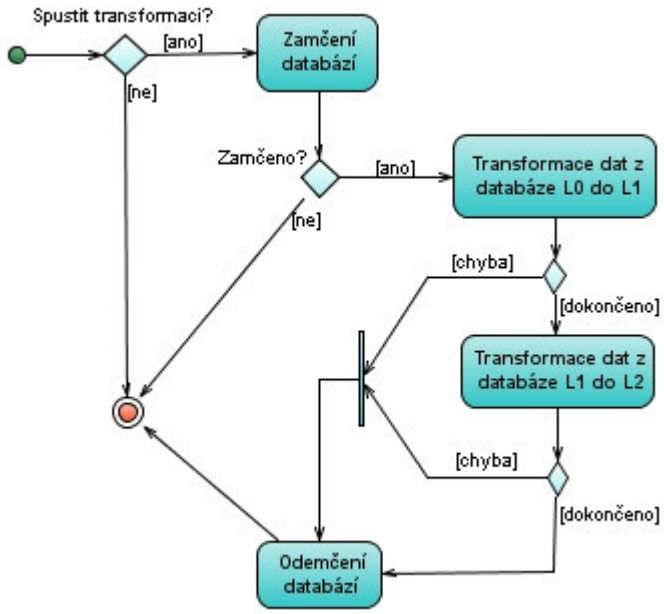
Při volbě jazyka pro realizaci transformace dat mezi databázemi jsem se rozhodovala mezi následujícími řešeními:

1. Použít jazyk databázového stroje a provádět konverzi na databázové úrovni. Mezi výhody tohoto řešení patří zejména rychlost celé transformace, protože odpadne konverze proměnných mezi serverem a klientskou aplikací a zmenší se komunikace mezi aplikací a serverem. Nevýhodou tohoto řešení je, že použitím jazyka konkrétní databáze se aplikace zaměří na tuto databázi a znemožní tak práci s ostatními databázovými servery, s kterými by aplikace mohla komunikovat díky použití již zmíněné technologie Hibernate. Komunikace s jinou databází, než pro kterou budou implementovány úložné procedury, by v případě tohoto řešení znamenalo přepsat celé konverze dat do jazyka druhé databáze. Protože implementace konverze dat je složitá a rozsáhlá, bylo by vytvoření podpory pro další databázi pracné a to i přes to, že jazyky používané některými databázemi jsou podobné. Další nevýhodou tohoto řešení je složitější způsob zápisu do logu a informování uživatele o nezpracovaných záznamech nebo případných chybách, které nastaly během transformace. Teoreticky by bylo možné tento zápis provádět do speciální tabulky, to by ale zabíralo další místo v databázi

2. Použít jazyk Java v kombinaci s technologií Hibernate. Mezi hlavní klady tohoto řešení patří zachování výhody komunikace s více databázemi získané použitím

Hibernate (Hibernate podporuje přibližně 20 základních databázových serverů). Další výhodou je možnost zápisu do log souboru, který je uložen na disku počítače, kde je aplikace spuštěna. Nevýhodou tohoto řešení oproti předchozímu způsobu, je větší časová náročnost operací způsobená komunikací mezi aplikací a databázovým serverem. Vzhledem k předpokladu, že statistická data budou do databáze importována po větších dávkách za delší období (např. měsíčně) a že spouštění konverze dat bude probíhat přes noc, není tato nevýhoda až tak omezující.

Nejdříve jsem zamýšlela použití prvního navrhovaného řešení, protože poskytuje rychlejší zpracování dat. Po opětovném zvážení výhod a nevýhod obou řešení, jsem se nakonec rozhodla pro řešení druhé. Pro porovnání jsem zvolila jazyk PL/pgSQL, protože je to jazyk databázového systému PostgreSQL, který jsem se rozhodla využívat jako primární. Dalším důvodem byla podobnost tohoto jazyka s jazykem PL/SQL, který je podporován databází Oracle.



obr. 26:Transformace L0 → L1 → L2

Aktivity diagram na obr. 26 seznamuje s průběhem transformace dat z databáze L0 až do databáze L2. Aby byla udržena konzistence dat a aby bylo zamezeno konfliktům v datech jsou během průběhu transformace databáze uzamčeny. Zámek je řešen pomocnou tabulkou v databázi L0, která zastává funkci semaforu. Hodnota 1, uložená v atributu tabulky lockdb informuje aplikaci o souběžně probíhající transakci, která vkládá, aktualizuje nebo maže data v databázích. Před spuštěním transformace se proto dotazem *„SELECT * from lockdb FOR UPDATE“* nejprve zjistí, zda jsou databáze odemčeny. Pokud ne, změní se tato hodnota na "zamčeno". Použitím explicitního zámku je zajištěno, že zámek nebude v době vyhodnocení dotazu a nastavení příznaku zámku změněn jinou transakcí. Pokud byl úspěšně nastaven příznak zámku pro danou transakci, spustí se převod dat, po jehož skončení jsou databáze opět odemčeny.

5.4.9.1 Transformace L0 → L1

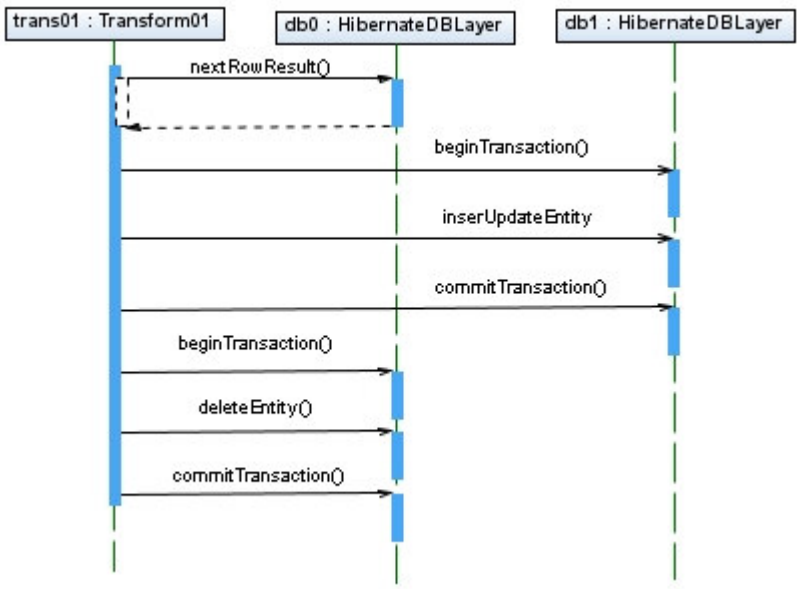
Konverze zahrnuje převod dat z databáze L0, která je použita jako dočasné uložiště dat, do databáze L1, která je datovým skladištěm. Data, která byla úspěšně uložena do databáze L1,

jsou z databáze L0 smazána. V opačném případě je v logu zaznamenána informace o chybném záznamu a záznam zůstává v databázi L0.

Během importu dat do databáze L0 jsou kontrolovány povinné položky záznamů, ale není již kontrolována duplicita dat, aby byl tento import co nejrychlejší. Databáze L1 obsahuje trvalá data určená k archivaci, proto je nutné, aby data v této databázi již byla konzistentní, čistá a neobsahovala duplicitní záznamy. Za duplicitní záznam je považován takový záznam, jehož všechny hodnoty, které určují jeho jednoznačnost, jsou stejné. Jedním z problémů, který je diskutován ve spojení s databázovými systémy, je vložení nového záznamu, který vznikne překlepem při zadávání údajů. I když se může jednat o malý překlep (např. mezera navíc, tečka, atd.) bude záznam do databáze vložen. Tento problém nebyl ještě uspokojivě vyřešen a jeho řešení by přesahovalo rozsah diplomové práce.

Průběh transformace dat mezi databázemi L0 a L1

Tabulky databáze L0 jsou postupně zpracovávány ve stanoveném pořadí (category, publisher, document, impactfactor, institution, provider, institutiondoc, price, statistic). Pořadí tabulek je stanoveno podle provázanosti dat v databázi L1 – např. nejdříve je importována informace o kategorii a poté, při zpracování dokumentu s touto kategorií, již může být správně nastaven cizí klíč do tabulky category. Před vložením nového záznamu do databáze L1 jsou dohledána data pro nastavení cizích klíčů a provedena kontrola úplnosti záznamu. Pokud vše proběhlo v pořádku, jsou data aktualizována nebo vložena do databáze L1 a příslušný záznam je smazán z databáze L0. V opačném případě je uložena informace o chybě v datech do logu a záznam nebude smazán z L0. Zpracování jednoho korektního záznamu je pro názornost zobrazeno pomocí sekvenčního diagramu na obr. 27. V diagramu je vidět práce s transakcemi. Commit transakce je volán po zpracování každého záznamu. Častější volání commitu transakce lze ospravedlnit potřebou detekovat duplicitní záznamy. Pokud bychom v transakcích zpracovávali několik záznamů najednou a mezi nimi se objevily duplicity, došlo by při jejím commitu k chybě a k následnému rollbacku transakce, čímž by všechny záznamy musely být zpracovány znovu. Také pro odstranění dat z databáze L0 je volán commit transakce pro každý záznam. Bylo by možné si pamatovat identifikátory záznamů, které byly úspěšně zpracovány nebo naopak, a pak smazat data tabulky najednou. Seznam identifikátorů záznamů může ale být hodně veliký a vést k přetečení paměti, proto jsem toto řešení nezvolila. Vzhledem k předpokladu, že transformace bude spouštěna v noci či v jiné vyhovující době, není častější commit transakcí, který zpomaluje převod dat, překážkou.



obr. 27: Sekvenční diagram – transformace L0 → L1 (záznam)

5.4.9.2 Transformace L1 → L2

Jakmile je L1 naplněna daty bude transformace pokračovat v převodu dat z aktualizované databáze L1 do databáze L2, která je určená pro získávání informací z dat pro účely reportů a hloubkové analýzy.

Rozhodovala jsem se mezi několika možnými způsoby, jak provést převod dat do databáze L2.

1. Naplnění vždy celé databáze znovu. Tento způsob je časově hodně náročný. Ve chvíli, kdy databáze L1 obsahuje velké množství dat, ve kterých došlo ke změně jen v rozsahu např. 0,5%, bylo by naplněním celé databáze L2 provedeno zbytečně velké množství operací navíc.

2. Inkrementální způsob plnění dat do databáze. Toto řešení se snaží odstranit nedostatky prvního způsobu řešení. Každý záznam databáze L1 obsahuje informaci, zda byl od posledního převodu dat aktualizován nebo nově vložen. Transformace dat do databáze L2 pak zpracovává jen záznamy s příznakem informujícím o jejich změně. Na první pohled toto řešení vypadá jako nejlepší. , Pracujeme jen se změněnými daty. Problém je však aktualizace faktových tabulek, které obsahují agregovaná data za celý rok. V této chvíli bychom např. nevěděli, zda danou hodnotu musíme do výsledné sumy přičíst jako úplně nový údaj nebo zda došlo pouze ke změně již existujícího údaje a je tudíž nutné odečíst starou hodnotu a přičíst novou. Pokud bychom si měli pamatovat u každého záznamu všechny staré a zároveň aktuální hodnoty statistických dat, došlo by k velkému nárůstu objemu dat v databázi L1. Při vypočítávání průměru je tato situace ještě komplikovanější, protože v dané chvíli neznáme celkový součet, ke kterému potřebujeme novou hodnotu přičíst a ani počet údajů, které jsme sečetli.

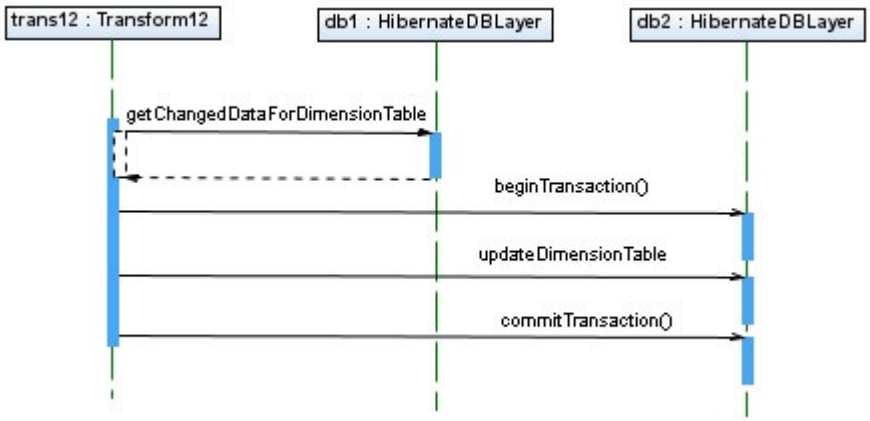
3. Kombinace inkrementálního způsobu plnění databáze s nahrazením dat pro roky, ve kterých došlo k aktualizaci statických dat. Tento způsob řeší nevýhody předchozích dvou variant zpracování dat. Dimenzionální tabulky se aktualizují plně inkrementální metodou. Pro faktové tabulky (tabulky obsahující statistická data) je použit následující postup. Zjistíme si roky, ve kterých došlo ke změně statistických dat. Pro tyto roky

jsou smazána data z faktových tabulek v databázi L2. Následně se data pro tyto roky načtou z L1 a uloží do faktových tabulek. Vzhledem k povaze statistických dat, je takto zpracován většinou aktuální rok.

Po zvážení uvedených výhod a nevýhod všech zmíněných způsobů řešení jsem zvolila třetí variantu.

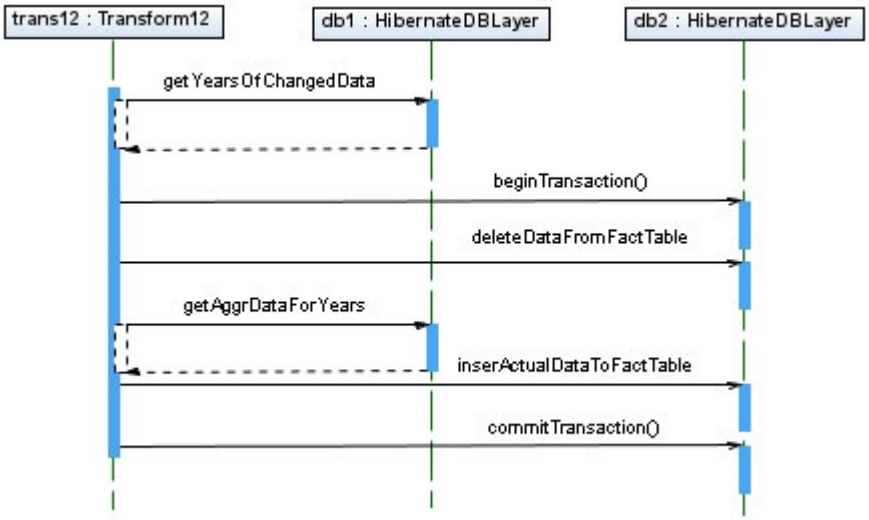
Průběh transformace dat mezi databázemi L1 a L2

Nejdříve proběhne aktualizace dimenzionálních tabulek. Pro každou dimenzionální tabulku jsou dohledány nové a změněné záznamy v databázi L1, které se poznají podle nastavené hodnoty atributu transformed. Během transformace se nekontroluje duplicita záznamů, protože již byla ošetřena v databázi L1. Stavový diagram na obr. 28 zobrazuje zpracování jedné dimenzionální tabulky.



obr. 28: Stavový diagram – transformace L1 → L2 (dimenzionální tabulka)

V druhé fázi jsou aktualizovány faktové tabulky. Z databáze L1 zjistíme roky, ve kterých došlo ke změně statistických dat. Při dohledávání těchto letopočtů jsou sledovány časové údaje u záznamů, které mají hodnotu atributu transformed nastavenou na 1 tzn. informují o změně či vložení záznamu. Z faktových tabulek jsou smazána data pro zjištěné roky a následně načtena a importována aktuální data pro tyto roky z databáze L1. Stavový diagram na obr. 29 znázorňuje zpracování faktové tabulky.

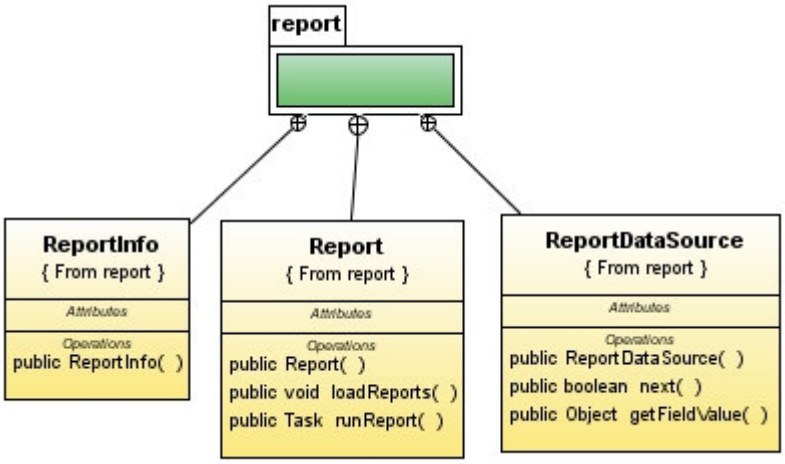


obr. 29: Stavový diagram – transformace L1 → L2 (faktová tabulka)

V poslední fázi transformace proběhne aktualizace dat v databázi L1. Všem záznamům je nastavena hodnota atributu transformed na 0 tzn. záznam byl zpracován a aktuální data jsou uložena v databázi L2.

5.4.10 Reporty

Sekce 4.7.1 seznamuje s reporty z pohledu uživatele, dělením reportů podle typů, parametry reportů, výstupními formáty. V této kapitole se reportům věnuji z pohledu návrhu jednotlivých tříd viz. diagram na obr. 30.



obr. 30: Report – návrh tříd

Reporty jsem se rozhodla generovat pomocí knihovny JasperReport (viz. kapitola 5.2.10). Třída ReportDataSource bude dědit od tříd JDataSource, která je určena pro správu datových zdrojů reportů. Je nutné implementovat její metodu next() pro získání dalšího záznamu z výsledku databázového dotazu, který byl sestaven na základě zadaných parametrů uživatelem, a metodu getFieldValue(…) pro získání hodnoty pro konkrétní pole v reportu. Třída ReportInfo obsahuje informace týkající se konkrétního reportu. Třída Report má na starosti

načtení dat o definovaných reportech z XML souboru report.xml a vytvoření výsledného reportu, který je zobrazen v novém dialogu pomocí třídy JasperViewer. Po zobrazení výsledného reportu je možné zvolit tisk nebo export reportu.

Ve výsledném reportu je možné informační zdroje třídit podle speciální hodnoty, která je dopočítána programem a vyhodnocuje investice do dokumentu. Návrhem výpočtu této hodnoty se zabývá následující část.

5.4.11 Hodnocení investic

Abychom získali precizní hodnocení investic do elektronických informačních zdrojů nestačí sledovat jen počet stažených plných textů nebo jen cenu stažení jednoho článku. Sledování jedné položky je pro objektivní hodnocení nedostačující. Při hodnocení je tudíž nutné počítat s více statistickými položkami a sledovat i další důležité aspekty jako např. priorita financování dokumentu, dostupnost.

Stěžejní fází je navrhnout postup, jak získat objektivní hodnocení investice do dokumentu.

Co vše by mělo být bráno při výpočtu hodnocení dokumentu v úvahu:

- Práce s více položkami
- Různá důležitost položek při hodnocení dokumentu
- Různý rozsah hodnot položek

Seznam položek, které jsou započítány do hodnocení dokumentu, obsahuje tabulka 10. Pro každou položku je uvedena její váha ve výpočtu. Váha představuje procentuální vyjádření a platí, že součet vah všech položek je roven 100%. Další důležitý údaj výpočtu je hodnocení dat sledovaných položek dokumentu pro konkrétní rok. Pro každou položku je možné stanovit minimální a maximální hodnotu, které nabývá. Získaný interval je pak nutné rozdělit na pět částí, které budou ohodnoceny 1 až 5, kde hodnocení 1 je nejhorší (např. počet stažených plných textů za rok je jen 10) a hodnocení 5 je nejlepší (např. počet stažených plných textů za rok je 3040). Hodnocení dat sledovaných položek je pak nastaveno podle toho, do kterého intervalu hodnota položky spadá.

Seznam položek, nastavení jejich vah dle důležitosti a rozdělení intervalů pro hodnocení položek jsem podle výsledků dotazníků, který byl vyplněn pracovníky knihoven.

Položka	***Váha (%)***	***Hodnocení (1 nejhorší až 5 nelepší)***
Počet stažení plných textů (viz. kapitola 4.1.5)	24	1 … <0, 50) 2 … <51, 200) 3 … <201, 800) 4 ... <801, 2000) 5 ... <2001, max>
Cena stažení (v korunách) (viz. kapitola 4.1.5)	14	1 ... <320, max> 2 ... <240, 320) 3 ... <160, 240) 4 ... <80, 160) 5 ... <0, 80)

Položka	*Váha (%)*	*Hodnocení (1 nejhorší až 5 nelepší)*
Impakt faktor (viz. kapitola 4.1.1)	12	2 ... <4,max> 3 ... <0,5, 4) 4 ... <0, 0,5)
Dostupnost (viz. kapitola 4.1.2)	10	1 ... <8, max> 2 ... {7} 3 ... {5,6} 4 ... {3,4} 5 ... {1,2}
Priorita financování dokumentu (viz. kapitola 4.1.2)	40	1 ... <5, max> 2 ... {4} 3 ... {3} 4 ... {2} 5 ... {1}

tabulka 10: Hodnocení investic – parametry

Jak je vidět z tabulky, největší vliv na výsledné hodnocení má priorita financování dokumentu. Během konzultací byl vznesen požadavek na větší flexibilitu hodnocení, aby bylo možné velice důležitý dokument i přes jeho malé využívání a velkou finanční nákladnost zařadit mezi top dokumenty. Požadovaná flexibilita bude řešena zavedením speciální položky SP pro každý dokument. Hodnota této položky bude implicitně 0. Pokud bude knihovník chtít daný dokument zvýhodnit změní implicitní hodnotu na číslo z intervalu <1, 10>. Speciální hodnota by měla být použita doopravdy jen ve výjimečných případech..

Pro získání konečného hodnocení dokumentu je nutné vypočítat dílčí výsledky jednotlivých položek a to podle vzorce DV = *V/100 * H*, kde V je váha položky a H je hodnocení. Součtem těchto dílčích výsledků SUM(DV) + SP, kde SP je výše zmíněná speciální hodnota, dostaneme konečné hodnocení dokumentu pro sledovaný rok. Získaný výsledek bude z intervalu (0, 4,88), pokud SP = 0. Tento interval byl na základě konzultací rozdělen na tři části a pro každou část bylo vyjádřeno, jak výhodné bylo dokument pořídit.

Dělení konečného výsledku

- (4, max) vynikající investice, dále předplácet
- (2,4> průměrná investice, možno dále předplácet, ale pokud se objeví něco zajímavějšího, bude průměrný titul nahrazen
- (0, 2> špatná investice, již nepředplácet

5.5 Webový klient

Webový klient zobrazuje výsledné reporty, umožňuje procházení kontingenčních tabulek, zobrazení grafů a export dat do různých formátů. Přístup k datům přes webové rozhraní nabízí několik výhod. Jednou ze stěžejních je fakt, že si uživatel nemusí instalovat aplikaci na vlastním počítači, ale je nainstalována centrálně a spravována pověřenou osobou.

Jako aplikační server jsem zvolila Apache Tomcat, který podporuje technologii JavaServer Pages. Podpora této technologie je nutná jak pro zobrazování dat v kontingenčních tabulkách, které je řešeno pomocí ROLAP serveru Mondrian, tak pro vytváření reportů, které je implementováno pomocí Javovské knihovny JasperReports.

5.5.1 Reporty

Reporty vytvářené a zobrazované prostřednictvím webového klienta jsou řešeny stejným způsobem jako u klienta GUI, což umožňuje použití již implementovaných tříd. Kromě tříd Report, ReportInfo a ReportDataSource (viz. sekce 5.4.10) jsou použity i další implementované třídy, které zajistí komunikaci s databází použitím technologie Hibernate.

Uživatel prostřednictvím formuláře definuje typ reportu, zadá parametry reportu, zvolí výstupní formát reportu a zobrazí výsledný report. Podle zvoleného typu reportu jsou zobrazeny příslušná pole formuláře.

5.5.2 Analýza dat

Pro analýzu dat je použit ROLAP server Mondrian, který ve spolupráci s prohlížečem JPivot umožňuje zobrazení dat v multidimenzionálním formátu v kontingenční tabulce. Aby bylo možné získávat data z relační databáze, je nutné pomocí jazyka XML definovat schéma multidimenzionální databáze. Schéma obsahuje logický model tvořený z kostek, hierarchií, členů a mapování tohoto modelu na relační databázi. V aplikaci je definováno pět multidimenzionálních kostek a každá odpovídá jednomu datovém tržišti databáze L2.

Prostřednictvím JSP stránek mohou uživatelé přistupovat k různým analýzám dat, které budou definované použitím jazyka MDX (viz. kapitola 5.2.7).

5.6 Popis implementace vybraných částí systému

Následující sekce seznamují s implementačními detaily vybraných částí systému. Zdrojové kódy všech tříd aplikace, skriptů, XML souborů, JSP dokumentů, atd. jsou k dispozici na přiloženém CD. Dále je na přiloženém CD k dispozici programátorská dokumentace.

5.6.1 Spuštění skriptů v jazyce Java

Aplikace pracuje se skriptovacím jazykem Jython. Jedná se o implementaci jazyka Python v jazyce Java. Skripty se používají pro stažení dat z Internetu a převod dat do XML souborů. Spouštění skriptu je implementován metodou runJythonScript(), jejíž kód doplněný podrobným komentářem je pro detailnější seznámení uveden níže.

```
public void runJythonScript(Task task, String scriptPath, JythonParam param)
throws FileNotFoundException, FeederException, IOException {
    // Nastavení cesty k modulům jazyka Python
    Properties props = new Properties();
    props.setProperty("python.path", ConstantsLibrarex.LIB_PYTHON);
    PythonInterpreter.initialize(System.getProperties(),props, null);
    // Vytvoření instance třídy PythonInterpreter pro spuštění kódu Pythonu
    PythonInterpreter pytInt = new PythonInterpreter();
    // Informování uživatele, že probíhá inicializace skriptu
    task.setInfoMessage(L10n.getString("Feeder.Script.Init"));
    // Předání parametrů skriptu
    pytInt.set(ConstantsLibrarex.PYTHON_PARAM, Py.java2py(param));
    // Načtení obsahu skriptu do řetězce
```

```
    StringBuffer contentsScript = DefaultFileWorker.getFileContents(scriptPath);
    // Přeložení řetězce s obsahem skriptu do bajtového kódu Javy
    PyCode pyCode = jythonScriptCompile(contentsScript, scriptPath);
    // Informování o spuštění skriptu
    task.setInfoMessage(L10n.getString("Feeder.Script.Run"));
    // Vykonání kompilovaného bajtového kódu
    jythonScriptExec(pyCode, pytInt);
    // Informování o dokončení nebo zastavení skriptu
    if (task.isCanceled()) {
         logger.debug("Execute script " + scriptName + "was stopped by user.");
         task.setInfoMessage(L10n.getString("Feeder.Script.Stop"));
    } else {
         task.setInfoMessage(L10n.getString("Feeder.Script.Finish"));
    }
  }
```

Vykonání skriptu probíhá ve vlastním vlákně, které je řešeno pomocí třídy Task. Metoda nejdříve nastaví cestu k modulům jazyka Python, což je ekvivalent nastavení systémové proměnné PYTHONPATH. Vlastní spouštění kódu jazyka Python z jazyka Java zajišťuje třída PythonInterpreter. Spouštěnému skriptu je možné předat vlastní parametry pomocí třídy JythonParam. Třída JythonParam obsahuje set a get metody pro práci s parametry skriptu a byla implementována speciálně pro tuto aplikaci. Důležitou proměnnou této třídy je proměnná canceled, která je při přerušení operace uživatelem nastavena na hodnotu true a umožní předat informaci o přerušení vykonávanému skriptu prostřednictvím metody isCanceled() třídy JythonParam. Pro přeložení skriptu do bajtového kódu Javy je volána metoda:

```
PyCode pyCode = __builtin__.compile(contentsScript.toString(), scriptPath, "exec");
```

Provedení kompilovaného bajtového kódu má na starosti metoda:

```
Py.exec(pyCode, pyInt.getLocals(), pyInt.getLocals());
```

V adresáři aplikace data/script/filter/default jsou připravené skripty pro generování jednotlivých typů XML souborů, které obsahují metody pro práci s datovým souborem, metody pro vytvoření a zápis do XML souboru a metodu pro kontrolu přerušení a následné ukončení skriptu. Pro vytvoření nového skriptu je implementována jen metoda pro zpracování řádků konkrétního datového souboru.

5.6.2 Editace velkých XML souborů

Jazyk XML jsem použila pro uložení nastavení aplikace, převod dat do jednotného formátu, uložení informací o filtrech a reportech. Vzhledem k odlišné velikosti a zpracování dat uložených v XML souborech jsem použila stromový i událostmi řízený přístup. Jako parser jsem zvolila Dom4j. Aby byla práce s XML maximálně zjednodušena, implementovala jsem třídy DOMXMLWork a SAXXMLWork, které poskytují metody pro načtení, editaci, mazání a vyhledávání v XML souborech. Kontrola struktury XML dokumentů je řešena pomocí DTD (Document Type Definition).

Následující kód seznamuje s editací velkých XML souborů, pro které nelze kvůli paměťové náročnosti použít rozhraní DOM.

```
    SAXReader saxReader = new SAXReader(true);
    XMLReader xmlReader = saxReader.getXMLReader();

    OutputFormat format = OutputFormat.createPrettyPrint();
    OutputStreamWriter out =
        new OutputStreamWriter(new FileOutputStream(newPathXml),"UTF-8");
    XMLWriter writer = new XMLWriter(out, format);

    // Vytvoření modifikátoru
```

```
SAXModifier modifier = new SAXModifier(xmlReader, true);
// Přidání modifikátoru, který bude volanán pro element specifikovaný XPath
modifier.addModifier(xPath,  elementModifier);
// Nastavení XMLWriteru,který bude použit pro zápis změněného dokumentu
modifier.setXMLWriter(writer);

FileInputStream in = new FileInputStream(pathXml);
InputStreamReader fileReader = new InputStreamReader(in, "UTF-8");
// Přečtení, modifikace a zapsání Dokumentu pomocí SAX
modifier.modify(fileReader);
out.close();
```

Modifikace velkých XML souborů jsem řešila pomocí třídy SAXModifier. Této třídě je zaregistrován modifikátor, který definuje změnu elementu určeného pomocí jazyka XPath. Třída, která obsahuje metody pro změnu elementu, implementuje rozhraní ElementModifier. Metoda modify() třídy SAXModifier sekvenčně zpracovává vstupní XML dokument, načítá element definovaný pomocí jazyka XPath, aktualizuje jej pomocí implementované funkce modifikátoru a ukládá do výsledného XML souboru.

5.6.3 Databázový dotaz

Pro zadávání databázového dotazu jsem zvolila criteria API, jeden z nabízených způsobů zadávání dotazů technologií Hibernate. Následující kód ukazuje nastavení kritérií dotazu, vykonání dotazu a získání identifikátoru výsledku, který lze použit pro postupné zpracování dat výsledku.

```
// Otevření nové session
session = sessionFactory.openSession();
// Vytvoření a nastavení kritérií dotazu
SelectQueryCriteria queryCriteria = new SelectQueryCriteria();
Criteria criteria = session.createCriteria(c);
// Spojení tabulek
criteria = queryCriteria.joinTable(criteria, joinTable);
// Projekce – SUM, MIN, COUNT, atd.
criteria = queryCriteria.projection(criteria, projection);
// Podmínky dotazu
criteria = queryCriteria.restriction(criteria, restriction);
// Odstranění duplicit
criteria = queryCriteria.distinct(criteria, distinct);
// Definování uspořádání
criteria = queryCriteria.order(criteria, order);
// Vykonání dotazu a vrácení identifikátoru výsledku
resultId = execSelect(session, criteria);
```

Implementovala jsem třídu SelectQueryCriteria, jejichž metody joinTable(…), projection(…), restriction(…), distinct(…) a order(…) třídy SleectQueryCriteria definují výsledný dotaz. Jako příklad uvádím hodnoty parametrů některých metod. Přesný popis těchto parametrů je k dispozici v JavaDoc u jednotlivých metod.

```
Object[] joinTable =
          {ConstantsLibrarex.JOIN, StatisticelectronicL1.RELATION, "rel"};
Object[] restriction = {ConstantsLibrarex.RESTRICTION_IN,
           StatisticelectronicL1.YEAR, null, null, yearList, null};
Object[] projection =
          {ConstantsLibrarex.PROJECTION_COUNT, StatisticelectronicL1.YEAR,
           ConstantsLibrarex.PROJECTION_SUM,
           StatisticelectronicL1.FULLTEXTPDF,
           ConstantsLibrarex.PROJECTION_GROUP, "rel." + RelationL1.ID};
```

5.6.4 Vytvoření reportu

Vytváření reportů jsem se rozhodla řešit pomocí knihovny JasperReports. Následující příklad seznamuje s použitím této knihovny a předáním dat reportu.

```
// Nastavení parametrů reportu
HashMap params = new HashMap();
params.put("LIBRARY", registredLibrary);
params.put("PARAMETERS",
     parametersReport.get(ConstantsLibrarex.REPORT_PARAMETERS));

// Získání identifikátoru výsledku dotazu sestaveného na základě parametrů
int resultId =
     database.conditionReportSelect(parametersReport, report.getType());

// Načtení kopilovaného návrhu reportu v souboru .jasper
FileInputStream fis = new FileInputStream(ConstantsLibrarex.REPORT_PATH + "/"
     + report.getFile() + ".jasper");
ObjectInputStream ois = new ObjectInputStream(fis);
JasperReport jasperReport = (JasperReport) ois.readObject();

//Předání dat reportu a vytvoření objektu pro zobrazení a tisk reportu
JasperPrint jasperPrint = JasperFillManager.fillReport(jasperReport,
     params, new ReportDataSource(database, resultId));
```

V dialogu aplikace zvolí uživatel typ reportu a vyplní jeho parametry. Informace o jednotlivých reportech jsou uloženy a načítány z XML souboru. Parametry jsou po načtení uloženy v instanci objektu Hashtable<Integer, Object>, kde každý parametr je přístupný podle číselného klíče. Databázový dotaz je vytvořen na základě uvedených parametrů a získaný identifikátor výsledku je předán třídě ReportDataSource, která implementuje interface JDataSource a je určena pro naplnění reportu daty. V třídě ReportDataSource je implementována metoda getFieldValue(JRField rField), která vrací hodnotu specifikovaného pole formuláře. Pole formuláře jsem pojmenovala složením názvu identifikujícího tabulku a názvu sloupce tabulky, což bylo důležité pro získání správné hodnoty pole. Definice vzhledu reportu je napsaná pomocí jazyka XML a uložena v .jrxml souboru. Pro vytvoření šablony reportu jsem použila nástroj iReport. Pro rychlejší získání výsledného reportu jsem v aplikaci použila již zkompilovanou šablonu reportu v .jasper souboru.

6 Testování

V průběhu implementace aplikace jsem testovala správné chování jednotlivých metod a komponent. Zaměřila jsem se na testování správných výstupů pro různá testovací data, kontrolu správnosti odchycení a zpracování výjimek, logiku, funkčnost GUI, atd. Tyto testy mi pomohly odhalit některé chyby, které jsem následně odstranila. V literatuře je tento typ testů nazýván jako „jednotkové testy“ [12].

Jednotlivé části systému jsem hned od začátku implementace kompilovala společně, i když nebyly dokončeny a samostatně otestovány. Vytvořila jsem si tak verzi klientské aplikace, která vždy odpovídala aktuálnímu stavu vývoje. Výhodou tohoto přístupu bylo, že jsem mohla průběžně testovat vzájemnou komunikaci jednotlivých komponent.

Jedním z nástrojů pro ladění programu je logování. Pro záznam činností programu jsem použila aplikační rozhraní API Log4j[35]. Parametry logování jsou nastaveny v externím textovém souboru, kde je možné definovat úrovně logování zpráv, formátování logovaných zpráv, výpis zpráv na standardní výstup nebo do souboru. Úrovně logování jsou DEBUG, ERROR, FATAL, INFO, WARN, OFF, ALL a umožňují odlišit typ zprávy např. ladící zprávu od chybové.

Výkonnost aplikace musí být akceptována koncovými uživateli, je tedy důležité provést i výkonnostní testy a testování zátěže. Pro testování jsem použila data poskytnutá ústřední knihovnou VŠCHT a implementovala skripty, které byly následně použity aplikací k převodu dat do XML souborů. Pracovala jsem s velkým objemem dat, abych zjistila jak se aplikace chová při velké zátěži. Pro sledování chování aplikace jsem použila trial verzi JProfiler[36]. JProfiler umožňuje sledování spotřeby paměti, vytížení procesoru a stavu vláken při běhu aplikace.

Vytvořila jsem si testovací úkol, který provádí následující operace:

1. Ze stránek společnosti SUWECO získá ISSN časopisů, které jsou přístupné čtenářům ústřední knihovny VŠCHT Praha. Jedná se o 1898 časopisů od poskytovatelů Elsevier, Wiley, Springer, Kluwer a Blackwell.

2. Z databáze Ulrich’s Periodicals Direktory stáhne pro každý časopis detailní informace např. titul, vydavatel, země vydání, cena, URL. Pro každého poskytovatele vytvoří CSV soubor s těmito údaji.

3. V první dávce zpracuje CSV soubory se základními údaji o dokumentech a vytvoří požadované XML soubory připravené pro import do databáze.

4. V druhé dávce uloží údaje z připravených souborů se statistickými daty do XML souborů. Spuštěno cca na 25000 záznamech.

5. Data z vytvořených XML souborů jsou importována do databáze L0.

6. Provede převod dat mezi databázemi, který naplnil databázi L1 a L2 novými daty.

Výsledek prvního naplnění databáze daty obsahuje tabulka 11.Velmi časově náročné bylo stažení základních dat z Internetu, kdy pro každý titul bylo nutné přistupovat k datům na novou stránku. Doba stažení dat z Internetu je závislá na kvalitě připojení.

[35] http://logging.apache.org/log4j/
[36] http://www.ej-technologies.com/products/jprofiler/overview.html

Typ operace	*Čas (h:m:s)*
Stažení dat z Internetu, jejich zpracování a uložení do CSV souborů	2:01:11
Generování XML souborů (vytvořeno 36 souborů s daty určenými pro import do databáze)	0:01:02
Import dat do databáze L0 (importováno cca 25000 záznamů)	0:00:34
Převod L0 → L1	0:12:41
Převod L1 → L2	0:06:39
CELKEM	*2:22:07*

tabulka 11: Časová náročnost naplnění databází základními daty

Tento testovací úkol zahrnoval naplnění databází základními daty, které bude v takovém rozsahu provedeno jedenkrát. Dále budou převážně zpracovávány statistické údaje. Výsledek zpracování souborů se statistickými daty (100000 záznamů), import do databáze L0 a následný převod dat mezi databázemi zobrazuje tabulka 12. Vzhledem k předpokladu, že úkol bude spouštěn v noci, je tento čas rovněž vyhovující.

Typ operace	*Čas (h:m:s)*
Generování XML souborů (vytvořeno 32 souborů se statistickými daty)	0:03:21
Import dat do databáze L0 (importováno cca 100000 záznamů)	0:02:16
Převod L0 → L1	1:18:11
Převod L1 → L2	0:31:56
CELKEM	*1:55:44*

tabulka 12: Časová náročnost zpracování statistických dat

Pokud bude klient GUI obsluhován několika správci na různých PC, nedojde ke zpomalení operací, které editují, importují či odstraňují data z databází. Aplikace neumožní paralelní zpuštění více těchto databázových operací.

Je plánováno nasazení aplikace na zkušební provoz v ústřední knihovně VŠCHT Praha, během kterého bude dále testována a postupně naplněna aktuálními daty knihovny.

7 Závěr

Cílem diplomové práce bylo analyzovat, navrhnout a implementovat systém, který bude knihovníkovi podporou při plánování investic a zkvalitňování knihovního fondu.

V první fázi byla pozornost věnována důkladnému seznámení s řešenou problematikou. Prostudováním řady informací a konzultacemi se zaměstnanci knihoven jsem se seznámila s procesy a postupy, které v knihovnách probíhají při akvizici a získala informace důležité pro specifikaci požadavků na aplikaci LibrarEX. Požadavky byly průběžně konzultovány s budoucími uživateli aplikace. Pomocí vizuálního modelování systému bylo podrobněji navrženo, jak se bude systém chovat při řešení vybraných požadavků.

Po dokončení analytické části byl řešen návrh architektury aplikace a hlavních částí systému. Volba technologie může návrh aplikace znatelně ovlivnit, proto jsem se již při návrhu seznámila s technologiemi, které byly použity pro implementaci. Vyhledávání vhodných technologií, knihoven a nástrojů, spolu s důkladným studiem jejich možností a práce s nimi se ukázalo významným přínosem pro výslednou aplikaci. Neméně důležitou částí návrhu byl návrh databáze. Vzhledem k požadavkům na aplikaci jsem se rozhodla pro řešení zahrnující návrh analytické databáze.

Implementace a testování výsledné aplikace bylo náplní další (poslední) fáze. Byl implementován klient s grafickým uživatelským rozhraním a webový klient. Byly implementovány všechny požadavky a vytvořen chybějící nástroj, který zjednodušuje knihovníkům rozhodování při investici finančních zdrojů. Výsledná aplikace je poskytnuta ústřední knihovně VŠCHT Praha, testovací provoz bude realizován od 1. 12. 2007 do 29. 2. 2008. Po skončení testovacího provozu proběhne vyhodnocení a konzultace s uživateli, které mohou přinést další zajímavé podněty pro rozšíření aplikace.

Výsledný software je zaměřen především na prácí s elektronickými a tištěnými informačními zdroji. Jedná se o dále rozšiřitelný produkt. Možné rozšíření, které v této práci již není řešené, je např. přidání sledování využití bibliografických databází, zaznamenávání aktuálního počtu čtenářů a rozšíření o výpočty s těmito údaji.

Závěrem lze říci, že v návaznosti na reakce zainteresovaných odborníků, se jedná o velmi prospěšnou aplikaci, která přináší knihovnám a dalším institucím v oblasti knihovnictví výraznou úsporu času a lidských zdrojů. Jednoznačně kladně je hodnocena přehlednost grafických výstupů, kompaktnost získaných dat a to zejména s ohledem na jejich další využití v budoucnosti. Všechny vytyčené cíle práce se podařilo bezezbytku splnit a práce naplnila očekávání.

8 Zdroje

[1] RESSLER, Miroslav. *Informační věda a knihovnictví* [online]. Verze 1.0. Praha : Vysoká škola chemicko–technologická v Praze ve spolupráci s Národní knihovnou ČR, c2006 [cit. 2007–05–04]. Dostupný z WWW: <http://vydavatelstvi.vscht.cz/knihy/uid_es–005/ebook.help.htm>.

[2] STÖCKLOVÁ, Anna. Informační studia a knihovnictví v elektronických textech I., Interaktivní modulární výukový systém na podporu informačního a knihovnického vzdělávání [CD–ROM]. Praha : ÚISK FF UK, 2001. 16 s.

[3] SMETANOVÁ, Dana, et al. Portál STM. *Národní knihovna* [online]. 2002, roč. 13, č. 4 [cit. 2007–05–04], s. 250–261. Dostupný z WWW: <http://knihovna.nkp.cz/NKKR0204/0204250.html>. ISSN 1214–0678.

[4] *Co jsou to konsorciální licence?* [online]. 2003 [cit. 2007–06–11]. Dostupný z WWW: <http://www.aip.cz/konsorcia_co.php>.

[5] *INFORUM : Zahraniční projekty k měření využívání online informačních zdrojů* [online]. c2007 [cit. 2007–05–07]. Dostupný z WWW: <http://www.inforum.cz/inforum2004/pdf/Katolicka_Barbora1.pdf>.

[6] *Ptejte se knihovny* [online]. [2004] [cit. 2007–06–04]. Dostupný z WWW: <http://www.ptejteseknihovny.cz>.

[7] POSPÍŠILOVÁ, Jindřiška. *Knihovny.cz* [online]. 2004 [cit. 2007–05–07]. Dostupný z WWW: <http://www.knihovny.cz/page.php3?page=coje_definice.htm >.

[8] *Databáze Národní knihovny ČR : KTD – Česká terminologická databáze knihovnictví a informační vědy (TDKIV)* [online]. c2004 [cit. 2007–06–05]. Dostupný z WWW: <http://sigma.nkp.cz/F/?func=file&file_name=find–a&local_base=ktd>.

[9] Impact factor: abychom věděli, o čem mluvíme. *Medicína* [online]. 2000, roč. VII, č. 6 [cit. 2007–05–20], s. 2–2. Dostupný z WWW: <http://www.zdrava–rodina.cz/med/med0600/med0600_4.html>.

[10] PALETA, Petr. *Co programátory ve škole neučí : Softwarové inženýrství v reálné praxi.* 1. vyd. Praha : Computer Press, 2003. 331 s. ISBN 80–251–0073–1.

[11] ARLOW, Jim, NEUSTAT, Ila. *UML a unifikovaný proces vývoje aplikací*. 1. vyd. Brno : Computer Press, 2003. 387 s. ISBN 80–7226–947–X.

[12] GUNDERLOY, Mike. *Z kodéra vývojářem : nástroje a techniky pro opravdové programátory*. 1. vyd. Brno : Computer Press, 2007. 287 s. ISBN 978–80–251–1517–6.

[13] LACKO , Luboslav. *Databáze : datové sklady, OLAP a dolování dat s příklady v Microsoft SQL Serveru a Oracle*. Brno : Computer Press, 2003. 488 s., 1 CD–ROM. ISBN 80–7226–969–0.

[14] HORÁK, Jiří, HORÁKOVÁ, Bronislava. Datové sklady a využití datové struktury typu hvězda pro prostorová data. In HORÁK, Jiří, DĚRGEL, Pavel, KAPIAS, Adrian. *Symposium GIS Ostrava 2007*. [s.l.] : VŠB–TUO 2007, 2007. Sekce 3: Správa prostorových dat a datové skladby. s. 1–26. Dostupný z WWW: <http://gis.vsb.cz/GIS_Ostrava/GIS_Ova_2007/sbornik/Referaty/Sekce3/hvezdaF4.pdf>. ISSN 1213–239X.

[15] HERNANDEZ, Michael J. *Návrh databází.* Jan Bouda. 1. vyd. Praha : GRADA, 2006. 408 s. ISBN 80–247–0900–7.

[16] *Pentaho : open Source Business Inteliigence* [online]. c2007 [cit. 2007–06–02]. Dostupný z WWW: <http://www.pentaho.com>.

[17] BRADLEY, N. *XML : Kompletní průvodce.* Jiří Bráza. Praha : Grada Publishing, 2000. 540 s. ISBN 80–7169–949–7.

A Obsah CD

A.1 Struktura adresářů

Adresáře	Obsah
`/LibrarEX/src`	Zdrojové kódy aplikace (klient GUI)
`/LibrarEX/doc`	Dokumentace aplikace (klient GUI]
`/LibrarEX/lib`	Knihovny třetích stran (klient GUI)
`/LibrarEX/config`	Konfigurace aplikace (klient GUI)
`/LibrarEX/data`	Data aplikace (klient GUI)
`/LibrarEX/log`	Log soubory aplikace (klient GUI)
`/LibrarEX_WEB/bin`	Přeložená aplikace (webový klient)
`/LibrarEX_WEB/src`	Zdrojové kódy aplikace (webový klient)
`/installer`	Instalační program aplikace (klient GUI)
`/software`	Použité programy třetích stran
`/doc`	Text diplomové práce, uživatelská příručka a přílohy

B Použité vývojové nástroje

Jako vývojový nástroj jsem zvolila Netbeans IDE. Hlavním důvodem jeho použití je editor Matisse, který je v Netbeans IDE verze 5 a vyšší již zabudován. Tento editor velice urychlí práci při vytváření dialogů aplikace. Dalším plusem pro Netbeans byl plugin pro modelování UML diagramů. Existuje již hodně nástrojů, které umožňují vytvářet UML diagramy, ale většinou se jedná o komerční produkty, které je možné otestovat pouze jako 30 denní verzi. Pro návrh databázového schématu jsem zvolila CASE Studio 2 demo verzi.

I když jsem se rozhodla pro vývojový nástroj Netbeans IDE, nakonec jsem pracovala i s vývojovým nástrojem Eclipse. Důvodem tohoto rozhodnutí byl plugin Hibernate Synchonizer určený právě pro vývojové prostředí Eclipse. Hibernace Synchonizer umožňuje generování XML a javovského kódu pro jednotlivé tabulky databáze. I přes prováděné další úpravy v těchto třídách bylo použití tohoto nástroje velkou pomocí a ušetřilo mi to psaní velkého množství get a set metod.

C Použité knihovny

S použitými nestandardními knihovnami seznamuje tabulka 13. Je vidět, že použití jedné technologie si v některých případech vynutí přidání více jak jedné knihovny.

Technologie	***Soubory***	***Popis***
Jyton	jython.jar + skripty Pythonu	Nástroj pro práci se skriptovacím jazykem Python.
Hibernate	hibernate3.jar, antlr–2.7.5H3.jar, cglib–2.1.jar, asm.jar, asm–attrs.jars, commons–collections–1.7.jar,	Systém pro objektově relační mapování.

Technologie	*Soubory*	*Popis*
	commons–logging–1.1.jar, jta.jar, dom4j–1.6.1.jar, log4j–1.2.14.jar	
Jasper reports	jasperreports–1.3.3.jar, poi–3.0–rc4–20070503.jar, commons–beanutils–1.7.jar, commons–collections–3.1.jar, commons–digester–1.7.jar, commons–logging–1.1.jar, xalan.jar, xercesImpl.jar, jfreechart–1.0.1.jar, itext–1.3.jar	Nástroj pro generování a zobrazování reportů.
Log4j	log4j–1.2.14.jar	Knihovna pro práci s logem.
Dom4j	dom4j–1.6.1.jar	Knihovna pro práci s XML soubory.
Look &Field	looks–2.0.4.jar	Knihovna pro definování vzhledu uživatelského prostředí.
jCalendar	jcalendar–1.3.1.jar	Knihovna pro práci s kalendářem.
Matisse	swing–layout–1.0.jar	Knihovna pro vytváření GUI v Javě.
Java Help	jh.jar	Systém pro tvorbu interaktivní nápovědy.
JDBC driver	postgresql–8.0–311.jdbc2.jar	JDBC ovladač pro PosgreSQL
OLAP	mondrian.jar, jpivot.jar	Knihovny pro analýzu dat.

tabulka 13: Použité knihovny

D Uživatelská příručka

Uživatelská příručka je dostupná na přiloženém CD v adresáři *doc* jako PDF dokument „uzivatelska_prirucka.pdf“.

Příručka v první části seznamuje se systémovými a hardwarovými požadavky aplikace, obsahuje popis instalace klienta s grafickým uživatelským rozhraním a popis instalace webového klienta aplikace LibrarEX.

Další část příručky je určena administrátorovi aplikace, který má na starosti správu vstupních dat, aktualizaci a zajištění jejich úplnosti v databázi. Administrátor pro vykonání uvedených úkolů má k dispozici klienta s grafickým uživatelským rozhraním. Příručka seznamuje s nastavením, ovládáním a možnostmi tohoto klienta. Kromě příručky může být pro zjištění potřebných postupů využita nápověda dostupná po spuštění aplikace.

Poslední část příručky je určena pro uživatele, který bude získávat informace z dat uložených v databázi prostřednictvím reportů, kontingenčních tabulek a různých grafů. Tento uživatel bude přistupovat k datům pomocí webového rozhraní.