

**BIBLIOGRÁFIAI ADATOK KONVERZIÓJÁNAK
TÁMOGATÁSA AZ XLS2XML
PROGRAM SEGÍTSÉGÉVEL**

Tartalomjegyzék

1. Bevezetés	3
2. A könyvtári gépesítés története és annak szükségszerűsége	6
3. A könyvtári automatizálás problémái	10
4. A Magyar Földrajzi Társaság Szakkönyvtára és annak problémája	12
5. Felhasznált technológiák ismertetése	15
5.1. A Java programozási környezet	15
5.2. Az XML bemutatása	16
6. XLS2XML konvertáló program bemutatása	24
7. Meglévő konvertáló programok hibája?!	30
8. Összefoglalás	32
Ábrajegyzék	35
Irodalomjegyzék	36
Mellékletek	38

„A könyvtár egy csoda. Ha valamire tényleg kíváncsi vagy, lépj oda a könyvespolchoz, és csak kövesd a megérzéseidet. A könyv a válasszal ott vár majd rád!”

(Lucien del Mar)

1. Bevezetés

A könyvtár, mint intézmény voltaképpen egyidős a civilizációval, a kommunikáció második forradalmával, az írással együtt született meg. Egy olyan megőrző intézmény jött létre, ahol a megszerzett tudást rendszerezve őrzik, és tovább adják az utókornak. Már az ókorban is léteztek könyvtárak (Ninive, Egyiptom, Görögország, Róma), ahol papirusztekercseket, agyagtáblákat gyűjtöttek.

A történelem során különböző könyvtárak alakultak ki. Előbb magánkönyvtárak jöttek létre, amelyek később nyilvánossá váltak, így megnyíltak a nagyközönség számára. A könyvtár mindig a társadalom igényeit elégítette ki, illetve ahhoz alkalmazkodott. A középkorban például visszaszorult a kolostorok falai közé, és vallási célokat szolgált. Később a reformáció és a polgárosodás korában demokratizálódott, ekkor jelentek meg a közkönyvtárak. Ehhez természetesen nélkülözhetetlen volt a kommunikáció új, immár harmadik forradalma, a nyomtatás feltalálása. Egyre több olyan, új nyomtatvány jelent meg, ami már sokak számára elérhető volt. A könyvtár alkalmazkodott mindezen fejlődésekhez, hogy az új eszmék terjesztőjévé válhasson.

A 19. századra az addigi, teljességre törekvő könyvtárak már csak bizonyos pontig lehettek enciklopédikusak, mert a nyomtatás feltalálása után egyszerűen nem lehetett összegyűjteni mindent, ami megjelent. Így alakultak ki a szakkönyvtárak. Más könyvtári épületekre volt szükség, új felhasználói rétegek jelentek meg és ezzel párhuzamosan a könyvtárosság ekkor vált igazán szakmává. Tehát a könyvtár sikeresen alkalmazkodott, és biztosította saját fennmaradását.

A 20. század új technikai forradalma hasonló volumenű változásokat hozott az emberiség életében, csak úgy, mint a nyomtatás megjelenése után. Még rövidebb idő alatt még több információhoz tudunk jutni, ha értünk a számítógéphez, pontosabban az Internethez. Most nem beszélhetünk új könyvtártípusról, hiszen az elektronikus könyvtárakkal csak a

hozzáférés új lehetősége jelent meg. A könyvtár azonban most is folyamatosan változik, alakul, ahogy eddig minden korban.^{1,2}

Mind a technika, mind a társadalmi igények fejlődésével összevetve a könyvtárak feladata mindig ugyanaz marad. Ez **a cél nem más, mint beszerezni, őrizni és hozzáférhetővé tenni a tudást, az információt, mindegy, hogy az milyen hordozón jelenik meg.** Az információ exponenciálisan növekvő mennyiségének a feldolgozása, visszakereshetősége a mai napig nehéz feladatok elé állítja a könyvtárakat. Az intézménynek tudomást kell szereznie a dokumentumról, be kell szereznie, tárolnia kell, és azok rendelkezésére kell bocsátania, akiknek a dokumentumra igénye, szüksége van. Természetesen csak azoknak lehet rá szüksége, akik tudnak a dokumentum létezéséről, és olyan ismereteket kapnak arról, amelyből el tudják dönteni, hogy szükségük van-e a benne foglalt információkra. Ez egy újabb funkcióhoz vezet: **a dokumentumok létezéséről és főbb tulajdonságaikról szóló metaadatokat a könyvtárnak rögzítenie és hozzáférhetővé kell tennie.** Ez a folyamat nem más, mint a feldolgozás.

Jól látható tehát, hogy az emberi civilizáció fejlődésével, illetve az információ előállításához szükséges technikák fejlődésével szükségszerűen fejlődnie kellett a könyvtárnak is. Nem csak a felhasználók kiszolgálása érdekében, hanem maga az információ feldolgozás hatékonyabbá tételének a szempontjából is. Ugyanis ez az alappillére a könyvtári munkafolyamatok hatékony elvégzésének. Azonban felvetődik egy probléma, ami az emberiség jelenlegi életében, sajnálatos módon jelen van: a pénz hiánya sok területen határt szab a fejlesztéseknek. Számos magyarországi könyvtár létezik, ahol nem valósult meg a könyvtári gépesítés folyamata. A történelmi áttekintésben is jól látható, elengedhetetlen, hogy a könyvtár, mint intézmény, lépést tudjon tartani a társadalmi, illetve technikai fejlődéssel, ezáltal hatékonyan szolgálhassa az emberiséget. Az emberiségnek pedig nem szabad engedni, hogy bármi is határt szabhasson a könyvtár és vele együtt az információáramlás fejlődésének.

Ezt a gondolatmenetet követve fogalmaztam meg a célokat. Feladatomban tűztem ki, hogy ha csekély szinten is, de hozzájárulhassak a gépesítési folyamathoz. Ezen belül segítsen a feldolgozási és a visszakereshetőségi folyamatok kialakítását olyan könyvtárakban, ahol ez a fejlődés különböző indokok miatt nem valósulhatott meg teljes egészében, illetve rosszabbik esetben el sem indult. Ennek következtében egy olyan program fejlesztésébe kezdtem, amely

¹ Könyvtárosok kézikönyve 1. / Szerk. Dr. Horváth Tibor, Papp István. – Budapest, 2003

² Könyvtárosok kézikönyve 2. / Szerk. Dr. Horváth Tibor, Papp István. – Budapest, 2003

ha nem is oldja meg a teljes gépesítési folyamatot szoftveres szinten, de nagymértékben elősegíti ezt a folyamatot.

A projektem keretein belül egy konkrét könyvtár problémája lebegett a szemem előtt. A Magyar Földrajzi Társaság könyvtári gyűjteményének feldolgozását már elkezdték a Debreceni Egyetem Informatikai Karának informatikus-könyvtáros hallgatói. Ezt a munkafolyamatot továbbgondolva egy olyan programot készítettem, mely az általuk létrehozott, bibliográfiai leírásokat tartalmazó Excel állományt, számítógép segítségével olyan formátummá konvertálja, amelyben adatbáziskezelői szinten bármiféle információ visszakeresése, szűrése megoldható.

A dolgozatom első három fejezetében egy rövid áttekintést teszek a könyvtárak automatizálásáról illetve annak problémaköreiről. A negyedik fejezetben általános áttekintést nyújtok a Magyar Földrajzi Társaságról továbbá az MFT Szakkönyvtár és a Könyvtárinformatikai Tanszék közös projektmunkájáról. Az ezeket követő fejezetekben az általam készített program bemutatására kerül sor, kiegészítve a felhasznált technológiák leírásával.

Köszönetnyilvánítás:

Ezúton szeretnék köszönetet nyilvánítani Dr. Boda Istvánnak a munkájáért, aki segített a TDK dolgozatom elkészítésében, valamint azoknak a hallgatóknak, akik részt vettek a Magyar Földrajzi Társaság könyvtári gyűjteményének feldolgozásában, illetve, hogy megosztották a témával kapcsolatos tapasztalataikat.

A TDK dolgozat elkészítését a TÁMOP-4.2.2/B-10/1-2010-0024 számú projekt támogatta.

A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósult meg.



SZÉCHENYI TERV

2. A könyvtári gépesítés története és annak szükségszerűsége

A nyugati világban már a hatvanas években megjelentek az első könyvtári számítógépes rendszerek, amelyeknek fő célja könyvjegyzékek készítése volt, később pedig a kölcsönzés adatforgalmának gépesítése. Ezt nevezhetjük a gépesítés első generációjának, aminek a mai napig kiható eredménye a katalogizáló rendszerek megalkotása volt. Hamar tudatosult ugyanis, hogy a gép segítségével előállított, olvasható rekordot később még számtalan más célra is fel lehet használni. Az angolszász könyvtárakban nagy tömegben kezdték előállítani a rekordokat, és belőlük számítógépes katalógusokat szerveztek. Hamarosan azonban jött az újabb felismerés, hogy a különböző könyvtárakban készített rekordok megosztása minden intézmény számára kölcsönösen előnyös lehet. Erre alakították ki a MARC szabványt, amelyet még a mai napig is használnak.³ Napjainkban több projekt is elindult a MARC formátum XML-ben történő leírására (például XMLMARC).⁴

A nyolcvanas években létrejött második generációt elsősorban két dolog jellemezte. Az egyik az integrált könyvtári rendszerek kialakulása, a másik pedig az elérhető áron lévő mikroszámítógépek megjelenése. Ezáltal fejlődtek a megfelelő operációs rendszerek, adatbáziskezelő, hálózati rendszerek, illetve a felhasználói felületek.

A későbbiekben azonban kiderült, hogy a már meglévő könyvtári rendszerek annyira operációs rendszer- és hardverfüggők, hogy további fejlesztésük ezen a módon már nem lehetséges, így viszont nem felelnek meg a változó könyvtári igényeknek. A könyvtári rendszerek harmadik generációjának legnagyobb előnye, hogy UNIX platform alatt futnak, és korlátlan számú terminálon lehet elérni a rendszerhez való hozzáférést.

Manapság az informatika tovább mutat minden eddigi fejlődésen. Megjelentek különböző platformok, vékony kliensek, illetve napról-napra újabb és újabb multimédiás eszközök „bukkannak fel”. A hálózatok és a World Wide Web fejlődésének köszönhetően sokkal hatékonyabb munkafolyamatok végezhetőek el, sokkal szélesebb körben.

A könyvtári gépesítésnek nagy múltja van Magyarországon is. Az első fejlesztések már a hatvanas években elkezdődtek, ezek egy része feledésbe merült, másik része közvetlenül vagy áttételesen ma is működik. Az Országos Széchényi Könyvtár az 1970-es

³ Mader Béla: Információ és intézményei. Hol tartunk, mi lesz velünk...

⁴ <http://xmlmarc.stanford.edu/>

években elsőként vezette be a számítógép használatát, amikor a Magyar Nemzeti Bibliográfiát géppel kezdte előállítani.⁵

A viszonylag hosszú alapozó időszak után a nyolcvanas évek végén indult meg a tényleges változás. Megjelentek a könyvtárak számára is elérhető áron beszerezhető személyi számítógépek. A dokumentumok számának növekedése, és az egyre nagyobb igény a korszerű, gyors, pontos információszolgáltatás iránt elengedhetetlenné tette a számítógépek használatát a könyvtárakban. A gépesített könyvtárak „hibrid” könyvtárnak számítanak. A hibrid kifejezésen olyan könyvtárat értünk, amelynek állománya papíralapú dokumentumokból áll, de a könyvtári munkafolyamatokat számítógép segítségével végzik. Azonban ezeket a könyvtárakat hívhatjuk automatizált könyvtáraknak is.⁶

A számítástechnika a hagyományos könyvtári munkák segítőjeként jelent meg a könyvtárakban, a katalóguscédulák készítésében az írógépet váltotta fel. Megjelentek az adatbázisok építését lehetővé tevő hardverek és szoftverek, majd kiépültek a helyi hálózatok. Innen már csak egy lépés volt, hogy az Internet elterjedése lehetővé tette a könyvtári katalógusok elérését a könyvtár falain kívül is. A '90-es években egyre több helyen megjelentek a korszerű integrált könyvtári számítógépes rendszerek, a multimédia valamint a World Wide Web.⁷

A fejlődés pedig tovább folytatódik, amely könyvtári szempontból sem elhanyagolható. Számtalan okból lehet szükséges a könyvtári nyilvántartás automatizálása. A legtöbb munkafolyamathoz hozzátartozik a rekordok pontos aktualizálása a megfelelő fájlokban. A kölcsönzési rendszernek pontosan tudnia kell, hogy melyik könyvet, mikor és ki kölcsönözte ki. A beszerzési adatoknak mutatniuk kell, hogy egy adott dokumentumból hány példányt rendeltek meg. Ezeknek a nyilvántartása általában unalmas, ismétlődő, mechanikus feladat. A számítógép ebben a munkában segíthet, még akkor is, ha összetett és bonyolult módon rendezett rekordokról van szó. A gépesítés a költséghatékonyságot is növeli, mivel javítja a pontosságot. Egyes munkákat alaposabban végezhetünk el, mint ahogy azt a munkaerővel való ellátottság szintje megengedi a manuális munkafolyamatok esetén, például az időszaki kiadványok meg nem érkezett számainak folyamatos reklamálása. Olyan dolgokat is elvégezhetünk számítógép segítségével, amit papíralapú állományban gyakorlatilag lehetetlen, ilyen például a jellemzők kombinációjára való keresés. Az automatizálás lehetővé

⁵ Kokas Károly: Könyvtárak az ezredforduló után

⁶ <http://ki.oszk.hu/kf/2010/10/schrader-andreas-%E2%80%93-sieweke-beate-hybrary-%E2%80%93-a-jovo-hibrid-konyvtara/>
[Letöltve: 2012, április]

⁷ Mader Béla: Információ és intézményei. Hol tartunk, mi lesz velünk...

teszi a rekordokhoz való decentralizált hozzáférést. A fiókkönyvtárban dolgozó könyvtáros könnyedén ellenőrizheti, hogy mit rendeltek meg a beszerzési osztályon, anélkül, hogy személyesen kellene felkeresnie a könyvtárat. A használó megtudhatja, hogy ki van-e kölcsönözve az általa keresett könyv, anélkül, hogy személyesen megnézné a könyvtár polcán.

Az automatizált könyvtárak nagy előnye az online katalógus, amely bárhol, bármikor elérhető. Mindezeket egybevetve az automatizált könyvtár komoly fejlődést jelent. A könyvtárak számára ez a jövő útja: a papíralapú dokumentumok és a számítógép kiegészítik egymást a könyvtárban. **Minden könyvtárnak el kell mozdulnia az informatika irányába,** mivel ez a szakterület megkönnyíti, elősegíti a könyvtári folyamatokat, így hatékonyabb kiszolgálást, munkavégzést eredményez.⁸

A másik fontos dolog, amiről említést kell tennem az a könyvári integrált rendszerek megjelenése. A számítógépek elterjedésével ugyanis majdnem az összes könyvtári folyamat megvalósítható lett informatikai környezetben. Az első könyvtári programok csak egy-egy részterületet gépesítettek, például a kölcsönzést vagy a katalógizálást. Hamarosan azonban megjelent az igény, hogy a könyvtárat, mint egységes, egész rendszert gépesítsék. Így jöttek létre az integrált könyvtári rendszerek. Az integrált könyvtári rendszerek fogalmát talán legegyszerűbben úgy lehet definiálni, hogy egy olyan rendszer, programcsomag, amely több modulból áll és az egyes könyvtári munkafolyamatokat segíti, gépesíti. Egymástól ugyan elkülönülnek, az egyes modulok külön végzik saját feladataikat, azonban egymással összefüggő rendszert alkotnak. Ebből kiindulva az integrált könyvtári rendszerek kettős feladatot teljesítenek. Egyrészt, hogy gépesítsék a könyvtári feladatokat, munkafolyamatokat, másrészt, hogy biztosítsák ezek között a kapcsolatot, megvalósítva ezzel az átjárhatóságot közöttük. A kettős funkció ellátást az úgynevezett mag adatbázis és egy relációs adatbáziskezelő rendszer biztosítja. Az adatbázis az adatok tárolásáért és visszakereshetőségért felel, valamint alkalmas arra, hogy összekapcsolja a modulokat és működjön a köztük lévő átjárhatóság. Az adatbáziskezelő rendszer táblákban tárolja az adatokat, mint például szerző, cím. Ezek között a táblák között létesít kapcsolatot, illetve lehetővé teszi, hogy felépüljön egy rekord. Ezt nevezzük relációs adatbáziskezelő rendszernek. Az adatbázisokban fellelhető tartalom lekérdezésére egy speciális nyelvet dolgoztak ki, az SQL-t (Structured Query Language). Ezt a nyelvet eredetileg az IBM dolgozta ki a saját

⁸ <http://www.bibl.u-szeged.hu/inf/szakdoli/2003/pappevi/fej4.htm> [Letöltve: 2012, április]

relációs adatbáziskezelő rendszeréhez, ma már azonban ez szabványos nyelv, amit minden adatbáziskezelőhöz használnak.

Az integrált könyvtári rendszereknek három fő csoportját különböztetjük meg. Az első típusba a nagy könyvtári rendszerek tartoznak, melyek összetett feladatok elvégzésére alkalmasak. Ilyeneket használnak például a nemzeti, illetve felsőoktatási könyvtárak. Magyarországon több magyar felsőoktatási intézmény könyvtára a CORVINA nevezetű rendszert használja. A második típusba az úgynevezett közepes könyvtárakat kiszolgáló rendszerek tartoznak. Ezeket a rendszereket főleg városi illetve szakkönyvtárak használják. A legismertebb ilyen rendszer Magyarországon a TEXTLIB. A harmadik csoportba a kisebb könyvtári rendszerek tartoznak, melyek alacsony árúak és egy PC-n is használhatóak. Leginkább az iskolai könyvtárak számára ideálisak. Ilyen rendszer a Szirén, mely magyar fejlesztésű és futtatható Windows illetve Linux operációs rendszerek alatt is.⁹

Az integrált könyvtári rendszerek általában öt fő modulból épülnek fel, melyek öt nagy területet gépesítenek. Ezek a következők: beszerzés/gyarapítás, katalogizálás, kölcsönzés, folyóirat-kezelés és a számítógépes katalógus. Ezzel az öt modullal gyakorlatilag az összes könyvtári folyamat lefedhető. A dolgozatom témáját tekintve a katalogizálási modult szükséges bővebben kifejteni. A kategorizálási modul az adatok, konkrétan a bibliográfiai adatok bevitelére szolgál. Minden fajta dokumentumot képes feldolgozni, ezt különböző űrlapok segítségével oldja meg. A feldolgozásra két lehetőség adott. Az első a kézi feldolgozás, ahol az űrlap megfelelő mezőibe manuálisan írják be a bibliográfiai adatokat. A másik lehetőség, amikor valamilyen katalógusban már fellelhető dokumentumot - mert ott már feldolgozták - átemelhetik, és csak hozzáírják saját példányadataikat. Jól látható, hogy az Internet használata a katalogizálás során is elengedhetetlen, hisz így munkát lehet megtakarítani. A katalogizálás során bibliográfiai rekordok készülnek, amihez általában a MARC (Machine Readable Cataloguing) formátumot használják. Ez a szabvány biztosítja a könyvtárak közötti adatcserét, ez teszi lehetővé a könyvtári hálózatok működését.¹⁰

⁹ <http://www.bibl.u-szeged.hu/inf/szakdoli/2003/pappevi/fej2.htm#intkvt> [Letöltve: 2012, április]

¹⁰ Bakonyi Géza - Kokas Károly: Könyvtári integrált rendszerek és hazai alkalmazásuk

3. A könyvtári automatizálás problémái

Az előző rövid áttekintéséből következően látható, hogy fontos a metainformáció, jelen esetben a könyvek és a bennük foglalt információk feltárása. Ebben a könyvtárnak van a legnagyobb szerepe, mint például feltárni egy dokumentumot, visszakereshetővé tenni azt, illetve kiszolgálni a felhasználókat. Ezek az úgymond alapfeladatok a történelem során más-más formában mindig megtalálhatóak voltak. A gépesítési folyamat csak segítette ezen folyamatok, illetve feladatok hatékonyságát. Azonban nem csak a hatékonyságot és precizitást, hanem a feldolgozással járó időt is lerövidítette. A szolgáltatásokat, mint például a kölcsönzést vagy információ visszakeresést szintén leegyszerűsítette. Gondolok itt arra, hogy egy felhasználónak elég csak a könyvtár honlapjára felmenni és az online katalógusban kikeresheti a könyvet, amit akár összetett keresés igénybevételével is végezhet. Fontos megjegyezni, hogy az összetett keresés számítógép nélkül lehetetlen. Ezen kívül a felhasználó előjegyezheti saját maga számára a kívánt dokumentumot és a továbbiakban már csak annyi feladata maradt, hogy személyesen megjelenjen az adott produktumért. Tehát fontos leszögezni, hogy az automatizálás valamint a számítógép segíti, javítja és hatékonyabbá teszi a könyvtári feladatok elvégzését. Mindez pedig szükséges a fejlődő világunkban. Itt nem csak az informatikai, technikai fejlődésről van szó, hanem magáról az ember, illetve az emberi szociális fejlődéséről, ami maga után vonja az információ illetve annak terjedésének fejlődését. Az interneten való tájékozódás, információszerzés mára alapvető lett az emberek számára. Mára már tudatosult az emberekben maga az információ fontossága, illetve annak hasznossága. Ez abból is látszik, hogy a XX. századra exponenciálisan megnőtt a nyilvánosan hozzáférhető információ mennyisége, persze ennek úgymond előfeltétele volt még az internet megjelenése is majdnem minden háztartásban. Ám ez rengeteg veszélyt hordoz magában, legfőképpen a hitelesség meglétét. Az interneten felelhető adatok nagy százaléka felülvizsgálatlan információ, nem hiteles, ami információtorzuláshoz vezet. A könyvtár egyik alapvető szerepe, hogy válogatott, hiteles információval szolgáljon a felhasználó számára. Ebből is látható, hogy mennyire fontos a technikai fejlődéssel megfogni az emberi, szociális fejlődés rohamosan növekvő - és valószínűleg soha nem csökkenő - terméket, magát az információt. Erre a legnagyszerűbb intézmény a könyvtár.¹¹

¹¹ Bakonyi Géza: Számítógép a könyvtárban, könyvtár a számítógépben

Tehát a technika, az informatikai háttér mára elérte azt a szintet, hogy tökéletes háttérrel adjon ezeknek a feladatoknak a megvalósításához. Ám sajnálatos módon még mindig van olyan tényező, ami határt szabhat ennek a fontos folyamatnak, hátráltathatja a fejlődést. Sok könyvtár számára még mindig elérhetetlen az automatizálás. Ez köszönhető részben a területi elhelyezkedésüknek, de legfőbb szempont, amely hátráltatja, illetve megakadályozza a fejlődési folyamatot az nem más, mint a pénzhiány. Ha pedig valahol nincs meg a kellően hatékony eszköz - gondolok itt az informatikára illetve az eszközparkra, mint a számítógépre, nem is beszélve az emberi erőforrásigényről - akkor a kívánt automatizálási folyamat nem kezdődhet meg. A nagyobb könyvtárakban az egyes könyvtári munkafolyamatok automatizálása már elkezdődött vagy végbement, ezek a számítógépes rendszerek kisebb, nagyobb sikerekkel működnek. Természetesen a kisebb községek könyvtáraiban, iskolai könyvtárakban is felmerült a számítógépes könyvtári nyilvántartások kialakításának igénye, azonban a megvalósításnak még sok helyen határt szab a szűkös anyagi keret. Számos helyen problémaként merült fel a hardverkörnyezet korszerűtlensége, annak hiányossága, rossz esetben pedig teljes hiánya. Előnyösebb az az eset, amikor a számítógép megtalálható a könyvtárakban, ám a hardverkörnyezet alacsony szintű és kis teljesítményű. Rosszabbik eset, ha fennáll ezeknek a teljes hiánya. Számos könyvtárban: iskolai könyvtárak, községek illetve kisebb városok könyvtáraiban az automatizálás még nem ment végbe.¹² A dolgozatomban szempontjából legmeghatározóbb példa erre egy a gépesítéssel kapcsolatos probléma, **amikor a katalógusok vezetése még mindig egy „sima” Excel fájlban történik.** Minden könyvtárgépesítés alapja a bibliográfiai rekordok elektronikus formában történő előállítása. Ehhez viszonylag egyszerű, és gyakorlatilag bármilyen számítógépen hozzáférhető programok is felhasználhatóak. Ilyen lehet az MS Excel. De ez csak az első lépés. Ebben az esetben ugyanis az információ visszakereshetősége elég bonyodalmas, sőt a nagyobb integrált rendszerekhez képest ez a megoldás szinte nem is működik. A megoldás pedig néha egyszerű dolgokban rejlik. Céлом volt egy olyan szoftver kidolgozása, amely azokat a könyvtárakat célozza meg, ahol még a gépesítési folyamat nem zajlott le, így az információ feldolgozás, tárolás megoldatlan problémát jelent, vagy ha jelen is van, elég kezdetleges cipőben jár, aminek következménye, hogy az információ visszakeresés nem kellőképpen hatékony. A konvertáló program elkészítése során a Magyar Földrajzi Társaság könyvtári gyűjteményének feldolgozásával kapcsolatos probléma megoldásosára törekedtem.

¹² Salgáné Medveczki Marianna: Az XML: új perspektívák a könyvtár-informatikában

4. A Magyar Földrajzi Társaság Szakkönyvtára és annak problémája

Az automatizálás hiányosságára számtalan példát lehetne felhozni, ám dolgozatomban, illetve az általam írt program elkészítése többek között egy konkrét könyvtárat kíván segíteni. A Debreceni Egyetem Informatikai Karának és a Magyar Földrajzi Társaság Szakkönyvtárának kapcsolata adta a program alap ötletét. Továbbá a Könyvtárinformatikai Tanszék hallgatói által végzett már két éves munka továbbvitelének a célja vezetett.



1. ábra: Az MFT emblémája

Forrás:

http://geo.science.unideb.hu/page/MFTprogram_2012.html

Az 1872-ben alapított Magyar Földrajzi Társaság (továbbiakban MFT) az egyik legrégebbi tudományos társaság hazánkban. Az „alapító atyák” fő célja szerint a magyarországi geográfiai eredmények fejlesztését és azok közkinccsé tételét, valamint a földrajz népszerűsítését igyekszik elősegíteni. Tagjai mindig is a magyar földrajztudomány legkiemelkedőbb alakjai voltak: tudósok, egyetemi professzorok, középiskolai és felsőoktatásbeli tanárok, valamint a társadalom bármely rétegéből kikerülő, a földrajz iránt érdeklődő személyek. A Magyar Földrajzi Társaság nonprofit közösség, mely kiemelten közhasznú szerepet tölt be. Tagjait a geográfia iránt érzett tisztelet és elkötelezettség vezérli. Napjainkra a világ változásával együtt a tudomány iránt támasztott elvárások és az általános társadalmi érdeklődés jellege is változott a földrajzzal kapcsolatban. Kutatási eredményeit számos területen hasznosítják, mint például a turizmus, természet- és környezetvédelem, közigazgatás. Az egyre színesedő területek művelőit és az ezek iránt érdeklődőket fogja össze az MFT rendezvényeivel és népszerűsítő munkáival. A fent leírt szándékok szolgálatára hozták létre az MFT Szakkönyvtárát, mely széles körben gyűjt földrajztudományi műveket. A könyvtár gyűjteményének sokszínűségét a gyarapító források sokfélesége adja. Főúri családok támogatása által hagyatékok és kéziratok is kerültek a **mintegy 100.000 kötetes gyűjteményébe**, ahol könyvek, időszaki kiadványok, térképek, kéziratok és levéltári anyagok

is megtalálhatóak. Muzeális dokumentumok nem tartoznak az intézmény gyűjtőkörébe, ezek az Érden található Magyar Földrajzi Múzeumba kerülnek át.¹³

A könyvtár számos problémával küzd. Terebélyes gyűjteménye számára szűkösnek bizonyul a könyvtárhelyiség által kínált férőhely. Bizonyos állományrészeket emiatt túlsúlyfoltosság jellemez, így vannak olyan dokumentumok, melyeket egyszerűen nem tud a könyvtár a legmegfelelőbb elhelyezésben tárolni. Mivel rendkívül értékes gyűjteményről van szó, érdemes lenne ezek számára nagyobb férőhelyet biztosítani. Az állomány kötetei csupán tárgyszavakkal vannak ellátva, speciális osztályozási rendszerrel nem tárták fel őket. A gyűjtemény formai feltárására már több kísérlet indult. Geoinformatikus hallgatók kezdték el az „A” betűjelű dokumentumok katalogizálását, és mintegy 2500 rekordot sikerült létrehozniuk. A későbbiekben egy szakképzett könyvtáros bevonásával a B209-es jelzetig kerültek leírásra a dokumentumok, s az adatokat Excel állományban tárolták el. Így közel kétszáz rekord készült el. A tanszék hallgatóinak célja a már elkezdett feltáró munka továbbfolytatása volt, mégpedig egy olyan elérhető rendszer segítségével, mely a jövőben is eszközt biztosít a dokumentumok feltárásához. Mivel a Huntéka rendszerhez való feldolgozói hozzáférés finanszírozására nincs a társaságnak anyagi forrásai, ezért a rekordokat ideiglenesen egy Excel táblázatban készítették el. A táblázatban az alábbiakban felsorolt adatelemek kerültek leírásra:

- Raktári jelzet
- Szerző(k)
- Közreműködő(k)
- Főcím
- Párhuzamos cím
- Alcím
- Kiadás jelzés
- Kiadás éve
- 1851 előtti-e (igen/nem)
- Kiadás helye
- Kiadó
- Terjedelem

¹³ <http://www.foldrajzitorsasag.hu/rolunk> [Letöltve: 2012, április]

- Sorozatcím, ISSN ; sorozati szám
- Terjesztési adatok
- Hiányzó tétel jelzése
- MFT példányszám (hány példány található belőle ebben a gyűjteményben)
- Megjegyzés¹⁴

A feltárást tekintve megfelelő eszköznek bizonyult az említett Excel fájl, azonban ahhoz, hogy ezt a fájlt fel lehessen tölteni a Huntéka vagy bármilyen más adatbázisba - ezzel is gyarapítani a rekordjainak számát és visszakereshetővé tenni a könyvtár produktumait - szükséges az Excel fájlban tárolt adatok átalakítása (konvertálása). Itt kapcsolódik be az én munkám e kiemelt jelentőségű munkafolyamatba. **Egy olyan programot írtam, mely képes Excel fájlokat konvertálni XML fájlokba.** Ha már elérhető az XML fájl, onnantól kezdve bármilyen adatbázisba feltölthető, ezáltal hatékonyabbá tehető a visszakeresés és teljes mértékben megoldható az összetett keresés, ami egy Excel fájlban lehetetlen az adatbázis rendszerek lekérdező nyelvéhez képest.

¹⁴ Az MFT könyvtárában történő munkavégzésről, a projektmunka keretein belül ott dolgozó, csoporttársaimtól informálódtam.

5. Felhasznált technológiák ismertetése

A felhasznált technológiák kiválasztása során törekedtem arra, hogy olyan környezetekben dolgozzak, illetve olyan technológiákat használjak fel, amelyeket a tanulmányaim során sikerült elsajátítanom. A jelenlegi fejezetben bemutatom a programozási környezetet, amelyben a programom írtam, illetve azokat a technológiákat, melyek részére készült ez a program. Bemutatásra kerül az XML (magyarul Kiterjeszthető Leíró Nyelv), valamint az XML Schema nyelv, illetve azon adatbáziskezelő rendszerek, programok, melyek kezelni tudják az XML fájlokat.

5.1 A Java programozási környezet

A Java egy általános célú, azaz szoftverírásban széles körben felhasználható, objektumorientált nyelv. Az objektumorientáltság azt jelenti, hogy a korábbi programozási nyelvektől eltérően nem a programozási műveleteken van a hangsúly, hanem az egymással kapcsolatban lévő programegységek hierarchiájának megalkotásán. Az objektumorientált gondolkodásmód lényegében a világ modellezését reprezentálja. Egy hétköznapi fogalom, például az autó értelmezhető egy osztály tagjaként, mégpedig az autók osztályának objektumaként. Az összes autó objektum rendelkezik az autókra jellemző tulajdonságokkal (például méret, szín) és cselekvési képességekkel (például gyorsulás, lassulás). Az objektumorientált nyelveknek fontos tulajdonsága továbbá az öröklődés. Ez teszi lehetővé az osztályok egymásból történő származtatását (például az autó osztály származhat a közlekedési eszközök osztályából, így megörökölheti a közlekedési eszközök osztályának képességeit).

A nyelv fejlesztését a Sun Microsystem végzi a '90-es évektől, a nyelv azóta is dinamikusan fejlődik, függetlenül attól, hogy 2009-ben az ORACLE felvásárolta a Sun Microsystem-t, így jelentős vállalati politika átalakuláson ment keresztül¹⁵. A Java egyre több feladat elvégzésre alkalmas, ennek köszönhetően széles körben alkalmazzák. A dinamikus fejlődés eredményeképp mára három Java platform van jelen a piacon. A Java Standard Edition (SE) a hagyományos asztali programok, appletek fejlesztését teszi lehetővé. Erre a Java már a kezdetektől fogva alkalmas volt. A Java Micro Edition (ME) a mobil eszközökre

¹⁵ http://itcafe.hu/hir/az_oracle_veszi_meg_a_sunt.html

történő fejlesztést teszi lehetővé. Az ME jellemzője, hogy nem rendelkezik minden az SE-ben megtalálható implementációval a céleszközök (mobilkészülékek) erőforrásainak korlátozottságai miatt, azonban tartalmaz olyan specifikációkat, amelyek csak az ME környezetében fordulnak elő. A harmadik egyben legnagyobb méretű kiadás a Java Enterprise Edition (EE), amely a több felhasználós vállalati fejlesztéseket teszi lehetővé.¹⁶

A Java jellemzője, hogy platform független, amelyet olyan módon valósít meg, hogy a programokat egy köztes nyelvre fordítja le, majd ez a kód egy úgynevezett futtató környezetben, a Java Virtuális Gépen fut. Ez a futtató környezet biztosítja, hogy a kód minden platformon megegyezően, ugyanúgy működjön. További jellemzője, hogy a Java programcsalád bizonyos tagjai ingyenesen elérhetőek. Így a NetBeans nevű fejlesztői környezet is ingyenesen elérhető, valamint a benne írt Java nyelven fejlesztett program is ingyenes, ez által szabadon felhasználható, terjeszthető.¹⁷ Jelen esetben ez rendkívül fontos tényező, mivel a programomat térítésmentesen szeretném felajánlani azon könyvtáraknak, ahol a gépesítési folyamatok elmaradtak.

5.2 Az XML bemutatása

Az előző fejezetekben bemutatott problémákhoz, valamint a programom tényleges ismertetése előtt fontosnak tartom az XML nyelvet bemutatni, mind általánosan, mind pedig a különböző adatbáziskezelő programok által várt XML fájlokon keresztül.

Az XML és a hozzá tartozó szabványos eljárások új lehetőséget biztosítanak különböző információk, szöveg és más médiák együttes kezeléséhez. Az XML a World Wide Web Consortium (W3C) által kifejlesztett hatékony információfeldolgozási illetve tárolási és megjelenítési eszköz. Az XML-t két nyelvre építették, elsősorban az SGML szabályaira, de a HTML jellemzőiből is átvett tulajdonságokkal egészítették ki, melyek megkönnyítik az információ interneten történő felhasználást.

Megjelenése hatalmas érdeklődést váltott ki az informatikusok világában. A nyelv sajátossága, hogy egyszerű szabályrendszerrel rendelkezik, ezáltal könnyen használható. Továbbá nyitott szabvány így azon kevés nyelvek között van, aminek nincs tulajdonosa és semmiféle gazdasági csoporthoz, szervezethez nem köthető, így hamar elterjedt és széles

¹⁶ <http://www.java.com/>

¹⁷ http://hu.wikipedia.org/wiki/Java_%28programoz%C3%A1si_nyelv%29

körben preferálják. **Az XML egy olyan metanyelv, mely segítségével további jelölőnyelvek készíthetők.** Ezek együtt információk tartalmi leírását teszik lehetővé, amely olvasható mind a számítógép, mind az ember számára. Fontos kiemelni azt, hogy az XML nem bináris adat, tehát ember számára is feldolgozható, olvasható.

Tökéletesen felhasználható a különböző médiákon megjelenített, több tagból felépített vagy félig felépített (más néven strukturált vagy fél strukturált) szövegek számára. Az XML nyelv segítségével leírt információ a médiához, illetve a célközönség követelményeinek megfelelően alakítható, formázható. Továbbá lehetőség nyílik a rendszerek, programok közötti információcserére. Azonban nélkülözhetetlen, hogy a kliensprogram értelmezni tudja a kapott információt és ne kelljen újra a kiszolgálóhoz kapcsolódnia.

Kétféle szerkezeetről beszélhetünk egy XML fájlban belül. Beszélhetünk logikai illetve fizikai szerkezeetről. A logikai szerkezet teszi lehetővé, hogy magát a dokumentumot nevesített egységekre bontsuk, amiket elemeknek nevezünk. A fizikai struktúrában lehetőség van az egyes dokumentumrészek (más néven egyedek) nevesítésére, valamint olyan különböző adatállományokban tárolni azokat, amelyekből maga az információ újra kinyerhető és más nem XML típusú adatokkal bővíthető, ilyen adat például egy kép fájl.

Az XML fájl nem előre meghatározott elemlistából épül fel, ezáltal könnyen bővíthető. A HTML nyelvvel szemben lehetőség van saját elemek deklarálására. Az egyes elemek egymásba ágyazhatóak, egy úgynevezett gyökérelembe ágyazva. A gyökérelemet talán úgy legegyszerűbb definiálni, hogy az a legmagasabb szinten elhelyezkedő elem, amelybe valamennyi elem beágyazható. Az XML nyelv elemei alkotják a definiált nyelv „szókincsét”, ezek egymáshoz kapcsolásának szabályszerűségei adják a nyelv „nyelvtanát”. Bárki létrehozhat egy saját nyelvet, dokumentumtípust, amennyiben betartja a megadott szintaktikai szabályokat. A formális szabályokat egy úgynevezett dokumentum típus definícióval adhatjuk meg, más néven egy DTD fájl segítségével, illetve ennek „modernebb” változatával, az XML Schema-val. Ez egy külön álló fájl, amit az XML fájlhoz kell kapcsolni. A DTD meghatározza az előfordulható elemeket, azok viszonyát, jellemzőit illetve az attribútumokat valamint a hozzájuk tartozó adatok típusát. Segítségével tehát előállíthatjuk az XML fájl logikai szerkezetét, gyakorlatilag a dokumentum felépítését, hierarchiáját és tagoltságát. Tagoltság alatt azt értjük, hogy az egyes elemeket mennyire bontjuk fel gyermek, más néven leszármazott elemekre. Az XML nyelv egyik legfőbb szabálya, ami a HTML nyelvre is igaz, hogy nyitó és záró jelölőelemekkel vesznek körbe egy adott információt. A

nyitó jelölőelem, illetve a záró jelölőelem, valamint a köztük lévő információ adja meg az elemi egységet. Tehát a jelölő elemek mindig párosával szerepelnek és „körbeölelik” az információt. Az egyes elemek egymásba ágyazhatóak, tetszőleges mélységben, azonban az egészet egy gyökérelembe kell ágyazni, zárni.

A nyelv szabályaiból láthatjuk, hogy bárki szabadon létrehozhat egy struktúrával, elemkészlettel rendelkező új, megfelelő nyelvet, azonban jó nyelvet létrehozni nem egyszerű feladat. Előregondoltan meg kell határozni a jelölőelemeket, azoknak kapcsolatát, beágyazásaikat, az attribútumokat illetve, hogy azok milyen típusú adatokkal tölthetők fel. Saját nyelvet kidolgozni tehát korántsem egyszerű feladat, ám **a nagyobb adatbázis kezelő rendszerek más-más szintaktikai felépítésű XML-t állítanak elő és a tökéletes használathoz is sajátos XML fájlt várnak.** Ilyen adatbázis kezelő rendszerek például az MS Access és a MySQL. Mindkettő XML fájl struktúrájával érdemes foglalkozni, mivel az általam kidolgozott program e kettő rendszer számára állít elő olyan XML fájlt, amit ezek a rendszerek - úgymond - elvárnak.¹⁸

Példaként itt mutatok be egy általános, a fent leírtakat követve saját dokumentum típus deklarációval (DTD) rendelkező XML fájlt.

Konyvek.xml (részlet)¹⁹:

```
<?xml version="1.0" encoding="windows-1250" standalone="yes"?>
<!DOCTYPE Konyvek [
<!ELEMENT Konyvek (Konyv)+>
<!ELEMENT Konyv ( Rakj, Szerz, Cím, Kiad, Kiadev, ISBN, Peld)>
<!ELEMENT Rakj (#PCDATA)>
<!ELEMENT Szerz (#PCDATA)>
<!ELEMENT Cim (#PCDATA)>
<!ELEMENT Kiad (#PCDATA)>
<!ELEMENT Kiadev (#PCDATA)>
<!ELEMENT ISBN (#PCDATA)>
<!ELEMENT Peld (#PCDATA)>
]>
```

¹⁸ http://www.w3schools.com/xml/xml_what_is.asp

¹⁹ A teljes forráskód megtalálható a 2. számú mellékletben

```

<konyvek>
<konyv>
<Rakj>1</Rakj>
<Szerz>Jókai Mór</Szerz>
<Cim>Az arany ember</Cim>
<Kiad>Presskontakt</Kiad>
<Kiadev>2008</Kiadev>
<ISBN>9789639503397</ISBN>
<Peld>2</Peld>
</konyv>
...
</konyvek>

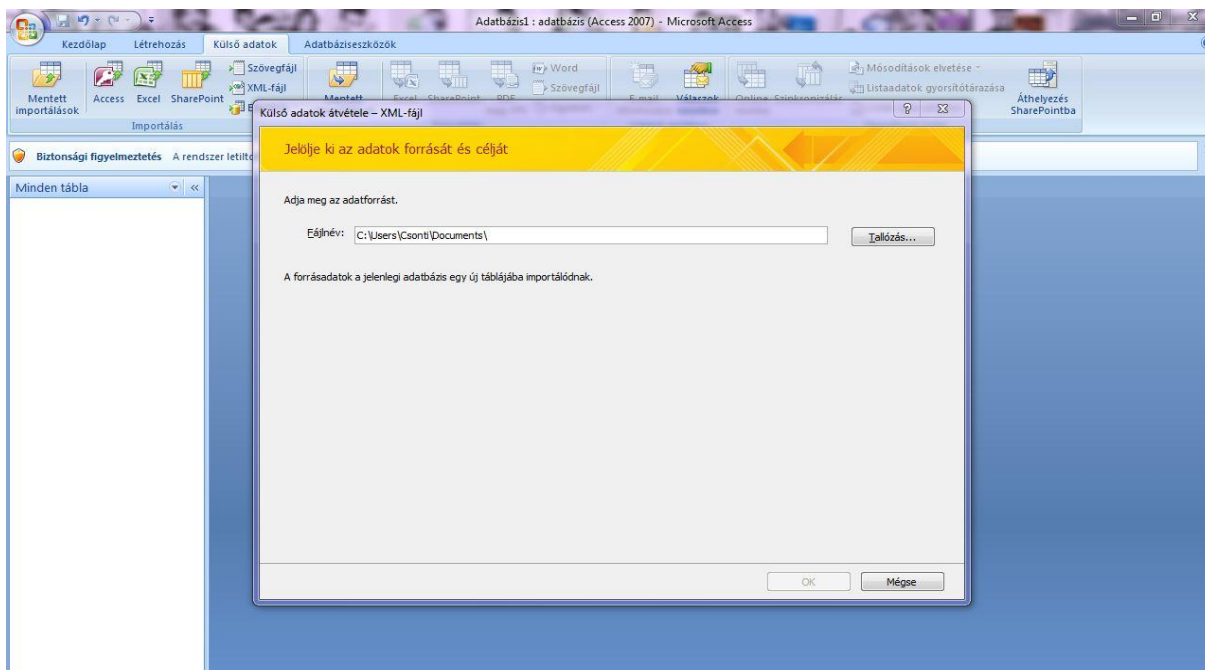
```

Látható, hogy az első sorban adtuk meg az XML fájl verzióját, valamint a kódolását. A „*standalone=yes*” parancssal meghatározzuk, hogy belső dokumentum típus deklarációval dolgozunk, ugyan is a DTD lehet belső, azaz az XML fájlal egybeépített, illetve lehet különálló fájl is. Utána meghatározzuk az elemeket, valamint azok egymásba ágyazhatóságának szabályait és a feltölthető adatok megnevezését, esetleg típusát.

XML – Access kapcsolata: Az Access által várt XML fájl felépítése

A Microsoft Access lehetővé teszi XML formátumú adatok importálását és exportálását. Az Access által várt XML fájl struktúrájának a legfontosabb sajátossága a gyökérelem meghatározása. A gyökérelem ebben az esetben a `<dataroot>`. Minden esetben, ha az Access adatbáziskezelő program segítségével exportálunk egy XML fájlt, akkor a gyökérelem `<dataroot>` lesz. A névtér pedig a következő: `dataroot xmlns:od="urn:schemas-microsoft-com:officedata"`. Ezeket a sajátosságokat ismerve szeretném röviden bemutatni, hogy is néz ki a Microsoft adatbáziskezelő programjába történő XML fájlimportálás.

Az XML-adatfájlok importálásához válasszuk a Fájl menü „*Külső adatok átvétele*” pontjának „*Importálás*” parancsát (2. ábra). A megjelenő importálási párbeszédpanelen kiválaszthatunk egy XML-dokumentumot vagy egy, az adatszerkezetet leíró XML-sémafájlt.



2. ábra: XML fájlok importálása az Access adatbáziskezelő rendszerben

Forrás: Saját készítés, 2012

Az Access programba egyszerre csak egy dokumentumot importálhatunk. Az adatok formátumának (vagy a szerkezetleíró fájl által átalakított formátumnak) az Access által kezelhetőnek kell lennie. Ennek függvényében fontos tehát, hogy egy olyan XML-t szolgáltatassunk a program számára, ami általa kezelhető. Ha az XML fájl megfelelő formátumú, az Access segítségével előállítható a tábla. Egy adattáblában a lekérdezőnyelv segítségével már megoldható bármiféle összetett keresés, ezáltal hatékonyabban lehet visszakeresni az információt.

Itt érdemes szót ejteni a XML sémákról is bővebben, mivel az Access is támogat egy séma formátumot, ami nem más, mint az XSD. Az XSD az XML standard séma. Az XML-sémák - így az XSD is - az adatszerkezetet olyan általános formátumban írják le, hogy az a felhasználók és az XML kezelésére alkalmas bármely program számára felismerhető legyen. A sémák határozzák meg az XML-adattáblán belüli szabályokat, az elemek neveit, az adattípusokat, az egyes elemek beágyazhatóságát, valamint az ezekhez elérhető attribútumokat. Az XSD a World Wide Web Consortium (W3C) által jóváhagyott szabvány az XML-dokumentumok típusa és szerkezete alapvető kialakításának leírásához.²⁰

Séma használatával biztosíthatjuk azt, hogy az Access programba való importáláshoz vagy az onnan más formátumba való exportáláshoz használt XML-dokumentumok a

²⁰ <http://www.w3schools.com/Schema/>

megfelelő adatokat adott struktúrában tartalmazzák. Ha az általunk használt sémát más könyvtárak rendelkezésére bocsátjuk, megfelelő szerkezetben küldhetik azután az adatokat. „Viszonzásul” használhatjuk az általuk használt szerkezetet a nekik szánt adatok továbbításakor. Példaként bemutatnám a fent leírt általános XML fájlt, átalakítva olyan fájlá, ami megfelel az Access által előírt szabályoknak.²¹

Konyvek.xml (részlet)²²:

```
<?xml version="1.0" encoding="windows-1250"?>
<dataroot      xmlns:od="urn:schemas-microsoft-com:officedata"      xmlns:xsi=
http://www.w3.org/2001/XMLSchema-instance      xsi:noNamespaceSchemaLocation=
"C:\Pelda\Konyvek_tulajdonsag.xsd">
  <Konyv>
    <Rakj>1</Rakj>
    <Szerz>Jókai Mór</Szerz>
    <Cim>Az arany ember</Cim>
    <Kiad>Presskontakt</Kiad>
    <Kiadev>2008</Kiadev>
    <ISBN>9789639503397</ISBN>
    <Peld>2</Peld>
  </ Konyv >
</dataroot>
```

Konyvek_tulajdonsag.xsd (részlet)²³:

```
<?xml version="1.0" encoding="windows-1250"?>
<xsd:schema      xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:od="urn:schemas-microsoft-com:officedata">
  <xsd:element name="dataroot">
    <xsd:complexType>
      <xsd:sequence>
```

²¹ <http://office.microsoft.com/en-us/access-help/using-xml-and-access-HA001034560.aspx>

²² A teljes forráskód megtalálható a 3. számú mellékletben

²³ A teljes forráskód megtalálható a 3. számú mellékletben

```

<xsd:element ref="Konyv" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="generated" type="xsd:dateTime"/>
</xsd:complexType>
</xsd:element>
<xsd:element name="konyv">
<xsd:annotation>
<xsd:appinfo>
</xsd:appinfo>
</xsd:annotation>
<xsd:complexType>
<xsd:sequence>
<xsd:element name="Rakj" minOccurs="unbounded" od:jetType="text"
od:sqlSType="Int" type="xsd:Int">
</xsd:element>
<xsd:element name="Szerz" minOccurs="unbounded" od:jetType="text"
od:sqlSType="string" type="xsd:string">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:maxLength value="30"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:element>
<xsd:element name="Cim" minOccurs="unbounded" od:jetType="text"
od:sqlSType="string" type="xsd:string">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:maxLength value="255"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:element>

```

XML – MySQL kapcsolata: A MySQL által várt XML fájl felépítése

Az Accesshez hasonlóan a MySQL is sajátos felépítésű XML fájlt vár. Azonban nem szimplán csak a gyökérelem megadásában tér el, hanem ezen felül az egyes elemek megnevezése is sajátos. Itt is változik a gyökérelem, mégpedig a `<list>` illetve `</list>` elemeket kell használnunk. A MySQL sajátos elemdeklarációt használ, azaz nem úgy határozunk meg egy elemet, mint egy egyszerű XML fájlban, hanem egy elem `<field-name="pelda">adat</field>` formátumként adható meg. Tehát az eddigi példákon keresztül bemutatott elemneveket most egy „name” attribútumhoz párosítjuk idézőjelben, majd ezután adhatjuk meg az elemhez tartozó adatot. Sajátos különlegesség ezen felül az egyes elemcsoportokhoz tartozó azonosító. Minden elemcsoportban az „anyaelem” után az első beágyazott elem tartalmazza az „anyaelem” azonosítóját. Ez az azonosító egy szám, mégpedig az elemcsoportok sorszámát adja meg. Példaként a kód egy részletét itt is bemutatnám, a teljes fájl kódja a mellékletben megtalálható.²⁴

Konyvek.xml (részlet)²⁵:

```
<?xml version="1.0" encoding="windows-1250"?>
<list>
  <konyv>
    <field name="konyv_id">1</field>
    <field name="Rakj">1</field>
    <field name="Szerz">Jókai Mór</field>
    <field name="Cim">Az arany ember</field>
    <field name="Kiad">Budapest Kiadó</field>
    <field name="Kiadev">2008</field>
    <field name="ISBN">9789639503397</field>
    <field name="Peld">2</field>
  </konyv>
```

²⁴ <http://www.databasejournal.com/features/mysql/article.php/3846526/Working-with-XML-Data-in-MySQL.htm>

²⁵ A teljes forráskód megtalálható a 4. számú mellékletben

6. XLS2XML konvertáló program bemutatása

Az előző fejezetekben bemutatott információk tudatában készítettem el a programom. Ismert az XML nyelvnek számos olyan, a könyvtári munka szempontjából előnyös tulajdonsága, amely lehetővé teszi az XML könyvtári területen történő sikeres felhasználását. A számítógép számára megkönnyíti az adatok generálását, olvasását, és biztosítja, hogy az adatszerkezet egyértelmű legyen. Biztosítja az információk többszöri felhasználását, kereshetőségét, számítógépes feldolgozhatóságát, információk összekötését, az információ karbantartását és a minőséget. Lehetővé teszi a bibliográfiai leírásban rejlő lehetőségek jobb kihasználását anélkül, hogy rontana annak azonosító funkcióján.²⁶ Az általam készített program egy konkrét Excel fájlt konvertál át egy konkrét XML fájlra, ami ezek után már betölthető egy adatbázis rendszerbe.

A program előre megszabott Excel fájl struktúrát feltételez (3. ábra). Mindez annyit tesz, hogy a táblázat első sorában, azaz a „header” elemekben, az oszlopok nevét kell megadni, mert később ezek adják meg az XML fájl azon elemeit, melyekbe beágyazásra kerül az oszlop többi mezője, azaz maga az információ. Egyetlen egy kikötést tettem, mégpedig, hogy az Excel fájl „header” mezőibe nem kerülhetnek központoszási karakterek (pl. „(){}[]-stb.). Úgymond, teljesen új, idegen dolgot nem vár el a program, hisz egy jól megszerkesztett Excel fájl ezt a struktúrát követeli.

	A	B	C	D	E	F	G
1	Rakj	Szerz	Cím	Kiad	Kiadev	ISBN	Peld
2	1	Jókai Mór	Az arany ember	Budapest Kiadó	2008	9789639503397	2
3	2	Alan Alexander Milne	Micimackó	Holló Kiadó	2010	9789631187342	3
4	3	Charlaine Harris	Inni és élni hagyni	Alma Kiadó	2009	9789632542775	1
5	4	George R. R. Martin	Trónok harca	Figure Kiadó	2008	9789633707784	3

3. ábra: Előre meghatározott struktúrával felépített Excel fájl

Forrás: Saját készítés, 2012

Mivel a program a Magyar Földrajzi Társaság könyvtárában történő projekt munka kereteiben belül vezetett Excel fájlt kívánja adatbáziskezelő rendszerek számára elérhetővé tenni, így azt az Excel fájlt vettem alapul, amikor meghatároztam a struktúrát.

A programot Java fejlesztői környezetben írtam. Elsősorban nem csak a tanulmányaim miatt választottam ezt a programozási környezetet. Fő célkitűzésem, hogy ezt a programot

²⁶ Salgáné Medveczki Marianna: Az XML: új perspektívák a könyvtár-informatikában

bármely könyvtár szabadon használja, magát a produktumot pedig kizárólag olyan program nyelven készíthetem el, mely nyílt forráskódú, tehát az általa létrehozott program szabad szoftver. Így maga a szoftver is szabadon terjeszthető, felhasználható.

Elsődleges célom az volt, hogy minél egyszerűbben működjön és a felület, úgymond „akadálymentes” legyen. Ennek céljából négy darab menüpont található, gomb formájában a főablakon. Ezek közül mindig az aktuálisan használható gomb aktív, ennek köszönhetően a program kezelése meglehetősen könnyű. Különösen fontosnak tartottam mind a funkcionális, mind a grafikai egyszerűséget, mivel a célkönyvtárak, melyek számára elsősorban készült ez a program, technikailag fejletlenebb könyvtárak. Ez a fejletlenség a pénzhiány mellett a regionális adottságok miatt is fennállhat. Nagyobb valószínűséggel az ilyen könyvtárakban – különösen hangsúlyoznám itt a kisebb térségi, községi könyvtárakat – a könyvtárosok számára az informatikai képzés még nem volt előírás, így bármiféle informatikai munkafolyamat elsajátítása bonyolultabb feladatot jelent. Ehhez igazodva szükséges, olyan mértékben könnyíteni helyzetüket az informatika terén, amilyen szinten csak lehetséges.

Ezeknek értelmében alakítottam ki az aktuális design-t és működési elvet. Mindig azon gombok aktívak, melyek funkciójára éppen sor kerül, valamint a gombok mérete is pont kényelmes, mindenki számára jól látható, de ezek mellett nem mondható zavarónak annak nagysága. Amikor a fájl betöltésére kerül a sor, mindig csak a betölthető fájlok típusa látható, így az Excel fájlokon kívül más dokumentum megnyitása konvertálásra nem lehetséges a programmal. Ezáltal kiküszöbölhetőek az alapvető felhasználói problémák, amik a kezelhetőségre vonatkozhatnak.

Alap állapotban csak a „*Betöltés*” gomb aktív (4. ábra). Kiválasztva be tudjuk tölteni a konvertálni kívánt XLS fájlt. A betöltés után lesz aktív a következő négy gomb: *XML*, *ACCESS*, *MySQL*, *Több tábla* (5. ábra). Ezen gombok segítenek magát a konvertálást elvégezni.

Az „*XML*” gomb (6. ábra) segítségével egy teljesen átlagos, úgymond alap XML fájlt tudunk létrehozni. Az Excel fájlok „header” elemei adják az elemeket, az oszlop többi tagjai pedig magát az információt, amivel feltöltjük az XML-t. Az általános XML fájl manuálisan tovább fejleszthető, szabadon formázható.



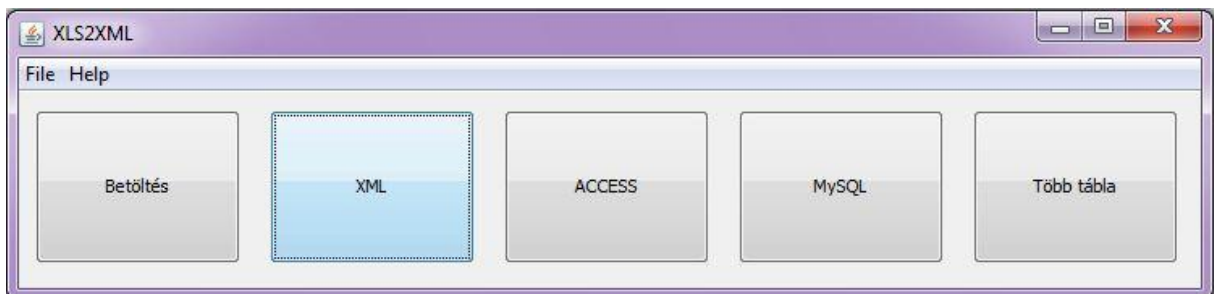
4. ábra: Alaphelyzetben csak a Betöltés gomb aktív

Forrás: Saját készítés, 2012



5. ábra: Excel fájl betöltése után, aktívá válnak a konvertáló gombok

Forrás: Saját készítés, 2012

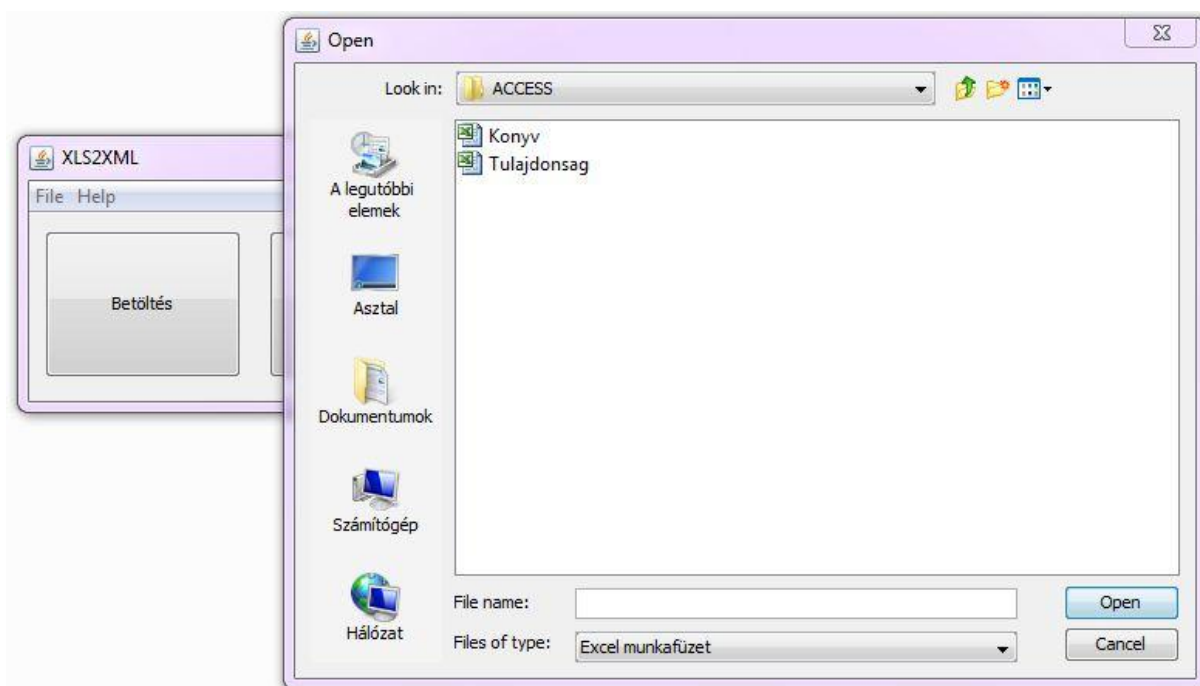


6. ábra: A kívánt platformra történő konvertálás esetén, a gombok színe kék lesz a kurzor rámutatásakor

Forrás: Saját készítés, 2012

Az „ACCESS” gomb lenyomásával egy Access kompatibilis XML fájlt tudunk előállítani. Ez azt jelenti, hogy az előállított XML fájl struktúráját tekintve megegyezik az Access által preferált XML strukturális szabályaival. Jelen esetben nem csak egy adott Excel fájlra van szükség, hanem kihasználva az Access adta lehetőségeket, séma fájlokat is létrehozhatunk, így egy olyan XML, XSD fájl párost kapunk a konvertálás után, ami már az általunk meghatározott adatrögzítési, adatmegjelenítési szabályokat is tartalmazza. Jelen esetben az XSD séma fájlról beszélünk, melynek előállításához szükség van egy második

Excel fájlra, amiben az eredeti Excel fájl „header” elemeire vonatkozóan kell megadnunk a szabályokat. Lehetőségünk van meghatározni előfordulásra, adattípusra illetve a tárolt információ hosszúságra vonatkozó adatokat, az XSD nyelvkészletével megegyezően. Az XSD előállításához szükséges Excel fájlt kétféleképpen adhatjuk meg. Első lehetőségként tárolhatjuk ezt a fájlt a konvertálásra váró Excel fájl mellett, azonban módosított elnevezéssel. Például ha adott egy konvertálni kívánt Excel fájl neve (*konyv.xls*), akkor az XSD előállításához szükséges Excel fájl nevének *kony_tulajdonsag.xls*-nek kell lennie. Ekkor a program automatikusan betölti mindkét Excel fájlt és előállítja az XML illetve XSD fájlokat. A második lehetőség, ha nem található meg a konvertálásra várt Excel fájl mellett a tulajdonságot leíró fájl, akkor automatikusan újra megjelenik a bekérés ablak az XSD fájlhoz szükséges Excel fájl megnyitásához.

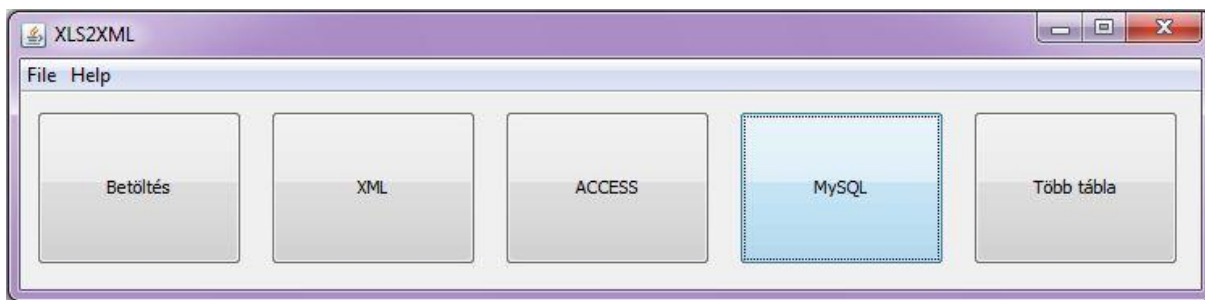


7. ábra: XML illetve XSD konvertálásához szükséges XLS fájlok bekérése

Forrás: Saját készítés, 2012

A fent megjelenített képen (7. ábra) jól látható, hogy a program csakis az XML fájlokat jeleníti meg. Mivel jelen esetben a *Konyv.xls* nevű fájlhoz nem társul *Konyv_tulajdonsag.xls* ezért az XSD fájlhoz szükséges konvertáláshoz újra meg fog jelenni ez a bekérő ablak a kívánt tulajdonság leírásokat tartalmazó, jelen esetben *Tulajdonsag.xls* nevű Excel fájl megnyitásához.

A harmadik gomb a „MySQL” (8. ábra). Ennek segítségével a MySQL számára tudunk előállítani XML fájlt, sajátos strukturális szabályainak megfelelően.



8. ábra: MySQL számára konvertálható XML fájl

Forrás: Saját készítés, 2012

Mivel az XML állomány rendkívül rugalmas, különböző adatbázis rendszerek számára konvertálható, bármilyen XML előállítható. Ennek okából tartottam szükségesnek, hogy a legfontosabb, adatbázis kezelő programok számára előállítsam az XML fájlokat. Az átkonvertált XML fájl betölthető az adatbáziskezelő programba, ahol megoldható számos szűrés, keresés, de legfőképpen az összetett kereséseket kell hangsúlyozni, ugyanis ez nem volt elérhető egy XLS fájlban.²⁷

Mint minden program készítése során, jelen esetben is időközben merültek fel javítási, bővítési lehetőségek. A program fejlesztése során felvetődött a kérdés, hogy ha a program alkalmas egy adott Excel fájlból készíteni egy adott XML fájlt, akkor előre megadott fájl struktúra alapján miért ne lehetne képes több XML fájlt létrehozni csupán egy Excel fájlból? Ezáltal az egyes XML fájlok külön beimportálhatóak egy adott adatbáziskezelő rendszerbe és az előre megadott Excel fájl struktúrájának köszönhetően kialakíthatjuk, milyen mezők tartoznak az egyes XML fájlokba. Ezáltal létrehozhatjuk a kapcsolódási pontokat (az úgynevezett elsődleges kulcsokat), melyek segítségével – a már beimportált XML fájlból létrejött – az adattáblák összekapcsolódhatnak.

A fent említett fájl struktúrára azért van szükség, hogy szintaktikailag megfelelő XML fájlokat állítson elő a program. Az általam kialakított Excel fájlt a következő képen (8. ábra) prezentálom:

²⁷ A program forráskódjának egy részlete megtalálható az 6. mellékletben

	A	B	C	D	E	F	G	H	I	J	K	L
1	Könyv	Rakj	Cím	Kiad	Kiadév	ISBN	Peld	Szerzo	Rakj	Nev	Szulido	Szulhely
2	Tablanev	Oszlop	Oszlop	Oszlop	Oszlop	Oszlop	Oszlop	Tablanev	Oszlop	Oszlop	Oszlop	Oszlop
3			1 Az arany ember	Budapest Kiadó	2008	9789639503397	2		1 Jókai Mór		1825	Komárom
4			2 Micimackó	Holló Kiadó	2010	9789631187342	3		2 Alan Alexander Milne		1882	London
5			3 Inni és élni hagyni	Alma Kiadó	2009	9789632542775	1		3 Charlaine Harris		1951	New York
6			4 Trónok harca	Figure Kiadó	2008	9789633707784	3		4 George R. R. Martin		1948	New Jersey

9. ábra: Több XML fájl előállítás egy darab Excel fájlból

Forrás: Saját készítés, 2012

Jól látható, hogy az adatmezőknek megszabott struktúrában kell elfoglalniuk helyüket (9. ábra). Az Excel táblázat első sora adja meg az XML fájl nevét illetve az XML fájlban előállítandó elemek nevét. A második sorban található az az információ, ami megadja, hogy az adott elem a táblára utal vagy a táblában található oszlopokra. A *Tablanev* típusú elem lesz az XML fájl neve tehát ez a név fog megjelenni – ha az XML fájlt betöltjük egy adatbáziskezelő programba (például Access) – mint táblanév. Az *Oszlop* elnevezésű elemek lesznek tehát az adattábla oszlopai és az alattuk elhelyezkedő elemek lesznek az oszlopokban található információk, azaz az egyes rekordok. Amennyiben tehát több XML fájlt akarunk előállítani, a több táblás adatbázis érdekében, akkor a következő gombbal tudjuk azt elvégezni (10. ábra):



10. ábra: Több XML fájl előállításához a Több tábla gombot kell alkalmazni

Forrás: Saját készítés, 2012

A *Több tábla* opció segítségével előállított XML fájlok megnyithatóak az MS Access program segítségével.

7. Meglévő konvertáló programok hibája?!

Az általam készített program képes részben kötött struktúrájú XLS fájlból XML, illetve igény esetén hozzátartozó XSD sémát konvertálni. Ennek segítségével az adatok betölthetők egy adott adatbáziskezelő rendszerbe. Azonban felvetődhet egy alapvető kérdés minden informatikában jártas személyben. Miben is nyújt ez újat? Miért nem jók a Microsoft termékek által nyújtott XML konvertálási lehetőségek? Szeretnék helyet adni dolgozatomban egy-két gyakran feltehető kérdés (GYIK) megválaszolásának.

A programom elkészítését megelőzően utána néztem a meglévő konvertáló programoknak. Elsőként is a Microsoft Office program csomag XML fájlok konvertálására, importálására, exportálására vonatkozó opciókat vizsgáltam meg. Ugyan lehetőség van XML fájlba elmenteni egy Excel fájlt a Microsoft Excel táblázatkezelő program segítségével, azonban ez az opció rendkívül nagy mennyiségű adatot generál magában, az Excel által előállított XML fájlban. Ilyen adatok például a lapon történő elhelyezkedése, nyomtatásra vonatkozó adatok és különböző, pozícióval kapcsolatos adatok. Egy rövid kódrészletet mellékelek, hogy prezentáljam miről is van itt szó. A teljes Excel által előállított XML fájl forráskódja a mellékletek között megtalálható.²⁸

```
<WorksheetOptions xmlns="urn:schemas-microsoft-com:office:excel">
  <PageSetup>
    <Header x:Margin="0.3"/>
    <Footer x:Margin="0.3"/>
    <PageMargins x:Bottom="0.75" x:Left="0.7" x:Right="0.7" x:Top="0.75"/>
  </PageSetup>
  <Print>
    <ValidPrinterInfo/>
    <PaperSizeIndex>9</PaperSizeIndex>
    <HorizontalResolution>200</HorizontalResolution>
    <VerticalResolution>200</VerticalResolution>
  </Print>
```

²⁸ 5. számú melléklet

```
<ProtectObjects>False</ProtectObjects>  
<ProtectScenarios>False</ProtectScenarios>  
</WorksheetOptions>  
</Worksheet>  
</Workbook>
```

Jól láthatóan ez a kódrészlet teljes egészében olyan utasításokat tartalmaz, melyek feleslegesek egy adatbáziskezelő program számára. Az Excel által előállított XML fájlt természetesen beimportáltam a Microsoft adatbáziskezelő programjába, az Access-be. A kapott ijesztő végeredmény ébresztett rá, hogy a könyvtárak számára ténylegesen felesleges, több mint, nyolcvan százalékban használhatatlan információkat tartalmaznak ezek az XML fájlok. Az Access programba való betöltés után létrejött több mint tíz tábla, melyek például a következő adatokat tartalmazzák: adatcellák elhelyezkedése az Excel fájlban belül; munkalapok információ telítettsége, az az egy munkafüzet (Sheet) mennyi információt hordoz továbbá nyomtatási adatok.

A következő lépés volt, hogy a már meglévő XML konvertáló programokat megvizsgáltam. A Google kereső által, az „XML konvertáló” illetve „XML fájlok konvertálása” kulcsszavak segítségével visszakeresett találatok között nem találtam az enyémhez hasonló, ingyenes, hatékony programot. Részben hasonló programok ugyan léteznek, azonban vagy nem konkrét adatbáziskezelő programok által várt XML fájlokat állítanak elő, vagy csak hogy nem megfizethetetlen áruk miatt nem éri meg használni azokat. Ingyenesen elérhető programot nem találtam. A kereső vagy egy áoldalra továbbított, illetve ténylegesen nem egy ingyenes verziót kaptam, hanem egy demo verziót, ami alkalmatlan volt a konvertálás elvégzésére. A kutatás végeredménye további pozitív megerősödést adott a programom elkészítésében, hisz több User is hasonló program után érdeklődött^{29,30}. Tehát nem csak a könyvtárak igénye, hanem másféle érdeklődés is jelentkezett már a témával kapcsolatban.

²⁹ <http://pcforum.hu/tudastar/45887/Xml+fajl+konvertalasa.html> [Letöltve: 2012. április]

³⁰ <http://tudastar.netacademia.net/default.aspx?upid=20001> [Letöltve: 2012. április]

8. Összefoglalás

Mind a könyvtári szellemi eszközök (mint például a szabványok), mind az informatika rohamos fejlődése nélkülözhetetlené tette a könyvtárak, mint intézmények fejlődését. A mai társadalomnak – melynek hatalmas szüksége van az információra – kötelező feladatának, célkitűzésének kellene tekintenie a könyvtárak segítését, fejlesztésének irányítását, ösztönzését. Nem szabadna engedni, hogy bármiféle akadály felmerüljön ezen folyamat végbemenetelében. Legyen az pénzübeli vagy regionális probléma.

Ezt a gondolatmenetet követően határoztam el, hogy, ha akár csekély mértékben is, de segítem a könyvtárak sikeres működését, fejlesztési folyamatát. A programom elkészítése után jól láthatóan, a Magyar Földrajzi Társaság könyvtári gyűjteményének feldolgozásán keresztül bebizonyosodott, hogy munkámra szükség van. Így, a keresésre lényegében alkalmatlan fájl formátumban megtalálható könyvtári katalógust sikeresen lehet átalakítani olyan fájl formátummá, amely egy adatbáziskezelő szoftver inputját képezheti. Ebben pedig már megoldható mindenféle keresés, legyen az egyszerű vagy összetett. A program egyszerű kezelhetőséggel bír, informatikai előtudás nélkül is bárki könnyedén alkalmazhatja. A következő két ábrán keresztül szeretném bemutatni az MFT könyvtár Excel fájlban tárolt könyvtári gyűjteményét (11. ábra) illetve az általam készített program segítségével átkonvertált XML fájlt (12. ábra), ezzel is prezentálva, hogy a kitűzött célt sikerült megvalósítani.

Proba [kompatibilis üzemmód] - Microsoft Excel										
Kezdőlap Beszúrás Lap elrendezése Képletok Adatok Korrekció Nézet										
Beillesztés Vágólap Arial 11 A B I U Sortoreléssel több sorba Általános 0% 000 0,00 Féltételes formázás Formázás Cellatípusok Beszúrás Törés Formátum Rendezés Keresés és szűrés Szerkesztés										
A66	A1136									
A	B	C	D	E	F	G	H	I	J	
Raktári számkjelzet	Szerző(k)	Közreműködő(k)	Főcím	Párhuzamos cím	Alcím	Kiadás jelzés	Kiadás éve	1851 előtti (igen/nem)*	Kiadás helye	
A1073			A handbook for travellers in Syria and Palestine		Including an account of the geography, history, antiquities, and inhabitants of these countries, the peninsula of Sinai, Edom, and the Syrian desert, with detailed descriptions of Jerusalem, Petra, Damascus, and Palmyra part II.	New and revised ed.	1868	nem	London ; Paris ; Malta ; Naples ; Trieste	Gal
A1074			A handbook for travellers in India and Ceylon		Including the provinces of Bengal, Bombay, and Madras, the Panjab, North-West Provinces, Rajputana, Central Provinces, Mysore, etc.) the Native States and Assam		1891	nem	London ; Calcutta	John
A1076			Hand-book to Calcutta		With a plan of the city		1892	nem	Calcutta	W.
A1078			Utah		Its people, resources, attractions, and institutions		[post 1909]	nem	Salt Lake City	Kia
A1077	Kelen Lajos és K. Fodor János		Kolozsvári kalauz				1902	nem		Uk
A1078	Tibor Baráth		Kolozsvár-Klausenburg das herz Siebenbürgens				1944	nem	Kolozsvár	
A1079	Kapitány Kálmán		Káriszbad				1908	nem	Budapest	
A1080	Kapitány Kálmán		Bécs				1908	nem	Budapest	
A1081	Kapitány Kálmán		Mérán és Arko				1908	nem	Budapest	
A1082	Ince Gábor		Új 'kís' tükör		Tíz magyarországi kép		1935	nem	Debrecen	
A1083	Ira Zubriczky Aladár		Vasárnapok a Tátrában				1911	nem	Budapest	Ist
A1084	Hungler József	szek. Somogyi Marianna	Veszprémi salák				1904	nem	Veszprém	V.
A1085	Wéber Pál		Feljegyzések		Róma, Amerika, Canada		1928	nem	Veszprém	
A1086										
A1087		Par les rédacteurs de Fortune	Le Japon				[1944]	nem	New York	C
A1088	Priestley, John Boynton	ford. Bér Erzsébet	Oroszországi utazás				[1947]	nem	Budapest	

11. ábra: Az MFT könyvtár Excel fájlban tárolt könyvtári gyűjteménye

Forrás: Saját készítés, 2012

```

<?xml version="1.0" encoding="UTF-8"?>
<könyvek>
<könyv>
<Raktáriszámjelzet>A1073</Raktáriszámjelzet>
<Szerzők/>
<Közreműködők/>
<Főcím>A handbook for travellers in Syria and Palestine</Főcím>
<Párhuzamoscím/>
<Alcím>Including an account of the geography, history, antiquities, and inhabitants of these countries, the peninsula of Sinai, Edom, and the Syrian desert; with detailed descriptions of Jerusalem, Petra, Damascus, and Palmyra part II.</Alcím>
<Kiadásjelzés>New and revised ed.</Kiadásjelzés>
<Kiadás éve>1868</Kiadás éve>
<1851 előtti igen/nem>nem</1851 előtti igen/nem>
<Kiadás helye>London ; Paris ; Malta ; Naples ; Trieste</Kiadás helye>
<Kiadó>John Murray, Galignani, Xavier, Muir, Dorant, Münster</Kiadó>
<NyomdaplHelnyomdászneve>vesszám/>
<Terjedelemoldalszámptillméretcm>275-619. p. : térk. , 19 cm + 1 térk. (39x36 cm)</Terjedelemoldalszámptillméretcm>
<Sorozatcím>ISSN sorozatcím/>
<Terjesztési adatok>ISBN kötés</Terjesztési adatok>ISBN kötés>
<Hiányzó tételek/>
<MFT példányszám/>
<Megjegyzés>választás>London : John Murray ; Paris : Galignani, Xavier ; Malta : Muir, Naples Dorant ; Trieste : Münster. - A térkép címe: Environs of Jerusalem, 1:200 000 </Megjegyzés>választás>
<Egységesített név/>
<Adatbázis/>
</könyv>
<könyv>
<Raktáriszámjelzet>A1074</Raktáriszámjelzet>
<Szerzők/>
<Közreműködők/>
<Főcím>A handbook for travellers in India and Ceylon</Főcím>
<Párhuzamoscím/>
<Alcím>Including the provinces of Bengal, Bombay, and Madras, (the Panjab, North-West Provinces, Rajputana, Central Provinces, Mysore, etc.) the Native States and Assam</Alcím>
<Kiadásjelzés>
<Kiadás éve>1891</Kiadás éve>
<1851 előtti igen/nem>nem</1851 előtti igen/nem>
<Kiadás helye>London ; Calcutta</Kiadás helye>
<Kiadó>John Murray, Thacker, Spink, & Co.</Kiadó>
<NyomdaplHelnyomdászneve>vesszám/>
<Terjedelemoldalszámptillméretcm>11x, 440 p. : ill. , térk. ; 19 cm + 1 térk. (21x34)</Terjedelemoldalszámptillméretcm>
<Sorozatcím>ISSN sorozatcím/>
<Terjesztési adatok>ISBN kötés</Terjesztési adatok>ISBN kötés>
<Hiányzó tételek/>
<MFT példányszám/>
<Megjegyzés>választás>A térképmell. címe: Island of Bombay, [Méretarány nélkül]</Megjegyzés>választás>
<Egységesített név/>
<Adatbázis/>
</könyv>
<könyv>
<Raktáriszámjelzet>A1075</Raktáriszámjelzet>
<Szerzők/>
<Közreműködők/>

```

12. ábra: Az MFT könyvtárának Excel fájlban tárolt könyvtári gyűjteményéből előállított XML fájl

Forrás: Saját készítés, 2012

Kikerülve a hátráltató tényezőket – mint a pénz hiánya vagy egy bonyolult munkafolyamat – egy jól átlátható megoldást sikerült létrehozni. Remélhetőleg számos könyvtár munkafolyamatát segíti majd a program.

Itt ragadnám meg az alkalmat, hogy a programot felajánljam azon könyvtárak részére, akik hasonló problémával küszködnek, mint a Magyar Földrajzi Társaság könyvtára.

Remélem sikerült hozzájárulnom a fejlesztési folyamathoz a könyvtár-informatika területen.

Ábrajegyzék

1. ábra: Az MFT emblémája.....	12
2. ábra: XML fájlok importálása az Access adatbáziskezelő rendszerben.....	20
3. ábra: Előre meghatározott struktúrával felépített Excel fájl	24
4. ábra: Alaphelyzetben csak a Betöltés gomb aktív	26
5. ábra: Excel fájl betöltése után, aktívvá válnak a konvertáló gombok	26
6. ábra: A kívánt platformra történő konvertálás esetén, a gombok színe kék lesz a kurzor rámutatásakor	26
7. ábra: XML illetve XSD konvertálásához szükséges XLS fájlok bekérése	27
8. ábra: MySQL számára konvertálható XML fájl	28
9. ábra: Több XML fájl előállítás egy darab Excel fájlból	29
10. ábra: Több XML fájl előállításához a Több tábla gombot kell alkalmazni	29
11. ábra: Az MFT könyvtár Excel fájlban tárolt könyvtári gyűjteménye.....	33
12. ábra: Az MFT könyvtárának Excel fájlban tárolt könyvtári gyűjteményéből előállított XML fájl.....	33

Irodalomjegyzék

1. Könyvtárosok kézikönyve 1. / Szerk. Dr. Horváth Tibor, Papp István. – Budapest, 2003
2. Könyvtárosok kézikönyve 2. / Szerk. Dr. Horváth Tibor, Papp István. – Budapest, 2003

Internetes gyűjtés

1. <http://www.bibl.u-szeged.hu/inf/szakdoli/2003/pappevi/fej1.htm#intkvt> [Letöltve: 2012, április]
2. <http://www.bibl.u-szeged.hu/inf/szakdoli/2003/pappevi/fej2.htm#intkvt> [Letöltve: 2012, április]
3. XMLMARC: <http://xmlmarc.stanford.edu/> [Letöltve: 2012, április]
4. Mader Béla: Információ és intézményei. Hol tartunk, mi lesz velünk...
<http://www.mek.iif.hu/porta/szint/muszaki/szamtech/wan/netwshop/netwsh95/madern95.hun> [Letöltve: 2012, április]
5. Tolnai György: A hazai könyvtárgépesítés a megváltozott világban. TMT, 1992/7-8. 303-309. p. http://tmt.omikk.bme.hu/show_news.html?id=3223&issue_id=413 [Letöltve: 2012, április]
6. Bakonyi Géza: Könyvtár és/vagy elektronikus könyvtár.
<http://www.mek.iif.hu/porta/szint/tarsad/konyvtar/ekonyvt/bege.hun> [Letöltve: 2012, április]
7. Bakonyi Géza: Számítógép a könyvtárban, könyvtár a számítógépben.
<http://www.mek.iif.hu/porta/szint/tarsad/konyvtar/automat/bakonyi2.hun> [Letöltve: 2012, április]
8. Beély Gábor: A könyvtár jövője, a jövő könyvtára.
<http://www.inco.hu/inco3/infoert/cikk1h.htm> [Letöltve: 2012, április]
9. Salgáné Medveczki Marianna: Az XML: új perspektívák a könyvtár-informatikában
http://tmt.omikk.bme.hu/show_news.html?id=3541&issue_id=448 [Letöltve: 2012, április]
10. Bakonyi Géza - Kokas Károly: Könyvtári integrált rendszerek és hazai alkalmazásuk
<http://mek.niif.hu/03000/03088/html/funkcioi.htm> [Letöltve: 2012, április]

11. Andreas Schrader és Beate Sieweke: Hibrary – The hybrid library of the future:
<http://ki.oszk.hu/kf/2010/10/schrader-andreas-%E2%80%93-sieweke-beate-hybrary-%E2%80%93-a-jovo-hibrid-konyvtara/> [Letöltve: 2012, április]
12. A Magyar Földrajzi Társaság hivatalos honlapja:
<http://www.foldrajzitasasag.hu/rolunk> [Letöltve: 2012, április]
13. A Java hivatalos honlapja: <http://www.java.com/> [Letöltve: 2012, április]
14. A Java programozási nyelv Wikipédias szócikke:
http://hu.wikipedia.org/wiki/Java_%28programoz%C3%A1si_nyelv%29 [Letöltve: 2012, április]
15. A Sun Microsystem felvásárlása:
http://itcafe.hu/hir/az_oracle_veszi_meg_a_sunt.html [Letöltve: 2012, április]
16. Az XML hivatalos honlapja: http://www.w3schools.com/xml/xml_what.asp
[Letöltve: 2012, április]
17. Az XSD Schema hivatalos honlapja: <http://www.w3schools.com/Schema/> [Letöltve: 2012, április]
18. A Microsoft által leírt Access és az XML kapcsolatának honlapja:
<http://office.microsoft.com/en-us/access-help/using-xml-and-access-HA001034560.aspx> [Letöltve: 2012, április]
19. A MySQL és az XML kapcsolatának leírása:
<http://www.databasejournal.com/features/mysql/article.php/3846526/Working-with-XML-Data-in-MySQL.htm> [Letöltve: 2012, április]
20. Internetes keresés meglévő konvertáló programok kutatásának céljából:
<http://pcforum.hu/tudastar/45887/Xml+fajl+konvertalasa.html> [Letöltve: 2012. április]
<http://tudastar.netacademia.net/default.aspx?upid=20001> [Letöltve: 2012. április]

Mellékletek

- I. számú melléklet: Felhasználói kézikönyv
- II. számú melléklet: A program által előállított általános XML fájl forrása
- III. számú melléklet: A program által, Access számára előállított XML illetve XSD séma fájlok forrása
- IV. számú melléklet: A program által, MySQL számára előállított XML fájl forrása
- V. számú melléklet: MS Excel táblázatkezelő programmal előállított XML fájl
- VI. számú melléklet: A program forráskódjának részlete

I. számú melléklet: Felhasználói kézikönyv

1. Kedves Felhasználó!

Az XLS2XML konvertáló program elsősorban könyvtári gépesítési folyamatok elősegítésére jött létre. Elsődleges cél olyan könyvtárak munkájának megkönnyítése, ahol az állomány nyilvántartást Excel segítségével, táblázatban tárolják. Új, adatbáziskezelő rendszerek számára is elfogadható dokumentumot írni manuálisan – a meglévő Excel alapján – nem csak hosszadalmas, de monoton munka is egyben. Ám a konvertáló program segítségével ez könnyen, másodpercek alatt megoldható, nagyszámú rekordok esetén is. Természetesen olyan személyek számára is elérhető a program, kiknek valamilyen oknál fogva – talán szintén adatbázis építés céljából – szükségük van Excel fájlok átalakítására, adatbáziskezelő rendszerek által olvasható XML fájlokra.

A program tehát Excel fájlokból állít elő XML fájlokat, adatvesztés és korlátozások nélkül. A következőkben szeretném bevezetni Önt a program rejtelmeibe. Cél, hogy minél egyszerűbben, minél hamarabb zökkenőmentesen tudja kezelni a programot.

2. Elvárt Excel fájlok

A program előre meghatározott felépítésű Excel fájlt kér bemenetként. Azaz a konvertálni kívánt Excel fájlt a program számára megfelelő módon kell kitölteni. Elsősorban ez nem szerkezeti felépítést tekintve, hanem inkább karakterhasználati szempontból fontos. A legfontosabb ebből a szempontból, hogy az Excel fájl mezőibe nem kerülhetnek központosági karaktereket (például ,(){}[]-).

Az Excel fájl első sora, az oszlopok neve (továbbiakban header elemek) adják meg az XML elem neveket. Az alattuk feltöltött mezők pedig a tartalmat, ami az egyes elemekbe feltöltésre kerül.

	A	B	C	D	E	F	G
1	Rakj	Szerz	Cím	Kiad	Kiadev	ISBN	Peld
2	1	Jókai Mór	Az arany ember	Budapest Kiadó	2008	9789639503397	2
3	2	Alan Alexander Milne	Micimackó	Holló Kiadó	2010	9789631187342	3
4	3	Charlaine Harris	Inni és élni hagyni	Alma Kiadó	2009	9789632542775	1
5	4	George R. R. Martin	Trónok harca	Figure Kiadó	2008	9789633707784	3

3. Kezelhetőség

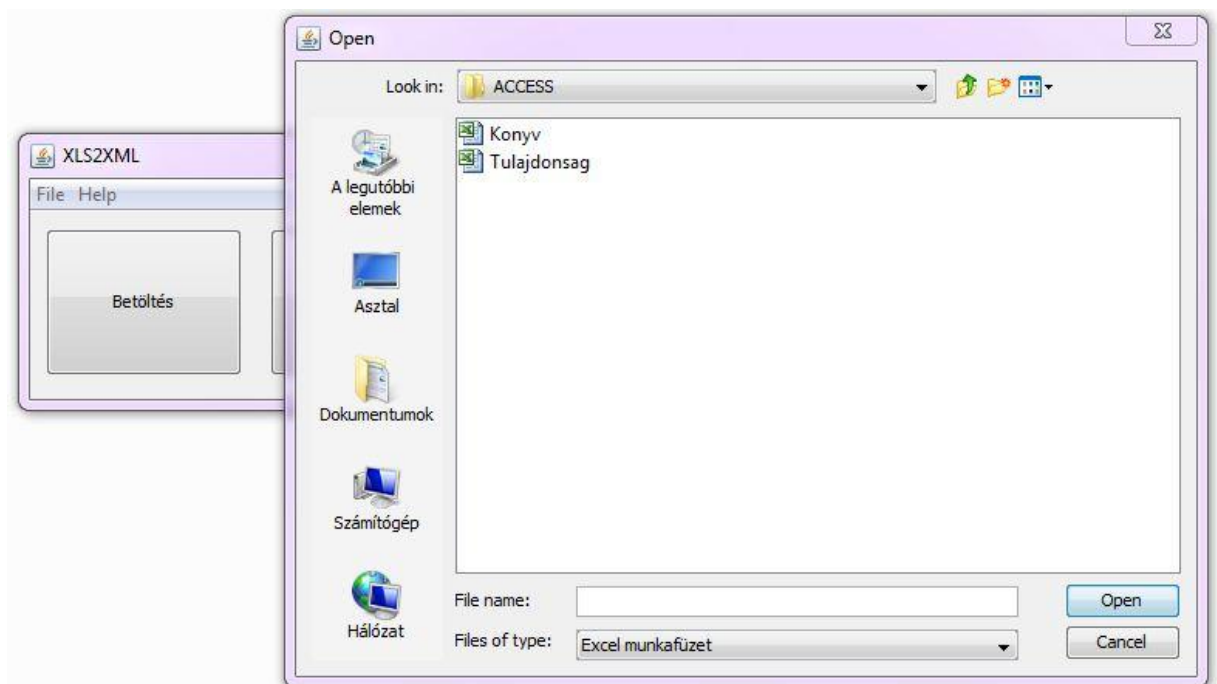
A program mindenki számára könnyű kezelhetőséget biztosít. A *konvertalo.jar* fájl segítségével futtatható az alkalmazás. Elindítása után a következő ablaknak kell megjelennie:



A fent látható gombok segítségével végezhető el a konvertálás egy adott Excel fájlra. A megfelelő gomb lenyomásával, a használni kívánt adatbáziskezelő rendszerek számára állítható elő XML fájl. Például ha egy, az Access program által előállított adattáblára van szükség akkor az *ACCESS* gomb segítségével hozható létre, az MS Access által olvasható XML fájl.

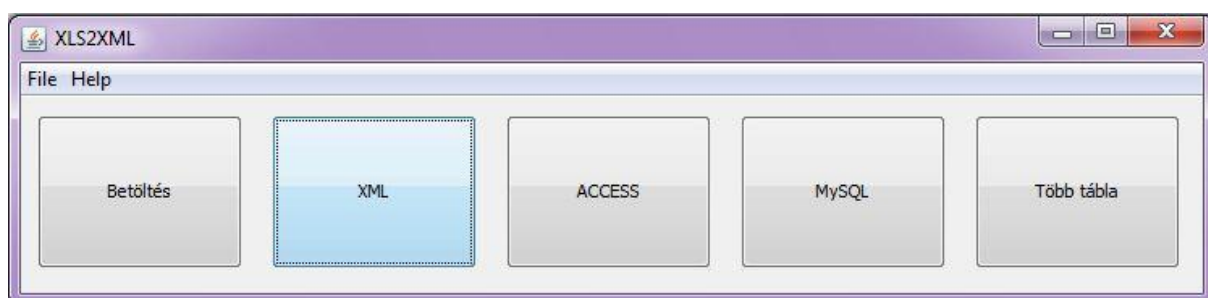
a. Betöltés gomb

A *Betöltés* gomb használatával kérhető be konvertálásra az Excel fájl. A gomb kiválasztásakor a következő ablaknak kell megjelennie:



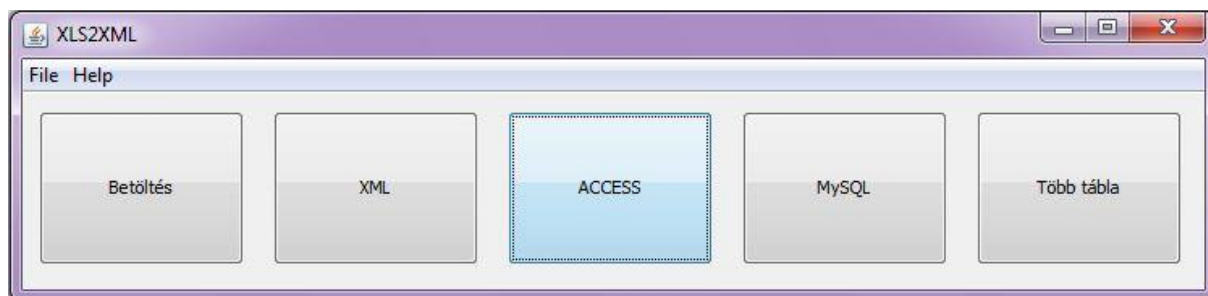
Az képen látható *Open* gomb segítségével véglegesíthető a bekérési művelet. A program úgy lett kialakítva, hogy a bekért fájl csak is Excel munkafüzet lehet, elkerülve a bekérésből adódó hibákat. A bekérést követően ismét az alapképernyő jelenik meg, ahol a további konvertálásokat lehet végre hajtani. Ezekről bővebben a *b,c,d,e pontban* lehet tájékozódni.

b. XML gomb



Az *XML* gomb segítségével egy általános XML fájl hozható létre. Az *XML* Gombra kattintva megjelenik az Excel fájlt bekérő ablak. A tallózás gomb segítségével megadhatjuk azt az Excel fájlt, amiből konvertálás útján XML fájl érhető el. Az XML fájl neve és elérési útja megegyezik az Excel fájl nevével és elérési útjával. Tehát, ha egy *Konyvtar.xls* a *D:\Dokumentumok\Katalogus\Konyvtar.xls* elérési úton érhető el akkor a létrehozott *Konyvtar.xml* fájl szintén a *D:\Dokumentumok\Katalogus\Konyvtar.xml* elérési úton lesz elérhető.

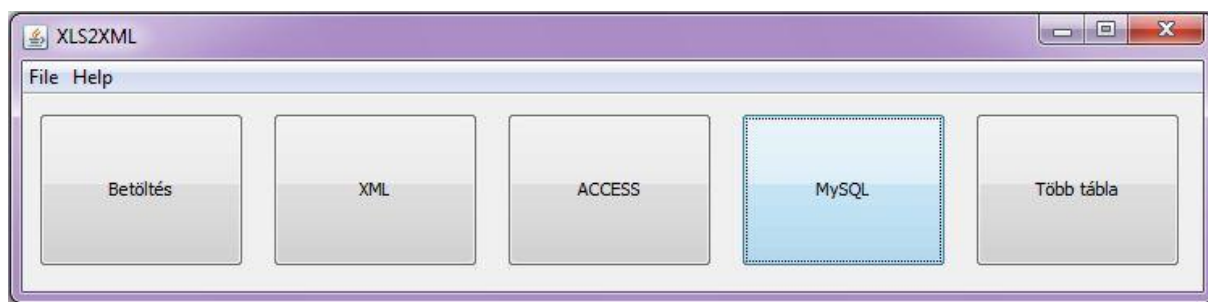
c. ACCESS gomb



Az *ACCESS* gomb segítségével a Microsoft Access adatbáziskezelő program számára olvasható illetve az általa elvárt felépítésű XML fájl hozható létre. Az *ACCESS* Gombra kattintva megjelenik az Excel fájlt bekérő ablak. A tallózás gomb

segítségével megadhatjuk azt az Excel fájlt amiből konvertálás útján XML fájl érhető el. Az XML fájl neve és elérési útja megegyezik az Excel fájl nevével és elérési útjával. Tehát, ha egy *Konyvtar.xls* a *D:\Dokumentumok\Katalogus\Konyvtar.xls* elérési úton érhető el akkor a létrehozott *Konyvtar.xml* fájl szintén a *D:\Dokumentumok\Katalogus\Konyvtar.xml* elérési úton lesz elérhető. Az Access számára létrehozott XML fájlokhoz lehetőség van stíluslapot, azaz XSD fájlt is létrehozni. Az XSD fájl segítségével adatfeltöltési megszorításokat adhatunk meg az egyes elemekre vonatkozóan. Az XSD fájlhoz szükséges Excel fájlt, automatikusan betölti a program az XML fájlhoz szükséges Excel fájl utána, ha nevük a következő módon van megadva: *Konyv.xls* – *Konyv_tulajdonsag.xls*. Ekkor a *Konyv.xls* fájlból jön létre az XML fájl, a *Konyv_tulajdonsag.xls* fájlból pedig az XSD fájl. Azonban ha két Excel fájl neve nem ilyen formában van megadva, akkor a program újra megjeleníti a fájl bekérő ablakot, ahol lehetőség van megadni az XSD fájl előállításához szükséges Excel fájlt. A létrehozott XML fájl Accessbe történő betöltése után, az adatbáziskezelő program automatikusan felismeri a hozzátartozó stíluslapot is.

d. MySQL gomb



A MySQL gomb segítségével a MySQL adatbáziskezelő rendszer számára olvasható illetve az általa elvárt felépítésű XML fájlt hozható létre. Az MySQL Gombra kattintva megjelenik az Excel fájlt bekérő ablak. A tállózás gomb segítségével megadhatjuk azt az Excel fájlt, amiből konvertálás útján XML fájl érhető el. Az XML fájl neve és elérési útja megegyezik az Excel fájl nevével és elérési útjával. Tehát, ha egy *Konyvtar.xls* a *D:\Dokumentumok\Katalogus\Konyvtar.xls*

elérési úton érhető el akkor a létrehozott *Konyvtar.xml* fájl szintén a *D:\Dokumentumok\Katalogus\Konyvtar.xml* elérési úton lesz elérhető.

e. Több Tábla gomb



A *Több Tábla* gomb segítségével egy adott Excel fájlból több XML fájl állítható elő. A több XML fájl segítségével, egy adatbáziskezelő rendszerbe történő importálásuk után több adattábla jön létre, melyek egy megadott közös elemen keresztül egymáshoz kapcsolódnak. A több XML fájlhoz szükséges Excel fájl is meghatározott struktúrát kíván. Az első sor elemei szintén XML elemek neveit adják meg, azonban itt adható meg a tábla név is. Az Excel fájl második sorában megadható, hogy az első sorban megnevezett elem az tábla név vagy egy XML elem neve. A harmadik sortól pedig szintén az elemekbe történő betöltendő adatok adhatóak meg.

	A	B	C	D	E	F	G	H	I	J	K	L
1	Könyv	Rakj	Cím	Kiad	Kiadév	ISBN	Peld	Szerzo	Rakj	Név	Szulido	Szulhely
2	Tablanev	Oszlop	Oszlop	Oszlop	Oszlop	Oszlop	Oszlop	Tablanev	Oszlop	Oszlop	Oszlop	Oszlop
3			1 Az arany ember	Budapest Kiadó	2008	9789639503397		2		1 Jókai Mór	1825	Komárom
4			2 Micimackó	Holló Kiadó	2010	9789631187342		3		2 Alan Alexander Milne	1882	London
5			3 Inni és élni hagyni	Alma Kiadó	2009	9789632542775		1		3 Charlaïne Harris	1951	New York
6			4 Trónok harca	Figure Kiadó	2008	9789633707784		3		4 George R. R. Martin	1948	New Jersey

Az XML fájlok neve ebben az esetben a *Tablanev* megnevezésű Excel elemek neve lesz. Az elérési útjuk pedig szintén megegyezik az Excel fájl elérési útjával. Tehát, ha egy *Konyvtar.xls* a *D:\Dokumentumok\Katalogus\Konyvtar.xls* elérési úton érhető el akkor a létrehozott *Konyvtar.xml*, *Szerzo.xml*, *Kiado.xml*, *Konyv.xml* fájlok szintén a *D:\Dokumentumok\Katalogus* mappában lesznek elérhetőek.

f. Kilépés

A program bármikor adatvesztés nélkül bezárható a program ablakának jobb felsősarkában található *X* gomb segítségével.

II. számú melléklet: A program által előállított általános XML fájl forrása

Konyvek.xml:

```
<?xml version="1.0" encoding="windows-1250" standalone="yes"?>
<!DOCTYPE Konyvek [
<!ELEMENT Konyvek (Konyv)+>
<!ELEMENT Konyv ( Rakj, Szerz, Cím, Kiad, Kiadev, ISBN, Peld)>
<!ELEMENT Rakj (#PCDATA)>
<!ELEMENT Szerz (#PCDATA)>
<!ELEMENT Cim (#PCDATA)>
<!ELEMENT Kiad (#PCDATA)>
<!ELEMENT Kiadev (#PCDATA)>
<!ELEMENT ISBN (#PCDATA)>
<!ELEMENT Peld (#PCDATA)>
]>
<konyvek>
<konyv>
<Rakj>1</Rakj>
<Szerz>Jókai Mór</Szerz>
<Cim>Az arany ember</Cim>
<Kiad>Presskontakt</Kiad>
<Kiadev>2008</Kiadev>
<ISBN>9789639503397</ISBN>
<Peld>2</Peld>
</konyv>
...
</konyvek>
```

III. számú melléklet: A program által, Access számára előállított XML illetve XSD séma fájlok forrása

Konyvek.xml:

```
<?xml version="1.0" encoding="windows-1250"?> <dataroot xmlns:od= "urn:schemas-
microsoft-com:officedata"      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\Pelda\ \Konyvek_tulajdonsag.xsd">
<Konyv>
<Rakj>1</ Rakj >
<Szerz>Jókai Mór</ Szerz >
<Cim>Az arany ember</ Cim >
<Kiad>Presskontakt</ Kiad >
<Kiadev>2008</ Kiadev >
<ISBN>9789639503397</ISBN>
<Peld>2</Peld>
</Konyv>
...
</dataroot>
```

Konyvek_tulajdonsag.xsd:

```
<?xml version="1.0" encoding="windows-1250"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:od="urn:schemas-
microsoft-com:officedata">
<xsd:element name="dataroot">
<xsd:complexType>
<xsd:sequence>
<xsd:element ref="Konyv" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="generated" type="xsd:dateTime"/>
</xsd:complexType>
</xsd:element>
```

```

<xsd:element name="konyv">
  <xsd:annotation>
    <xsd:appinfo>
    </xsd:appinfo>
  </xsd:annotation>
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="Rakj" minOccurs="unbounded" od:jetType="text" od:sqlSType="Int"
type="xsd:Int">
    </xsd:element>
    <xsd:element name="Szerz" minOccurs="unbounded" od:jetType="text"
od:sqlSType="string" type="xsd:string">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:maxLength value="30"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="Cim" minOccurs="unbounded" od:jetType="text"
od:sqlSType="string" type="xsd:string">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:maxLength value="255"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element name="Kiad" minOccurs="unbounded" od:jetType="text"
od:sqlSType="string" type="xsd:string">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:maxLength value="255"/>
        </xsd:restriction>

```

```

</xsd:simpleType>
</xsd:element>
<xsd:element      name="Kiadev"      minOccurs="unbounded"      od:jetType="text"
od:sqlSType="gYear" type="xsd:gYear">
</xsd:element>
<xsd:element      name="ISBN"      minOccurs="unbounded"      od:jetType="text"
od:sqlSType="String" type="xsd:String">
<xsd:simpleType>
<xsd:restriction base="xsd:String">
<xsd:maxLength value="13"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:element>
<xsd:element name="Peld" minOccurs="unbounded" od:jetType="text" od:sqlSType="Int"
type="xsd:Int">
</xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:schema>

```

IV. számú melléklet: A program által, MySQL számára előállított XML fájl forrása

Konyvek.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<list>

<konyv>

<field name="konyv_id">1</field>

<field name="Rakj">1</field>

<field name="Szerz">Jókai Mór</field>

<field name="Cim">Az arany ember</field>

<field name="Kiad">Budapest Kiadó</field>

<field name="Kiadev">2008</field>

<field name="ISBN">9789639503397</field>

<field name="Peld">2</field>

</konyv>

...

</list>
```

V. számú melléklet: MS Excel táblázatkezelő programmal előállított XML fájl

```
<?xml version="1.0"?>

<?mso-application progid="Excel.Sheet"?>

<Workbook xmlns="urn:schemas-microsoft-com:office:spreadsheet"
  xmlns:o="urn:schemas-microsoft-com:office:office"
  xmlns:x="urn:schemas-microsoft-com:office:excel"
  xmlns:ss="urn:schemas-microsoft-com:office:spreadsheet"
  xmlns:html="http://www.w3.org/TR/REC-html40">

  <DocumentProperties xmlns="urn:schemas-microsoft-com:office:office">

    <Created>2006-10-17T13:40:18Z</Created>

    <LastSaved>2012-02-27T10:37:37Z</LastSaved>

    <Version>12.00</Version>

  </DocumentProperties>

  <OfficeDocumentSettings xmlns="urn:schemas-microsoft-com:office:office">

    <RemovePersonalInformation/>

  </OfficeDocumentSettings>

  <ExcelWorkbook xmlns="urn:schemas-microsoft-com:office:excel">

    <WindowHeight>8010</WindowHeight>

    <WindowWidth>15120</WindowWidth>

    <WindowTopX>120</WindowTopX>

    <WindowTopY>105</WindowTopY>

    <ProtectStructure>False</ProtectStructure>

    <ProtectWindows>False</ProtectWindows>

  </ExcelWorkbook>

  <Styles>
```

```

<Style ss:ID="Default" ss:Name="Normal">
  <Alignment ss:Vertical="Bottom"/>
  <Borders/>
  <Font ss:FontName="Calibri" x:CharSet="238" x:Family="Swiss" ss:Size="11"
    ss:Color="#000000"/>
  <Interior/>
  <NumberFormat/>
  <Protection/>
</Style>
<Style ss:ID="s16">
  <Alignment ss:Horizontal="Center" ss:Vertical="Center"/>
</Style>
<Style ss:ID="s17">
  <Alignment ss:Horizontal="Center" ss:Vertical="Center"/>
  <NumberFormat ss:Format="0"/>
</Style>
</Styles>
<Worksheet ss:Name="Munka1">
  <Table ss:ExpandedColumnCount="7" ss:ExpandedRowCount="5" x:FullColumns="1"
    x:FullRows="1" ss:DefaultRowHeight="15">
    <Column ss:Width="24.75"/>
    <Column ss:Width="107.25"/>
    <Column ss:Width="90.75"/>
    <Column ss:Width="78"/>
    <Column ss:AutoFitWidth="0" ss:Width="96"/>
    <Column ss:Width="74.25"/>
  <Row>

```

```

<Cell ss:StyleID="s16"><Data ss:Type="String">Rakj</Data></Cell>
<Cell ss:StyleID="s16"><Data ss:Type="String">Szerz</Data></Cell>
<Cell ss:StyleID="s16"><Data ss:Type="String">Cim</Data></Cell>
<Cell ss:StyleID="s16"><Data ss:Type="String">Kiad</Data></Cell>
<Cell ss:StyleID="s16"><Data ss:Type="String">Kiadev</Data></Cell>
<Cell ss:StyleID="s16"><Data ss:Type="String">ISBN</Data></Cell>
<Cell ss:StyleID="s16"><Data ss:Type="String">Peld</Data></Cell>
</Row>
<Row>
<Cell ss:StyleID="s16"><Data ss:Type="Number">1</Data></Cell>
<Cell ss:StyleID="s16"><Data ss:Type="String">Jókai Mór</Data></Cell>
<Cell ss:StyleID="s16"><Data ss:Type="String">Az arany ember</Data></Cell>
<Cell><Data ss:Type="String">Budapest Kiadó</Data></Cell>
<Cell ss:StyleID="s16"><Data ss:Type="Number">2008</Data></Cell>
<Cell ss:StyleID="s17"><Data ss:Type="Number">9789639503397</Data></Cell>
<Cell><Data ss:Type="Number">2</Data></Cell>
</Row>
<Row>
<Cell ss:StyleID="s16"><Data ss:Type="Number">2</Data></Cell>
<Cell ss:StyleID="s16"><Data ss:Type="String">Alan Alexander Milne</Data></Cell>
<Cell ss:StyleID="s16"><Data ss:Type="String">Micimackó</Data></Cell>
<Cell><Data ss:Type="String">Holló Kiadó</Data></Cell>
<Cell ss:StyleID="s16"><Data ss:Type="Number">2010</Data></Cell>
<Cell ss:StyleID="s17"><Data ss:Type="Number">9789631187342</Data></Cell>
<Cell><Data ss:Type="Number">3</Data></Cell>
</Row>
<Row>

```

```

<Cell ss:StyleID="s16"><Data ss:Type="Number">3</Data></Cell>
<Cell ss:StyleID="s16"><Data ss:Type="String">Charlaine Harris</Data></Cell>
<Cell ss:StyleID="s16"><Data ss:Type="String">Inni és élni hagyni</Data></Cell>
<Cell><Data ss:Type="String">Alma Kiadó</Data></Cell>
<Cell ss:StyleID="s16"><Data ss:Type="Number">2009</Data></Cell>
<Cell ss:StyleID="s17"><Data ss:Type="Number">9789632542775</Data></Cell>
<Cell><Data ss:Type="Number">1</Data></Cell>
</Row>
<Row>
<Cell ss:StyleID="s16"><Data ss:Type="Number">4</Data></Cell>
<Cell ss:StyleID="s16"><Data ss:Type="String">George R. R. Martin</Data></Cell>
<Cell ss:StyleID="s16"><Data ss:Type="String">Trónok harca</Data></Cell>
<Cell><Data ss:Type="String">Figure Kiadó</Data></Cell>
<Cell ss:StyleID="s16"><Data ss:Type="Number">2008</Data></Cell>
<Cell ss:StyleID="s17"><Data ss:Type="Number">9789633707784</Data></Cell>
<Cell><Data ss:Type="Number">3</Data></Cell>
</Row>
</Table>
<WorksheetOptions xmlns="urn:schemas-microsoft-com:office:excel">
<PageSetup>
<Header x:Margin="0.3"/>
<Footer x:Margin="0.3"/>
<PageMargins x:Bottom="0.75" x:Left="0.7" x:Right="0.7" x:Top="0.75"/>
</PageSetup>
<Print>
<ValidPrinterInfo/>
<PaperSizeIndex>9</PaperSizeIndex>

```

```

<HorizontalResolution>200</HorizontalResolution>

<VerticalResolution>200</VerticalResolution>

</Print>

<Selected/>

<Panes>

<Pane>

<Number>3</Number>

<ActiveRow>19</ActiveRow>

<ActiveCol>2</ActiveCol>

</Pane>

</Panes>

<ProtectObjects>False</ProtectObjects>

<ProtectScenarios>False</ProtectScenarios>

</WorksheetOptions>

</Worksheet>

<Worksheet ss:Name="Munka2">

<Table ss:ExpandedColumnCount="1" ss:ExpandedRowCount="1" x:FullColumns="1"
x:FullRows="1" ss:DefaultRowHeight="15">

</Table>

<WorksheetOptions xmlns="urn:schemas-microsoft-com:office:excel">

<PageSetup>

<Header x:Margin="0.3"/>

<Footer x:Margin="0.3"/>

<PageMargins x:Bottom="0.75" x:Left="0.7" x:Right="0.7" x:Top="0.75"/>

</PageSetup>

<Print>

<ValidPrinterInfo/>

```

```

<PaperSizeIndex>9</PaperSizeIndex>

<HorizontalResolution>200</HorizontalResolution>

<VerticalResolution>200</VerticalResolution>

</Print>

<ProtectObjects>False</ProtectObjects>

<ProtectScenarios>False</ProtectScenarios>

</WorksheetOptions>

</Worksheet>

<Worksheet ss:Name="Munka3">

<Table ss:ExpandedColumnCount="1" ss:ExpandedRowCount="1" x:FullColumns="1"
x:FullRows="1" ss:DefaultRowHeight="15">

</Table>

<WorksheetOptions xmlns="urn:schemas-microsoft-com:office:excel">

<PageSetup>

<Header x:Margin="0.3"/>

<Footer x:Margin="0.3"/>

<PageMargins x:Bottom="0.75" x:Left="0.7" x:Right="0.7" x:Top="0.75"/>

</PageSetup>

<Print>

<ValidPrinterInfo/>

<PaperSizeIndex>9</PaperSizeIndex>

<HorizontalResolution>200</HorizontalResolution>

<VerticalResolution>200</VerticalResolution>

</Print>

<ProtectObjects>False</ProtectObjects>

<ProtectScenarios>False</ProtectScenarios>

</WorksheetOptions>

```

</Worksheet> </Workbook>

VI. számú melléklet: A program forráskódjának részlete

```
//  
// Konvertálás XML-be gomb  
  
//  
// Ha sikeres a betöltés  
if (workbook != null){  
  
    // Megszámoljuk mennyi sor van a munkafüzetben  
    Sheet sheet = workbook.getSheet(0);  
    int RowCount = sheet.getRows();  
  
    // Létrehozzuk az xml vázát  
    DocumentBuilderFactory dbfac = DocumentBuilderFactory.newInstance();  
    DocumentBuilder docBuilder = dbfac.newDocumentBuilder();  
    doc = docBuilder.newDocument();  
    Element root = doc.createElement("konyvek");  
    doc.appendChild(root);  
  
    // Végigmegyünk a munkafüzeten, és létrehozzuk az xml-ben a megfelelő tag-eket  
    // illetve beleírjuk a munkafüzetben található adatokat a tag-ek közé  
    for (int i=1; i<RowCount; i++){  
        Element konyv = doc.createElement("konyv");  
        root.appendChild(konyv);  
        for (int j=0; j<sheet.getColumns(); j++){  
            Element elm = null;  
            try{  
                //OptionPane.showMessageDialog(frame, sheet.getCell(j,0).getContents());  
                String str = sheet.getCell(j,0).getContents().replace(" ", "");  
                str = str.replace(";","");  
                str = str.replace(":", "");  
                str = str.replace("[", "");
```

```

        str = str.replace("]", "");
        str = str.replace("(", "");
        str = str.replace(")", "");
        str = str.replace(".", "");
        str = str.replace(",", "");
        str = str.replace("+", "");
        str = str.replace("\\", "");
        str = str.replace("-", "");
        str = str.replace("-", "");
        str = str.replace("*", "");
        str = str.replace("\\", "");
        str = str.replace("/", "");
        if (str.startsWith("1")) {
            str = "_" + str;
        }
        elm = doc.createElement(str);
    } catch (Exception ex) {
    }
    konyv.appendChild(elm);
    Text text = doc.createTextNode(sheet.getCell(j,i).getContents());
    elm.appendChild(text);
}
}

```

```

//
// Konvertálás ACCESS-be gomb
//

...
//Betöltjük a munkafüzetet
...
// Ha sikeres a betöltés
if (workbook != null){

    // Megszámoljuk mennyi sor van a munkafüzetben
    Sheet sheet = workbook.getSheet(0);
    int RowCount = sheet.getRows();

    // Létrehozzuk az xml vázát
    DocumentBuilderFactory dbfac = DocumentBuilderFactory.newInstance();
    DocumentBuilder docBuilder = dbfac.newDocumentBuilder();
    doc = docBuilder.newDocument();
    Element root = doc.createElement("dataroot");
    root.setAttribute("xmlns:od", "urn:schemas-microsoft-com:officedata");
    root.setAttribute("xmlns:xsi", "http://www.w3.org/2001/XMLSchema-instance");
    String filename = XLSfile.getAbsolutePath().toString().split("\\.")[0];
    filename += "_tulajdonsag.xsd";
    root.setAttribute("xsi:noNamespaceSchemaLocation", filename);

    DateFormat dateFormat = new SimpleDateFormat("yyyy/MM/dd HH:mm:ss");
    Date date = new Date();
    root.setAttribute("generated", dateFormat.format(date));
    doc.appendChild(root);

    // Végigmegyünk a munkafüzetben, és létrehozzuk az xml-ben a megfelelő tag-eket
    // illetve beleírjuk a munkafüzetben található adatokat a tag-ek közé
    for (int i=1; i<RowCount; i++){

```

```

Element konyv = doc.createElement("KONYV");
root.appendChild(konyv);
for (int j=0; j<sheet.getColumns(); j++){
    Element elm = null;
    try{
        //OptionPane.showMessageDialog(frame, sheet.getCell(j,0).getContents());
        String str = sheet.getCell(j,0).getContents().replace(" ", "");
        str = str.replace(";","");
        str = str.replace(":","");
        str = str.replace("[","");
        str = str.replace("]","");
        str = str.replace("(","");
        str = str.replace(")","");
        str = str.replace(".", "");
        str = str.replace(",","");
        str = str.replace("+","");
        str = str.replace("\\","");
        str = str.replace("-","");
        str = str.replace("-","");
        str = str.replace("*","");
        str = str.replace("\\","");
        str = str.replace("/","");
        if (str.startsWith("1")) {
            str = "_ " + str;
        }
        elm = doc.createElement(str.toUpperCase());
    } catch (Exception ex) {
    }
    konyv.appendChild(elm);
    Text text = doc.createTextNode(sheet.getCell(j,i).getContents());
    elm.appendChild(text);
}

```

```

    }
    //Generáljuk az XSD fájlt is
    GenerateXSD();
} catch (IOException e) {
    JOptionPane.showMessageDialog(frame, "Mentés közben hiba történt!");
}
} else {
    ...
    // Betöltjük a munkafüzetet
    Workbook workbook = null;
    JFrame frame = new JFrame();
    try {
        WorkbookSettings ws = new WorkbookSettings();
        ws.setEncoding("Cp1252");
        workbook = Workbook.getWorkbook(P_XLSFile,ws);
    } catch (IOException ex) {
        Logger.getLogger(KonvertaloView.class.getName()).log(Level.SEVERE, null, ex);

        // Létrehozzuk a file betöltő ablakot
        final JFileChooser fc = new JFileChooser();
        // Beállítunk egy filtert, hogy az xls fájlokat mutassa
        fc.setFileFilter(new ExtensionFilter("Excel munkafüzet", ".xls"));
        int returnVal = fc.showOpenDialog(frame);
        ...
        // Ha sikeres a betöltés
        if (workbook != null){

            // Megszámoljuk mennyi sor van a munkafüzetben
            Sheet sheet = workbook.getSheet(0);
            int RowCount = sheet.getRows();
            int ColCount = sheet.getColumns();

```

```

String xmlString = "<?xml version=\"1.0\" encoding=\"windows-1250\"?>" +
System.getProperty("line.separator");
    xmlString += "<xsd:schema xmlns:xsd=\"http://www.w3.org/2001/XMLSchema\"
xmlns:od=\"urn:schemas-microsoft-com:officedata\">" +
System.getProperty("line.separator");
    xmlString += "<xsd:element name=\"dataroot\">" +
System.getProperty("line.separator");
    xmlString += "<xsd:complexType>" + System.getProperty("line.separator");
    xmlString += "<xsd:sequence>" + System.getProperty("line.separator");
    String minOccurs = sheet.getCell(0,1).getContents().substring(11);
    String maxOccurs = sheet.getCell(0,2).getContents().substring(11);
    xmlString += "<xsd:element ref=\"\" + sheet.getCell(0,0).getContents() + \"\"
minOccurs=\"\"+ minOccurs + \"\" maxOccurs=\"\"+ maxOccurs + \"\"/>" +
System.getProperty("line.separator");
    xmlString += "</xsd:sequence>" + System.getProperty("line.separator");
    xmlString += "<xsd:attribute name=\"generated\" type=\"xsd:dateTime\"/>" +
System.getProperty("line.separator");
    xmlString += "</xsd:complexType>" + System.getProperty("line.separator");
    xmlString += "</xsd:element>" + System.getProperty("line.separator");
    xmlString += "<xsd:element name=\"konyv\">" +
System.getProperty("line.separator");
    xmlString += "<xsd:annotation>" + System.getProperty("line.separator");
    xmlString += "<xsd:appinfo>" + System.getProperty("line.separator");
    xmlString += "</xsd:appinfo>" + System.getProperty("line.separator");
    xmlString += "</xsd:annotation>" + System.getProperty("line.separator");
    xmlString += "<xsd:complexType>" + System.getProperty("line.separator");
    xmlString += "<xsd:sequence>" + System.getProperty("line.separator");
    for (int i=1; i<ColCount; i++){
        String type = "";
        minOccurs = sheet.getCell(i,1).getContents().substring(11);
        minOccurs = sheet.getCell(i,2).getContents().substring(11);
        String sqlType = sheet.getCell(i,3).getContents().substring(5);

```

```

String maxVal = null;
if (sheet.getCell(i,4).getContents() != ""){
    maxVal = sheet.getCell(i,4).getContents().substring(7);
}
if (sqlType.equalsIgnoreCase("Type=int")){
    type = "longinteger";
} else {
    type = "text";
}
xmlString += "<xsd:element name=\"" + sheet.getCell(i,0).getContents() + "\"
minOccurs=\"" + minOccurs + "\" od:jetType=\"" + type + "\" od:sqlSType=\"" + sqlType + "\"
type=\"" + sqlType + "\">" + System.getProperty("line.separator");
if (maxVal != null){
    xmlString += "<xsd:simpleType>" + System.getProperty("line.separator");
    xmlString += "<xsd:restriction base=\"" + sqlType + "\">" +
System.getProperty("line.separator");
    xmlString += "<xsd:maxLength value=\"" + maxVal + "\"/>" +
System.getProperty("line.separator");
    xmlString += "</xsd:restriction>" + System.getProperty("line.separator");
    xmlString += "</xsd:simpleType>" + System.getProperty("line.separator");
    xmlString += "</xsd:element>" + System.getProperty("line.separator");
} else xmlString += "</xsd:element>" + System.getProperty("line.separator");

}

xmlString += "</xsd:sequence>" + System.getProperty("line.separator");
xmlString += "</xsd:complexType>" + System.getProperty("line.separator");
xmlString += "</xsd:element>" + System.getProperty("line.separator");
xmlString += "</xsd:schema>" + System.getProperty("line.separator");
...

```

```

//
// Konvertálás MySQL-be gomb
//
...
// Ha sikeres a betöltés
if (workbook != null){

    // Megszámoljuk mennyi sor van a munkafüzetben
    Sheet sheet = workbook.getSheet(0);
    int RowCount = sheet.getRows();

    // Létrehozzuk az xml vázát
    DocumentBuilderFactory dbfac = DocumentBuilderFactory.newInstance();
    DocumentBuilder docBuilder = dbfac.newDocumentBuilder();
    doc = docBuilder.newDocument();
    Element root = doc.createElement("list");
    doc.appendChild(root);

    // Végigmegyünk a munkafüzeten, és létrehozzuk az xml-ben a megfelelő tag-eket
    // illetve beleírjuk a munkafüzetben található adatokat a tag-ek közé
    for (int i=1; i<RowCount; i++){
        Element konyv = doc.createElement("konyv");
        root.appendChild(konyv);
        Element elmid = doc.createElement("field");
        elmid.setAttribute("name", "konyv_id");
        konyv.appendChild(elmid);
        Text textid = doc.createTextNode(Integer.toString(i));
        elmid.appendChild(textid);
        for (int j=0; j<sheet.getColumns(); j++){
            Element elm = doc.createElement("field");
            elm.setAttribute("name", sheet.getCell(j,0).getContents().replace(" ", ""));
            konyv.appendChild(elm);
        }
    }
}

```

```
        Text text = doc.createTextNode(sheet.getCell(j,i).getContents());
        elm.appendChild(text);
    }
}
...
```

```

//
// Konvertálás XML-be gomb (Több tábla)
//
// Ha sikeres a betöltés
if (workbook != null){
    Sheet sheet = workbook.getSheet(0);

    // Meg kell keresni az összes 'Tablanev'-et
    int pos = 0;
    for (int i=1; i<sheet.getColumns()-1;i++){
        if (sheet.getCell(i, 1).getContents().equals("Tablanev")){
            pos++;
        }
    }
    pos++;
    int tablanevek[] = new int[pos];
    tablanevek[0] = 1;
    pos = 1;
    for (int i=1; i<sheet.getColumns()-1;i++){
        if (sheet.getCell(i, 1).getContents().equals("Tablanev")){
            tablanevek[pos] = i;
            pos++;
        }
    }
    for (int k=0; k<tablanevek.length;k++){
        int start = 1;
        int end = sheet.getColumns();
        String filename = null;
        if (k > 0) {
            start = tablanevek[k]+1;
        }
        if (k < tablanevek.length-1){

```

```

        end = tablanevek[k+1];
    }

```

// Létrehozzuk az xml vázát

```

DocumentBuilderFactory dbfac = DocumentBuilderFactory.newInstance();
DocumentBuilder docBuilder = dbfac.newDocumentBuilder();
doc = docBuilder.newDocument();
Element root = doc.createElement("konyvek");
doc.appendChild(root);

```

// Végigmegyünk a munkafüzeten, és létrehozzuk az xml-ben a megfelelő tag-eket

// illetve beleírjuk a munkafüzetben található adatokat a tag-ek közé

```

for (int j=2; j<sheet.getRows(); j++){
    filename = sheet.getCell(start-1,0).getContents();
    Element konyv = doc.createElement(sheet.getCell(start-1,0).getContents());
    root.appendChild(konyv);
    for (int i=start; i<end; i++){
        Element elm = null;
        try{
            String str = sheet.getCell(i,0).getContents().replace(" ", "");
            str = str.replace("; ", "");
            str = str.replace(":", "");
            str = str.replace("[", "");
            str = str.replace("]", "");
            str = str.replace("(", "");
            str = str.replace(")", "");
            str = str.replace(".", "");
            str = str.replace(",", "");
            str = str.replace("+", "");
            str = str.replace("\\", "");
            str = str.replace("-", "");
            str = str.replace("-", "");

```

```

        str = str.replace("*", "");
        str = str.replace("\\", "");
        str = str.replace("/", "");
        if (str.startsWith("1")) {
            str = "_" + str;
        }
        elm = doc.createElement(str);
    } catch (Exception ex) {
    }
    konyv.appendChild(elm);
    Text text = doc.createTextNode(sheet.getCell(i,j).getContents());
    elm.appendChild(text);
}
}
...

```