A UNICODE használata a bibliográfiai adatcserében

Horváth Ádám –
Lakatosné Takács Rita

Bevezetés

A bibliográfiai adatcsere formátuma valamilyen MARC formátum. A különféle MARC kézikönyvek részletesen szabályozzák, hogy a bibliográfiai adatokat milyen mezőkben, hogyan kell elhelyezni. Tartalmaznak pusztán számítástechnikai jellegű előírásokat is, például a mezők sorrendjéről a MARC rekordok mutató részében. Nem írják azonban elő, hogy az adatokat milyen kódkészletben kell szolgáltatni. Az alkalmazott kódkészletet a MARC állományokat kísérő szöveges dokumentációban kell közölni a cserepartnerrel.

A mágneses adathordozón tárolt szövegek számítástechnikai alapegysége az oktett. Az oktetten alapuló karaktertáblákban maximum 256 féle karaktert lehet megkülönböztetni. A világon előforduló karakterek száma tízezres nagyságrendű. A kettő között feszülő ellentét különösen érzékenyen érinti a könyvtárakat, melyek a világ minden tájáról gyűjtik irodalmukat. Hogyan lehet ilyen körülmények között többnyelvű szövegeket számítógéppel kezelni? Csakis kerülő úton. A 256-os korlát feloldására számtalan módszer született. Ezek egyike sem terjedt el széles körben. Ahány ház, annyi szokás. Az adatcsere mindenképpen karakterkonverzióval jár: annyiféle konverzióval, ahányféle cserepartnerünk van.

1991-ben a Unicode konzorcium kiadta a Unicode szabvány első verzióját. A Unicode 16 biten ábrázolja a karaktereket. Egy ilyen tábla több mint 65 000 karakter befogadására alkalmas. A Unicode elterjedése megszüntethetné a mai helyzet kuszaságát. Egy 16 bites ábrázolásra való áttérés azonban nem egyszerű feladat. Biztató jelek ellenére még nagyon messze vagyunk a Unicode általános elterjedésétől.

A következő fejezetekben áttekintjük a bibliográfia adatcsere összetevőit, majd bemutatjuk a Unicode alkalmazásának lehetőségét.

Kódkészletek

Az oktetten alapuló karaktertáblák között két nagy csoport alakult ki: az ASCII és az EBCDIC. Könyvtáros szempontból két lényeges különbség

van a kettő között. Az ASCII táblában gyakorlatilag csak 32 vezérlő karakter van, míg az EBCDIC-ben 64, ennek megfelelően az ASCII-ban több hely marad a tényleges karakterek számára. Ezen kívül az ASCII táblában a számok megelőzik a betűket, az EBCDIC-ben pedig fordítva. A könyvtári besorolás szempontjából az ASCII megoldása kedvezőbb.

Még mindig az oktett keretei között maradva, vannak ún. gyártói szabványok, melyeket széles körben való elterjedtségük miatt tekinthetünk szabványnak. Az IBM kidolgozta a PC DOS operációs rendszer számára a nemzeti karaktertáblákat. Ezek segítségével elérhető, hogy az egyes nemzetek saját nemzeti karaktereiket láthatják viszont a képernyőn. Egy-egy „nemzeti" karaktertábla több nemzetet szolgál. Így például a számunkra kitalált 852-es tábla a magyar mellett a cseh és a szlovák karaktereket is tartalmazza. De természetesen egy tábla nem szolgálhatja az összes európai nemzetet, ezért aztán a mi kódtáblánkba nem fértek bele például a francia karakterek. A PC DOS-os, tehát ASCII nemzeti tábláknak megvannak a megfelelői EBCDIC-ben is. Az OSZK például a 870-es EBCDIC táblát használja alap karaktertáblaként.

Az oktett határait kitágító módszerek némelyike nemzetközi vagy nemzeti szabvány szintjére is emelkedett. Ilyen például az *ISO 2022*-es szabvány, melynek magyar megfelelője a *„Számítástechnikai kódrendszerek: Kódkiterjesztés"* címet viseli. Ez a szabvány leírja, hogy hogyan lehet 7 és 8 bites környezetben az alap karaktertáblát a felhasználó által tervezett táblákkal lecserélni, illetve az alap és a felhasználói táblák között szabadon mozogni. A módszer segítségével elvileg korlátlan számú tábla között lehet váltani. Tiszta megvalósítása alig ismert.

A kódkiterjesztés elterjedt módszere a karaktersorozatok használata. NIIF körökben jól ismert az ún. gizmós ábrázolás. Az alap karaktertáblába nem tartozó karaktereket három karakteren fejezzük ki. A @ jelzi, hogy most valami speciálisan értelmezendő karakterek következnek. A @ karakter utáni két karakter együttesen reprezentálja az alap táblából hiányzó karaktert. Elvileg így nagyon sok karaktert kifejezhetünk. A gizmós karaktereket tartalmazó szöveg nehezen olvasható. A gizmós ábrázolást megengedő programok általában úgy viselkednek, hogy a gizmót és az azt követő karaktert kérésre nem jeleníti meg a szöveg kiírásakor. Az így megmaradó karakter utalni szokott arra a karakterre, melyet a gizmós ábrázolás részeként kifejez. Ezzel a megkötéssel a variációk száma már lecsökken. (Itt jegyezzük meg, hogy az OSZK-ban használt DOBIS/LIBIS belső karakterábrázolása is ennek a kódolási típusnak egyfajta, bonyolultabb változata.)

Ugyancsak elterjedt módszer a repülő ékezetek használata. Ebben az esetben olyan karaktertáblát terveznek, mely tartalmazza az alapkaraktereket és a leggyakrabban használt ékezeteket. Az ékezetes karaktereket aztán két oktetten ábrázolják. Először jön az ékezet, majd az alapbetű. A megfelelően felkészített programok az így rögzített szövegek megjelenítésekor a kettőt egymásra nyomtatják. Ezzel a módszerrel sem lehet mindent megoldani, mert az összes alapkaraktert és az összes ékezetet nem lehet egy táblázatban felsorolni.

A könyvtári szoftverek világában nem terjedt el egyik módszer sem kizárólagosan. Sőt ennél is rosszabb a helyzet: ahány szoftver annyiféle kódkiterjesztés. Ha csak Magyarországra szűkítjük le a kérdést, akkor azt láthatjuk, hogy a hazánkban elterjedt integrált rendszerek szinte mindegyike más módszert használ. Ráadásul a módszerek általában olyanok, hogy a kódkiterjesztés tábláit maguknak a felhasználóknak kell elkészíteniük. Így még az azonos módon dolgozó szoftverek konkrét alkalmazásai is más és más tartalmú táblákat használnak. Egy-egy konkrét alkalmazáson belül maradva nincs is különösebb baj: a szoftverek ha különböző módokon is, de általában jól, a felhasználó megelégedésre oldják meg az extra karakterek tárolását és kezelését. A baj akkor kezdődik, amikor az adatokat ki kell cserélni, mivel nincs két rendszer, mely ugyanazt a kódkészletet használná.

Unicode

A Unicode első verzióját az amerikai Unicode konzorcium készítette el és adta ki *„The Unicode Standard: Worldwide Character Encoding, Version 1.0"* címmel. A kidolgozásban többek között a Research Library Group, a Microsoft, az IBM, a Novell, Sun Microsystem, a WordPerfect,

a Xerox szakemberei vettek részt. A szabványt nevezik *UCS–2*-nek (Universial Character Set – 2 byte) is. A Unicode szabvány részhalmaza az ISO által 10646-nak nevezett szabványtervezetnek, mely 32 biten kódolja a karaktereket. Ez utóbbit nevezik *UCS–4*-nek is.

A Unicode egy 16 bites fix hosszúságú kódokat tartalmazó szabvány. Jelenleg 28 000 karaktert tartalmaz a maximálisan kiadható több mint 65 000-ből. A teljes kódtartomány a karakterek ábrázolására van fenntartva. A szabvány ugyanakkor kompatibilitási okokból tartalmazza a legelterjedtebb szabványokban leírt karaktereket és vezérlő kódokat.

Az ékezetes karaktereket megtaláljuk a Unicode-ban statikus, előre definiált formában, ha azok előfordultak a fent említett szabványok valamelyikében. Pl. az előre elkészített *Á* betű Unicode-ja *U+00C1*. * A Unicode szabvány felsorolja az összes ékezetet, melyek segítségével bármelyik ékezetes karakter előállítható. Az ékezetek az alapkarakter után következnek. Ez a repülő ékezetes módszer. Az *Á* repülő ékezetes kódja: *U+0041* + *U+0301*, ahol *U+0041* az *A* betű kódja, a *U+0301* a ferde ékezeté (*acute*). Egy alapkarakterre több ékezet is feltehető.

A Unicode bevezetése nem egyszerű feladat. Az adatbázisok nagyságát rögtön megkétszerezi, nem állnak maradéktalanul rendelkezésre a megjelenítéshez szükséges betűkészletek és nem megoldott a Unicode karakterek bevitele sem, hogy csak a legnyilvánvalóbb nehézségeket említsük.

A Unicode-ra épülő alkalmazások száma a szabvány megjelenését követő tétova évek után egyre erőteljesebben növekszik. A Microsoft operációs rendszerekben régóta jelen van, igaz a felhasználó elől elrejtve. A Novell is Unicode-ot használ a fájlnevek tárolására. A programnyelvekben egyre több Unicode-ot kezelő rutin jelenik meg. Piacra kerültek az első Unicode-os szövegszerkesztők, mint például a WinCalis. Az Interneten az *UTF–8* kódolás kezd terjedni, mely a Unicode szabvány megvalósítása változó hosszúságú kódokkal. A Unicode-os WWW böngészők is az UTF–8 kódolást tudják kezelni.

Könyvtári szoftverek készítői kezdetben azon az álláspontón voltak, hogy a Unicode kezelése az operációs rendszer feladata, és az alkalmazói szoftverbe csak ezután kerülhet be. Tipikus helyzet: mindenki a másiktól várja az első lépés megtételét. Mára azonban módosulni látszanak a vélemények és egyre több cégről hallani, hogy tervezik a Unicode bevezetését saját hatáskörükön belül. Hiába létezik tehát a Unicode, könyvtári megjelenésére még mindenképp várnunk kell egy-két évet.

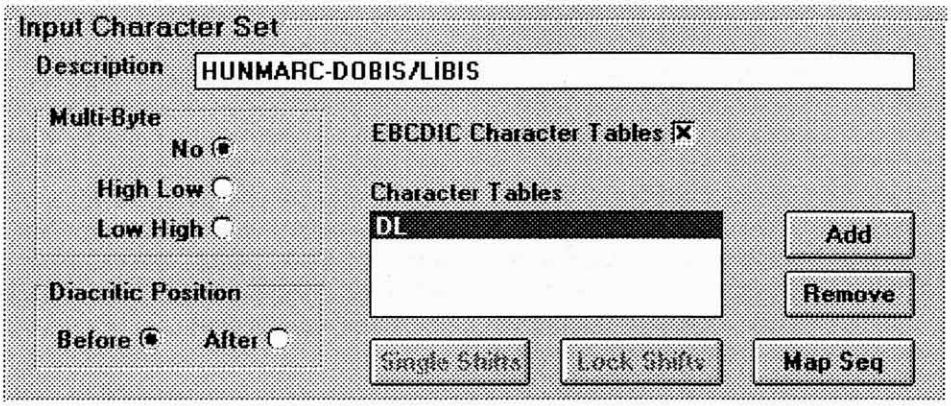
Adatcsere-formátum

Minden MARC formátum – jelen esetben a bibliográfiai adatcsere formátumot értve MARC alatt – összetevői: rekordszerkezet, a tartalomjelölők és maga az adattartalom. Megegyezés csak a rekordszerkezetben történt, az *ISO 2709*-et (*„Format for bibliographic information interchange on magnetic tape"*) szabványként fogadták el az adatrekordok szerkezetére vonatkozóan. Ezt a szabványt a Magyar Szabványügyi Hivatal is honosította: *„MSZ 193/1-83 Mágnesszalagos bibliográfiai adatcsere formátuma. A rekordok szerkezete"* címen.

Egy ISO 2709-es szerkezetű rekord három fő részből áll: rekordfejből, a mutatóból és az adatmezőkből. A rekordfejben a mutató rész felépítésére vonatkozóan is szerepel információ. A mutató részben van leírva, hogy a rekord adatmezők részében milyen hívójelű mezők milyen hosszan és hol találhatóak. Maguk az adatok az adatmezők részben foglalnak helyet. Az ISO 2709 nem, de mind a USMARC, mind a UNIMARC és természetesen a HUNMARC is előírja, hogy a mutató részben az adatmezőket hívójelük növekvő sorrendjében kell felsorolni. Az adatrészben a mezőknek nem kell ebben a sorrendben következniük, hiszen a rekordon belüli helyüket a mutató részből egyértelműen meg lehet határozni.

Sem az ISO 2709 szabvány, sem a MARC kézikönyvek – mint a bevezetőben már volt róla szó – nem írják elő az alkalmazandó karakterkészletet.

* A Unicode karakterekre a szabványos hivatkozás a következő: *U + a Unicode karakter hexadecimális megfelelője.*



1. ábra

CHASE

A CHASE egy általános célú karakterkonvertáló program. Jellegzetessége, hogy képes oktettes karaktereket Unicode-ra konvertálni. Az oktettes karakterkészlet lehet ASCII és EBCDIC és ezeken belül:

- ISO 2022-es szerkezetű,
- repülő ékezetes,
- karaktersorozatos kódkiterjesztést tartalmazó.

A Unicode-os ékezetes karakterek lehetnek előre definiált típusúak és repülő ékezetesek. A CHASE szoftverrel sima szövegfájlokat és ISO 2709-es fájlokat is lehet konvertálni. Ez utóbbi esetben a CHASE a rekordfejet is, a mutatót is Unicode-osítja és a mutatóban a kezdőpozíciókat és hosszakat is átszámolja az adatmezők konverziója eredményének megfelelően.

Az oda-vissza konverzió egy konverziós tábla alapján történik. A konverziós tábla készítését külön program segíti. A CHASE program DOS alatt fut, de már készül a UNIX-os verziója. A táblaszerkesztő Windows-os program. A konverziós tábla maga – amit a táblaszerkesztő létrehoz – egy egyszerű szövegfájl, melyet bármilyen szövegszerkesztőbe be lehet olvasni.

A CHASE az Európai Unió COBRA kutatási programjának keretében készült. A CHASE mozaikszó, melynek feloldása: **Cha**racter **S**et for **E**urope. A szoftver tesztelésében az OSZK is részt vett.

A táblaszerkesztő program

A karakterkészlet definiálása

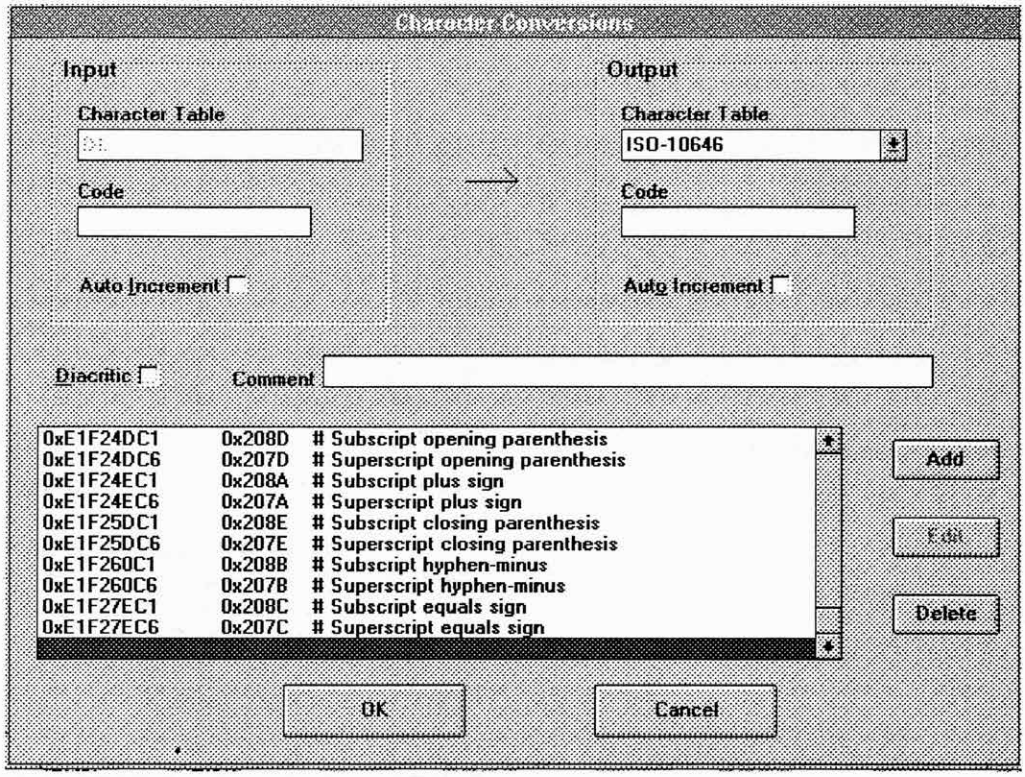
A táblaszerkesztő program segítségével tudjuk definiálni a konverzió input és output oldalának jellegzetességeit. Maga a tábla ezért két, egymástól jól elkülönülő részre osztható, amelyek egymás alatt helyezkednek el. A CHASE egy általános konvertáló program, mely egyaránt képes egy bájtos* valamint egy és több bájtos kódkészletek között konvertálni, ezért mindkét táblarészt külön-külön ki kell tölteni.

Először is meg kell neveznünk a konvertálásban résztvevő kódkészleteket, vagyis ki kell töltenünk a *„Description"* mezőt, melybe a konvertálandó karakterkészlet megnevezésére szolgáló szöveges információ kerül (*1. ábra*).

Miután megneveztük a konverzióban résztvevő karakterkészleteket, ezek jellemzése a következő mezők kitöltésével történik:

- „Multi-Byte",
- „Diacritic Position",
- „EBCDIC Character Tables",
- „Character Tables".

* A később következő angol nyelvű ábrákhoz jobban illik a bájt szó. Ezért ettől a fejezettől az oktettről áttérünk a bájt megnevezésre.



2. ábra

- *Multi-Byte*

A *„No"* kijelölésével jelezzük, hogy a definiálandó karakterkészlet egy bájtos. Amennyiben két bájtos karakterkészletet szeretnénk definiálni a *„High Low"* vagy pedig a *„Low High"* gombokat kell kiválasztanunk, ezzel jelezve a bájtok tárolási sorrendjét. (*1. ábra*)

Diacritic Position

Abban az esetben, ha a definiálandó kódolás használja a repülő ékezetes kódkiterjesztést, a programnak tudnia kell, hogy az ékezetek követik vagy megelőzik az alapkaraktert. A repülő ékezetek általában megelőzik az alapkaraktert. A Unicode azonban szakított ezzel a hagyománnyal és a repülő ékezetet az alapkarakter után helyezi el, vagyis az *„After"*-t kell kiválasztanunk. A konverzió definiálásakor adhatjuk meg, hogy mely karaktereket tekintse a program ékezeteknek. Amennyiben a definiált karakterkészletben nincsenek repülő ékezetek – mint ahogyan a DOBIS/LIBIS táblában sincs –, akkor a kijelölésnek nincs jelentősége.

- *EBCDIC Character Tables*

Amennyiben a definiálandó karakterkészlet EBCDIC kódolású, így az ezt jelölő négyzetet ki kell jelölnünk.

- *Character Tables*

Az Input, illetve az Output karakterkészlet állhat egy vagy – a helyi karakterkészlet sajátosságainak megfelelően – több táblából, melyeket a *„Character Tables"* mezőben sorolunk fel. A tábla neve a konverziós program számára szolgáló belső azonosító, ezért tetszőleges elnevezéssel utalhatunk a táblára (pl. DOBIS/LIBIS helyett csak *„DL"*-nek neveztük el a táblánkat). A táblákat az *„Add"* és a *„Remove"* gombokkal lehet hozzáadni, illetve eltávolítani a mezőből.

Az ISO 2022-es szabvány esetén a *„Single Shift"* és a *„Lock Shift"* gombokkal hozzá lehet rendelni az egyes táblákhoz a megfelelő egyszeres vagy reteszelt átváltó karaktersorozatokat. EBCDIC szabvány esetén, amennyiben karakterkészletünk több táblából áll, az egyes táblákat lehívó karaktersorozatokat a *„Map Seq"* gomb se-

gítségével lehet megadni. A *„Map Seq"* gomb használata ASCII kódkészlet esetén is megengedett.

Ha az output karakterkészlet Unicode, akkor a *„Character Tables"* mezőben mindig egy tábla (vagyis maga a Unicode tábla) szerepel, amelyet mi *„ISO–10646"*-tal jelöltünk. A táblához természetesen nem tartozik átváltó karaktersorozat. A *„Diacritic Position"* pedig mindig *„After"*. A bájtok tárolási sorrendjét (*„High Low"* / *„Low High"*) a helyi operációs rendszer sajátosságainak megfelelően kell kiválasztani.

A konverzióban részt vevő kódkészletek definiálása után elmenthetjük a beállításokat, kiléphetünk a táblaszerkesztőből, vagy megkezdhetjük a konverziós tábla kitöltését.

A konverzió definiálása

A konverziót definiáló ablakban (*2. ábra*) látható az előzőekben leírt és kiválasztott input és az output karaktertábla megnevezése. Most már csupán a kódokat kell bevinnünk. Először a kódokat írjuk be hexadecimális formában, majd kitöltjük a *„Comment"* mezőt: ide kerül a karakter szabványos megnevezése. (*2. ábra*)

Az *„Auto Increment"* funkciót akkor érdemes kiválasztanunk, ha a kódokat folyamatosan, értéküket mindig eggyel növelve kívánjuk bevinni. Ilyenkor ugyanis a program automatikusan az előző érték után következő kódot ajánlja fel.

Amennyiben a helyi kódkészletben repülő ékezetet használunk, akkor a program számára jeleznünk kell, hogy melyik kód az ékezet; ezt a *„Diacritic"* bejelölésével tehetjük meg.

A redőnyös ablakban láthatjuk* a korábbi bevitelek eredményeképpen létrejött konverziós táblát, melyhez hozzáadhatjuk a következő karakter leírását, illetve amelyből törölhetjük a téves vagy feleslegessé vált kódokat. Az így bevitt kódok összességeként hozzuk létre azt a **.chs* kiterjesztésű szövegfájlt, melyet a konverziós program felhasznál.

* Az ábrán éppen karaktersorozat - Unicode konverzió definiálása látható.

Karakterkonverzió

A konverziót a CHASE program végzi. A program indítása:

„chase -st hsldobis.chs -i hsldobis.iso -o hsldobis.uni"

```
Conversion status . . . . . - Successful
Input file . . . . . - hsldobis.iso
Output file . . . . . - hsldobis.uni
Translation table . . . . . - hsldobis.chs
ISO 2709 . . . . . - Yes
Number of records . . . . . - 646
Initialisation time . . . . . - 0 seconds
Conversion time . . . . . - 4 seconds
Avg time per record . . . . . - 0.0062 seconds
```

Mint láthatjuk, parancssor argumentumokkal adhatjuk meg az input, output és a konverziós táblát tartalmazó fájlokat. Az *„s"* kapcsolóval kértük a statisztika kiírását. Fordított irányú konverzió esetén a hívás: *„chase -sxt hsldobis.chs -i hsldobis.uni -o hsldobis.iso"*.

A konverziós tábla készítésének rejtelmei

A konverziós tábla kitöltésének általános leírása után érdemes a felmerült problémák közül egy-kettőt azok általunk választott megoldását megemlítenünk, megosztva ezzel saját tapasztalatainkat, és talán ötletet adva más, helyi sajátosságok kezeléséhez.

- *Repülő ékezet vagy előre definiált karakter (combined / pre-combined)*

Mint az előzőekben láttuk, a Unicode szabvány szerint az ékezetes karaktereket kétféleképpen is le lehet írni. Mi minden esetben a repülő ékezetes megoldást választottuk, mert ezzel a módszerrel azokat a karaktereket is le lehet írni, melyeket a Unicode szabvány nem sorol fel előre definiált formában.

- *Indexek*

A DOBIS/LIBIS karakterkészletében külön szerepel a magyar ábécé valamennyi betűjének, valamint a számoknak és egyes szimbólumoknak alsó, illetve felső indexként való leírása is. A Unicode szabvány azonban ezzel a problémával tudatosan nem foglalkozik, azt szövegszerkesztési

feladatnak tekinti, így külön kódolását sem végezte el. Vannak ugyan kivételek: a számok, valamint bizonyos szimbólumok (melyek meglévő szabványokban is szerepelnek) alsó és felső indexe helyet kapott a Unicode táblában. Ezek a kódok azonban nem összetettek, nem magából a karakterből + az alsó vagy felső index jelölésére szolgáló kódból állnak, ezért utólagos képzésük nem lehetséges. A Unicode-ban azonban vannak úgynevezett szabad helyek (Private Use Area), melyeket éppen az ilyen egyéni megoldások számára hagytak üresen; itt szabadon definiálhattuk saját, a Unicode-ok között nem szereplő karaktereinket. Ez a terület a Unicode tábla *E800* és *FDFF* kódhelyei között található. A szabad helyek felhasználását legalább országos szinten összehangoltan kellene végezni.

Bibliográfiai adatszolgáltatás az OSZK-ban

Az Országos Széchényi Könyvtár (OSZK) a *„Nemzeti Bibliográfia: Könyvek Bibliográfiája"* című periodikum anyagát kéthetes periodicitással HUNMARC formátumban szolgáltatja előfizetői számára. A MARC rekordok a könyvtár integrált rendszeréből, a DOBIS/LIBIS-ből származnak. Jelenleg négyféle, széles körben elterjedt karakterkészlettel állítjuk elő.* Közös jellemzője mind a négy kódolásnak, hogy a DOBIS/LIBIS belső karakterkészletét a legteljesebbnek szánt változatban sem tudjuk maradéktalanul átmenteni. A többiben szándékosak az elhagyások: olyan rendszerek számára készültek, melyek nem ismerik a kódkiterjesztés módszerét.

Véleményünk és tapasztalataink szerint a CHASE programra támaszkodva be lehetne vezetni a Unicode-os HUNMARC adatszolgáltatást. Ezzel biztosítani lehetne, hogy a felhasználó a DOBIS/LIBIS, vagy bármely más adatszolgáltató saját rendszerének teljes karakterkészletét megkapja, és ezek után maga döntsön arról, hogy abból mit vesz át abban az esetben, ha nincs szüksége a teljes készletre, hiszen a Unicode-ról a helyi készletre való konverzió a felhasználás helyszínén történne.

Amennyiben el tudnánk érni, hogy a hazai bibliográfiai adatcsere mindig Unicode-ban történjen, akkor az összes magyar könyvtárnak ebben a vonatkozásban csak egy konverziós táblával kellene rendelkeznie: a helyi karakterkészletről Unicode-ra és Unicode-ról a helyi karakterkészletre konvertáló egyetlen táblára.

* A Nemzeti Bibliográfia füzeteit ASCII+ (közismertebb nevén: OCLC karakterkészlet), 852+ (a 852-es kódtáblára alapozott gizmós karakterkészlet), ISO 8859/1, ISO 8859/2 karakterkészletekben szolgáltatjuk.